Human Pose Estimation for Robot Safety

Zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

von der KIT-Fakultät für Informatik
des Karlsruher Instituts für Technologie (KIT)

genehmigte

Dissertation

von

Patrick Stefan Schlosser

Tag der mündlichen Prüfung: 28.10.2024

1. Referent: Prof. Dr. Tamim Asfour
2. Referent: Dr. Torsten Kröger

To my family, friends and colleagues.

Without you, this would have not been possible.

Abstract

Human Pose Estimation for Robot Safety

In modern industrial manufacturing, the interaction of humans with robots becomes increasingly important, as each can add individual strengths to the manufacturing process. To be able to work together, humans and robots may no longer be physically separated through the likes of safety fences. Without such a physical separation, human safety needs to be ensured by other means. A standard procedure is to monitor the distance between the human and the robot, stopping the robot safely if the human comes too close. Typically, distance monitoring is done through a safety laser scanner, which only monitors the location of human legs. A more complete picture could be obtained through human pose estimation, which detects keypoints across the whole human body.

The use of human pose estimation in its current form for such safety-critical, industrial robot applications is prohibited by several factors. Most important are the requirements of relevant safety standards for a sufficiently low error rate as well as hard real-time capability. Both must be achieved before human pose estimation can be used in safety-critical industrial applications. In this work, both of these factors are addressed. Methods to reduce errors in human pose estimation are introduced, which are used to detect and filter out incorrect keypoint predictions from human pose estimation methods. As each predicted keypoint position is affected by a measurement error of some degree, it is also necessary to know the worst-case magnitude of the measurement error for a keypoint prediction. Only then can the potential inaccuracy of a keypoint prediction be considered by a safety-critical application. Therefore, additional methods are introduced to predict an upper bound for the measurement error of each individual keypoint prediction. As a factor that can significantly increase the occurrence of incorrect results, noise is specifically considered in this work. Its negative effect on human pose estimation and methods of this work is highlighted. To counteract its negative effect, several countermeasures are investigated. In addition to the previous contributions that focus more or less directly on the error rate in human pose estimation, the hard real-time capability is also addressed. To achieve hard real-time capability, a method is introduced that bridges the time between the arrival of human pose estimation results and allows for a safe, hard real-time capable distance calculation between human and robot.

Most methods are evaluated through experiments on the MPII Human Pose dataset. Methods for error reduction were able to significantly reduce previously undetected, false human pose estimation results. The prediction of correct and high-quality upper bounds for the measurement error of keypoints predictions was possible in most cases. Furthermore, the impact of noise on human pose estimation and the methods of this work was greatly reduced. The proposed method for achieving hard real-time capability was evaluated through a theoretical comparison against a safety laser scanner. The results indicate an advantage over this de-facto industrial standard for distance monitoring.

Zusammenfassung

Human Pose Estimation für sichere Robotik

Für die moderne industrielle Fertigung wird die Interaktion zwischen Mensch und Roboter zunehmen wichtig, da so beide ihre individuelle Stärken zum Fertigungsprozess beitragen können. Um die Zusammenarbeit zwischen Mensch und Roboter zu ermöglichen, dürfen sie nicht physisch getrennt werden, wie es bei Sicherheitszäunen und ähnlichen Vorrichtungen der Fall ist. Ohne physische Trennung muss die Sicherheit des Menschen jedoch nach wie vor gewährleistet werden. Ein gängiges Vorgehen um dieses Ziel zu erreichen ist die Abstandsüberwachung zwischen Menschen und Roboter, bei der der Roboter vollständig gestoppt wird, falls der Mensch ihm zu nahe kommt. Üblicherweise wird die Abstandsüberwachung mittels eines Sicherheits-Laserscanners realisiert, welcher lediglich die Position der menschlichen Beine detektiert. Um ein vollständigeres Bild der Position des gesamten menschlichen Körpers zu erhalten, könnten Verfahren zur menschlichen Posenschätzung - der Human Pose Estimation - eingesetzt werden, welche Schlüsselpunkte des gesamten Körpers - sogenannte Keypoints - detektieren.

Aktuell ist der Einsatz von Human Pose Estimation in sicherheitskritischen Teilen industrieller Robotikanwendungen aufgrund verschiedener Faktoren noch nicht möglich. Maßgeblich sind vor allem zwei Faktoren, die sich aus relevanten Sicherheitsnormen ergeben: Eine Fehlerquote, die gering genug ist, und harte Echtzeitfähigkeit. Im Rahmen dieser Arbeit werden beide Punkte adressiert. Es werden Methoden zur Fehlerreduktion in der Human Pose Estimation vorgestellt, welche zur Detektion und dem Entfernen von falschen Ergebnissen für die Position von Keypoints eingesetzt werden. Darüber hinaus ist jede Position die für einen Keypoint bestimmt wird mit einem Messfehler variabler Größe behaftet. Damit die entsprechende Ungenauigkeit bei der Lokalisierung von Keypoints im Rahmen sicherheitskritischer Anwendungen berücksichtigt werden kann, ist es nötig, dass der größtmögliche Messfehler bekannt ist. Um dieses Problem zu lösen werden zusätzliche Verfahren eingeführt, die für jede berechnete Position eines Keypoints eine individuelle Obergrenze für den Messfehler bestimmen. Ein Faktor der das Auftreten falscher Ergebnisse stark begünstigen kann ist das Vorhandensein von Rauschen, welches daher im Rahmen dieser Arbeit ebenfalls behandelt wird. Es werden verschiedene Maßnahmen untersucht, um den negativen Effekt von Rauschen auf Human Pose Estimation und die im Rahmen dieser Arbeit vorgestellten Methoden zu verringern. Zusätzlich zu bisherigen Beiträgen, die direkt oder indirekt auf die Verringerung der Fehleranfälligkeit abzielen, wird das Problem der harten Echtzeitfähigkeit behandelt. Hierfür wird ein Verfahren eingeführt, das die Zeit zwischen einzelnen Ergebnissen der Human Pose Estimation sicher überbrückt. Darüber hinaus ermöglicht das Verfahren die hart echtzeitfähige Distanzberechnung zwischen Mensch und Roboter.

Der Großteil aller Verfahren wird auf dem MPII Human Pose Datensatz evaluiert. Methoden zur Fehlerreduktion waren in der Lage die Menge bisher unentdeckter, falscher

Ergebnisse für Keypoints signifikant zu reduzieren. Beim Bestimmen individueller Obergrenzen für den Messfehler war es in den meisten Fällen möglich korrekte und qualitativ hochwertige Obergrenzen zu berechnen. Der Einfluss von Rauschen auf Human Pose Estimation und die Verfahren dieser Arbeit konnte durch geeignete Gegenmaßnahmen signifikant reduziert werden. Das vorgeschlagene Verfahren zur Echtzeitfähigkeit wurde im Gegensatz zu den anderen Verfahren durch einen theoretischen Vergleich mit einem Sicherheits-Laserscanner evaluiert. Dieser theoretische Vergleich zeigte Vorteile des Verfahrens gegenüber dem Sicherheits-Laserscanner auf, welcher de-facto ein Industriestandard ist.

Contents

1 Introduction

The manufacturing of goods has come a long way since its beginning: Initially, artisans manufactured individual goods using simple tools. This individual manufacturing practice was increasingly replaced by a more standardised manufacturing process and product design, first through the introduction of machines and later through the use of assembly lines, sacrificing product customization for higher productivity. This process reached its peak through the introduction of industrial robots in such assembly lines. These robots further increased the productivity and efficiency of the manufacturing process.

In contrast to most other machinery, robots had the drawback that humans could not work safely alongside them due to their rather unpredictable movements and the high forces they can exert. This led to the physical separation of humans and robots, most commonly through the use of safety fences, preventing humans from entering the robot workspace. As a result, the product had to leave the restricted robot working space whenever the human had to perform a task and it had to reenter it for the robots to continue. Naturally, this facilitates long sequences of either robot or human tasks, to keep these transitions to a minimum. This makes the production process more rigid and prevents the use of human dexterity and adaptability for single steps during the robotic task sequences in an economical way.

Nowadays, manufacturers aim to increase product individualization while maintaining productivity. This shall be achieved by breaking with the paradigm of human-robot separation in factories to increase the flexibility of the production process and to combine the advantages of humans and robots in human-robot collaboration. However, this desire for a shared human-robot workspace poses new challenges to human safety, which must be ensured at any time. One way to address the danger of robots in a shared environment is to use robots that limit the force they can exert on humans, thus preventing injuries in the event of a collision. However, such robots are only suitable for a subset of industrial tasks. In the general case, it has to be assumed that the robot can injure the human if a collision occurs. Here, it is necessary to measure and monitor the human's position and to intervene before a collision occurs, e. g., by safely stopping the robot when the human-robot distance becomes too small.

When safeguarding a robot by monitoring the human-robot distance, the human detection system must be reliable enough. This means that the system must not endanger the human, e. g., because its results are incorrect or delayed. The strict requirements for such systems are formalized in laws and standards and must be met before any system can be deployed. This has led to the development of several specialized safety systems used for human detection in safety-critical scenarios and imposes a high barrier to the introduction of new technologies in this domain.

1.1 Motivation

Traditional safety devices for detecting an approaching human include e. g., safety mats, light curtains, and laser scanners, where the latter can be viewed as a de facto industry standard nowadays. A safety laser scanner can be used to monitor a two-dimensional area called a *detection zone* [46], determining whether people and/or objects are inside. This is especially useful when the monitored area is located around the robot's workspace and is parallel to the directions from which humans can approach. In such a case, the robot's behavior can be safely adjusted based on the distance of the human to the robot's workspace. One common and safe way to adapt the robot's behavior is *speed and separation monitoring* (SSM) [48, 51], where the robot is slowed down as a human approaches, up to a full stop of the robot if the human comes too close.

When using a safety laser scanner for SSM, the distance between the human and the robot's workspace is evaluated based on the intersection of the human with the two-dimensional scan plane. This approach has two drawbacks:

1. A human entering the two-dimensional detection zone is detected at the position where his body intersects with the detection zone. However, an intersection in two-dimensional space is not representative of the whole body and thus not of the true distance between the human and the robot's workspace. For safety, the worst case has to be assumed: The human is significantly closer to the robot than the detected position suggests, i. e., the human stretches his hand towards the robot. As a result, the robot has to slow down and stop earlier than it would be necessary in most cases, since the full, three-dimensional position of the body is not known.

2. A laser scanner indiscriminately detects any intersection of a large enough object or being within the detection zone. This means that not only humans, but also autonomous vehicles and other objects can trigger a slowdown or safety stop when entering the detection zone, resulting in unnecessary loss of productivity.

Recent developments in safety technology try to address these shortcomings: The PILZ SafetyEYE® [97] is a safe camera system that can be placed above a work cell that requires monitoring. In contrast to the safety laser scanner, it can monitor 3D space, resolving the first downside. Safety radar systems like safeRS [114] go in a similar direction: They can monitor a 3D volume directly in front of them, originating from the radar device and contained by horizontal and vertical aperture angles. Similar to the SafetyEYE®, they address the first drawback. In the case of safeRS, the second drawback is also partially addressed by the capability to filter out falling objects [114]. The most recent addition to the field of safety technology is the Veo FreeMove® [126] safety system. It uses multiple time-of-flight cameras to monitor a robot work cell in 3D. Furthermore, it can discriminate some large, tracked objects, like workpieces, from humans, thus not only addressing the first drawback but also starting to work on the second one.

Potential solutions that could address both drawbacks sufficiently can be found in the field of computer vision. One of these potential solutions is *human pose estimation* (HPE). Methods from this domain detect important keypoints of the human body, which are distributed all over the human body and thus approximate the body position as a whole. Furthermore, the methods in this domain are explicitly designed for human detection and do not suffer from the risk of detecting objects and other entities due to the design of the

detection mechanism. However, recent methods based on neural networks are neither reliable enough nor suitable for hard real-time capability, making them currently inapplicable in the safety domain.

1.2 Goal

The goal of this work is to bring human pose estimation closer to being applicable in the safety domain by making human pose estimation methods more reliable. The focus lies on the SSM application, where human pose estimation could perform the task of determining a human's position. Achieving higher reliability involves addressing several safety-relevant aspects, including (i) the reduction and/or detection of previously undetected errors made by human pose estimators, (ii) the determination of upper bounds for the distance between keypoint detections and actual keypoint locations (i. e., an estimation of measurement errors) and (iii) the achievement of hard real-time capability.

Whenever possible, these aspects are addressed for the basic task of 2D single-person human pose estimation in single images. The whole area of human pose estimation is much broader: The task can be performed in 2D and 3D, for single or multiple people, and based on single data points or data sequences. However, since little attention has been paid to the safety of human pose estimation so far, it is reasonable to address the task in its most fundamental form before considering more advanced variants, which bring their own opportunities for improving reliability, but also additional challenges.

Throughout this work, the following research questions are addressed:

- How can previously undetected errors of a human pose estimator be detected, ideally in a general way without making any assumptions about the internal functionality of the human pose estimator?
- How can an upper bound for the distance between a keypoint detection and the actual keypoint location be determined?
- How can noise occur and affect human pose estimation in safety-critical industrial robot applications, and how can its impact on human pose estimation be handled?
- Assuming that the typical human pose estimator will not be capable of hard real-time, how can hard real-time capability be ensured for human-robot distance monitoring in SSM based on human pose estimation results?

1.3 Outline

The content of this thesis is organised as follows:

Fundamentals: In Chapter 2, the common foundation for different aspects of this work is laid out by giving an overview of the requirements and principles for safety in industrial applications. This includes an introduction to relevant laws and associated safety standards. It is highlighted how these standards apply, and which requirements arise from them that have to be considered for this work.

Related Work: In Chapter 3, an overview of current research from several research areas related to this work is given. Recent methods, datasets, and evaluation metrics for human pose estimation are reviewed, highlighting differences and similarities. Next, different approaches to robot safety are discussed, especially focusing on the use of human pose estimation in such applications. Conceptual shortcomings of selected human-pose-estimation-based approaches regarding legal compliance are highlighted. Last, methods from the field of neural network uncertainty are introduced, which are a means of assessing the reliability of neural network results.

Error Reduction: In Chapter 4, diversity is introduced as a way to reduce errors of recent human pose estimators, without making any assumptions about their outputs apart from keypoint positions. A threshold-based comparison of results is performed for multiple results from different networks, and mismatching detections are discarded. A method for calculating these comparison thresholds, based on a general human body model and redundant keypoint detections, is proposed. Furthermore, methods for error detection based on heatmap activations are introduced and investigated. Experiments are performed to evaluate the potential of the methods regarding the elimination of human pose estimation errors.

Measurement Error Estimation: In Chapter 5, the common definition of correctness in human pose estimation is put into question for safety purposes. It is replaced by a formulation based on the prediction of measurement errors with additional quality criteria ensuring meaningful results. This allows the prediction of an area containing the keypoint at inference time in contrast to simple point predictions with unknown error magnitudes. Several ways for predicting the measurement error alongside the keypoint position are investigated and evaluated. Proposed methods build either directly on the current measurement error or on the prediction of probability distributions.

Impact and Handling of Noise: In Chapter 6, the necessity of handling noise in safety-critical industrial applications is explored, based on special conditions that apply there. Due to the existence of closely defined environmental conditions and a mandatory risk assessment, it can be assumed that only previously known types of noise can occur in safety-critical industrial applications – making countermeasures against specific noise types a viable choice. Through experiments, the impact of noise on classic human pose estimation and the previously introduced adaptations is shown, and countermeasures in the form of denoising and training against noise are examined.

Real-time capable distance monitoring: In Chapter 7, hard real-time capable human-robot distance monitoring is explored, without the need for a real-time capable human pose estimator. Instead, a bridging algorithm for previously detected human body keypoints is introduced, adapting them over time in a way that is compliant with safety standards. Through these alterations as well as human and robot volume modeling based on spherical cones, it is possible to calculate distances between humans and robots in hard real-time under the assumption that the number of humans and robots is limited. Furthermore, the methodology ensures continuous distance changes.

Discussion, Outlook and Conclusion: In Chapter 8, the methods and results regarding error reduction, measurement error estimation, and hard real-time capability are discussed with respect to the ultimate goal of making human pose estimation applicable as a part of safety-critical industrial applications. Achievements are highlighted, together with further necessary and optional improvements and additions.

2 Fundamentals

2.1 Safety-Related Laws

The contents of this work are primarily motivated by the legal requirements for robot and machine safety, which are imposed by laws and standards. However, laws and associated safety requirements can vary from country to country, making it necessary to choose one consistent definition for further investigation. For this work, the safety requirements of the European Union (EU) are chosen for two reasons: (i) the EU is well-known for pursuing a high level of safety when it comes to the operation of robots and machines, hence being a suitable choice for safety requirements, and (ii) the EU has put considerable effort into harmonizing safety-related laws of different member states [26], thereby creating a legally binding definition of safety which is applied in many countries.

Within the European Union, there are five different types of legislation, which are defined by the Article 288 TFEU [29] and further elaborated by official EU sources [20]:

1. **Regulations:** A legally binding legislation for all member states. It defines what has to be done and must be directly and entirely applied.
2. **Directives:** Another legally binding legislation for all member states. It defines common goals that must be achieved but are enforced through national legislation.
3. **Decisions:** A legally binding legislation very similar to the regulation that is limited in its scope, e. g., applies to a single member state only.
4. **Recommendations:** A legally non-binding legislation with limited scope. Used to communicate the position and desired action of the EU regarding specific topics.
5. **Opinions:** Similar to recommendations, but only used to communicate the position of individual EU institutions and the likes on a topic without a desired action.

Safety-related legislation in the EU is typically pursued through directives [25, 26, 27, 28]. This is also the case for the safety of machinery, which is formalized in the Directive 2006/42/EC of the European Parliament and of the Council [26] on the European level, commonly referred to as Machinery Directive. Taking Germany as an example, the required national laws implementing the directive are the Product Safety Act [31] and the Ninth Ordinance on the Product Safety Act [32].

To ensure safety, the Machinery Directive [26] defines in Appendix I the *essential health and safety requirements* (EHSRs) that must be fulfilled for a machine if they apply. Whether or not a specific EHSR applies depends almost always on potential hazards originating from the machine. This means that most EHSRs are directly linked to hazards. A notable exception is the general requirement for a risk assessment that always applies. One of the goals of the risk assessment is to identify all potential hazards of a machine, which in turn indicate the relevant EHSRs. Once identified, the relevant EHSRs must be

fulfilled. To help with meeting the EHSRs, the Machinery Directive provides the option to employ so-called *harmonized standards*. When applied correctly, they provide the *presumption of conformity* with the EHSRs covered by them [26], hence being a helpful tool for achieving conformity with legal requirements. Defined at the European level [21], they provide a coherent technical means across all member states for approaching EHSRs.

2.2 Harmonized Standards

Throughout this section, a more detailed introduction to harmonized standards and their application will be given. A European harmonized standard is a technical specification that was created at the request of the European Commission to a standardization organization (CEN, CENELEC, or ETSI) with the goal of assisting the fulfillment of legal requirements [21, 26]. Such standards can be applied for this purpose as soon as they have been published in the Official Journal of the European Union, leading to the presumption of conformity with legal requirements when done right [26]. In the context of the Machinery Directive, central legal requirements are the EHSRs that must be met to achieve safety. This involves several steps [26]:

1. Carry out a *risk assessment* during which hazards of the machine are identified.
2. Identify the EHSRs corresponding to those hazards.
3. Employ measures to meet the EHSRs e. g., apply harmonized standards.

The role of harmonized standards in these steps is to provide solutions that help to meet the EHSRs in the form of e. g., processes that must be followed or properties a machine must have. In simple words: While the Machinery Directive defines (through EHSRs) *which* goals have to be fulfilled, the harmonized standards outline *how* to fulfill these goals through technical procedures, measures, and properties.

After clarifying what harmonized standards can be used for, the question of how to apply them correctly remains. This is especially interesting, as the Machinery Directive has a vast amount of harmonized standards [22] but gives little to no guidance on how to use them, apart from the requirement that the harmonized standards have to cover EHSRs for the presumption of conformity [26]. More details on the application of harmonized standards can be found in the ISO 12100 standard [47]. This standard is harmonized with the machinery directive and introduces a general structure for harmonized standards of the Machinery Directive [47]: Each harmonized standard is either classified as *type A, B* or *C standard*, with B further distinguishing between *B1* (for safety aspects) and *B2* (for safety equipment) standards. The most general form is the type A standard, which introduces e. g., general principles for safety. It is both broad in scope of topics and devices it applies to, but not highly detailed. In contrast, type B standards are detailed and broad with respect to the kinds of machines they apply to, but the safety-relevant topic is narrow in its scope (e. g., one kind of safety aspect or equipment). On the contrary, type C standards are narrow with respect to the kinds of machines they address (e. g., single kind of machine) but are broad and detailed in the types of safety aspects they address. When applying standards, the type C standard always has priority if one exists for the machine [47]. In turn, this means if no type C standard exists or some safety-critical aspects are not covered by the type C standard, then type A and B standards have to be applied. In the following, standards relevant to this work will be introduced.

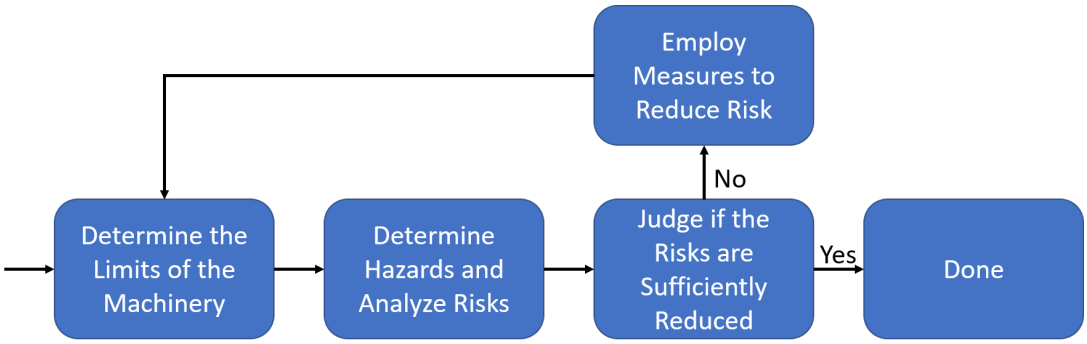


Figure 2.1: ISO 12100 [47] process for determining and handling risks (simplified).

2.2.1 ISO 12100

The ISO 12100 standard [47] is the only type A standard for the safety of machinery that is currently in effect [22]. Thus, it is the central document that details the general principles and procedures for machine safety. Its most important contribution is a procedure for determining hazards and reducing risk, which is a fleshed-out version of the risk assessment steps of the 1. General Principle in Annex I of the Machinery Directive [26].

Not all the details of this procedure are required for this work, a more rudimentary understanding will suffice. A sufficiently simplified version of it is shown in Figure 2.1. The (iteratively performed) steps highlighted there comprise the following activities [47]:

1. **Determine the Limits of the Machinery:** The *limits of the machinery* are determined first, including e. g., potential interactions with the machine, application areas, and environmental conditions. Intended use and foreseeable misuse have to be considered. The limits form the foundation for assessing hazards and risks.
2. **Determine Hazards and Analyze Risks:** Next, potential hazards have to be identified based on the limits of the machinery. Normal operation, foreseeable misuse, and defective machine behavior (e. g., induced by external factors) shall be considered. For each identified hazard, the associated risk has to be determined, based on the probability of occurrence and the potential damage inflicted.
3. **Judge if the Risks are Sufficiently Reduced:** It has to be judged whether each risk is sufficiently low, or if additional risk reduction measures are necessary.
4. **Employ Measures to Reduce Risks:** If the risks are not sufficiently low, measures for risk reduction must be taken. These are (i) changing the machine's design, (ii) adding external technical safety measures, and (iii) informing the user about the risk (e. g., warning signs or noise). For the given order, the first viable solution has to be taken to eliminate risks as early as possible. Then, the whole process is repeated because of potential new limits, hazards, and risks.

The mandatory nature of this procedure leads to unique conditions that set the industrial application of robots and machines apart from an application in the wild: (environmental) conditions that the machine can encounter during operation are closely defined by the limits of the machinery, and all hazards with non-neglectable risk that require handling are identified. This guides the choice of risk-reduction measures to achieve safety. However, no detailed risk-reduction measures are introduced by ISO 12100. These are subject to specialized type B and C standards, such as those introduced in the following.

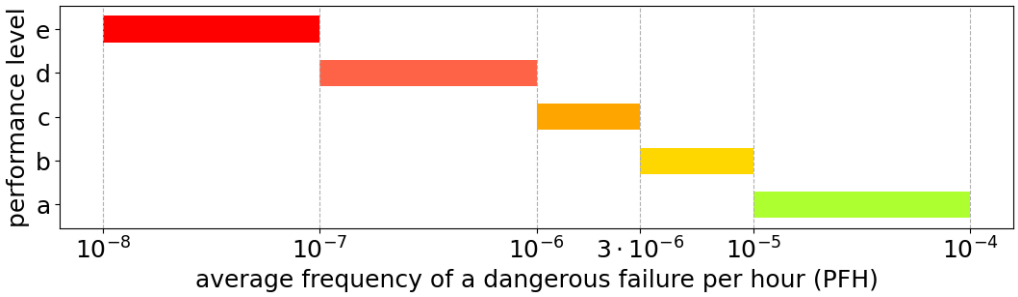


Figure 2.2: Performance levels and their right-open PFH intervals as in ISO 13849-1 [53].

2.2.2 ISO 13849

The ISO 13849 standard is a B1 standard split in two parts: ISO 13849-1 [53] and ISO 13849-2 [50], that deal with the design and validation of safety-related parts of control systems respectively. Within the context of these standards, a part of a control system is considered safety-related when it is used as part of a *safety function*, which in turn is a function employed as a measure for risk reduction with the property that the risk is increased if it malfunctions. An example of such a safety function would be the safe adjustment of a robot's speed based on a sensor output monitoring the human-robot distance. Here, a malfunction could lead to a prohibitive high speed of the robot in the vicinity of a human. The ISO 13849-1 standard covers a broad range of requirements for and aspects of safety functions. Those regarding reliability are of special interest for this work.

The reliability of a safety function is indicated through its *performance level*, as introduced by ISO 13849-1 [53]. It categorizes safety functions by five different levels a-e, with a having the weakest and e having the strongest demands for reliability. To assess the reliability, the *average frequency of a dangerous failure per hour* (PFH) is employed. Figure 2.2 shows the relation between performance level and PFH. The performance level required for a specific safety function depends on several factors, e. g., the severity of possible injuries and the ability to prevent an injury even if the safety functions fail (e. g., if the human can leave the danger zone in time). Guidance on determining the necessary performance level is available in ISO 13849-1, while other standards like ISO 10218-1 [48] demand a certain performance level for all safety functions used in a specific context.

Another relevant aspect discussed by ISO 13849-1 is the use of *redundancy* and *diversity*, where the latter can be described as a special form of redundancy that requires the same functionality to be provided in different ways (e. g., use of different sensor data types, different algorithms, ...). Both concepts are introduced as useful tools for increasing the reliability of safety functions by decreasing the probability that a system malfunctions, e. g., by detecting errors through the comparison of redundant or diverse results. Diversity is furthermore highlighted as an effective measure against common cause failures, where the same cause leads to a malfunction of separate (redundant) parts of a safety function.

2.2.3 ISO 13855

The standard ISO 13855 [46] is a type B1 standard that is focused on the correct placement of a variety of protective equipment that is mostly but not exclusively non-separating. A notable portion of the standard is dedicated to electro-sensitive protective equipment,

Assumed movement speed K	Applicable for
2.0 m/s	Hands and arms
1.6 m/s	Remaining body and walking motion

Table 2.1: Movement speeds defined by ISO 13855 [46] as well as their target application.

which includes the likes of laser scanners and camera systems. Such devices are used to ensure that a *minimum distance* between a robot and a human is kept, and that the robot can be brought to a safe stop if this distance gets violated. To make sure a safe stop can be adequately achieved, the standard introduces different ways to calculate the minimum distance, based on the sensor placement and further information. For this work, two elements of ISO 13855 are especially relevant: The way safety distances are calculated, as well as the way human movement is factored into this calculation.

Regarding human motion, the standard introduces proven in-use constant values that can be applied to calculate the distance a human can traverse in a certain amount of time. Depending on the placement of protective equipment, two different values for the human speed K are used: 2.0 m/s whenever the detection of the hand or arm is crucial, and 1.6 m/s whenever other parts of the body are detected. The draft for an updated version of ISO 13855 [52] explicitly highlights them as values for hand/arm and walking motion respectively. Table 2.1 also displays these values. Depending on this speed, the necessary minimum distance that has to be kept is calculated by the following formula [46]:

$$S = (K \times T) + C \tag{2.1}$$

Hereby, the factor K is the aforementioned human movement speed that is assumed, T is the worst-case time that is needed to detect the human and subsequently fully stop the robot, and C is an additional factor that indicates how far a human can bypass protective equipment before he is detected (e. g., by reaching over a plane monitored by a laser scanner). Overall, this formula gives a general way for calculating safety distances, which can be further detailed out for specific applications, as done by e. g., ISO/TS 15066 [51].

2.2.4 ISO 10218 and ISO/TS 15066

The standard ISO 10218 is a type C standard exclusively focused on industrial robots, split into two parts. The first part, ISO 10218-1 [48], focuses on the safety of industrial robots themselves, while the second part, ISO 10218-2 [49], focuses on safety aspects that arise when integrating them into their final application. For this work, important aspects of ISO 10218 include the definition and demand for safety-relevant functionalities of industrial robots, as well as the formulation of possible human-robot interaction modes together with necessary requirements for them.

Looking at *human-robot collaboration* (HRC), ISO 10218-1 outlines four potential ways to enable a human to share a common workspace with the robot [48]:

1. **Safety-Rated Monitored Stop:** In this HRC mode, it is necessary to detect when the human is inside the shared workspace. When the human enters, the robot has to stop completely, and remain this way until the human leaves.
2. **Hand Guiding:** In this HRC mode, the human can move the robot by hand while its speed is limited in a safe way.

3. **Speed and Separation Monitoring (SSM):** In contrast to the safety-rated monitored stop, this HRC mode allows the robot to continue operation while both, human and robot, are inside the shared workspace. A safe distance between human and robot must be maintained, otherwise the robot must be stopped. This makes it necessary to detect the human and calculate the human-robot distance in a safe way.
4. **Power and Force Limiting:** The last mode for HRC allows humans and robots to share a workspace without an immediate stop. This is achieved through a safe way of limiting the force that the robot can exert on the human. It is necessary to monitor the robot's (potential) forces and to stop the robot safely if limits are exceeded.

SSM is the scenario motivating this work. For it to be used in the HRC context, both ISO 10218-1 and ISO 10218-2 require the safety functions used to implement it to have PL d (see Figure 2.2). For the calculation of the necessary safety distance, ISO 10218-1 refers to the contents of the previously presented ISO 13855. For further information on how to actually implement the SSM mode, both ISO 10281-1 and ISO 10281-2 point towards the (not harmonized) technical standard ISO/TS 15066 [51].

Overall, ISO/TS 15066 puts very few restrictions on the implementation of SSM, e. g., by not directly limiting the kinds of devices that can be employed. In any way, a (minimum) *protective separation distance* between human and robot has to be maintained, otherwise the robot has to be stopped in a safe way. Mandatory prerequisites regarding the robot include a safe way to limit the robot's speed, a safe way to stop the robot, and, depending on the way SSM is implemented, a safe way to limit the area the robot can reach. Regarding associated safety functions, PL d is mandated. To calculate the protective separation distance, ISO/TS 15066 expands on Eq. (2.1) and introduces the following formula [51]:

$$S_p(t_0) = S_h + S_r + S_s + C + Z_d + Z_r \tag{2.2}$$

In this formula, $S_p(t_0)$ refers to the protective separation distance for a certain point in time t_0, indicating a potential time dependence. Whether or not $S_p(t_0)$ can actually change over time is up to how the SSM application is designed, with ISO/TS 15066 leaving both possibilities open. The factors contributing to $S_p(t_0)$ are defined as follows [51]: S_h is the distance the human can traverse in the direction of the robot during the time that is needed to detect the human and then fully stop the robot. S_r is the distance that the robot can traverse in the direction of the human during the time necessary for human detection and signaling the robot to stop. Similarly, S_s is the distance the robot can traverse towards the human from start to finish of the stopping process. C denotes a surcharge to the protective separation distance for body parts that are not detected (see ISO 13855). Z_d and Z_r are both surcharges for the (maximum) measurement error in human and robot position respectively. It is left open whether S_h, S_r, and S_s employ dynamic speed functions or static speed limits for movement speeds, but worst-case assumptions have to be made.

For this work, the following conclusions can be drawn from ISO 10218 and ISO/TS 15066 for the use of human detection approaches as part of a safety function in SSM:

1. The approach must have PL d, meaning it must fulfill high requirements regarding its reliability under foreseeable conditions during operation, as shown in Figure 2.2.
2. The approach must be capable of hard real-time, as the time the human detection process takes must be limited to allow the calculation of S_h and S_r.
3. There must be a way to determine the worst-case measurement error of the human detection approach to supply the value Z_d for the protective separation distance.

3 Related Work

3.1 Human Pose Estimation

Throughout this section, an overview of the task of human pose estimation and its different forms will be given, together with an introduction to different datasets and evaluation metrics. Afterward, selected human pose estimation approaches with high relevance for this work will be examined in more detail. Due to the overwhelming performance advantage of deep-learning-based approaches compared to handcrafted ones on popular datasets [115], the latter will not be part of this section. The overall focus of this chapter lies on the problem of 2D single-person human pose estimation on single images and its application in more advanced tasks, as it is the primary form of human pose estimation used in this work.

3.1.1 General

The task of human pose estimation aims at 'estimating the configuration of the human body' [115], which is in recent approaches typically done through the estimation of the position of important human body keypoints [12, 78, 79, 86, 115, 131, 143, 144, 146]. Hence, in most cases, human pose estimation constitutes a localization problem of a fixed, predefined set of points. This broad definition comprises a variety of more specialized task definitions for human pose estimation, which are used to distinguish human pose estimation approaches alongside the employed methodology. The first widely used criterion to distinguish between human pose estimation approaches is whether they are aimed at *2D* or *3D human pose estimation* [35, 147], with entire surveys/reviews focused on one of both areas [18, 130]. This separation is further reinforced by popular human pose estimation datasets, which have either a clear focus on 2D human pose estimation [3, 58, 70, 104] or 3D human pose estimation [55, 127]. In this work, the separation of human pose estimation approaches into 2D and 3D for classification will also be employed.

For the field of 2D human pose estimation, there exists a widely agreed on taxonomy for approaches [18, 35, 147]: First, approaches are distinguished based on whether they aim to detect a single person (2D *single-person* human pose estimation) or multiple people (2D *multi-person* human pose estimation) in a given image. In the field of 2D single-person human pose estimation, the task of the human pose estimator is to only infer the keypoint coordinates belonging to a single individual, either from a single-person image or an image crop of a single person. In contrast, the keypoints of an arbitrary number of individuals has to be detected in 2D multi-person human pose estimation. This does not only include the detection of keypoint positions and the correct number of individuals, but also the association of each keypoint with a specific individual.

Looking closer at 2D single-person human pose estimation, approaches for this problem are typically categorized by the prediction mechanism they employ [18, 35, 147]: either

direct keypoint regression or *heatmap prediction*. When performing direct keypoint regression, the neural network is structured in a way that it directly calculates a result $\hat{\boldsymbol{y}}$ that contains the estimated N keypoint positions for a given individual, thus $\hat{\boldsymbol{y}} \in \mathbb{R}^{N\times 2}$. Networks for this task start by processing spatial image data in the form of an input image or image crop. At some point in the network, the spatial dimensions are typically dropped, with one or more fully connected layers being employed before the final result is produced [12, 78, 125]. Typically, these kinds of networks are trained in a way that minimizes the difference between annotated and predicted keypoint positions (end-to-end training), which can be achieved through the application of a suitable loss function, like the L_2-loss [12, 125]. Let, $\boldsymbol{y} \in \mathbb{R}^{N\times 2}$ denote the annotated keypoint positions, and $\boldsymbol{y_i} \in \mathbb{R}^2$ the position of the i-th keypoint, with a likewise definition of $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ for predictions. Then, the loss for a single person can be formulated as [125]:

$$L_2(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \sum_{i=1}^{N} ||\boldsymbol{y_i} - \hat{\boldsymbol{y}}_{\boldsymbol{i}}||_2^2 = \sum_{i=1}^{N} (\boldsymbol{y_i} - \hat{\boldsymbol{y}}_{\boldsymbol{i}})^2 \tag{3.1}$$

The alternative to direct keypoint regression is heatmap prediction [18, 35, 147]. While approaches for direct keypoint regression sacrifice the spatial dimensions of the input image at some point in the prediction process, the heatmap-based approaches retain them until the final outputs of the network are produced. These outputs are the name-giving heatmaps. They have the same spatial axes as the input image, however, the actual spatial resolution may differ (for example, Newell et al. [86] use an image resolution of 256×256 and a heatmap resolution of 64×64 while operating on the MPII Human Pose dataset [3]). Typically, one heatmap $\hat{\boldsymbol{h}}_{\boldsymbol{i}}$ for each keypoint i of the human is produced. The contents of a heatmap for a single keypoint are pixel-wise pseudo-probability scores, indicating how likely the keypoint is located at each pixel location. The final keypoint location required for the human pose estimation task has to be inferred from the heatmap in post-processing through a post-processing function f_p. The simplest approach to realize this would be to use the heatmap coordinates with the highest heatmap value – thus the highest keypoint probability – and to backproject the coordinates into the original image. Mathematically, this could be expressed as follows for a single keypoint i:

$$\begin{aligned} \hat{x}_i, \hat{y}_i &= f_p(\hat{\boldsymbol{h}}_{\boldsymbol{i}}) \\ \text{e. g., } \hat{x}_i, \hat{y}_i &= T^{-1}(\underset{u,v}{\text{argmax}}(\hat{\boldsymbol{h}}_{\boldsymbol{i}}[u, v])) \end{aligned} \tag{3.2}$$

In this equation, T refers to a transformation from the original image to the heatmap, hence T^{-1} backprojects the keypoint locations from the heatmap into the original image. Using the location of the heatmap maximum as a foundation for inferring the final keypoint position is widespread in heatmap-based approaches [35], with Newell et al. [86] for example using a combination of the maximum position and of the position of the highest-valued neighboring pixel. To ensure that the heatmap-based approaches learn meaningful pseudo-probabilities, a suitable ground truth heatmap $\boldsymbol{h}_{\boldsymbol{i}}$ is necessary for every keypoint i during the learning process. These are generated by applying a spatial filter – most often a 2D Gaussian – at the location of the keypoints in the heatmaps [35, 147]. This procedure rewards the neural network for heatmap activation $\hat{\boldsymbol{h}}_{\boldsymbol{i}}[u, v] > 0.0$ not only directly at the location of the keypoint, but also in its vicinity. The neural network is then trained to predict these heatmaps by minimizing the difference between predicted and ground truth heatmaps, e. g., through the application of the *mean squared error* (MSE) [86, 117, 147],

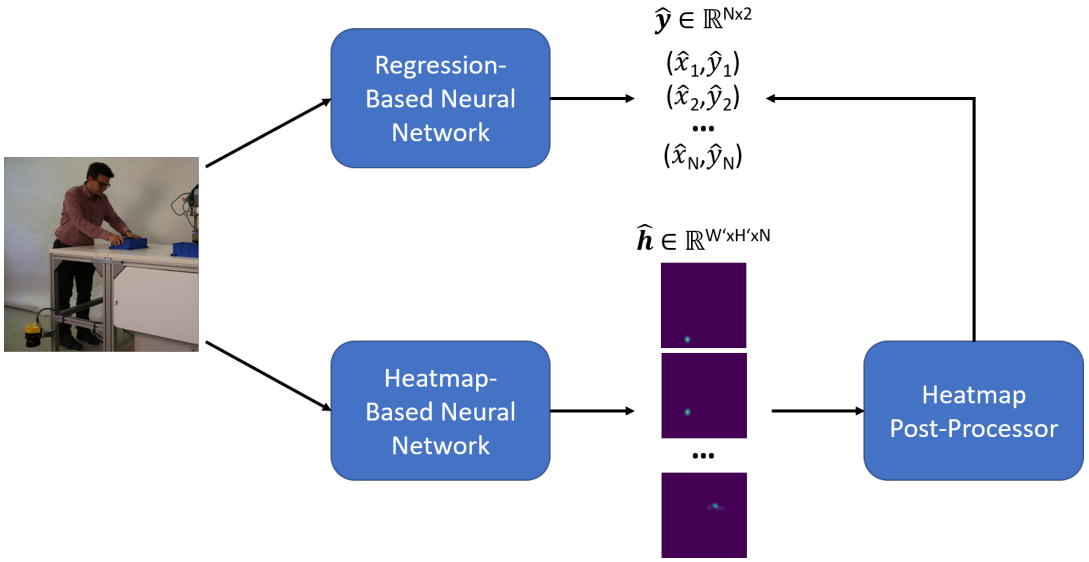


Figure 3.1: Illustration of a regression and heatmap-based approach, both processing an input image. The regression-based approach directly produces its output $\hat{\boldsymbol{y}}$ in form of coordinate tuples $(\hat{x}_i, \hat{y}_i)$ with $i \in [1, 2, ..., N]$ for each of N human keypoints. In contrast, the heatmap-based approach produces an output $\hat{\boldsymbol{h}}$ of N heatmaps, each having width W' and height H'. The keypoint positions have to be inferred through a separate post-processing step.

a variant of the L_2 loss that is normalized based on the number of elements it is calculated for. In case of using the MSE, the loss function for the predicted and annotated heatmaps $\hat{\boldsymbol{h}}$ and $\boldsymbol{h}$ of a single input is calculated as follows, with W' and H' denoting width and height of the heatmaps and $\hat{\boldsymbol{h}}, \boldsymbol{h} \in \mathbb{R}^{W' \times H' \times N}$:

$$\text{MSE}(\boldsymbol{h}, \hat{\boldsymbol{h}}) = \frac{1}{N \cdot W' \cdot H'} \sum_{i=1}^{N} \sum_{u=1}^{W'} \sum_{v=1}^{H'} (\boldsymbol{h_i}[u, v] - \hat{\boldsymbol{h}}_{\boldsymbol{i}}[u, v])^2 \tag{3.3}$$

When comparing both approaches from a theoretical standpoint, each has some advantages over the other. Direct regression allows for a highly precise prediction of the keypoint position due to direct coordinate regression, while heatmap-based approaches are limited by the discretization and resolution of the heatmap [18, 147]. Another advantage of regression-based approaches is the possibility to train them in an end-to-end fashion [18, 35, 147] – a property that heatmap-based approaches usually lack due to the use of non-differentiable post-processing steps like the application of argmax as displayed in Eq. (3.2) [35]. On the other hand, the spatial nature of heatmaps with non-zero values near the keypoint location offers better supervision during training [35, 147]. In practice, heatmap-based approaches are typically superior in terms of keypoint prediction performance [35, 147] and are used more often [147], hence being the dominant strategy of the last years. The keypoint prediction with both approaches is illustrated in Figure 3.1.

Next, the task of 2D multi-person human pose estimation will be examined in greater detail [18, 35, 147]. Compared to the single-person case, multiple people are present in an image, and image crops containing single individuals are not available by default. This leads to additional challenges: (i) the number of keypoints and total human poses that have to be detected is no longer known for an image, and (ii) it is not sufficient to

only detect all keypoints in an image, but it is also necessary to associate them with the different individuals. In general, there are two ways in which this problem can be tackled: *top-down* and *bottom-up*. Top-down approaches try to resolve the problem by locating all individuals in an image first and then predict the keypoints for them individually. In contrast, bottom-up approaches try to predict all keypoints present in an image first, and then group them based on the individual they belong to. This separation of approaches is also used to classify 2D multi-person approaches. Not covered by this taxonomy is a small set of relatively new approaches [89, 111, 121], that try to break with the two-stage paradigm laid down by both top-down and bottom-up approaches. Instead, these approaches aim to solve the problem directly with a single stage in end-to-end fashion.

First, a more detailed description of the top-down approach will be given [18, 35, 147]. As mentioned before, these approaches have two tasks to solve, (i) the localization of different individuals in an image and (ii) the prediction of keypoints for each of these individuals. These tasks are solved one after another through a two-stage pipeline. The location of different individuals is obtained in a first step, where a human detector is applied to the input image. Then, a single-person human pose estimator is applied at the position of each human. Many approaches [9, 67, 68, 117, 134, 137] realize this procedure by employing an established bounding-box object detector like faster RCNN [101] or MegDet [95] and focus their efforts on the (single-person) human pose estimator that is applied inside the detected bounding boxes afterward. An exception to this is e. g., the work of Feng et al. [30], who focused on obtaining a more accurate localization of the individuals in an image before predicting the final keypoint locations, as a good localization of the whole individual is important for the actual human pose estimation afterward. Overall, top-down approaches can be seen as natural extension of 2D single-person human pose estimation with an additional step for human localization.

Second, the bottom-up approach will be introduced [18, 35, 147]. Here, the two tasks that have to be solved are (i) the localization of all keypoints present in an image, and (ii) the association of these keypoints with different individuals. Just like top-down approaches, bottom-up approaches solve these problems in consecutive steps. However, contrary to top-down approaches, the outputs necessary to solve both consecutive tasks are typically calculated by a single neural network instead of two separate ones. Approaches like Openpose [10] or PifPaf [63] employ a single neural network to predict two sets of spatial maps: One set that is first used to obtain the keypoint locations in the image, and another set that is used afterward to predict the connectivity of keypoints used for the association of keypoints with individuals. Both approaches use specialized architectures to achieve this. Instead of focusing on keypoint connectivity, Newell et al. [87] predict labels indicative of human instances (called associative embeddings) directly, also through an additional set of spatial maps. These labels are one-dimensional vectors, and the distance between predicted label vectors is used to associate keypoints with individuals. In contrast to PifPaf and OpenPose, no highly specialized architecture is required. It is sufficient to adjust the output layers and training procedures of existing heatmap-based single-person human pose estimation approaches, as e. g., done by HigherHRNet [14].

Looking at the overall relevance of 2D single-person human pose estimation for the multi-person task, it can be said that the direct application of existing methods is highly relevant for top-down approaches, with a powerful single-person human pose estimator being crucial for the success of the multi-person method. In contrast, the direct application of a single-person method trained to detect a single set of keypoints is not possible for bottom-up methods. However, with slight adjustments (e. g., training heatmap-based approaches

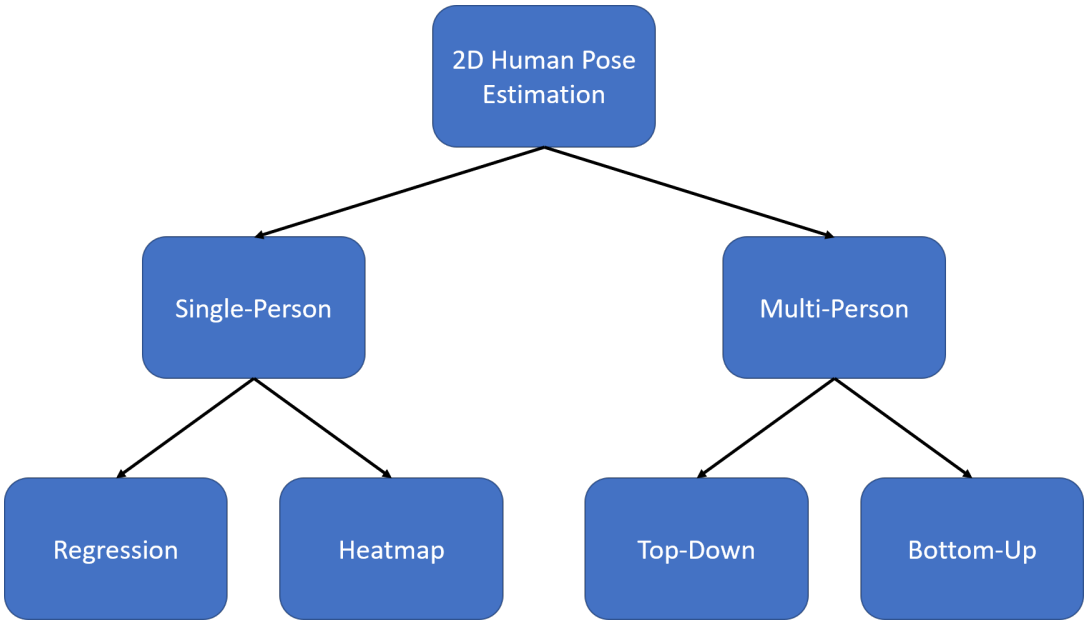


Figure 3.2: The taxonomy of 2D human pose estimation outlined by multiple surveys and reviews [18, 35, 147] that is also used in this work.

to detect all keypoint instances instead of a single one [87]) and minor extensions (like associative embeddings), it is possible to make some of them applicable. However, in the presence of more specialized architectures and necessary adjustments, they are not as relevant for bottom-up approaches as for top-down approaches.

A full overview of the presented taxonomy for 2D human pose estimation used in this work is shown in Figure 3.2. One additional factor that can be used to distinguish human pose estimation methods not covered by this taxonomy is whether the input data are *single images* (like in the previously presented cases) or *sequences of images* (videos). Following the definitions from the PoseTrack dataset [4], human pose estimation in videos is the task of determining the poses of humans in a single image (called key frame), when preceding and subsequent images from a video are available. This task is not to be confused with the more advanced problem of pose tracking, where multiple images from videos are available as input as well. However, instead of predicting poses in single images, the task involves the prediction of temporal consistent poses across all images from an image sequence, as well as the consistent association of poses (and keypoints) with individuals [4]. For the problem of human pose estimation in videos, the simple solution would be to employ methods working on single images, discarding the temporal information. However, to achieve better results, specialized approaches that incorporate temporal information were developed. To incorporate motion between different frames into the prediction process, some methods rely on separately calculated motion representations like optical flow, for example to align outputs for different images [96] or to support the prediction process itself directly [56]. Other approaches [74, 75] directly process multiple images from the image sequence to predict results for the key frame.

Although this work is primarily focused on 2D single-person human pose estimation, a brief overview of 3D human pose estimation will be given, with the goal of presenting the research area of human pose estimation as a whole and highlighting potential applications of 2D methods in this advanced task. Like 2D human pose estimation, the task of 3D human pose estimation aims at the correct localization of important human body keypoints

such as joints, with the difference that it is now performed in 3D space [130, 147]. In contrast to 2D human pose estimation, there is no single taxonomy that is widely agreed on. For example, on the first level, Zheng et al. [147] distinguish whether 3D human pose estimation is performed on images (including both, single images and image sequences) or on other sources, Wang et al. [130] distinguish between the use of single images and image sequences, and Gamra et al. [35] distinguish between single-person and multi-person methods. Despite these differences in building up a taxonomy, common factors are used to categorize different methods in these works. Thus, instead of building up yet another taxonomy in this work, these factors will be quickly discussed in the following.

As a first factor to classify 3D human pose estimation approaches, the general strategy used to obtain keypoints will be discussed. Overall, three different methods can be distinguished [130, 147]. First is *direct regression*, where the 3D position of keypoints is directly predicted. For example, Zhang et al. [144] propose a neural network architecture consisting of two parts, a convolutional neural network (CNN) for extracting image features from multiple input images taken from different views, and a transformer network to predict 3D keypoint positions directly based on the extracted features. As a feature extractor, they employ the existing 2D human pose estimation architecture of Xiao et al. [134]. Second is the *lifting from 2D to 3D* strategy. Approaches following this strategy use a two-stage procedure, where a 2D human pose estimator is employed in the first stage to obtain a 2D pose representation, which is afterward used in a second stage to infer 3D keypoints. A well-known example of such approaches is the work of Martinez et al. [80]. For calculating 3D keypoints, they build upon the 2D keypoint positions obtained from an existing 2D human pose detector. These 2D keypoint positions serve as only input for their simple network architecture for predicting 3D keypoint locations, consisting of two fully connected layers and two residual blocks based on fully connected layers. Having 2D keypoint locations as only input for the network that predicts the 3D locations makes this approach highly reliant on the 2D human pose estimator. Third is the use of *parametric body models* that represent the body shape through a 3D mesh, with the SMPL model [76] being a popular example of such models. Instead of predicting the location of keypoints, the parameters of the parametric body model [60] or the resulting mesh [62] are predicted. Then, 3D keypoint locations can be obtained, e. g., through the application of a learned regression matrix on the vertices of the body mesh produced by the parametric body model [76]. An example of approaches that employ parametric body models is the work of Kolotouros et al. [62], who use a CNN for extracting features from images that are used as input for a graph neural network, in which each node calculates the location of one vertex of the SMPL mesh model. 3D keypoint locations are inferred through the official regression matrix provided by SMPL. Another example is the work of Kamazawa et al. [60]: they also extract features from images by using a CNN first, but then infer SMPL parameters from the image features by using a regressor that iteratively corrects an initial estimate of these parameters. To obtain 3D keypoint locations, a regressor is used on the mesh model which is calculated from the predicted SMPL parameters.

Another distinguishing factor (that also appears in 2D human pose estimation) is whether methods are aimed at detecting a single person or multiple people [35, 130, 147]. Methods for 3D single-person human pose estimation are limited in a way that they can only predict the pose information of a single individual. Using the previously introduced method of Kolotouros et al. [62] as an example, the graph neural network is constructed in such a way that it predicts exactly one position for every vertex of the SMPL mesh model, making it by design unable to predict multiple positions/meshes and thus results for multiple

people. In contrast, 3D multi-person methods must be able to detect multiple people in the network input. Just as in the 2D case, 3D multi-person approaches can further be split into top-down and bottom-up approaches. Again, top-down approaches have to localize single human instances in input images first, and then predict 3D keypoint locations for these single human instances. For example, Moon et al. [84] follow this paradigm by employing a human bounding box detector to locate humans in the input image, subsequently cropping the input image to single human instances using the detected bounding boxes. Afterward, their 3D keypoint detection approach is applied to every cropped image, detecting a root as well as root-relative pose information for a single human instance per crop. A special case of the top-down paradigm can occur when multiple images from different views are used as input: Here, the same person can be present in multiple images, making it necessary to associate different detections of the same person with one another. This problem is for example solved by Dong et al. [23] by clustering image crops of single persons in different images based on their appearance in the image crop as well as their 2D pose, which is detected through a 2D human pose estimator. In contrast to top-down approaches, bottom-up approaches predict keypoint locations for all persons in the input data, and then have to solve the association problem of keypoints to person instances – a similar definition to the 2D case. For example, Zhen et al. [146] calculate multiple intermediate results. Heatmaps are used to obtain the location of all keypoints first, then the keypoints are associated with individuals using part affinity fields and root depth maps, resulting in 2D poses. 3D poses are afterward calculated based on root depth maps and part relative-depth maps. Apart from top-down and bottom-up processing, there exist approaches that omit both paradigms by aiming directly at the prediction of multiple 3D poses, like Zhang et al. [144].

A unique factor for 3D human pose estimation is whether camera data is available from a *single view* or from *multiple views* [130, 147]. Many methods focus on the problem of reconstructing 3D human poses from images taken by a single camera, hence having only a single view of the human to work with. This task is hard due to two factors: First, the problem of inferring the depth of a human pose from a single image has no unique solution, as the same 2D human pose can be created by projecting different 3D human poses into a 2D image. Second, taking images from a single view is prone to (partial) occlusion of the human. For example, severe occlusion can hinder the precise localization of 2D keypoints, in which case 3D single-view human pose estimation methods that rely on precise 2D keypoint locations like Martinez et al. [80] are set up to fail. Having images taken from multiple cameras with different views can help to alleviate both single-view problems, with occlusions being not necessarily present in all views and with triangulation or more advanced methods [23] being available to obtain depth information more reliably. However, processing images from multiple views can also add additional problems that need to be solved, like the identification of the same individual in multiple images from different views [23]. The previously introduced methods of Zhang et al. [144], who collect and jointly process image features extracted from multiple views, and Dong et al. [23], who generate and process cluster of image crops from different views associated with the same individual, are examples of how images from multiple views can be leveraged for 3D human pose estimation.

The final distinction discussed in this section concerns the *input data* used by different methods [130, 147]. These can be single images, image sequences, other data like IMU information, or combinations thereof. Previously introduced examples belonged to the single images category, where one image from a single view or multiple, simultaneously

taken images from multiple views are processed without taking temporal information and context into consideration. Working with image sequences from a single or multiple views offers exactly that temporal context. For example, Pavllo et al. [93] pursue a general strategy similar to Martinez et al. [80] by predicting 3D keypoint positions directly from previously extracted 2D keypoint positions. In contrast, Pavllo et al. do not build on a single set of 2D keypoints as input, but use 2D keypoint locations obtained from a sequence of 243 images. Temporal information is exploited by stacking the 243 sets of 2D keypoint positions along one axis and by processing them through a CNN that employs dilated convolutions along this axis to finally produce a single set of 3D keypoints. Apart from distinguishing between the use of single images and image sequences, other input source or combination of inputs are also used. For example, Zhou et al. [149] calculate point clouds from depth images and estimate 3D poses based on these point clouds, and Macard et al. [127] leveraged a combination of IMU and image data to obtain the ground truth 3D poses for their 3D Poses in the Wild dataset. However, such methods are not as widely used as pure image-based methods [147].

Summarizing the contents of this general overview with respect to this work, 2D single-person human pose estimation from single images is the most fundamental problem of human pose estimation. However, advanced tasks are not completely disconnected from it, as methods from this domain can be found in various tasks, like in 2D multi-person human pose estimation through top-down approaches and two-stage 3D human pose estimation. Thus, a reliable 2D single-person human pose detector benefits not only the task of 2D single-person human pose estimation, but has also relevance for 3D and multi-person human pose estimation.

3.1.2 Datasets

As the current state-of-the-art in human pose estimation is dominated by deep learning, vast amounts of data are necessary for training and evaluation. To perform both tasks in a standardized way, a variety of datasets is available. With respect to the focus of this work, only 2D human pose estimation datasets for human pose estimation from single images will be covered in the following. In this domain, a variety of older datasets like the *Leeds Sport Dataset* (LSP) [58] together with its extended variant (extended LSP) [59] and the *Frames Labelled in Cinema* (FLIC) dataset [104] exist. They were frequently used by the pioneering deep learning approaches for human pose estimation [86, 98, 123, 125, 131]. However, these datasets are lacking with respect to the needs of modern deep learning approaches in at least one regard: the available amount of data is too small and/or the subjects and activities displayed are not diverse enough. For example, the original LSP dataset contains only 2000 images, all focused on sporting activities. FLIC has with 5003 images substantially more data, however, this data is only taken from 30 different movies, severely limiting the diversity displayed in the images. Extended LSP contains the most images, however, these are still limited to the sports domain, showing no large variety of settings and activities. Over time, these smaller or limited-in-scope dataset were gradually replaced by two popular large-scale human pose estimation datasets: the *MPII Human Pose dataset* (MPII) [3] as well as *Microsoft Common Objects in COntext* (MS COCO) [70], which are frequently used by more modern approaches [10, 78, 86, 111, 117]. Apart from MPII and MS COCO, there exist further large-scale human pose estimation datasets, like the *Look into Person dataset* (50.462 images) [69] and the *AI Challenger dataset* (300.000 images) [133]. However, these are by

far not as commonly used. Thus, a more detailed overview of MPII and MS COCO will be given in the following:

MPII Human Pose Dataset [3]: The MPII Human Pose dataset (MPII) was the first large-scale human pose estimation dataset für 2D human pose estimation that featured both, a large number of samples, as well as a big variety in displayed activities. In its current form[1], the dataset contains approximately 25.000 images with a total of about 40.000 annotated individuals performing 410 different activities. For 2D single-person human pose estimation, a total of almost 29.000 training samples together with annotations is publicly available, while the remaining samples are used as test set, withholding ground truth annotations for keypoint positions. In any case, the approximate location of individual people is provided and can be used, making the localization of individuals unnecessary. A further division of the training set into training and validation set has been performed by Newell et al. [86] by using 3000 images from the training set only for validation during training. To evaluate the performance for 2D single-person human pose estimation, a metric called PCKh is used. Here, detections are classified as correct or incorrect based on their Euclidean distance to the corresponding ground truth annotations and a threshold calculated from an annotated head bounding box (a more detailed introduction will be given in Section 3.1.3). The dataset also features a 2D multi-person human pose estimation task, where keypoints have to be detected and assigned to multiple people in an image without using the approximate locations of these individuals. The official evaluation script for the multi-person case is hereby based on the evaluation procedure proposed by Pishchulin et al. [98]. Detections are assigned to the ground truth based on the highest per-instance PCKh score, and the final evaluation of approaches is then performed based on the mean average precision (mAP).

Microsoft COCO [70]: The public[2] Microsoft Common Objects in COntext (MS COCO) dataset is large-scale and not only focused on human pose estimation but also additional tasks like object detection and scene segmentation. For the human pose estimation task, the dataset features about 200.000 images containing 250.000 annotated people in diverse environments and activities. In contrast to MPII, it does not provide the coarse location of individual human instances in the image, but instead explicitly forbids to use such data during evaluation. Thus, the problem of human pose estimation on MS COCO includes both: The detection of all individuals in an image together with the position of their associated keypoints – the classic multi-person human pose estimation task. From an evaluation perspective, MS COCO treats the human pose estimation problem like an object detection problem, with the difference, that a similarity measure specific to human poses is employed to measure how well a detected pose fits an annotated one. This measure is called Object Keypoint Similarity (OKS), a score between 0.0 and 1.0 that indicates how well a predicted pose reflects the annotated pose under consideration of the difficulty to correctly estimate the position of certain types of keypoints (a detailed introduction to OKS will be given in Section 3.1.3). Based on the OKS for human poses together with a predicted confidence score for each human detection, standard evaluation metrics like average precision and average recall are employed to evaluate different methods.

Although MS COCO is a larger dataset than MPII, it is solely focused on 2D multi-person human pose estimation. Thus, the MPII Human Pose dataset with its focus on 2D single-person human pose estimation will be used throughout this work.

[1] available at `http://human-pose.mpi-inf.mpg.de`
[2] available at `https://cocodataset.org/#home`

3.1.3 Evaluation Metrics

A standardized evaluation of human pose estimation requires standardized evaluation metrics in addition to commonly used datasets like those from Section 3.1.2. In the following, a selection of 2D human pose estimation evaluation metrics will be introduced and discussed, including those typically employed on the MPII and MS COCO datasets:

Percentage of Correct Parts (PCP) [24]: The PCP metric is an evaluation metric that does not aim at directly measuring how well single keypoints were detected, but instead focuses on correctly detected body parts defined by pairs of keypoints, whose direct connection is indicative of the body part. A body part with annotated endpoints $(\boldsymbol{p_{i,1}}, \boldsymbol{p_{i,2}})$ is considered detected correctly, if the distance of both detected endpoints $(\hat{\boldsymbol{p}}_{\boldsymbol{i,1}}, \hat{\boldsymbol{p}}_{\boldsymbol{i,2}})$ from their respective annotated counterparts is less than a constant fraction $c \in [0.1, 0.5]$ of the annotated body parts length. Obviously, this metric can only consider body parts with annotated endpoints, thus let $\boldsymbol{v_i}$ be $True$, if both endpoints $p_{i,1}$ and $p_{i,2}$ are annotated, and $False$ otherwise. Expressed as a formula, the PCP for a single human pose with N body parts is:

$$\text{PCP} = \frac{100}{\sum_{i=1}^{N} \delta(\boldsymbol{v_i})} \sum_{i=1}^{N} \delta(\boldsymbol{v_i})\delta(\frac{||\boldsymbol{p_{i,1}} - \hat{\boldsymbol{p}}_{\boldsymbol{i,1}}||_2}{||\boldsymbol{p_{i,1}} - \boldsymbol{p_{i,2}}||_2} \leq c)\delta(\frac{||\boldsymbol{p_{i,2}} - \hat{\boldsymbol{p}}_{\boldsymbol{i,2}}||_2}{||\boldsymbol{p_{i,1}} - \boldsymbol{p_{i,2}}||_2} \leq c) \tag{3.4}$$

In this formula, δ constitutes a function to convert boolean values as follows:

$$\delta(\boldsymbol{v_i}) = \begin{cases} 0, & \text{if } \boldsymbol{v_i} = False \\ 1, & \text{otherwise} \end{cases} \tag{3.5}$$

The PCP metric has the obvious downside, that the difficulty of detecting a certain human body part is highly influenced by the length of the ground truth body part, which can vary greatly even among instances of the same kind of body part, due to the projection from 3D into 2D when taking pictures – a fact that is e. g., criticized by Andriluka et al. [3].

Mean-based Percentage of Correct Parts (PCPm)[3]: The PCPm metric proposed by Andriluka et al. [3] is an evolution of the PCP metric that aims to decrease the inherent correlation of body part detection difficulty with annotated body part length when using PCP. This is done by replacing the pose- and part-specific part lengths used by PCP through pose-agnostic and part-specific part lengths $\bar{\boldsymbol{l}}_{\boldsymbol{i}}$. The part-specific value $\bar{\boldsymbol{l}}_{\boldsymbol{i}}$ for part i is calculated as the mean of all part lengths of part i in the test set. Then, the resulting constant $\bar{\boldsymbol{l}}_{\boldsymbol{i}}$ is applied during testing every time part i is evaluated. In addition, only $c = 0.5$ is used, meaning that a detected body part is now considered correct, if the distance of both detected endpoints to their annotated counterparts is within 50% of the mean annotated part length. Based on Eq. (3.4) for PCP, this leads to the following formula for PCPm (for a single pose):

$$\text{PCPm} = \frac{100}{\sum_{i=1}^{N} \delta(\boldsymbol{v_i})} \sum_{i=1}^{N} \delta(\boldsymbol{v_i})\delta(\frac{||\boldsymbol{p_{i,1}} - \hat{\boldsymbol{p}}_{\boldsymbol{i,1}}||_2}{\bar{\boldsymbol{l}}_{\boldsymbol{i}}} \leq c)\delta(\frac{||\boldsymbol{p_{i,2}} - \hat{\boldsymbol{p}}_{\boldsymbol{i,2}}||_2}{\bar{\boldsymbol{l}}_{\boldsymbol{i}}} \leq c) \tag{3.6}$$

Percentage of Correct Keypoints (PCK) [139]: In contrast to PCP and PCPm, PCK is the first introduced metric that focuses on keypoint detections directly. To this end,

it focuses on the distance between annotated keypoints $\boldsymbol{y_i}$ and detected keypoints $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$. A detected keypoint is considered correct, if this distance is at most a constant fraction c of the size of an annotated bounding box of the human the keypoint belongs to. For the size of the bounding box with height h and width w, the maximum side length $\max(h, w)$ is used. Depending on the dataset, proposed values for c include 0.1 and 0.2 [139]. Just like PCP and PCPm, the PCK score can only be evaluated when a ground truth annotation is available, thus let $\boldsymbol{v_i}$ be $True$ is the i-th keypoint of a person is annotated, and $False$ otherwise. Using these introduced variables, the PCK score for a single pose with N keypoints can be expressed as follows:

$$\text{PCK} = \frac{100}{\sum_{i=1}^{N} \delta(\boldsymbol{v_i})} \sum_{i=1}^{N} \delta(\boldsymbol{v_i}) \delta(\frac{||\boldsymbol{y_i} - \hat{\boldsymbol{y}}_{\boldsymbol{i}}||_2}{\max(h, w)} \leq c) \tag{3.7}$$

A downside of this metric is the dependence on the size of the annotated human bounding box, as Andriluka et al. [3] criticize. It is highly susceptible to changes in human posture: For example, sitting instead of standing upright can already reduce the height h by about half, which in turn makes it about twice as hard to detect human keypoints correctly (assuming $h > w$ when standing and sitting).

Head-based Percentage of Correct Keypoints (PCKh) [3]: The PCKh metric, which is the official evaluation metric for 2D single-person human pose estimation on MPII, was proposed by Andriluka et al. [3] as a variant of the PCK metric. The goal was to eliminate the influence of the human posture on the distance-based thresholding during evaluation, to e. g., prevent the aforementioned increase in difficulty when sitting instead of standing. To do so, they no longer base their calculations on the size of an annotated human bounding box, but on a fraction of the size of the head inferred from an annotated head bounding box. In contrast to the size of the overall human bounding box, this measure does not change much for different postures. Let s_h denote the size of the head and c the constant fraction of it being used, which is commonly set to 0.5 [147]. Then, based on Eq. (3.7), the PCKh score for a single human pose with N keypoints can be expressed as:

$$\text{PCKh} = \frac{100}{\sum_{i=1}^{N} \delta(\boldsymbol{v_i})} \sum_{i=1}^{N} \delta(\boldsymbol{v_i}) \delta(\frac{||\boldsymbol{y_i} - \hat{\boldsymbol{y}}_{\boldsymbol{i}}||_2}{s_h} \leq c) \tag{3.8}$$

Object Keypoint Similarity (OKS) [70]: The OKS metric is part of the evaluation procedure on the MS COCO dataset. The goal of this metric is to assess how well a detected pose resembles an annotated pose, based on the distance between annotated keypoints $\boldsymbol{y_i}$ and detected keypoints $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$. While this is so far similar to PCK and PCKh, the OKS score differs in two significant ways: First, the evaluation metric does not work with percentages of keypoints classified as correct or incorrect, and does not distinguish between correct and incorrect at all. Instead, a similarity measure with values between 0 and 1 is calculated, indicating through a floating point number how well poses are matches. Second, for the first time, the difficulty of detecting different kinds of keypoints is reflected by the evaluation metric through the use of keypoint type specific standard deviations $\overline{\boldsymbol{\sigma}}_{\boldsymbol{i}}$ obtained under consideration of people's scale in an image from multiple human annotations. The higher the $\overline{\boldsymbol{\sigma}}_{\boldsymbol{i}}$ value, the lower the negative impact of a certain distance on the OKS score. Another factor influencing the OKS score is the scale of the person s_p (annotated). The effect on the OKS is similar to the effect of s_h on the PCKh, with higher values of s_p leading to a lower negative impact of a calculated distance between keypoints

$\boldsymbol{y_i}$ and $\hat{\boldsymbol{y}}_i$ on the OKS score. In this section's notation, the official formula for the OKS score from Lin et al. [70] can be expressed as follows for a single pose with N keypoints:

$$\text{OKS} = \frac{1}{\sum_{i=1}^{N} \delta(\boldsymbol{v_i})} \sum_{i=1}^{N} \delta(\boldsymbol{v_i}) e^{\left(\frac{-||\boldsymbol{y_i} - \hat{\boldsymbol{y}}_i||_2^2}{2s_p^2(2\overline{\boldsymbol{\sigma}}_i)^2}\right)} \tag{3.9}$$

Average Precision (AP): The previously introduced evaluation metrics provide measures only tailored towards the evaluation of 2D single-person human pose estimation – it is assumed that exactly one human instance with annotated keypoints exists per image, making it necessary to predict a single location for each keypoint. This is no longer true in the 2D multi-person setting, as the number of people (and thus the number of keypoints to detect) is no longer known. With the number of individuals and keypoints no longer known, it becomes possible to produce incorrect detection for which no corresponding annotation exists. In this setting, both MPII [3] and MS COCO [70] build upon the long-standing metrics of (average) precision and (average) recall for evaluation. For their calculation, these metrics need three kinds of results: (i) *true positives* (TP), where a matching ground truth exists for a prediction, (ii) *false positives* (FP), where no matching ground truth exists for a prediction, and (iii) *false negatives* (FN), where no matching prediction exists for a ground truth. Whether a detection matches a ground truth or not, is decided on MPII and MS COCO through the PCKh score and OKS score respectively. In both cases, each ground truth can only have one matching prediction. Then, with # denoting the amount of something, a general definition of *precision* and *recall* is:

$$\begin{aligned} \text{Precision} &= \frac{\#\text{TP}}{\#\text{TP} + \#\text{FP}} \\ \text{Recall} &= \frac{\#\text{TP}}{\#\text{TP} + \#\text{FN}} \end{aligned} \tag{3.10}$$

However, when directly used like this, these metrics have the downside that a decision has to be made which human pose estimation results have to be kept for evaluation, with the contraindicating goals of keeping as many as possible to maximize #TP and thus the recall, and to keep only a few selected result which are very likely to be correct to minimize #FP and thereby maximizing the precision.

Instead, a better way to evaluate 2D multi-person human pose estimation performance is the *average precision* (AP) metric, which is the primary evaluation metric for this task used by MPII and MS COCO. For its calculation, an additional confidence score is required for each detection (both MPII and MS COCO utilize scores at the human instance level). Based on this confidence score, it can be decided which 2D multi-person human pose estimation results shall be kept and which shall be discarded. Instead of evaluating just a single threshold for confidence, recall and precision are plotted against one another, forming a *precision-recall curve* (PRC). At each point, this curve shows the precision value that can be achieved for a certain recall value, with different confidence thresholds being employed to obtain each recall value. The *area under the precision-recall curve* (AUPR) is indicative of the performance, and is represented by the average of the precision values along this curve – the AP metric. To maximize AP, there is not only a human pose estimator necessary that detects all annotated human instances and their keypoints precisely but also a way to obtain suitable confidence scores that allow for a separation into correct and incorrect detections.

3.1.4 Selected Approaches

After giving a general introduction to the overall topic of human pose estimation as well as to datasets and evaluation metrics used in 2D human pose estimation, several approaches with high relevance to this work shall be presented in greater detail. As this work does not aim at directly improving performance on evaluation metrics like PCKh but investigates other problems like the identification of incorrect keypoint predictions and the estimation of measurement errors, approaches and experiments should be based on broadly used neural network architectures to ensure good generalization. Two such approaches are the *stacked hourglass* (HG) model proposed by Newell et al. [86] as well as the *High-Resolution Net* (HRNet) from Sun et al. [117]. Both are prominent milestones for the task of 2D single-person human pose estimation, with many works using them as a foundation for their own architectures and modifications [8, 14, 15, 16, 57, 85, 116, 138, 140, 141]. Their importance as well as the common basis they provide for many approaches make them an ideal choice for the investigations of this work. Therefore they will be introduced in greater detail. A third approach that will be highlighted is the work of Wang et al. [129]. For the field of 2D human pose estimation, they were the first to touch on the topic of robustness against noise - a topic that was up to this point neither covered by other research nor through existing benchmarks. To improve robustness against noise, they proposed a data augmentation method called *Adversarial Augmentation Mix* (AdvMix).

Stacked Hourglass (HG) Model [86]: The stacked hourglass model proposed by Newell et al. [86] belongs to the category of heatmap-based 2D single-person human pose estimation methods. The core idea behind the stacked hourglass model is to consecutively apply encoder-decoder-inspired blocks called hourglasses for the prediction and improvement of human pose data in the form of heatmaps. Hereby, a single hourglass consists of three different types of components: 2×2 max pooling layers to half the spatial resolution of inputs in both dimensions, 2×2 nearest neighbor upsampling layers to double the spatial resolution of inputs in both dimensions as well as residual blocks (a concept introduced by He et al. [41]) for data processing. When data enters an hourglass, it is first processed by repeated downsampling and application of residual blocks, with the aim of processing the data at different resolutions. Before every downsampling operation, data is branched off to a skip connection that preserves the spatial resolution and processes the data through a residual block. After reaching the lowest spatial resolution, data is repeatedly upsampled and processed by residual blocks, with data of the same spatial resolution being added from the skip connections. This procedure aims to join features extracted at different resolutions together. To gain a better understanding of the hourglass, the schematics are shown in Figure 3.3. The residual blocks employed by the hourglass consist of a 1×1 convolution, followed by a 3×3 convolution and another 1×1 convolution. The first two convolutions have 128 filters, while the last one uses 256. At the end, features before and after the processing through convolutions are added up. To connect (or rather stack) hourglasses and to calculate intermediate results, additional layers are necessary between individual hourglasses. To this end, a residual block together with several 1×1 convolutions is used in the official implementation[3]: Specifically, the hourglass output is processed by a residual block followed by a 1×1 convolution with 256 filters. The result is processed by another 1×1 convolution with N filters to produce heatmaps (the intermediate result), where N is the number of keypoints. Afterward, the heatmaps as well as the features before heatmap prediction are separately processed by 1×1 convolutions to

[3] available at `https://github.com/princeton-vl/pose-hg-train`

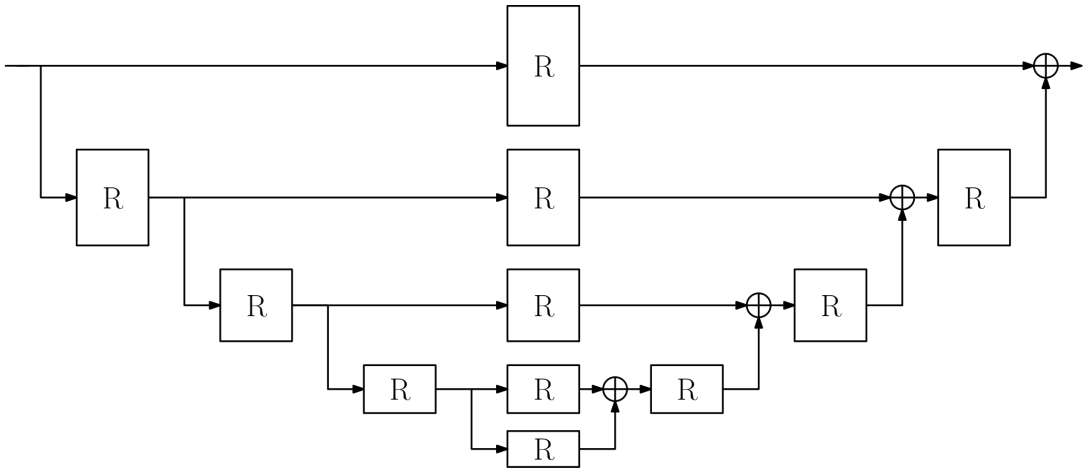


Figure 3.3: An illustration of the architecture of a single hourglass used in the stacked hourglass model. Blocks marked with R denote residual modules, lines moving down denote 2×2 max pooling operations, lines moving up denote 2×2 nearest neighbor upsampling and the plus sign stands for an add operation. Note that this illustration is reflective of the final hourglass structure used in Newell's official implementation of the stacked hourglass model and that it slightly differs from the figure presented by Newell et al. [86] for a single hourglass.

bring them back to a feature dimension of 256. Then, they are added up with features extracted right before the hourglass was applied to form the input for the next hourglass. See Figure 3.4 for an illustration. If the hourglass is the last hourglass of the stacked hourglass model, only the first residual block and 1×1 convolution, as well as the 1×1 convolution for heatmap prediction are applied. For training and evaluation on MPII, Newell et al. use inputs of size 256×256 for their network. Before the first hourglass stack, these inputs are preprocessed and downsampled through several preprocessing layers, including convolutions, residual blocks, and max pooling, to bring the spatial resolution down to 64×64. Apart from the neural network itself, the data augmentation strategy pursued by Newell et al. during training will also be highlighted, as it will also be used throughout this work. When training on MPII, they first crop individuals from the full image by using bounding boxes derived from the annotated position of individuals and resize them to fit the input size of the network. As a first augmentation, before cropping, the size of this bounding box is adjusted to a value between 75% and 125%. After cropping, two other augmentations are performed: Flipping (left-right) of the image with a chance of 50%, as well as a rotation of the image using an angle between -30° and +30°. As ground truth, heatmaps are generated from annotated keypoints by applying a Gaussian kernel at the location of the keypoints at heatmap resolution (64×64). The loss is then calculated using the MSE between all predictions (intermediate and final) and the ground truth heatmaps.

High-Resolution Net (HRNet) [117]: The High-Resolution Net proposed by Sun et al. follows the same core belief as Newell et al. [86] that data processing at different resolutions is crucial for good human pose estimation results. However, instead of repeatedly decreasing and increasing the resolution to accumulate information from different resolutions, Sun et al. want to process data at different resolutions in parallel, including the exchange of information between these resolutions. To this end, they proposed a network architecture that iteratively adds data processing at lower resolutions, while fully maintaining branches that work on higher resolutions. Specifics will be given based upon the

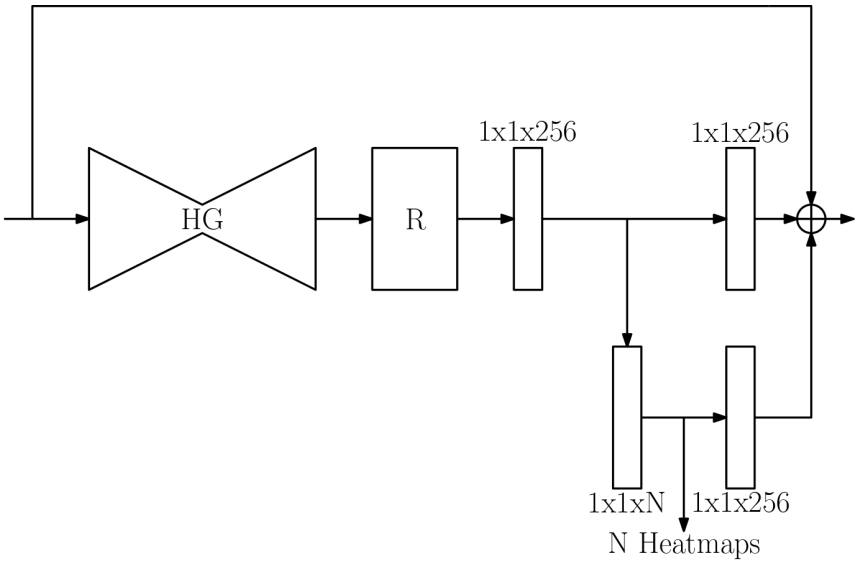


Figure 3.4: Illustration of one hourglass stack (hourglass + intermediate layers) from Newell et al. [86], consisting of one hourglass HG, one additional residual block R, as well as several 1×1 convolutions, denoted through $1 \times 1 \times F$, where F is the amount of filters used in the layer (either 256 or the amount of keypoint N.). This illustration displays the layers used in the official implementation. If the hourglass stack is the last stack, only elements on the path to N Heatmaps will be present.

work of Sun et al. [117] and the official implementation[4]: First, they leverage two 3×3 convolutions with stride two, each decreasing the spatial resolution of the input in both dimensions by half. The resulting spatial resolution is the highest spatial resolution that will be used throughout the network – in the case of MPII, the input will have a size of 256×256, resulting in a resolution of 64×64 for the highest resolution branch. After these initial convolutions, four stages are added. The first stage works on the highest resolution only and consists of four residual modules, each consisting of a 1×1 convolution, a 3×3 convolution as well as another 1×1 convolution. Features from before the first convolution are either added directly to the results of the last convolution, or, in the case of mismatching feature dimensions, another 1×1 convolution is used to make them match. Whenever advancing to a stage of higher order, a new branch with half the spatial resolution in both directions is added first by applying a 3×3 convolution with stride two on the output of the lowest resolution branch of the previous stage. In addition to decreasing the spatial resolution, the size of the feature dimension is doubled. In contrast to the first stage, stages 2-4 consist of so-called exchange blocks, where each exchange block features four residual blocks on every branch as well as one exchange unit. The residual blocks differ from those of stage one, as they only use two 3×3 convolutions instead of two 1×1 and one 3×3 convolution. The following exchange unit is used to exchange features across all scales. For each scale, features are first gathered from all scales and then either processed by an identity function (if the source spatial resolution equals the target spatial resolution), a 1×1 convolution for adjusting the feature dimension together with nearest neighbor upsampling to adjust the spatial dimension (if the source spatial resolution is smaller than the target spatial resolution) or 3×3 convolution(s) with stride two to adjust spatial resolution and feature resolution simultaneously (if source spatial resolution is larger than target resolution). To aggregate the features, gathered and processed

[4] see `https://github.com/leoxiaobin/deep-high-resolution-net.pytorch`

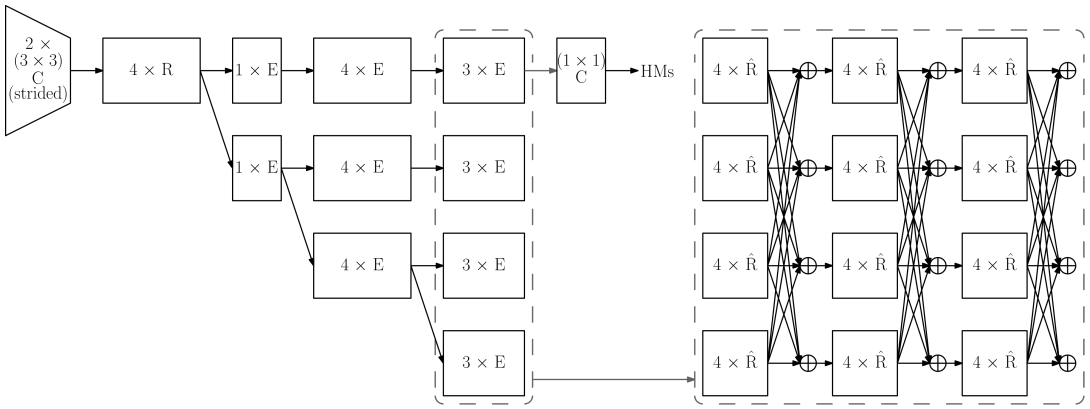


Figure 3.5: An illustration of the HRNet architectures based on the paper and official implementation of Sun et al. [117]. For better understanding, the exchange blocks (E) that span over all resolutions are written out for every resolution, with the three exchange blocks of the last layer being shown in detail to the right. In the image, C denotes a convolution, E denotes an exchange block, R denotes a residual block with two 1×1 convolutions and one 3×3 convolution, and $\hat{\text{R}}$ denotes a residual block with two 3×3 convolutions. Horizontal arrows represent identity functions, arrows pointing down stand for 3×3 convolutions with stride two that double the feature dimensions (one convolution for each branch it moves down), and arrows pointing upward stand for a 1×1 convolution that halves the feature dimension per branch the arrow moves up as well as an nearest neighbor upsampling that doubles the spatial dimension per branch the arrow moves up.

features for each target resolution are added up before they are passed to the next layer. Stages 2-4 use one, four and three exchange blocks respectively. For the prediction of keypoint positions, the features from the highest resolution branch after the last exchange block are used, and processed by a further 1×1 convolution to produce one heatmap per keypoint. An illustration of the network architecture can be found in Figure 3.5. Typical feature dimensions are 32 and 48 on the highest branch in stages 2-4.

Adversarial Augmentation Mix (AdvMix) [129]: Although the AdvMix method proposed by Wang et el. [129] is not used in this work, the background surrounding it is of high relevance. Only recently, Wang et al. were the first to spotlight the fact that the impact of different kinds of noise (e. g., Gaussian, fog, ...) on methods of 2D human pose estimation is severely neglected in current research on the topic, with the work being the first to investigate countermeasures against noise for modern deep-learning based human pose estimators. Their work focuses on a broad variety of noise types that can occur in the wild. They highlight the susceptibility of human pose estimation methods to these noise types through experiments, showing a considerable decrease in performance metrics like PCKh and AP. As anticipation of potential noise types for a scenario in the wild is hardly possible as previously unforeseen events can happen, they focus on improving the robustness of human pose estimation against noise through an adversarial image augmentation strategy they call AdvMix, achieving performance improvements for almost all previously unknown kinds of noise (improvements for all kinds of noise when combined with the stylized images approach [37]) that are investigated, while not being able to eliminate the impact completely. The primary benefit of the approach is in its generalization capability towards previously unknown noise types corrupting images: Their experiments showed that training with image corruptions from a subset of the investigated noise types usually led to better results for these noise types during test time compared to AdvMix

(and optional usage of stylized images). However, results did not generalize as well to all kinds of previously unseen noise types, with AdvMix having the better results most of the time. In addition, the performance on clean data was negatively impacted too. Putting the research of Wang et al. into the perspective of this work, the major motivation for having a strategy like AdvMix is the operation of 2D human pose estimation in an uncontrolled environment where previously unknown types of noise can occur. However, this is not true for the setting that is investigated here, as the industrial environment is highly controlled (outlined in Section 2.2.1). Thus, alternative strategies more tailored towards noise types that can occur in such a setting could prove beneficial (based on the results obtained from training on a subset of noise types) if the negative impact on clean data performance can be removed or minimized.

3.2 Robot Safety

This work focuses on the task of speed and separation monitoring as a motivational example. Looking at SSM from a more generalized perspective, it can be seen as a specific methodology for collision avoidance aimed at humans, where the avoidance is pursued through slowdowns and if necessary a full stop of the robot, based on a monitored safety distance. For the general task of collision avoidance in robotics, be it with humans or arbitrary obstacles, a variety of approaches exists [33, 66, 71, 72, 73, 83, 88, 91, 100, 102, 105, 119, 136, 142]. Apart from a few exceptions, these approaches do not only focus on collision avoidance but also try to achieve an efficient robotic task or movement completion. Generally speaking, collision avoidance approaches consist of two main parts: One part that is responsible for perceiving the environment (perception), with the goal of locating static and moving obstacles in the vicinity of the robot, and the actual collision avoidance part, which acts on the obstacle information obtained from perception. In the following, an overview of different strategies to tackle collision avoidance in recent research will be given, followed by an overview of how human and/or obstacle locations are obtained for these approaches. Afterward, a closer look at two approaches that tackle the motivational example of speed and separation monitoring by using human pose estimation will be given.

In the broad field of collision avoidance, several general strategies for tackling the task exist. One strategy is to approach collision avoidance as an optimization problem [88, 142], where the movement of the robot is determined by solving the optimization problem, and the collision avoidance is performed through imposing constraints on the optimization problem. This can e. g., have the form of constraints on the joint velocity [88] while obtaining the optimal target joint velocity for the next time step. Another set of strategies [33, 136] for approaching the task is inspired by the idea of potential fields: the application of a repulsive force on the robot (originating from obstacles) that is considered during movement. This allows for reaching a goal while avoiding obstacles. For example, Xu et al. [136] move their robot towards a target based on an attractive force vector. For each limb of the human body, the distance to the robot is monitored. For each limb that violates a predefined safety distance, a repulsive force vector is added to the attractive one, repelling the robot from the body part, while still considering the attraction towards the target. Furthermore, it is possible to combine the idea of potential fields with optimization as performed by Liu et al. [71], who apply a repulsive velocity (obtained based on the

calculation of a repulsive force) to the end effector, while collision avoidance for the remaining links of the robot is tackled as an optimization problem with constraints. Based on recent advances in machine learning, the use of reinforcement learning for collision avoidance while completing tasks was explored [73]. To this end, Liu et al. [73] used a deep reinforcement learning approach that rewarded the completion of a task, while penalizing violations of a safety distance to obstacles as well as a delayed completion of tasks to facilitate both, safety and efficient task execution. Other approaches that strive to avoid obstacles while aiming to reach a target or completing a task include for example the use of sampling-based path replanning as a reaction to moving obstacles [91], the formulation of the problem in closed form [83] or the modulation of dynamic systems through a suitable modulation matrix [105]. However, collision avoidance is not limited to performing evasive maneuvers while continuously aiming for a target, it can also be viewed as a separate problem. In this regard, several approaches do not modify the robot's movements, but aim to adjust its overall speed and stop it in time, before a collision occurs [66, 102, 119]. For example, Rosenstrauch et al. [102] monitor the distance between humans and robots and employ a function for scaling the robot's speed down as soon as a certain distance threshold is violated, with a scaling of the velocity to 0 within a safety margin around the robot. Another approach to collision avoidance is presented in the work of Reardon et al. [100], who operate on a more abstract action planning level with a state machine. A distance-based reward function determines whether the robot should take a productive action or a safety action next.

No matter which kind of approach to collision avoidance is pursued, each of them builds on the reliable detection of humans or (more general) obstacles in the vicinity of the robot. Typical devices that are employed for perception include depth sensors [33, 71], RGB/RGB-D cameras [88, 105, 119, 136], camera systems with associated drivers that directly offer human pose information [73, 100, 102] and whole vision-based human tracking system (like e. g., Vicon) [2, 66, 83], which can e. g., be further extended through the use of inertial measurement units (IMUs) [2]. From these sensors and through optional further processing steps, the input data required for different approaches can be obtained. The input data can e. g., include RGB images [136], depth images [33] or point clouds [71, 105], the position of human keypoints and/or associated limbs [100, 102] or more elaborate representations of the human pose and body [66]. Based on the input data, the localization of obstacles is performed or the required location is already given. In any case, obtaining inaccurate or incorrect obstacle locations can become decremental for the following task of collision avoidance. For example, Xu et al. [136] build their approach on the location of 3D human joints that they obtain through their own perception pipeline from RGB images. The safety of the human is pursued through the use of repelling force vectors emitted from the position of human body links. In case of a severely incorrect localization of the human body, the behavior of the system might become completely unpredictable, while some smaller localization error (e. g., slightly above the end effector while in reality being slightly below) could send the robot straight into the human, hereby provoking a collision. Another example is Roenstrauch et al. [102], who work directly on the 3D keypoint positions obtained with the help of a Kinect V2 sensor. Based on these positions, distances between the robot and the human are calculated and used to safely control the speed of the robot, slowing it down when the human is close. In this case, if the closest point with respect to the robot is detected further away than it is, the robot can move at a higher speed than what would actually be allowed, increasing the probability of collisions. To improve the reliability and accuracy of the perception, some

works employ additional measures. Examples of such measures are the use of Kalman filters [71, 105], checks to verify and if necessary undo previous actions taken in iterative perception procedures [136] or the additional use of IMUs (for accurate results under occlusion) [2]. However, the reliability and accuracy of results from the perception part are rarely assessed and later reflected in the collision avoidance method. For example, a perception method is sometimes just declared to be (sufficiently) accurate [66] without an assessment of measurement errors. In other cases, measurement uncertainties in the range of a few centimeters [142] to a few millimeters [2] are declared, but without proof that this will always (or almost always) be the case in the target application under different conditions. Collision avoidance methods also rarely account for measurement uncertainties or outright incorrect results from the perception, with exceptions being e. g., the work of Zanchettin et al. [142], who allow the incorporation of the magnitude of measurement errors directly into their overall optimization problem, as well as the work of Reardon et al. [100], who do not use the measured position of the human directly to obtain a human's state, but employ a state-based probability function instead to account for e. g., uncertainties in the measurement.

In the following, a more detailed look at the works of Svarny et al. [119] and Rosenstrauch et al. [102] will be taken, as they (i) use human pose estimation (without an extensive tracking system) to determine the distance between human and robot, (ii) pursue the application of speed and separation monitoring, and (iii) refer to European safety standards. In the case of Svarny et al. [119], their work is motivated by the monitoring of the protective separation distance S_p as defined by ISO/TS 15066 between human and robot. Instead of the typical realization through laser scanners and safety zone monitoring, the distance between human keypoints and robot keypoints is monitored. Hereby, human keypoints are obtained through human pose estimation. In particular, the authors apply OpenPose [10] to the 2D images of an RGB-D camera and project the resulting 2D keypoints into 3D through an associated point cloud obtained from the camera. For the robot, keypoint positions are calculated as well using the forward kinematics. As keypoint positions are only a discretization of the full body, they further propose to alter the protective separation distance S_p to a keypoint separation distance S_d, which incorporates additional surcharges on S_p to account for the discretization. In particular, specific surcharge values are applied for every human and robot keypoint. These are calculated by assigning each part of the human or robot body to the closest keypoint, and then using the maximum Euclidean distance between a keypoint and its assigned body parts as the keypoint-specific surcharge value. In addition, further keypoint-specific surcharges are applied to realize higher safety distances to certain keypoints. Overall, this means that different distance surcharges have to be considered for each combination of human and robot keypoints, which leads to individual values for S_d for each of these combinations. Concerning e. g., measurement uncertainties and reaction times that can influence S_d, the authors point out that these can already be captured by the original safety distance S_p. However, for their work, they do not point out a methodology to obtain these values for human pose estimation, hence the proposed approach would require an additional method that tackles these problems before it becomes applicable in practice. In their experiments, they ignore this problem by using a constant value for S_p. Furthermore, the authors do not go into any detail on how safety could be maintained in the case of missing keypoint detections, although their methodology explicitly builds on the availability of all human and robot keypoints for pair-wise distance calculations. Similar to Saverny et al., Rosenstrauch et al. [102] also motivate their work based on the speed and separation monitoring task from ISO/TS 15066. How-

ever, their focus lies on an implementation of SSM that adjusts the robot's speed through a continuous function, based on the human-robot distance. They employ a skeleton tracker on data from a Kinect V2 sensor to obtain the position of human keypoints, and calculate the currently relevant human-robot distance as the shortest distance between all human and robot joints. For scaling the robot's speed, two distance thresholds are defined: A minimum distance threshold mandating a full stop of the robot if the human-robot distance becomes smaller and a maximum distance threshold that allows the robot to operate at full speed if the current human-robot distance is larger. Between these two thresholds, linear speed scaling is employed, based on the current distance (from 1.0 at the maximum threshold to 0.0 at the minimum threshold). In experiments, they use two constant values for the thresholds, derived from the robot arm's length. This does not reflect the methodology used for the calculation of the minimum separation distance S_p in ISO/TS 15066, and neither considers values for the body dimensions of the human, nor any uncertainties, reaction times, and more. For practical application, these factors would have to be calculated and considered. Concerning missing keypoint detections that would prohibit distance calculations, they propose to stop the robot, thus achieving a safe state. However, stopping the robot every time a single keypoint detection is missing could severely impact productivity, thus alternative solutions to maintain a safe state would be preferable.

3.3 Uncertainty Estimation for Neural Networks

Whenever neural networks are utilized as a safety-critical part of an application, the reliability of the neural network results is paramount. However, the results $\hat{\boldsymbol{y}}$ produced by the neural network for a given input $\boldsymbol{x}$ can not be viewed as reliable in general, as the whole process that leads to their calculation is affected by uncertainties, originating e. g., from the training data, the neural network training and each input sample $\boldsymbol{x}$ [36, 44]. This makes it necessary to assess the reliability of all neural network outputs on an individual basis. The field of *uncertainty estimation* for (deep) neural networks explicitly focuses on assessing the reliability of neural network outputs by estimating their uncertainty, which in turn can be expressed through different uncertainty metrics, depending on the task and output of the neural network. In the following, a general introduction to uncertainty estimation for neural networks will be given, including a closer look at uncertainty and its sources, metrics for measuring uncertainty, as well as strategies that are employed to obtain uncertainty estimates. Afterward, the assessment of the reliability of human pose estimation results in recent methods will be discussed.

3.3.1 General

When talking about uncertainty in the context of neural networks, two major types of uncertainty are frequently distinguished: *aleatoric uncertainty*, referring to the part of the uncertainty inherent to a problem (unavoidable), and *epistemic uncertainty*, which stems from an imperfect method to approach the problem (avoidable) [44]. A simple example of aleatoric uncertainty would be the prediction of the result of a dice roll: while using all information available for the dice like imbalances could lead to one outcome being favored over the others, no 100% accurate prediction can be made for each dice roll, as the outcome is inherently ambiguous. On the other hand, a simple example of a source of

epistemic uncertainty would be the use of a linear classifier for a non-linear classification problem – uncertainty in this case stems from the use of a method incapable of providing an optimal solution for the problem, and uncertainty could be reduced or completely removed through the use of a non-linear classifier. Closer resembling the notions used with neural networks, Gawlikowski et al. [36] refers to aleatoric uncertainty as *data uncertainty* (viewing data as the only source of irreducible uncertainty) and epistemic uncertainty as *model uncertainty* – a notion that is adopted in this work. Model uncertainty is directly linked to the model's ability to correctly find the best mapping from all potential inputs of its input space to their respective results in the output space [44]. Multiple factors contribute to the model uncertainty, with prominent ones being presented in the following. First, the training of neural networks is typically performed on a subset of all potential inputs, optimizing the network for mapping training inputs to training outputs. However, this training procedure does not ensure that the learned mapping is also optimal for all potential inputs, leading to increased uncertainty the smaller and less representative the amount of training data is [36, 44]. Another contributing factor to the model uncertainty is the model architecture – given that enough training data is available, an incorrect architecture can still prohibit that an optimal mapping between inputs and outputs is found, as its structure is unable to express this mapping [36, 44]. The training procedure of neural networks can further contribute to the model uncertainty, as it is typically a stochastic process that does not guarantee that an optimal mapping for the input data is found [36]. In contrast to model uncertainty, data uncertainty is attributed to the process of data collection through measurement by Gawlikowski et al. [36]: When data is collected, only some characteristics of the real world are captured, leading to a loss of information, while the data that is collected is further effected by factors like noise. In turn, this can lead to e. g., the same or very similar input data being representative of multiple different outputs, thus containing inherent uncertainty about the correct output. Apart from aleatoric and epistimec uncertainty, there is also the uncertainty that is inherent to a single neural network output $\hat{\boldsymbol{y}}$ for a given input $\boldsymbol{x}$, called *predictive uncertainty* [36, 44]. Typically, this uncertainty should contain both, data and model uncertainty, however, the term is always used when the uncertainty for a prediction is calculated, whether it captures all uncertainties or not. With respect to this work, the predictive uncertainty is important, as the reliability of individual keypoint predictions shall be assessed.

To express the uncertainty of a neural network output, different task-dependent measures of uncertainty are employed throughout the literature. For regression problems, a common approach is to express the uncertainty of a prediction through the use of Gaussian distributions, using the mean as the point prediction of the regression task and the variance to represent the uncertainty of this point prediction [34, 65]. Using the variance, uncertainty can also be expressed as an interval over regression values around the mean. Intervals encapsulating the potential solution of a regression task can also be predicted directly [94], without relying on a Gaussian distribution first. Furthermore, the use of a confidence score ranging from 0.0 to 1.0 to indicate the reliability of a regression result has also been proposed [112]. In the field of classification, using scores ranging from 0.0 to 1.0 to indicate the reliability of a result is way more common, as the maximum class probability in such a task is a natural measure of a neural network's trust in its result [36], and can be obtained from a single neural network or (for better results) from e. g., an ensemble of neural networks [65]. To improve the uncertainty assessment in classification, the variance of multiple softmax predictions for the same input can be used [36]. Another option is the use of a confidence score, predicted in addition to the softmax clas-

sification score to account for e. g., overconfidence in the predicted class due to an input far away from the training distribution [19]. To fully capture the uncertainty of a neural network output, it can be necessary to combine different measures, as different measures and procedures for their calculation may only be representative of either data or model uncertainty, while also being influenced by the other [36]. For a more comprehensive and detailed overview of uncertainty measures, please refer to Gawlikowski et al. [36].

So far, methods for obtaining uncertainty estimates have not been discussed yet. To that end, the taxonomy for uncertainty estimation approaches introduced by Gawlikowski et al. [36] will be adopted, who distinguish between four different kinds of approaches: (i) the direct prediction of uncertainty estimates through a single, deterministic neural network (*single network deterministic methods*), (ii) the prediction of uncertainty through the application of Bayesian probability theory to neural networks (*Bayesian neural networks*), (iii) the prediction of uncertainties from multiple neural network outputs calculated by different neural networks (*ensemble methods*) as well as (iv) the prediction of uncertainties based on different outputs from one neural network obtained through different augmentations of the test data (*test-time augmentation methods*). In the following, these approaches will be introduced in greater detail.

Single Network Deterministic Methods [36]: The first category of approaches that is defined by Gawlikowski et al. [36] encompasses all approaches that fulfill two criteria: (i) a single neural network is used for uncertainty estimation, and (ii) the behavior of the neural network is deterministic, meaning that not only the neural network itself is deterministic, but also that no nondeterministic behavior is induced, e. g., through multiple forward passes with differently augmented data. With these restrictions in place for methods of this category, there are only two ways to obtain uncertainty estimates. First, the neural network can be designed in a way that it predicts both – the final prediction for a specific task as well as an estimate for the uncertainty of that prediction, and in doing so, linking the actual prediction task with the uncertainty estimation. A very simple example from Gawlikowski et al. are classification methods, that derive the predicted class as the class with the highest probability over a probability distribution, while the actual probability for the class can be put out as a basic uncertainty measure. The second way to handle uncertainty estimates in the current category is to separate the actual prediction task from the estimation of the associated uncertainty by applying one method to predict results, and another, separate one, to predict uncertainties. An abstract schematic illustration of how single network deterministic methods work can be found in Figure 3.6. Looking at some exemplary methods belonging to this domain, DeVries et al. [19] approach the problem of uncertainty in classification tasks by not only predicting class probabilities, but by extending the neural network architecture by an additional branch that predicts a confidence score for the current result. This branch is explicitly trained in a way to have high confidence if the neural network is sure about its current prediction, i. e., thinks it knows the data and the correct result, and to express low confidence, when the data and their correct interpretation is unknown. This especially aims at identifying incorrect results for data that was not or only poorly represented by the training data, for which high classification probabilities can occur despite them being incorrect, making the probability distribution of the classification task an unsuitable uncertainty measure in these cases. Another approach belonging to the current category is the work of Oala et al. [90]. They aim to capture uncertainty in form of an interval predicted around the current prediction result. To obtain this interval, they define an additional interval neural network, where each parameter is not defined as a single value, but as an interval through a lower and

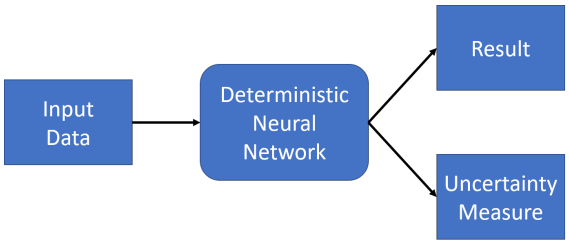


Figure 3.6: Exemplary, abstract visualization of a method belonging to the single network deterministic methods category of Gawlikowski et al. [36]. Input data is processed by a deterministic neural network that directly produces the result for the input, as well as an output to measure the predictive uncertainty of that result.

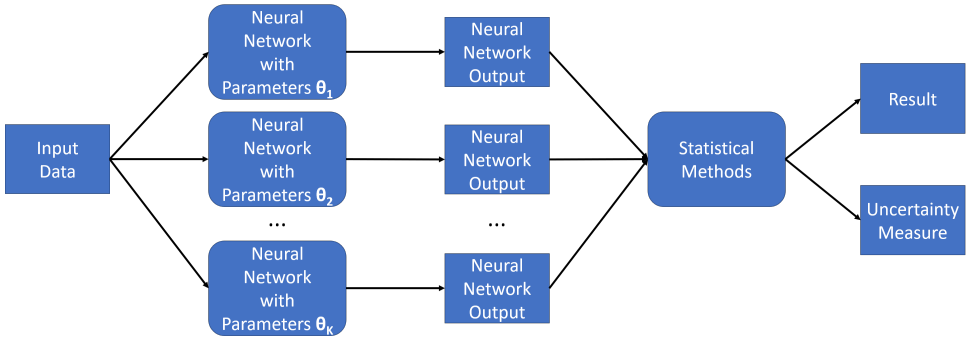


Figure 3.7: Exemplary, abstract visualization of a method belonging to the Bayesian neural network category of Gawlikowski et al. [36]. Input data is processed by the same neural network multiple times with different parameters $\boldsymbol{\theta_1}$, $\boldsymbol{\theta_2}$, ..., $\boldsymbol{\theta_K}$ that have been drawn from an approximation of the posterior distribution $p(\boldsymbol{\theta}|\boldsymbol{D})$. Each time, a different neural network result is produced, which are afterward processed through statistical methods to obtain the final result as well as an estimate of the predictive uncertainty.

upper bound. First, the interval neural network is created as an exact copy of a pretrained neural network, with lower and upper bound for all parameters being the same as the singular parametric values of the original network. In the following, it is trained to produce meaningful intervals, that always contain the parameter values of the original network, in the end resulting in an interval produced for every prediction of the original network when applied to the same input. An example of the full separation of predicting the result for a task and predicting the uncertainty is the work of Raghu et al. [99]. They employ an unmodified classification model for a classification task, and run a completely separate neural network on the same input data to obtain uncertainty estimates for the current prediction. They explore the prediction of different uncertainty measure like the variance, whereas the ground truth uncertainty for training is obtained from multiple annotations made for the same input.

Bayesian Neural Networks [36]: The second category used by Gawlikowski et al. [36] encapsulates all approaches that build upon the principles of Bayesian probability theory. While other neural networks are designed and trained in a way that aims to find the single best set of parameters to solve a specific problem (maximum likelihood), Bayesian methods incorporate a probabilistic view by treating network parameters as well as neural network outputs as probability distributions. Specifically, the distribution of the neural network parameters $\boldsymbol{\theta}$ is modeled through the use of Bayes theorem as a posterior distribution that depends on the training data $\boldsymbol{D}$, while the output distribution for a neural

network output $\hat{\boldsymbol{y}}$ is modeled as a probability that is conditional on both, the current input $\boldsymbol{x}$ as well as the training data $\boldsymbol{D}$ [36]:

$$\begin{aligned} p(\boldsymbol{\theta}|\boldsymbol{D}) &= \frac{p(\boldsymbol{D}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\boldsymbol{D})} \\ p(\hat{\boldsymbol{y}}|\boldsymbol{x}, \boldsymbol{D}) &= \int p(\hat{\boldsymbol{y}}|\boldsymbol{x}, \boldsymbol{\theta})p(\boldsymbol{\theta}|\boldsymbol{D})\, d\boldsymbol{\theta} \end{aligned} \tag{3.11}$$

In this equation, $p(\hat{\boldsymbol{y}}|\boldsymbol{x}, \boldsymbol{\theta})$ takes the role of data uncertainty, while the model uncertainty is explicitly represented by $p(\boldsymbol{\theta}|\boldsymbol{D})$ [36]. While forming a strong theoretical foundation, a direct calculation of $p(\hat{\boldsymbol{y}}|\boldsymbol{x}, \boldsymbol{D})$ is typically not possible [36], thus Bayesian neural networks encompass approximate methods. Gawlikowski et al. further distinguish three different ways of how this approximation can be obtained: (i) variational inference methods that try to fit a parametric distribution $q(\boldsymbol{\theta})$ as close as possible to the true posterior distribution of the network parameters $p(\boldsymbol{\theta}|\boldsymbol{D})$, which can be done by minimizing the evidence lower bound of the Kullback-Leibler divergence $\mathrm{KL}(q(\boldsymbol{\theta})||p(\boldsymbol{\theta}|\boldsymbol{D}))$ between both distributions, so that $q(\boldsymbol{\theta})$ becomes a good approximation of $p(\boldsymbol{\theta}|\boldsymbol{D})$, (ii) sampling methods that create one sample after another for potential values of $\boldsymbol{\theta}$, with the entirety of created samples representing an arbitrary, non-parametric probability distribution, and (iii) Laplace approximation methods that simplify the true posterior distribution of $p(\boldsymbol{\theta}|\boldsymbol{D})$ by assuming it is a multivariate normal distribution, where the mean is the most likely set of parameters (maximum a posteriori estimate of the probability distribution, which can e. g., be obtained through traditional neural network training), whereas the uncertainty of the distribution is represented by a Hessian matrix, with the main challenge for methods from the field being the approximation of the Hessian. Independent from the method ((i) - (iii)) through which the distribution of model parameters is approximated, it is then possible to evaluate statistical measures of uncertainty like the variance by performing multiple forward passes through the neural network with the same input, each time using another set of sampled parameters $\boldsymbol{\theta_k}$ from the probability distribution, with k denoting the k-th forward pass. See Figure 3.7 for an illustration. Looking at examples in the category of Bayesian neural network, one prominent approach is *Monte Carlo dropout* (MC dropout), which was proposed by Gal et al. [34]. In their work, they make two major contributions: First, they prove that training a standard neural network with dropout layers (i. e., applying a Bernoulli distribution to intermediate outputs of individual neural network layers, which sets individual elements to zero with a certain probability) is equivalent to fitting a distribution $q(\boldsymbol{\theta})$ to approximate the true posterior distribution $p(\boldsymbol{\theta}|\boldsymbol{D})$ by minimizing the Kullback-Leibler divergence, as it is done in the category of variational inference methods. Second, they show that a final prediction together with measures for the predictive uncertainty of the output can be obtained through gathering multiple results from the same neural network through multiple forward passes with different samples of the Bernoulli distribution being applied, leading to different results in every forward pass. The final result is calculated as the mean of all results obtained that way, while the predictive variance is represented as the variance of the samples, while also taking a hyperparameter for the the model precision into account.

Ensemble Methods [36]: The third category presented by Gawlikowski et al. [36] includes all approaches that use multiple neural networks as a *neural network ensemble* to obtain predictions and uncertainty estimates. Specifically, this means that each of multiple neural networks is presented with the same input $\boldsymbol{x}$ and produces its own output $\hat{\boldsymbol{y}}_{\boldsymbol{k}}$, with k indicating the k-th neural network that is used. Having obtained multiple outputs,

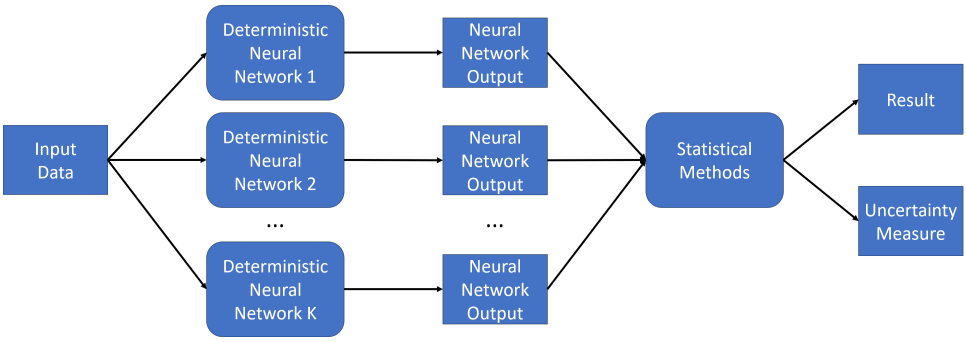


Figure 3.8: Exemplary, abstract visualization of a method belonging to the ensemble methods category of Gawlikowski et al. [36]. Input data is processed by multiple different (structure-wise/weight-wise/...) deterministic neural networks. Each time, a different neural network result is produced, which are afterward processed through statistical methods to obtain the final result as well as an estimate of the predictive uncertainty.

statistical methods can be applied, e. g., to calculate the mean and, as an uncertainty measure, the variance. See Figure 3.8 for a general illustration. For this procedure it is crucial that the employed neural networks do not operate exactly the same, so that they do not produce the same outputs for the same inputs. Especially when uncertainty is high, the models should expose different behavior. To achieve this varying behavior, it turned out that using the same model multiple times while initializing model parameters in a random way, together with the randomness introduced by training on shuffled minibatches, is sufficient for ensembles, as e. g., highlighted by Lakshminarayanan et al. [65]. However, further beneficial diversity can be introduced to the ensembles by other means. For example, Herron et al. [42] highlight the benefits of using different neural network architectures in ensembles to further increase the ensemble's diversity. A prominent example of uncertainty estimation through ensemble methods is the work of Lakshminarayanan et al. [65]. For regression, they use a neural network architecture that already predicts regression results and associated uncertainty measures by using two outputs that reflect mean and variance. To further improve this estimate, they employ an ensemble of neural networks by using the same neural network architecture multiple times, but with different randomized initialization as well as shuffling with minibatches during training. To obtain the predictive uncertainty, they pass the same input once through each neural network, and aggregate the individual outputs of means and variances as a Gaussian mixture model. As final prediction, they represent the Gaussian mixture model through a single Gaussian distribution, which has a single variance that can serve as predictive uncertainty measure.

Test-Time Augmentation Methods [36]: The last category introduced by Gawlikowski et al. [36] encapsulate all methods that employ data augmentation at test time to deal with uncertainty. The general idea in this category is to induce diverse behavior into a single, deterministic neural network by creating a range of different neural network inputs from a single input $\boldsymbol{x}$ through different data augmentations. Different results for an input $\boldsymbol{x}$ are created by passing the different, augmented versions of $\boldsymbol{x}$ through the neural network. With different results available, statistical methods can be employed to calculate the final result and associated uncertainty measures. See Figure 3.9 for an illustration. An important point when performing test-time augmentation that is emphasized by Gawlikowski et al. [36] is that data augmentation at test time should be used in a way that only creates new data belonging to the training distribution. Simple examples of augmentations that can be used during test time are the same that have been applied during training. For

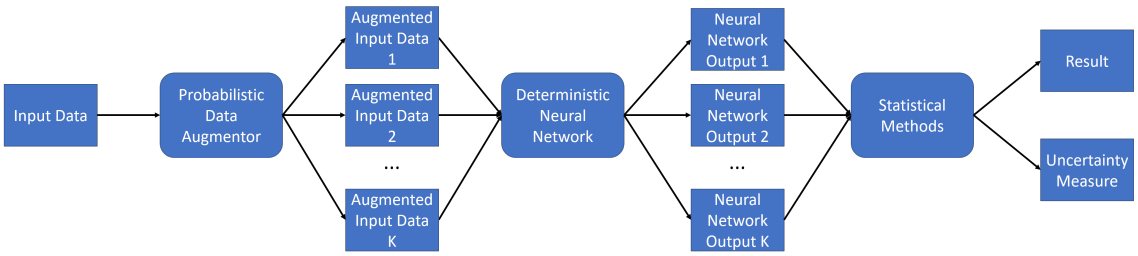


Figure 3.9: Exemplary, abstract visualization of a method belonging to the test-time augmentation methods category of Gawlikowski et al. [36]. Input data is augmented multiple times using non-deterministic augmentation methods to produce a set of differently augmented input data. Each of them is processed by the same deterministic neural network, producing a different output each time. Afterward, these outputs are processed through statistical methods to obtain the final result and an estimate of the predictive uncertainty.

a further investigation on augmentations during test time that are beneficial vs. harmful, Gawlikowski et al. point to the work of Shanmugam et al. [110] on the topic. An example of a test-time augmentation approach is e. g., the work of Wang et al. [128], who use test-time augmentation as a method to calculate the data uncertainty in an image segmentation task. In their work, they use the same augmentations (noise and spatial transformations) during training and test time. Through repeated forward passes with different augmentations, they generate multiple segmentation results. For every pixel, the frequency of each segmentation class is recorded (based on the highest probability in each forward pass), and the entropy is subsequently calculated as an uncertainty measure based on the class frequency.

3.3.2 Human Pose Estimation

In human pose estimation, assessing the (un)certainty of results, be it individual keypoints or the overall resulting pose, is not an integral part of all tasks. Concerning 2D single-person human pose estimation, the common evaluation metrics introduced in Section 3.1.3 do not even offer a way to include the confidence of results into the final score. It is only important whether the detection is correct with respect to the ground truth, while confidence in the detection does not play a role at all, and delivering no keypoint detection for an annotated keypoint is treated equal to supplying an incorrect keypoint detection. In conclusion, methods aiming at 2D single-person human pose estimation do not necessarily produce outputs that could be used to assess the uncertainty of keypoint detections. While heatmap-based methods have a natural way to express uncertainty through the pseudo-probability scores of the heatmaps, the same is not true for regression-based methods, which in some cases only calculate the estimated position of keypoints without further measures [12, 118, 125].

In contrast to single-person human pose estimation, it is an integral part of multi-person human pose estimation to assess how certain a neural network is about its results. For 2D multi-person human pose estimation, both predominant datasets – MPII and MS COCO (see Section 3.1.2) – require a confidence score per detected person expressing how certain the neural network is that the detection is correct. This confidence score is required for the calculation of core performance metrics like the precision-recall curve or (mean) average precision. Therefore, a good estimation of a neural network's (un)certainty about

its detections on a human instance level is crucial for attaining good performance on these datasets. Looking at top-down multi-person human pose estimation methods consisting of an object detector and a separate single-person human pose estimator, the straightforward approach to obtaining confidence scores would be to use confidence scores already supplied by the object detector for its current detection, hereby alleviating the human pose estimator from producing any estimates itself. However, Papandreou et al. [92] discovered that this leads to suboptimal results, and suggested using the average over the maximum of activation maps for each individual keypoint instead, with activation maps being their proposed adaptation of heatmaps. More commonly used is a refinement of this approach, where the confidence scores of single keypoints (e. g., from heatmaps) are combined with the score from the object detector [9, 13, 67], hereby loosely following the idea of ensembles by incorporating predictions from more than one network, however, without explicitly assessing deviations between predictions. An example of such approaches is the work of Chen et al. [13], who calculate the final score for a human pose based on the average of individual keypoint confidence scores (obtained from heatmaps), multiplied by the confidence score associated with the bounding box that is obtained from a object detector. Completely independent of the presence of an object detector, some human pose estimation approaches from the bottom-up domain assess the confidence at human instance level only based on confidence values supplied by the heatmaps [14, 87][5]. For example, Cheng et al. [14] first extract the confidence scores of keypoint-specific heatmaps at the locations where the individual keypoints were detected for a human instance. Then, the confidence value at the human instance level is calculated as the mean over these confidence values. All previously presented approaches so far depend on the presence of heatmaps to obtain the final confidence score, however, these are not present when regression approaches are used. In the absence of heatmaps, some approaches directly aim to predict confidence scores, either on the human instance level or per keypoint [79, 111]. For example, the regression-based transformer approach of Mao et al. [79] supplies per-keypoint confidence scores for the final confidence calculation by predicting the parameters of a probability distribution for each keypoint. Mean and scale are used as parameters for a Laplacian density function, from which the keypoint confidence is obtained as the probability that the actual keypoint locations falls within an interval focused at the mean, that is defined through a fixed offset from the mean in both directions.

Works from the previous section primarily focus on achieving good performance with respect to the evaluation metrics of MS COCO and other datasets, with the calculation of confidence scores being only a smaller part of this procedure. However, there are also works that explicitly focus on the uncertainty assessment in human pose estimation. Bramlage et al. [7] explore two different methods to obtain estimates for both, data and model uncertainty. The first approach builds upon the prediction of the parameters of a Gaussian distribution using maximum a-posteriori inference, with the mean representing the keypoint position and the variance representing data uncertainty. To obtain a measure for the model uncertainty, they apply Monte Carlo dropout and calculate the variance over the predicted mean values in each forward pass. Their second approach is the application of deep evidential regression (introduced by Amini et al. [1]) to the problem of human pose estimation. While their first approach predicts the most likely values for the mean and variance of a Gaussian, deep evidential regression aims to predict the parame-

[5] not mentioned in the respective papers directly, but evident from the code supplied by the corresponding official repositories available at `https://github.com/princeton-vl/pose-ae-demo` and `https://github.com/HRNet/HigherHRNet-Human-Pose-Estimation`

ters of a normal-inverse-gamma distribution which describes the mean and variance of the Gaussian distribution as a distribution. An upside of this approach is, that all three, final keypoint position, data uncertainty, and model uncertainty, can be directly calculated from the predicted parameters of the normal-inverse-gamma distribution, without the need for sampling multiple results like it is the case with Monte Carlo dropout. In any case, confidence intervals reflecting uncertainty for predicted keypoint positions can be inferred from the obtained standard deviations. To obtain better confidence intervals from the predicted standard deviations, an additional recalibration step is performed that multiplies standard deviations with a factor obtained from a learned function for which the output depends on the current quantile of data that shall be covered. Gundavarapu et al. [40] model human pose estimation as a problem of predicting a multivariate Gaussian distribution. Keypoint positions are represented as the mean values, while the covariance matrix is expressive of the uncertainty for such a distribution. Instead of predicting the covariance matrix directly, they aim to obtain the inverse of it, the precision matrix, by predicting the parameters for the Cholesky decomposition of the precision matrix. In turn, the precision matrix and thus also the covariance matrix can be calculated from the neural network outputs. As a final uncertainty measure obtained from the covariance matrix, they investigate both, the overall entropy as well as the variance for different human body joints. Through experiments, they show that their uncertainty measure correlates with the amount of occlusion that is present (which in turn correlates with increased data uncertainty). Further experiments highlight the capability of the uncertainty measure to identify images of the MPII human pose dataset for which their human pose estimator fails to produce correct results, as well as the ability to identify images for which the predicted poses are correct.

Methods that incorporate uncertainty can also be found in other tasks that leverage human pose estimation. For example, aiming at the prediction of the 3D position of humans from 2D keypoints in a single, monocular image, Bertoni et al. [6] leverage a Laplace distribution to capture the data uncertainty inherent to the task of inferring the depth using 2D keypoints in an image from a single camera, while Monte Carlo dropout is performed to incorporate the model uncertainty. Their work focuses only on the step from 2D to 3D, with their proposed architecture using 2D keypoint positions as input, which they obtain from existing 2D human pose estimation methods. Their network is designed to calculate the mean and scale of a Laplace distribution from the 2D keypoints used as input, where the mean represents the estimated 3D position and the scale is representative of the uncertainty. To implement MC Dropout, the network contains the required dropout layers. To obtain a final measure for predictive uncertainty which contains both, data and model uncertainty, they calculate multiple results with active dropout at test time, while also sampling multiple times from the Laplace distribution to obtain depth values. Then, the variance over these results is calculated to represent the predictive uncertainty.

4 Error Reduction

In this chapter, the first of the four central points of this work will be discussed: *How can previously undetected errors of a human pose estimator be detected?* The presence of incorrect results itself is a property that is inherent to treating human pose estimation as a point estimation problem that is solved by neural networks. As previously outlined, neural networks suffer from model uncertainty, induced by factors like the model structure and training procedure, that affects their capability to predict correct results. On the other hand, the formulation of human pose estimation as a point estimation problem, where the only necessary result per keypoint is a single set of coordinates indicating the keypoint's position, is prone to incorrect results whenever there is large enough positional ambiguity in the input that cannot be resolved. However, the presence of such errors is not an acceptable option for safety-critical applications like SSM, as a keypoint localized at an incorrect position poses a potential threat to safety. Throughout this chapter, it is ruled out of the question to change the human pose estimation problem to something else than a pure point estimation problem, as this is the way how the problem is approached on the common 2D single-person human pose estimation datasets. A solution to the problem is the identification of potentially incorrect results that can not be trusted for further use in safety-critical applications. Their identification opens up the use of secondary safety measures which can be employed whenever an unreliable, potentially incorrect result is identified. For example, these could stop the safety-critical application or safely bridge the time until another reliable result for a keypoint is available. This makes the identification of potentially incorrect results a crucial part for the safe use of human pose estimation in safety-critical applications.

As in most cases throughout this work, the problem of identifying incorrect keypoint predictions from neural networks is investigated and pursued for the fundamental task of 2D single-person human pose estimation. Furthermore, the approach for identifying incorrect results shall make as little assumptions as possible about the neural network employed for human pose estimation to ensure broad applicability. Ideally, this means that a method for error detection solely depends on the keypoint positions calculated by the neural network and no further outputs and/or intermediate results that can deviate between neural network architectures and approaches. Although the primary goal is to identify previously undetected errors, this should be achieved while maintaining a low false positive rate (correct detections that are falsely labeled as erroneous). This is necessary to avoid that safety measures building upon human pose estimation results are practically rendered useless when almost every result is labeled as potentially incorrect, and secondary safety measures have to be used all the time to ensure safety. In the following, the problem of error detection in human pose estimation together with definitions like the correctness of a keypoint will be further formalized. Next, the application of safety engineering and neural network uncertainty estimation principles to this problem will be discussed, before presenting potential solutions and their evaluation. The content presented in this chapter is primarily based on previously published work of the author on the application of diverse neural network ensembles for error identification in human pose estimation [106].

4.1 Problem Definition

The defined goal is to find methods that are capable of identifying erroneous human pose estimation results at inference time, whereby erroneous means that a predicted keypoint position would be considered incorrect. As perfectly predicting the correctness of every detected keypoint position is a highly challenging task, it can be relaxed to determining whether a result is *reliable* (correct) or *unreliable* (potentially incorrect). This puts the emphasis on retaining a set of correct keypoint predictions in the reliable category, while the unreliable category pools incorrect results as well as correct results without enough evidence to support their correctness. This relaxation is suitable for retaining safety when sufficient additional safety measures are used whenever an unreliable result is identified.

To obtain a formal definition of the error detection problem for 2D single-person human pose estimation, let $f : \mathbb{R}^{W\times H\times 3} \mapsto \mathbb{R}^{N\times 2}$ denote the function that is realized by a trained neural network for human pose estimation. It maps an input, a 3-channel color image with width W and height H, to a total of N coordinate pairs in image coordinates, with N being the total number of keypoints that shall be localized. Let $\hat{\boldsymbol{y}} = f(\boldsymbol{x}) \in \mathbb{R}^{N\times 2}$ be the keypoint positions predicted by the neural network for a single input image $\boldsymbol{x} \in \mathbb{R}^{W\times H\times 3}$, and let $\boldsymbol{y} \in \mathbb{R}^{N\times 2}$ denote the true (ground truth) locations of the keypoints. Furthermore, let $g : \mathbb{R}^{N\times 2} \times \mathbb{R}^{N\times 2} \times \_ \mapsto \{0,1\}^N$ denote the general definition of a function used to evaluate the neural network output. It calculates the evaluation result $\boldsymbol{b} = g(\hat{\boldsymbol{y}}, \boldsymbol{y}, \boldsymbol{D_g})$ based on the predicted and true 2D keypoint positions, as well as additional data $\boldsymbol{D_g}$ of arbitrary type and shape (dependent on specific evaluation function). The result $\boldsymbol{b} \in \{0,1\}^N$ contains the value 1 at position i, if the i-th keypoint prediction $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ for input $\boldsymbol{x}$ is correct with respect to the true location $\boldsymbol{y}_{\boldsymbol{i}}$, and 0 otherwise. The function g and thus also $\boldsymbol{b}$ have the downside, that they can only be applied and calculated when the true keypoint locations and potential additional data $\boldsymbol{D_g}$ are known (e. g., from human annotations on a dataset), however, this is not the case when the human pose estimator shall be deployed in a practical application. To assess at inference time whether a result is correct or not, it is necessary to define a function $g' : \_ \mapsto \{0,1\}^N$ that mimics the behavior of g and calculates an output $\hat{\boldsymbol{b}} \in \{0,1\}^N$ solely based on data that is available at inference time, including e. g., the input image and neural network results, while explicitly excluding the likes of human annotations. The ideal result of g' is $\hat{\boldsymbol{b}}_{\boldsymbol{i}} = \boldsymbol{b}_{\boldsymbol{i}}$ for every given keypoint prediction $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$. For safety, it is also sufficient to aim for $\hat{\boldsymbol{b}}_{\boldsymbol{i}} \leq \boldsymbol{b}_{\boldsymbol{i}}$, where $\hat{\boldsymbol{b}}_{\boldsymbol{i}} = 0$ now indicates that the results for the i-th keypoint is unreliable (might be correct or incorrect but should not be used or trusted), while $\hat{\boldsymbol{b}}_{\boldsymbol{i}} = 1$ indicates that the result is reliable (assumed to be correct). Only $\hat{\boldsymbol{b}}_{\boldsymbol{i}} > \boldsymbol{b}_{\boldsymbol{i}}$ is safety-critical, as the i-th keypoint position would be considered correct while it is not. To assess the performance of a human pose estimator, together with a realization of g' that makes a statement about the reliability of the predicted keypoint positions, three potential states for a predicted keypoint $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ have to be considered:

$$\hat{\boldsymbol{y}}_{\boldsymbol{i}} = f(\boldsymbol{x})_i \text{ is } \begin{cases} \text{Correct}, & \text{if } \boldsymbol{b}_{\boldsymbol{i}} = 1 \;\&\; \hat{\boldsymbol{b}}_{\boldsymbol{i}} = 1 \\ \text{False}, & \text{if } \boldsymbol{b}_{\boldsymbol{i}} = 0 \;\&\; \hat{\boldsymbol{b}}_{\boldsymbol{i}} = 1 \\ \text{Uncertain}, & \text{if } \hat{\boldsymbol{b}}_{\boldsymbol{i}} = 0 \end{cases} \tag{4.1}$$

With an ideal realization of g', all keypoint predictions that are incorrect with respect to g would fall into the Uncertain category, while all correct keypoint predictions with respect to g would fall into the Correct category. In practice, a good realization of g' should feature a very low amount of results in the False category, while maintaining a high amount of

results in the Correct category. Eliminating (almost) all incorrect results at the cost of leaving almost no results in the Correct category is not desirable, as that would mean that almost all results would fall into the Uncertain category. This would mean that almost no results could be used, which would severely limit practical usability.

To enable the implementation of a function g', the actual evaluation function g that determines whether a detected keypoint position is correct or not should be known beforehand. For common 2D single-person human pose estimation datasets [3, 24, 139], the categorization of keypoint (or part) detections into correct or incorrect is done by thresholding the Euclidean distance between predicted and human-annotated positions with thresholds derived from human annotations. As the experiments in this work will be performed on the MPII Human Pose dataset [3], the same correctness definition that is employed there will be used, which is part of Eq. (3.8) for the calculation of the PCKh score. Transferred into the notation of this chapter, the function for assessing correctness on the MPII dataset g_{MPII} can be written as follows for the i-th keypoint:

$$g_{MPII}(\hat{\boldsymbol{y}}, \boldsymbol{y}, s_h)_i = \delta(\frac{||\hat{\boldsymbol{y}}_{\boldsymbol{i}} - \boldsymbol{y}_{\boldsymbol{i}}||_2}{s_h} \leq c_{MPII}) = \delta(||\hat{\boldsymbol{y}}_{\boldsymbol{i}} - \boldsymbol{y}_{\boldsymbol{i}}||_2 \leq c_{MPII} \cdot s_h) \tag{4.2}$$

Here, s_h (the head-based, annotated normalization distance that changes between different human instances) takes the place of the additional data $\boldsymbol{D_g}$, and it is not available at inference time, just as the annotated location $\boldsymbol{y_i}$. The constant c_{MPII} is known and always the same, typically 0.5. It will be the goal to find a function g'_{MPII} that classifies a result as reliable when it is correct with respect to Eq. (4.2) and as unreliable otherwise. Also note that Eq. (4.2) omits the case that human annotations can miss for certain keypoints, something that is considered in Eq. (3.8) from which it is derived. On datasets, these missing keypoints typically originate from the human annotator being unable to annotate them. The standard procedure is to omit these missing annotations (as it is e. g., also done in Eq. (3.8)), and further detail on this case will be provided when experiments are performed.

4.2 Discussion of Safety Engineering and Neural Network Uncertainty Concepts

Throughout this section, the concepts and strategies from safety engineering as well as from neural network uncertainty estimation will be discussed in the context of their potential for obtaining a function g' that is capable of identifying potentially incorrect results. This shall be done under the following limitations imposed by the problem definition:

1. No assumptions about the neural network for 2D single-person human pose estimation shall be made, except that it solves a point estimation problem. This means that the only outputs that shall be considered are the predicted coordinates for keypoints.
2. Only data available at inference time may be processed. Most and foremost, this includes the 2D input image, as well as keypoint coordinates obtained from the human pose estimation network.
3. Only a single input image is available for the human pose estimator, and no other, additional input data.

Safety Engineering Concepts: First, safety engineering concepts from safety standards will be discussed for the problem of identifying incorrect human pose estimation results. Specifically, the concepts of redundancy and diversity from Section 2.2.2 will be assessed:

1. **Redundancy:** The concept of redundancy itself, where an implementation that fulfills a given functionality is simply duplicated, is not a useful option for detecting incorrect human pose estimation results. The majority of neural networks for human pose estimation are deterministic in nature (see e. g., the representative networks introduced in depth in Section 3.1.4), thus running the same neural network multiple times on the same input leads to the same result (in absence of hardware malfunctions which are not considered here). Having the same result multiple times opens up no options for error identification, thus redundancy is considered unsuitable.

2. **Diversity:** In contrast to mere redundancy, diversity is highly promising. Instead of identical implementations, functionally different implementations are used, meaning that different results for the same input can occur. This opens up the possibility to compare results for the identification of errors, with deviations being indicative of malfunctions. Previously introduced error sources affecting neural networks for human pose estimation include factors like ambiguity or lack of evidence regarding the keypoint position in the 2D input image, as well as factors like the neural network structure or training. Using different neural network architectures as well as a randomized training procedure should prevent that errors caused by architecture and training are identical across multiple networks. As both, architecture and training, are aspects of the neural network's implementation, this claim is further supported by standard IEC 61508-7 [45], which lists diversity as a measure against software implementation errors through result comparison. With respect to ambiguity or lack of evidence in the input, it can be expected that highly different neural networks react in different ways. In case of ambiguity where different, highly likely candidates for the keypoint position exist, it should be likely that different neural networks predict different candidates, again enabling a comparison-based error detection. The same should be true when a lack of evidence occurs and no likely candidate for the keypoint position exists. Overall, this leads to the conclusion that a diversity-based comparison of results should be effective for identifying incorrect human pose estimation results. As the correctness of a keypoint detection is defined based on spatial vicinity, the comparison of results should also be vicinity-based. This makes it necessary to obtain a suitable distance-based threshold from data available at inference time, posing an additional challenge. From a theoretical perspective, the only slightly limiting factor for the concept is that diversity in inputs can not be achieved due to the problem definition mandating a single input.

Neural Network Uncertainty Estimation Methods: Next, the four general methods for assessing uncertainty in neural networks from Section 3.3 will be assessed regarding their capability to identify incorrect human pose estimation results under the given limitations:

1. **Ensemble Methods:** The concept of using an ensemble of neural networks for assessing uncertainty is closely related to using diversity for error detection. In this strategy, multiple neural networks are used to predict the same output for a given input, with uncertainty metrics like the variance being calculated over the results. For human pose estimation, the variance of multiple predicted keypoint positions could be assessed, which is indicative of the distance between them. This can be seen as a distance-based comparison, where high variance is indicative of an

error. Using a threshold on the variance can enable a binary decision whether an error occurred or not. If architecturally different neural networks and randomized training procedures are used for the ensemble, this is almost the same as using diversity, making the strategy useful for the problem in a similar way. For such an ensemble, the field of uncertainty estimation overlaps with safety engineering.

2. **Bayesian Neural Networks:** The assessment of uncertainty through Bayesian neural network approaches works similar to ensemble methods, with uncertainty measures like the variance being calculated over multiple different neural network outputs. Different outputs are obtained by using different weights that are sampled from weight distributions for the same neural network. Compared to an ensemble with architecturally different neural networks, this has the downside that the neural network architecture poses a single point of failure. Furthermore, Bayesian methods make requirements regarding the neural network architecture (e. g., the presence of dropout layers for Monte-Carlo dropout [34]), hereby violating the first limitation. Thus they are considered less viable than ensembles or diversity for the problem.

3. **Test-Time Augmentation Methods:** Methods from the field of test-time augmentation also assess uncertainty similar to Bayesian methods and ensembles based on multiple results. The difference lies in how these results are created. In contrast to using multiple networks or sampled weights, different neural network behavior is induced through different augmentations of the input at inference time. This provokes different results from a single neural network with fixed weights, but suffers from a single point of failure by using only a single neural network. Furthermore, augmentations of the input can be considered a form of noise on the input, which conflicts with the aim of limiting the impact of noise. Therefore, this approach is also considered less viable than ensembles and diversity.

4. **Single Network Deterministic Methods:** The strategy of using single, deterministic neural networks for predicting neural network uncertainty is highly different from the other three strategies, as neither multiple results nor statistical methods are used. Instead, the uncertainty measure is directly predicted by a neural network. For the given problem, this could e. g., be done by using a separate neural network that calculates an uncertainty value from a predicted keypoint position and the input image. Alternatively, the neural network for human pose estimation itself could predict an uncertainty value for each of its predicted positions. Thresholding on the uncertainty value could be used to decide whether a result is correct or not. Both ideas have only a single network responsible for the uncertainty assessment, hence there is a single point of failure. Having the neural network itself predict uncertainty values would further violate the first limitation. However, this violation could be kept small by using the broadly available heatmaps as uncertainty measure. These are already a useful tool for calculating confidence scores for human detections in multi-person human pose estimation (see Section 3.3.2).

Concluding the look at safety engineering concepts and neural network uncertainty estimation methods, two approaches look promising: First is a *combination of diversity and neural network ensembles*, as both share similar ideas for detecting errors likewise assessing uncertainty while fulfilling all introduced limitations. Hence, their combination makes an ideal candidate to realize g', while also unifying both worlds. Second is having a look at *heatmaps as a realization of direct uncertainty estimation*, as it is the only strategy that does not rely on a comparison or statistical evaluation of multiple results.

4.3 Human Pose Estimation and Error Detection with Neural Network Ensembles

Throughout this section, the first idea of combining diversity and neural network ensembles for the realization of a function g' that separates results into reliable and unreliable will be further explored. To this end, an ensemble architecture based on two or more diverse neural networks is proposed. Its task is to predict both, final keypoint positions as well as a binary output for each keypoint that indicates whether the predicted position is reliable or unreliable. This architecture will be called a *diverse neural network ensemble*. First, the proposed architecture of the diverse neural network ensemble will be introduced, followed by the methods for obtaining final keypoint positions as well as the binary reliable/unreliable classifications. The latter performs a distance-based comparison between multiple predicted keypoint positions, which requires a suitable threshold. Last, a method to obtain such a threshold from data available at inference time will be introduced.

4.3.1 Method Design

The proposed diverse neural network ensemble will require two building blocks. The first building block is a set of *two or more diverse neural networks*. Each of them will be used to process the same input image and produce an individual predicted position for every keypoint. No other outputs are produced. As a result, every keypoint will have as many different predicted positions as diverse neural networks are employed. The second building block is a *comparison module*. Based on all individual keypoint predictions from the diverse neural networks, it is responsible for producing the final predicted position for every keypoint, alongside a binary classification for every keypoint that indicates whether the predicted position is reliable or unreliable. Together, both building blocks form the pipeline of the diverse neural network ensemble as depicted in Figure 4.1. The pipeline has characteristics of both, a typical ensemble of neural networks for uncertainty estimation and the safety concept of diversity. Processing the same, single input through multiple neural networks with some differences is most in line with typical ensembles, while diversity would have further preferred a different kind of input for every neural network. However, this was ruled out by the problem definition. In contrast, the comparison module is more in line with the concept of diversity by making a binary decision that indicates whether a result is reliable and may be used further, or if it is unreliable. An ensemble of neural networks for uncertainty estimation would have preferred an output that indicates the degree of uncertainty instead of a binary decision.

Next, the individual building blocks are introduced in greater detail, starting with diverse neural networks. Multiple neural networks are considered diverse in this work, if (i) their architectures are different, and (ii) their training procedures are randomized (e. g., randomized weight initialization, randomized data augmentation, ...) or completely different. Although the literature considers differences in training (especially randomized weight initialization) sufficient for uncertainty estimation with ensembles [36], different architectures can be beneficial [42] and are more in line with the concept of diversity by increasing functional differences. Therefore, diverse neural networks shall have both. Furthermore, they must treat human pose estimation as a point estimation problem by realizing a function $f_k : \mathbb{R}^{W \times H \times 3} \mapsto \mathbb{R}^{N \times 2}$ that predicts keypoint coordinates $\hat{\boldsymbol{y}}_{\boldsymbol{k}} = f_k(\boldsymbol{x})$ from a color image $\boldsymbol{x}$. Hereby, $k \in [1, ..., K]$ denotes the k-th of K diverse neural networks.

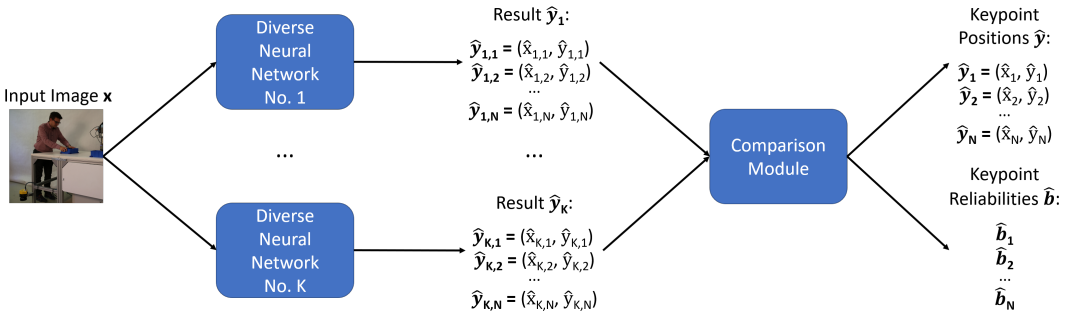


Figure 4.1: The diverse neural network ensemble pipeline. An input image is processed by $K \geq 2$ diverse neural networks, each producing a result $\hat{\boldsymbol{y}}_{\boldsymbol{k}}$ for the keypoint positions. From these results, the comparison module predicts the final keypoint positions $\hat{\boldsymbol{y}}$ as well as their reliable/unreliable classification $\hat{\boldsymbol{b}}$. Based on author's figure from [106].

The second individual building block is the comparison module that needs to realize two functionalities: calculation of the final keypoint positions as well as the binary classification into reliable or unreliable. The latter will be discussed first. Classification shall be performed based on the safety concept of diversity, which means that all results calculated in different ways should be the same to be reliable and used further [45]. However, perfect similarity of multiple predicted keypoint positions is not viable: not even humans are able to agree on a single, perfect keypoint position, which is e. g., highlighted by the average standard deviation of human annotations used by the OKS score (see Section 3.1.3). Instead, close enough spatial vicinity will be used to determine whether multiple keypoint positions match or not – a practice that is also used by many human pose estimation datasets to evaluate predictions against annotations (see Section 3.1.3). As later experiments and evaluations will be performed on the MPII Human Pose dataset, the dataset's vicinity-based comparison from the evaluation metric (see Eq. (4.2)) is a suitable choice as a foundation for determining whether multiple predicted keypoint positions from diverse neural networks match or not. Two major adaptations to this foundation are required. First, the human-annotated threshold s_h can no longer be used and must be replaced with an approximation $\hat{s}_h$ that can be calculated at inference time. Second, the distance-based comparison was only defined for a single pair of keypoint positions, while now a total of K predicted keypoint positions exist which shall all match. To change this, pairwise comparisons between all potential pairs of keypoint positions can be performed, with all of them matching if all pairs match. For a formal definition, let $\hat{\boldsymbol{y}}_{\boldsymbol{k},\boldsymbol{i}} = f_k(\boldsymbol{x})_i$ denote the predicted position for the i-th keypoint from the k-th diverse neural network, $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ the final keypoint position resulting from them, and c_R a selectable constant that takes the role of c_{MPII} from Eq. (4.2). Then, reliable and unreliable are determined as follows:

$$\hat{\boldsymbol{y}}_{\boldsymbol{i}} \text{ is } \begin{cases} \text{Reliable}, & \text{if } \sum_{l=1}^{K-1} \sum_{k=l+1}^{K} \delta(||\hat{\boldsymbol{y}}_{\boldsymbol{k},\boldsymbol{i}} - \hat{\boldsymbol{y}}_{\boldsymbol{l},\boldsymbol{i}}||_2 \leq c_R \cdot \hat{s}_h) = \dfrac{K(K-1)}{2} \\ \text{Unreliable}, & \text{else} \end{cases} \tag{4.3}$$

This comparison can be formulated as an approximation $g'_{R,MPII}$ of Eq. (4.2):

$$g'_{R,MPII}([\hat{\boldsymbol{y}}_{\boldsymbol{1}}, ..., \hat{\boldsymbol{y}}_{\boldsymbol{K}}], s_h)_i = \delta((\sum_{l=1}^{K-1} \sum_{k=l+1}^{K} \delta(||\hat{\boldsymbol{y}}_{\boldsymbol{k},\boldsymbol{i}} - \hat{\boldsymbol{y}}_{\boldsymbol{l},\boldsymbol{i}}||_2 \leq c_R \cdot \hat{s}_h)) = \frac{K^2 - K}{2}) \tag{4.4}$$

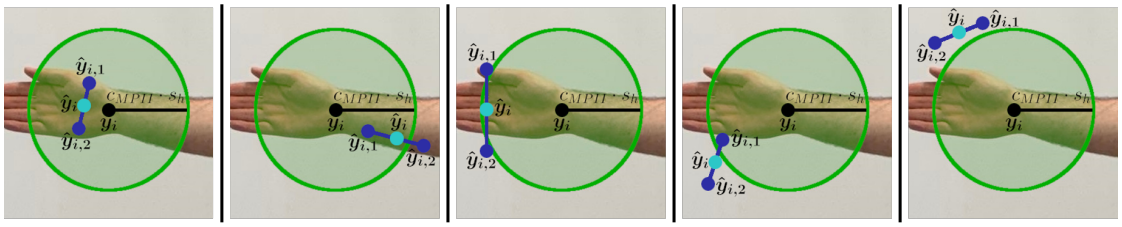


Figure 4.2: Exemplary illustration of potential outcomes when calculating the final keypoint position $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ (cyan) as average over K keypoint predictions $\hat{\boldsymbol{y}}_{\boldsymbol{k,i}}$ (blue, here K=2). All keypoint detections inside the green circle are considered correct for ground truth position $\boldsymbol{y}_{\boldsymbol{i}}$ and distance threshold $c_{MPII} \cdot s_h$ (black). Based on author's figure from [106].

In this equation, each element $\hat{\boldsymbol{y}}_{\boldsymbol{k}}$ of $[\hat{\boldsymbol{y}}_{\mathbf{1}}, ..., \hat{\boldsymbol{y}}_{\boldsymbol{K}}]$ contains all predicted keypoints of the k-th diverse neural network. If the equation evaluates to 1, the output is reliable, while 0 means unreliable. The calculation of the required approximation $\hat{s}_h$ will be subject to Section 4.3.2. Whether Eq. (4.4) is suitable for detecting incorrect results or not will be determined through an isolated proof of concept and further experiments.

Apart from the reliable/unreliable classification, the final keypoint positions must be calculated based on the individual predictions from the diverse neural networks. A common practice among ensembles of neural networks is to average the results [36], which will also be done here. Apart from being a common practice, averaging the results also makes sense for the given problem: Cases in which the average produces very poor results should be identified through the binary classification by labeling the resulting keypoint positions as unreliable. An example of a very poor result from averaging would be if two locations far away from each other exist that could both be the location of the keypoint with high likelihood. In such a case, multiple predictions are expected to be distributed around these two locations, while the average is at a position in-between that makes no sense. However, this also means that not all keypoints are in the vicinity of one another. Hence, the resulting (mean) keypoint position is classified as unreliable, preventing further use. On the other hand, if the binary classification yields reliable, only keypoints in the vicinity of one another exist, making the average keypoint position a good approximation of individual results. Therefore, the final keypoint position $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ for the i-th keypoint is calculated as follows:

$$\hat{\boldsymbol{y}}_{\boldsymbol{i}} = \frac{1}{K} \sum_{k=1}^{K} \hat{\boldsymbol{y}}_{\boldsymbol{k,i}} \tag{4.5}$$

While $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ is a meaningful final keypoint prediction, it can either be correct or incorrect with respect to g_{MPII}, (partially) depending on whether the individual predicted keypoints $\hat{\boldsymbol{y}}_{\boldsymbol{k,i}}$ where correct or not. A selection of potential outcomes is presented in Figure 4.2. It can only be guaranteed that $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ is correct, if all $\hat{\boldsymbol{y}}_{\boldsymbol{k,i}}$ where correct. In this case, all $\hat{\boldsymbol{y}}_{\boldsymbol{k,i}}$ lie within a 1D sphere (a circle) with center $\boldsymbol{y}_{\boldsymbol{i}}$ (the ground truth position) and radius $c_{MPII} \cdot s_h$ (the threshold for correctness), which constitutes a convex geometry. As every convex combination (like the mean) of points within a convex geometry falls within the convex geometry, this is also true for $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$, meaning that it is also correct. In every other case, $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ can either be correct or incorrect. Although $g'_{R,MPII}$ can thus not guarantee that the average $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ is correct if labeled reliable, the same is true for every single prediction of $\hat{\boldsymbol{y}}_{\boldsymbol{k,i}}$. Therefore, the use of $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ is not invalidated by the potential of incorrect results, as the alternative of using one of the multiple available predictions yields no theoretical benefit.

4.3.2 Threshold Calculation at Inference Time

To enable an application of the diverse neural network ensemble at inference time, it is necessary to calculate and use an approximate $\hat{s}_h$ instead of the human-annotated s_h for distance thresholding as in Eq. (4.4). This calculation must be performed based on data available at inference time. To be in line with functional safety principles, the calculation of $\hat{s}_h$ should be performed in a way that does not introduce potential new errors when it is used as a substitute for s_h. New errors can only be introduced when $\hat{s}_h > s_h$, as a larger threshold leads to fewer results being labeled as unreliable, with the potential of additional incorrect results being labeled as reliable. Therefore, approximate values of $\hat{s}_h$ should be the same or more conservative than the respective values for s_h, meaning $\hat{s}_h \leq s_h$. In this case, the same amount or more results will be labeled as unreliable, however, no additional incorrect results will be labeled as reliable.

As a reference point, the calculation of s_h values will be introduced first. Each is obtained from an annotated head bounding box with width w and height h as follows [3]:

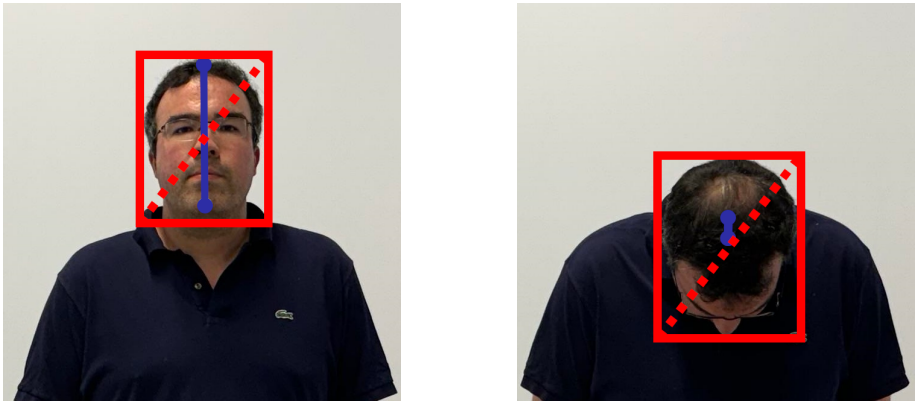
$$s_h = 0.6 \cdot \sqrt{w^2 + h^2} \tag{4.6}$$

The straightforward way to get an approximation $\hat{s}_h$ for this would be to employ a head bounding box detector and perform the same calculation using the width and height of detected bounding boxes. However, the bounding box detector is a potential additional, unchecked source of errors, that can easily lead to less conservative $\hat{s}_h$ values whenever the predicted bounding box is too large. A safer approach for obtaining $\hat{s}_h$ would be to leverage the diverse neural networks, hereby building on diversity again. As each of them only predicts keypoint positions, the calculation of $\hat{s}_h$ must be based on them.

Keypoint positions on their own are not very useful for calculating $\hat{s}_h$. However, typical keypoints include joints, which means that certain pairs of keypoints define rigid body parts and can be used to calculate their length. On the MPII dataset, two keypoints are especially useful for conservatively approximating $\hat{s}_h$: the upper neck and the top of the head keypoints. They define the start- and endpoint of the head itself, therefore they should be close to the border of a bounding box encapsulating the head closely. Although these points do not define the diagonal of the head bounding box used in Eq. (4.6) directly, the distance l_H between them can be seen as a conservative approximation of the diagonal as (i) they lie within in the bounding box, meaning l_H is smaller than the diagonal, and (ii) they are usually close to the bounding box border (exceptions being discussed later). Furthermore, both keypoints have the highest correct detection rates among all keypoints according to official MPII data[1]. While the use of these two keypoints alone might seem to be a good idea at first, the distance between both points can vanish due to the projection from 3D into 2D when taking an image. In cases where the 3D keypoint locations are aligned closely to the same ray from the camera, their 2D projection will be at approximately the same position, leading to a distance close to 0 and thus a bad approximation of $\hat{s}_h$. Figure 4.3 illustrates where l_H leads to a good vs. a bad approximation.

To resolve this issue, an additional distance obtained from other keypoints is necessary, with the requirement that this distance cannot vanish simultaneously. Furthermore, a relation between this distance and l_H should be established, such that the new distance can also be mapped to a meaningful approximation of $\hat{s}_h$. A suitable choice for these keypoints

[1] available at `http://human-pose.mpi-inf.mpg.de/#results`



(a) Good approximation **(b)** Bad approximation

Figure 4.3: Illustration of using the distance (blue line) between the upper neck and top of the head keypoints (blue dots) as an approximation of the diagonal (red dotted line) of a head bounding box (red rectangle). On the left side, a good conservative approximation is achieved. However, on the right side, the distance between both keypoints vanishes due to the projection into 2D, leading to a conservative yet very poor (way too small) approximation of the bounding box diagonal.

are the shoulder keypoints, with their distance to one another being called l_S. As the upper neck is very close to the line connecting both shoulders, l_S can only vanish at the same time as l_H, when the shoulder keypoints are closely aligned along the same line as the upper neck and top of the head keypoints (assuming the camera has a reasonable distance of a few meters to the human). However, due to the spatial dimensions of the body around these keypoints, this is highly unlikely – for an illustration, see Figure 4.4. Therefore, both distances should not vanish at the same time. A further point that speaks for the shoulder keypoints is that they are detected correctly more often than most other keypoints according to official MPII data, except for the aforementioned top of the head and upper neck keypoints. To put l_H into relation to l_S, the work of Winter [132] can be applied, which contains a model of body part lengths relative to the body height. This enables the inference of a relation between shoulder width and head height, which is approximately 2:1 based on the body model used by Winter. This will be used to put l_S into context of l_H by assuming $0.5 \cdot l_S = l_H$ (the ideal case where body parts are not foreshortened due to projection into 2D). Without leveraging the diverse neural networks, an early idea for the approximation $\hat{s}_h$ with respect to Eq. (4.6) could be $\hat{s}_h = 0.6 \cdot \max(0.5 \cdot l_S, l_H)$, with the maximum being used to keep the distance which has lost less length due to the projection into 2D. However, if these distances are calculated from e. g., only a single of the diverse neural networks, both l_H and l_S are prone to errors due to incorrectly localized keypoints making up the respective body parts. This can result in distances being too large, which in turn can lead to values for $\hat{s}_h$ that are too large and thus safety-critical.

The remaining question is how to incorporate the diverse neural networks into the calculation of $\hat{s}_h$ to decrease the risk of safety-critical errors. Similar to the comparison between keypoints, for which $\hat{s}_h$ is required, this will be done through a comparison-based approach. For each of the diverse neural networks, both length values will be calculated from their keypoint predictions, producing the values $l_{H,k}$ and $l_{S,k}$ for the k-th of K diverse neural networks. Comparison will be performed between different values for $l_{H,k}$ and $l_{S,k}$ respectively. The final value for l_H will be calculated as average over all values

(a) Vanished head

(b) Vanished shoulder

(c) Vanished shoulder, head aligned

Figure 4.4: Comparison of the distance between the two shoulder keypoints (orange) with the distance between the upper neck and top of the head keypoints (blue). Whenever the shoulder or head distance (almost) completely vanishes, the other one does not. This is even true in example (c), where the head is aligned with the line connecting the shoulder keypoints until the body blocks further alignment.

$l_{H,k}$, and the validity will be assessed based on the difference between the largest and smallest value for $l_{H,k}$, resulting in a variable v_H which is True if the largest value for $l_{H,k}$ is at most 50% larger than the smallest value for $l_{H,k}$:

$$\begin{aligned} v_H &= \begin{cases} \text{True}, & \text{if } 1.5 \cdot \min(\{l_{H,k} | k \in [1, ..., K]\}) \geq \max(\{l_{H,k} | k \in [1, ..., K]\}) \\ \text{False}, & \text{else} \end{cases} \\ l_H &= \frac{1}{K} \sum_{k=1}^{K} l_{H,k} \end{aligned} \tag{4.7}$$

Hereby, the formulation of v_H is loosely inspired by the PCP metric (see Section 3.1.3), where a fraction of the length of annotated body parts is used to assess whether a prediction is correct for a given ground truth annotation, with 0.5 being a common choice for the fraction [3, 147]. The same fraction is used here to define the maximum allowed deviation between multiple lengths for the same body part: The largest length may be at most 50% larger than the smallest. Definitions similar to those for v_H and l_H will be used for calculating v_S and l_S based on multiple $l_{S,k}$ values. Whenever v_S or v_H are False, it means that the corresponding value for l_S respectively l_H should not be used. The situation that both are False can occur too, thus a conservative fallback value is necessary. This value will be called $\overline{s}_{h,min}$ and is calculated as the average over the 5% smallest values for s_h, with the goal that $\overline{s}_{h,min}$ is highly conservative, but not as conservative as the smallest value. Building on Eq. (4.6) and Eq. (4.7), this leads to the final formula for $\hat{s}_h$:

$$\hat{s}_h = \max(0.6 \cdot 0.5 \cdot \delta(v_S) \cdot l_S, 0.6 \cdot \delta(v_H) \cdot l_H, \overline{s}_{h,min}) \tag{4.8}$$

The maximum is used to get the largest, non-vanishing conservative approximation of $\hat{s}_h$ that is considered valid. Validity is ensured by $\delta(v_S)$ and $\delta(v_H)$, which set the corresponding calculated lengths l_S and l_H to 0 if they are not valid. This means that they are ignored by the $\max$ function as $\overline{s}_{h,min} > 0$. Now, the approximation $\hat{s}_h$ can be calculated at runtime, with a safe and conservative estimation of $\hat{s}_h$ being aided by the diverse neural networks under the application of the principle of diversity for error reduction. An evaluation of the methods for obtaining $\hat{s}_h$, as well as additional experiments with the calculated $\hat{s}_h$ values, will be provided in the experimental section.

4.4 Heatmap-Based Error Detection

After introducing the diverse neural network ensemble approach for detecting potentially incorrect results in the previous section, this section is dedicated to the detection of incorrect results based on the contents of predicted heatmaps $\hat{\boldsymbol{h}}$. Heatmaps are intermediate outputs of heatmap-based neural networks, from which final keypoint positions $\hat{\boldsymbol{y}}$ are inferred. Their use constitutes a violation of the first limitation from the problem definition in Section 4.2 which says that keypoint coordinates shall be the only neural network outputs to be considered. The reason behind this limitation is to ensure the broad applicability of error detection methods. With this goal in mind, the use of heatmaps is only a slight violation of this limitation, due to heatmap-based methods being widely adopted for 2D single-person human pose estimation (see Section 3.1.1). In contrast to this slight violation, the use of heatmaps offers the opportunity to investigate error detection approaches that are highly different from the diverse neural network ensemble. This difference is that neither a comparison nor a statistical evaluation of multiple results has to be performed. Instead, the reliability of individual keypoints can be directly assessed based on the associated heatmaps. Such approaches can be categorized as single network deterministic methods in uncertainty estimation, the category from Section 3.3.1 that has the largest difference from all others. This makes heatmap-based methods worthy of investigation.

4.4.1 Heatmaps as Uncertainty Measure

The concept of heatmaps was previously introduced in Section 3.1.1. The primary purpose of a single predicted heatmap $\hat{\boldsymbol{h}}_i$ for the i-th keypoint is the identification of the keypoint's location. At each location, the heatmap contains a pseudo-probability score that indicates how likely it is that the keypoint is located at this location, with the final keypoint position typically being inferred based on the location of the heatmap maximum. With respect to uncertainty, the magnitude of the maximum can be seen as indicative of how certain the neural network is in its result for the keypoint. Maximum values from heatmaps have also been successfully used as (part of) confidence scores. An example is the use for expressing the confidence into an entire human detection in multi-person human pose estimation, where e. g., the average of maximum heatmap activations across different keypoints [92] or the combination of this average with other elements [9, 13, 67] was used. Furthermore, Newell et al. [86] showed on the MPII Human Pose dataset that the maximum (or mean) heatmap scores are suitable for deciding whether an individual keypoint has an annotated position or misses one (human was not able to determine the position). Therefore, it is a straightforward and reasonable idea to employ the established concept of using the heatmap maximum as confidence measure to define a function g' that separates results into reliable and unreliable, as well as to explore other heatmap-based functions.

The use of heatmaps for defining a function g' has some differences to the examples mentioned above. The separation into annotated and not annotated keypoints performed by Newell et al. is a much simpler problem, as the absence of a keypoint annotation is influenced by factors like heavy occlusion or the keypoint being outside the image [86]. This means the image strongly lacks evidence for the keypoint's location, meaning heatmaps with very low values can be expected. This can also have an influence when average maximum heatmap scores are used as (part of) confidence scores for whole human detections in multi-person human pose estimation, e. g., when a previously used person de-

tector cropped off parts of a human, leading to a bad detection that is scored accordingly. However, the relation between maximum heatmap scores and detection quality for existing keypoints also plays a role. For defining a function g' that approximates g_{MPII} from Eq. (4.2), the most important heatmap property is that the magnitude of a heatmap's scores is indicative of the distance to the annotation keypoint position, as g_{MPII} builds upon the distance between predicted and annotated keypoint positions. This is typically the case when the standard training procedure for heatmap-based methods as introduced in Section 3.1.1 is used. Then, the predicted heatmap aims to reproduce a ground truth heatmap that is calculated by applying a 2D Gaussian at the annotated keypoint position, which makes the Euclidean distance part of every score. Thus, a low Euclidean distance to the annotation can be assumed for a keypoint prediction with high associated maximum heatmap score, while the opposite is true for a low maximum score. This reinforces the assumption that heatmaps are useful for defining a function g'.

When using g_{MPII} to determine the correctness of a keypoint prediction on the MPII Human Pose dataset, there is not only a dependency on the Euclidean distance, but also on the annotated value s_h, which is different for every person. This highlights a conceptual shortcoming of using heatmaps to define a function g' to approximate g_{MPII}, as predicted heatmap scores are only assumed to be indicative of the Euclidean distance and not of s_h, making it impossible to decide whether a keypoint prediction is correct or not even if the Euclidean distance would be perfectly represented by the predicted score. However, lower Euclidean distances can be assumed for higher predicted heatmap scores, thus increasing the chance of fulfilling g_{MPII} for an unknown s_h.

4.4.2 Method Design

Two ways of using the predicted heatmaps for assessing keypoint reliability will be investigated: The first one will use the *heatmap score* at the predicted keypoint position, the second one will assess *how well a predicted heatmap matches the expected heatmap* for a predicted keypoint position, thus considering more information than only the maximum.

The first method employs the established principle of using the heatmap maximum. It is practically similar to what Newell et al. [86] does, however, the approach is applied to identify incorrect keypoint results instead of those with missing annotations. The final predicted keypoint position $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ for the i-th keypoint is obtained as the position of the maximum in the corresponding heatmap $\hat{\boldsymbol{h}}_{\boldsymbol{i}}$, backprojected into the original image through a transformation T^{-1}. The maximum $\hat{\boldsymbol{h}}_{\boldsymbol{max,i}}$ of the i-th heatmap expresses the reliability:

$$\hat{\boldsymbol{y}}_{\boldsymbol{i}} = T^{-1}(\underset{u,v}{\text{argmax}}(\hat{\boldsymbol{h}}_{\boldsymbol{i}}[u,v])), \;\; \hat{\boldsymbol{h}}_{\boldsymbol{max,i}} = \max(\hat{\boldsymbol{h}}_{\boldsymbol{i}}) \tag{4.9}$$

A realization of g'_{MPIII} based on $\hat{\boldsymbol{h}}_{\boldsymbol{max,i}}$, which will be called $g'_{H-Max,MPIII}$, can then be realized by thresholding over $\hat{\boldsymbol{h}}_{\boldsymbol{max,i}}$:

$$g'_{H-Max,MPIII}(\hat{\boldsymbol{h}}_{\boldsymbol{max}})_i = \delta(\hat{\boldsymbol{h}}_{\boldsymbol{max,i}} \geq c_{max}) \tag{4.10}$$

This leaves finding a suitable constant for c_{max} up to debate, which could e. g., be done statistically based on data. Furthermore, the approach of using the heatmap maximum has the downside that major parts of the heatmap are ignored, which could still contain relevant information regarding the reliability of results. For example, multiple high heatmap

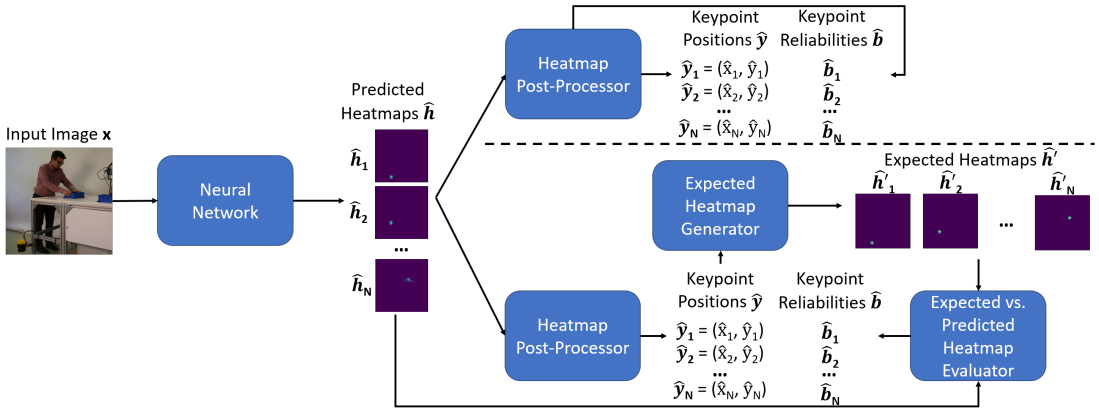


Figure 4.5: Pipelines of the two heatmap-based methods that build on predicted heatmaps $\hat{\boldsymbol{h}}$. The upper branch shows the method building upon the heatmap maximum for predicting reliability, where keypoint position and binary reliable/unreliable classification can be directly obtained from the predicted heatmaps. The lower branch illustrates the second method. Only the final keypoint positions $\hat{\boldsymbol{y}}$ are directly inferred from the predicted heatmaps. From these positions, the expected heatmaps are generated and compared to the predicted ones to obtain the binary classification into reliable and unreliable.

values in different places could indicate multiple likely keypoint candidates. To capture this information, a suitable method is required. Using the mean of the scores in a single heatmap (like Newell et al.) seems not suitable for the given problem, as multiple low scores could lead to the same mean as a single high score.

Instead, it will be leveraged that the heatmap $\hat{\boldsymbol{h}}_{\boldsymbol{i}}$ should look like a heatmap created during training if the neural network is very certain about its result $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$. Deviations like a smaller maximum score or multiple high scores in different locations are indicative of uncertainty. To assess these deviations, a new heatmap $\hat{\boldsymbol{h}}'_{\boldsymbol{i}}$ will be created through the same process that would have been used to create the ground truth heatmap $\boldsymbol{h}_{\boldsymbol{i}}$ during training, but with $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ replacing $\boldsymbol{y}_{\boldsymbol{i}}$ in the process. This leads to $\hat{\boldsymbol{h}}'_{\boldsymbol{i}}$ being a perfectly shaped heatmap, and exactly the training target $\boldsymbol{h}_{\boldsymbol{i}}$ if $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ equals the ground truth annotation $\boldsymbol{y}_{\boldsymbol{i}}$. To assess the degree of deviation between $\hat{\boldsymbol{h}}_{\boldsymbol{i}}$ and $\hat{\boldsymbol{h}}'_{\boldsymbol{i}}$, the mean squared error (MSE) can be applied to obtain a measure $\hat{\boldsymbol{h}}_{\boldsymbol{mse,i}}$ of uncertainty that can be used for thresholding:

$$\hat{\boldsymbol{h}}_{\boldsymbol{mse,i}} = \mathrm{MSE}(\hat{\boldsymbol{h}}_{\boldsymbol{i}}, \hat{\boldsymbol{h}}'_{\boldsymbol{i}}) \tag{4.11}$$

A low score for $\hat{\boldsymbol{h}}_{\boldsymbol{mse,i}}$ correlates with high confidence of the neural network in its result. To obtain a very low score, it is not only necessary to have a high heatmap maximum similar to the maximum in ground truth heatmaps, but other factors impacting reliability must also be absent, like multiple high heatmap values in different locations. A function for binary reliable/unreliable classification $g'_{H-MSE,MPII}$ can again be defined through thresholding $\hat{\boldsymbol{h}}_{\boldsymbol{mse,i}}$ with a constant threshold c_{mse}:

$$g'_{H-MSE,MPIII}(\hat{\boldsymbol{h}}, \hat{\boldsymbol{h}}')_i = \delta(\hat{\boldsymbol{h}}_{\boldsymbol{mse,i}} \leq c_{mse}) \tag{4.12}$$

Again, it is necessary to find a value for the constant c_{mse}, e. g., statistically. For a quick reference, Figure 4.5 shows the pipelines realized by both methods from this section.

4.5 Experiments

To evaluate the proposed approaches for the identification of potentially incorrect results, experiments will be performed on the MPII Human Pose dataset. The goal of the experiments is to show how good the different methods are at filtering out incorrect results while retaining a high rate of correct detections. To this end, it is necessary to modify the standard evaluation metric of the dataset, the PCKh score. Previously introduced for a single human with N keypoints in Eq. (3.8), this equation will now be presented in a form that accounts for a total of M inputs with one human each [3]:

$$\text{PCKh} = \frac{100}{\sum_{j=1}^{M}\sum_{i=1}^{N}\delta(v_{j,i})}\sum_{j=1}^{M}\sum_{i=1}^{N}\delta(v_{j,i})\delta(\frac{||\boldsymbol{y_{j,i}} - \hat{\boldsymbol{y}}_{\boldsymbol{j,i}}||_2}{s_{h,j}} \leq c_{MPII}) \tag{4.13}$$

Index j denotes the j-th of M inputs and index i still denotes the i-th of N keypoints. The variable $s_{h,j}$ is the human-specific head-based distance used in thresholding for the j-th input, while $v_{j,i}$ denotes whether a keypoint is annotated ($True$) or not ($False$).

This equation is not suitable for evaluating the methods of this chapter, as it (i) does not consider reliable/unreliable classification and (ii) does not account for the fact that the percentage of incorrect results is no longer implicitly defined by the percentage of correct ones. Therefore, three new evaluation metrics will be defined with respect to Eq. (4.1): the *percentage of correct keypoints* (C), *false keypoints* (F) and *uncertain keypoints* (UC). With $\hat{\boldsymbol{b}}$ denoting the binary classification result from a realization of g' that predicts 1 when the output is reliable and 0 otherwise (1 will be assumed for all $\hat{\boldsymbol{b}}$ values if no classification function is used), this leads to the following equations for M test samples:

$$\begin{aligned}
\text{C} &= \frac{100}{\sum_{j=1}^{M}\sum_{i=1}^{N}\delta(v_{j,i})}\sum_{j=1}^{M}\sum_{i=1}^{N}\delta(v_{j,i})\hat{\boldsymbol{b}}_{\boldsymbol{j,i}}\delta(\frac{||\boldsymbol{y_{j,i}} - \hat{\boldsymbol{y}}_{\boldsymbol{j,i}}||_2}{s_{h,j}} \leq c_{MPII}) \\
\text{F} &= \frac{100}{\sum_{j=1}^{M}\sum_{i=1}^{N}\delta(v_{j,i})}\sum_{j=1}^{M}\sum_{i=1}^{N}\delta(v_{j,i})\hat{\boldsymbol{b}}_{\boldsymbol{j,i}}\delta(\frac{||\boldsymbol{y_{j,i}} - \hat{\boldsymbol{y}}_{\boldsymbol{j,i}}||_2}{s_{h,j}} > c_{MPII}) \\
\text{UC} &= \frac{100}{\sum_{j=1}^{M}\sum_{i=1}^{N}\delta(v_{j,i})}\sum_{j=1}^{M}\sum_{i=1}^{N}\delta(v_{j,i})(1 - \hat{\boldsymbol{b}}_{\boldsymbol{j,i}})
\end{aligned} \tag{4.14}$$

Results without corresponding keypoint annotations are not considered in these equations, as it is common practice during evaluation on MPII. One could argue that an evaluation of keypoints without annotations would be useful, as they should be classified as unreliable as even humans failed to annotate them. However, such keypoints suffer from various problems on the MPII dataset. First, missing keypoint annotations are heavily biased. On the validation split used by Newell et al. [86], 54.5% of missing annotations are from ankle keypoints, while 85.3% are either from ankle or knee keypoints. Second, one of the major reasons for missing keypoints (as highlighted by Newell et al.) is that keypoints are outside the image. With respect to practical application, this should not happen as the safety-critical working area should be well-covered by sensors. Furthermore, there can be a most likely position for the cut-off keypoint in the image: the point where the associated body limb leaves the image, especially when the keypoint is cut off closely (e. g., the end of the forearm if the wrist is cut off). Thus, keypoints without annotations will be omitted.

To perform the evaluation, the MPII dataset needs to be split into training, validation, and test data, which will be called train/validation/test split. The official dataset only features a training and test split, with the latter being kept private. Newell et al. [86] used parts of the official training data to form a separate validation split of approximately 3000 samples, which they also employed for evaluation in some of their experiments that were not using the PCKh metric. Their used training and validation split are publicly available[2]. In this work, their validation split will be used as test split for evaluation. Furthermore, 1000 samples are removed from their training split to form a new validation split, with the remainder of their training split serving as the training split of this work.

Throughout the experimental section, two neural networks will be repeatedly used for evaluating the different approaches: a stacked hourglass model (HG) as proposed by Newell et al. [86] as well as a High-Resolution Net (HRNet) as proposed by Sun et al. [117]. A detailed introduction to these neural networks was given in Section 3.1.4, which also highlighted that they are the foundation or a common building block of many advanced works, making them an ideal candidate for experimental evaluations. Regarding their diversity, the architectures share some similarities as both predict heatmaps and process data at different spatial resolutions. However, the HG does that through repeated down- and upsampling, while the HRNet processes data across different resolutions simultaneously and subsequently adds lower resolutions. Thus, a fair share of architectural diversity is present among some similarities. For the hourglass model, a variant with four stacked hourglass blocks and standard feature dimensions will be used, which is called 4-HG. For HRNet, the standard architecture with a feature dimension of 32 in the highest resolution branch will be used and called HRNet-W32. Both will be trained with almost the same training procedure on the training split of MPII. In each case, training will be performed for 200 epochs. From a further 10 epochs, the best weights are determined based on validation split performance (PCKh score). Weights are randomly initialized and the dataset is shuffled in-between epochs. To obtain the input images for the neural networks, the images of the dataset are cropped around individual humans using the dataset's annotated human bounding boxes, as it is common practice for single-person human pose estimation on MPII. Data augmentation during training is performed analogously to Newell et al. [86]: Input images are rotated randomly in-between $-30°$ and $30°$ as well as horizontally flipped with a chance of 50%, and human bounding boxes are randomly scaled before cropping by a factor between 0.75 and 1.25. Image crops are then rescaled to the input size of 256×256 used by both networks. As optimizer, RMSprop [122] is used with a leaning rate of $2.5e^{-4}$ and gradient clipping. A batch size of 8 is used for HG, while 32 is used for HRNet (for stability during training). The training split is shuffled and weights are randomly initialized. Ground truth heatmaps of size 64×64 are used. For the i-th keypoint with annotated position $\boldsymbol{y}_i$, the corresponding heatmap $\boldsymbol{h_i}$ is created by projecting the keypoint's position into the heatmap with a transformation T, yielding $(x_i', y_i') = \boldsymbol{y_i'} = T(\boldsymbol{y}_i)$. Then, a 2D Gaussian with standard deviation σ is applied. The heatmap value at pixel coordinates (u, v) is determined as follows:

$$\boldsymbol{h_i}[u, v] = e^{-(\frac{(x_i' - u)^2}{2\sigma^2} + \frac{(y_i' - v)^2}{2\sigma^2})} \tag{4.15}$$

The mean squared error between the predicted and ground truth heatmaps will be used as loss function. In the case of 4-HG, the loss is also applied to intermediate results (and not only the final heatmap predictions), as Newell et al. [86] use intermediate supervision.

[2] see train.h5/valid.h5 available at `https://github.com/princeton-vl/pose-hg-train`

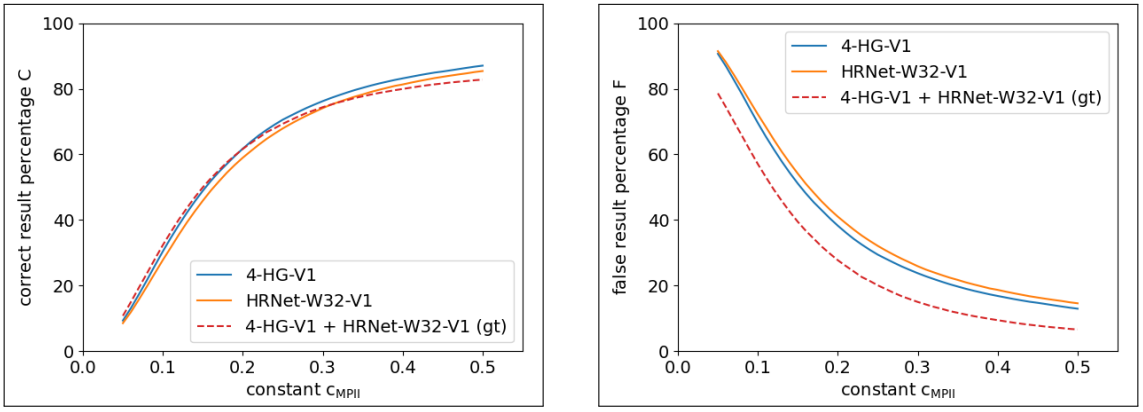
For missing keypoint annotations, two different training approaches will be explored: (i) using empty heatmaps (all zeros) as ground truth and (ii) ignoring the keypoints during training. In case of (ii), all annotated keypoints falling outside an augmented and cropped training image will also be ignored to enforce a distinct peak in every used ground truth heatmap. The suffixes V1 and V2 will be used to indicate whether (i) or (ii) is used. Predicted keypoint positions will be obtained from final predicted heatmaps by backprojecting the position of each heatmap's maximum into the original image (as in Eq. (3.2)).

4.5.1 Proof of Concept: Diverse Neural Network Ensemble

First, a proof of concept will be provided for the core mechanism behind the diverse neural network ensemble: the reliable/unreliable classification performed by $g'_{R,MPII}$ (see Eq. (4.4)). It builds on two major assumptions: (i) individual results of diverse neural networks in case of an incorrect final keypoint prediction are different enough for the distance-based comparison to work, and (ii) the conservative approximation of s_h through $\hat{s}_h$ is possible at inference time through the procedure from Section 4.3.2.

Both assumptions will be assessed, starting with the first one. To show that the distance-based comparison is suitable for the identification of incorrect results, experiments will be performed in isolation of the approximation procedure for $\hat{s}_h$, which is an additional source of errors. Therefore, $\hat{s}_h$ will be replaced with the value it aims to approximate: s_h itself. Furthermore, the constant $c_R = 0.5$ will be used for $g'_{R,MPII}$, as this is the common choice for its equivalent $c_{MPII} = 0.5$ in the PCKh metric [3, 147]. Both 4-HG-V1 and HRNet-W32-V1 will be used for keypoint prediction, serving as the diverse neural networks. Experiments will be performed on the test split with the evaluation metrics introduced in Eq. (4.14). Fig 4.6 shows the experimental results for c_{MPII} values between 0.05 and 0.5. The application of the diverse neural network ensemble significantly decreased the percentage of false detections compared to the individual neural networks without an error detection mechanism. All the while, a high percentage of correct results was retained, even including a minor increase in correct results over the individual networks for small c_{MPII} values. The minor increase in correct results can be explained by incorrect results from individual neural networks contributing to a correct final keypoint prediction of the ensemble (as illustrated in Figure 4.2). The loss of correct results in other cases, e. g., for $c_{MPII} = 0.5$, is also a logical result, as $g'_{R,MPII}$ is expected to mark some results as unreliable that would otherwise have been correct. Looking at the common choice of $c_{MPII} = 0.5$, a notable decrease of almost 50% in false results from 12.9% (lowest of both networks) to 6.6% was achieved, while the percentage of correct results only decreased from 85.4% (lowest of both networks) to 82.8%. These results highlight the capability of the distance-based comparison to identify incorrect results, suggesting that the diverse neural network ensemble could be a valuable tool for error detection.

To further solidify this claim, the second assumption must be verified by showing that s_h can be conservatively approximated through $\hat{s}_h$ at inference time through the procedure from Section 4.3.2 and especially Eq. (4.8). The required constant $\overline{s}_{h,min}$ is calculated as the average over the smallest 5% of annotated s_h values, using only data from the training and validation split to avoid a bias towards the test split. This results in a rounded value of $\overline{s}_{h,min} = 25$ pixels. Adding the subscript j to indicate the j-th input, it will be evaluated how often a conservative approximation $\hat{s}_{h,j} \leq s_{h,j}$ is achieved. The same two networks as previously will be used to supply keypoint predictions for the approximation procedure.



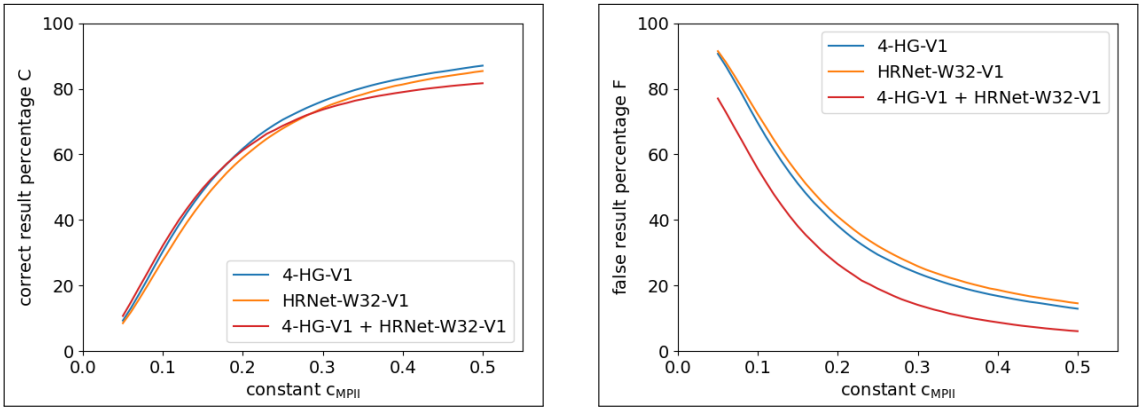
(a) Correct results **(b)** False results

Figure 4.6: Evaluation results on the test split for a diverse neural network ensemble consisting of 4-HG-V1 and HRNet-W32-V1, as well as for the individual networks. In the ensemble, annotated ground truth values $s_{h,j}$ are used instead of $\hat{s}_{h,j}$, indicated through the suffix (gt) in the plots. Furthermore, $c_R = 0.5$ is used for the ensemble.

On the joined training and validation split, 96.71% conservative approximations were achieved, while 95.37% were achieved on the test split. To evaluate whether the non-conservative approximations with $\hat{s}_{h,j} > s_{h,j}$ are far too large or close by, it is evaluated how many values of $\hat{s}_{h,j}$ are still in the vicinity of $s_{h,j}$. This is done by looking at the percentage of results that are smaller or equal to a 50% increase of $s_{h,j}$, thus fulfilling $\hat{s}_{h,j} \leq 1.5 \cdot s_{h,j}$. This was the case for 99.73% of all approximations on the joined training and validation split, and for 99.76% on the test split. In conclusion, a conservative approximation was possible in more than 95% of all cases, while large errors almost never occur ($< 0.3\%$). Thus, the second assumption holds true, hereby validating the diverse neural network approach and clearing the way for more extensive experiments.

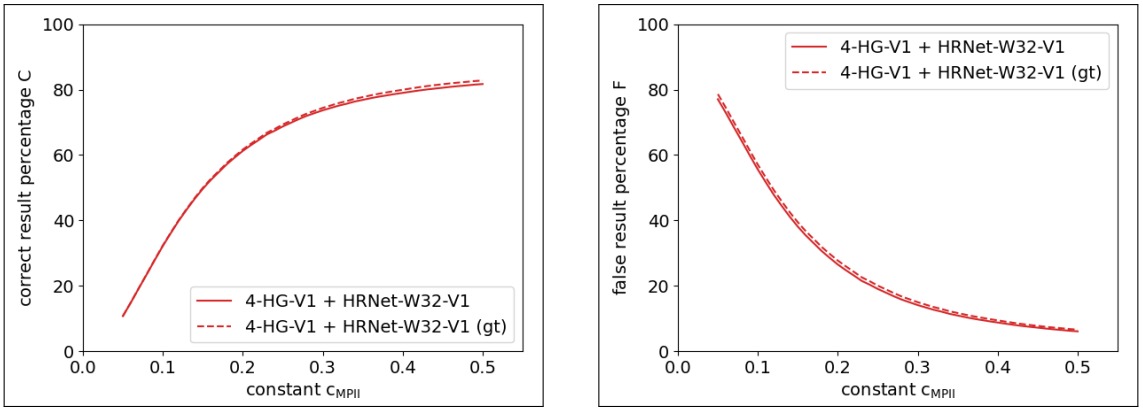
4.5.2 Evaluation of Diverse Neural Network Ensembles

With the strategy for calculating $\hat{s}_{h,j}$ being validated, experiments can be performed to show how well the diverse neural network ensemble performs in its intended form using $\hat{s}_{h,j}$ in $g'_{R,MPII}$ for calculating $\hat{\boldsymbol{b}}_{\boldsymbol{j,i}}$ values. However, $\hat{s}_{h,j} \leq s_{h,j}$ being true in most cases does also mean that $g'_{R,MPII}$ will be more strict in cases where the value is smaller, most likely leading to more unreliable classifications and thus a higher UC percentage, which means less correct and/or false results. In turn, if too many correct results would be lost, then practical usability would be hampered. Figure 4.7 shows the repetition of the proof of concept experiment, but with $\hat{s}_{h,j}$. At $c_{MPII} = 0.5$, correct results were decreased from 85.4% to 81.7%, while false results were decreased from 12.9% to 6.1%. This means that 1.84 false results were eliminated per lost correct result, showing the effectiveness of the diverse neural network ensemble at eliminating previously undetected, incorrect results. Furthermore, Figure 4.8 shows a direct comparison of using approximation $\hat{s}_{h,j}$ vs. annotation $s_{h,j}$ in $g'_{R,MPII}$ for the otherwise same diverse neural network ensemble. The differences are minimal: For $c_{MPII} = 0.5$, the percentage of correct (C) results decreased from 82.8% to 81.7%, while the percentage of false (F) results also decreased from 6.6% to 6.1%. Losses in both metrics are expected behavior when using a conservative approx-



(a) Correct results **(b)** False results

Figure 4.7: Results for the repetition of the experiment from Figure 4.6 using the same ensemble and individual neural networks. The only difference in the experimental setting is that $\hat{s}_{h,j}$ values are used by the ensemble as originally intended.



(a) Correct results **(b)** False results

Figure 4.8: Experimental results comparing the use of approximations $\hat{s}_{h,j}$ (available at inference time) with the use of the ground truth annotations $s_{h,j}$ (suffix (gt)) in a diverse neural network ensemble employing 4-HG-V1 and HRNet-W32-V1. Differences in performance are minimal.

imation. The fact that the loss in correct results is only small further highlights that the conservative approximation $\hat{s}_{h,j}$ is a suitable choice.

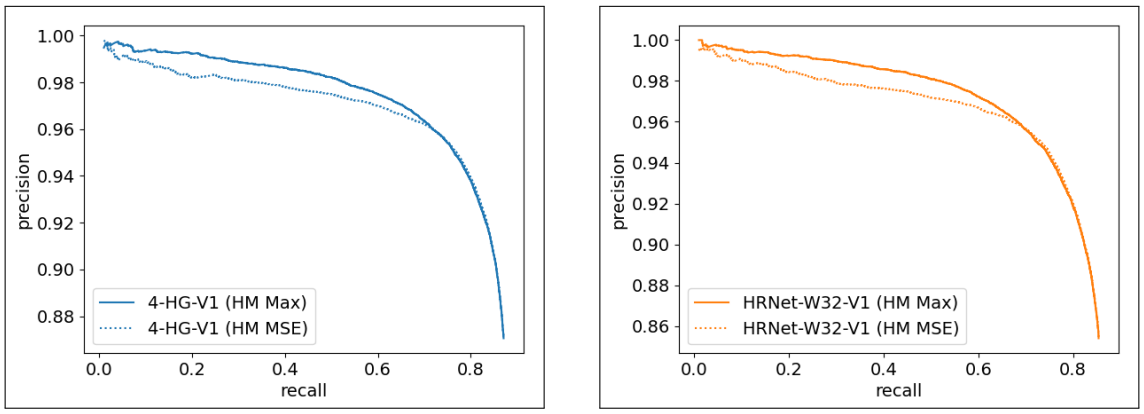
Next, different variants of the diverse neural network ensemble with respect to the involved diverse neural networks are evaluated. To this end, the following neural networks are employed: 4-HG-V1, 4-HG-V2, HRNet-W32-V1, HRNet-W32-V2, as well as the network of Zhang et al. [143], which will be called FPD-HG. On one hand, FPD-HG is highly similar to 4-HG-V1 and 4-HG-V2, as it is an hourglass variant with four hourglass stacks but decreased feature dimensions (256 down to 128). On the other hand, its training is more dissimilar, as the name-giving *Fast Pose Distillation* (FPD) [143] training procedure is employed that aims at a knowledge transfer from a teacher to a student model. Therefore, adding FPD-HG to the mix of neural networks will allow a closer examination of the effect of differences in *training* and *architecture*. For FPD-HG, official experimental results on the test split made available[3] by Zhang et al. [143] will be used. Experiments

[3] see `https://github.com/ilovepose/fast-human-pose-estimation.pytorch`

Method	Head		Shoulder		Elbow		Wrist		Hip		Knee		Ankle		Mean		
	C	F	C	F	C	F	C	F	C	F	C	F	C	F	C	F	UC
4-HG-V1	97.0	3.0	94.7	5.3	87.7	12.3	82.0	18.0	86.2	13.8	80.9	19.1	77.1	22.9	87.1	12.9	0.0
4-HG-V2	96.9	3.1	94.5	5.5	87.8	12.2	81.8	18.2	86.8	13.2	82.0	18.0	78.9	21.1	87.4	12.6	0.0
FPD-HG [143]	97.4	2.6	95.5	4.5	89.0	11.0	84.3	15.7	88.9	11.1	84.1	15.9	80.7	19.3	89.0	11.0	0.0
HRNet-W32-V1	96.7	3.3	93.9	6.1	85.1	14.9	80.4	19.6	84.7	15.3	78.4	21.6	74.7	25.3	85.4	14.6	0.0
HRNet-W32-V2	96.3	3.7	93.8	6.2	85.3	14.7	79.9	20.1	84.0	16.0	79.0	21.0	76.1	23.9	85.4	14.6	0.0
4-HG-V1 + 4-HG-V2	96.2	1.5	93.1	3.2	84.3	6.6	77.1	8.4	83.8	9.4	76.5	9.5	72.3	9.7	83.9	6.7	9.4
4-HG-V1 + FPD-HG	96.3	1.5	93.3	3.0	85.0	6.6	78.5	8.7	85.1	8.4	77.9	9.1	73.4	9.6	84.8	6.5	8.7
4-HG-V2 + FPD-HG	96.5	1.7	93.4	3.3	85.1	6.6	78.7	8.5	85.9	8.4	78.7	8.8	74.5	9.3	85.3	6.5	8.2
HRNet-W32-V1 + HRNet-W32-V2	95.5	1.6	91.5	3.1	80.4	6.6	74.3	8.4	79.8	9.2	72.4	8.6	68.2	9.9	81.0	6.6	12.4
4-HG-V1 + HRNet-W32-V1	95.8	1.5	92.3	2.9	81.2	6.4	75.3	8.1	80.7	8.5	73.0	7.9	68.5	8.4	81.7	6.1	12.2
4-HG-V2 + HRNet-W32-V1	95.9	1.5	91.8	3.0	81.6	6.3	75.0	8.1	81.7	8.2	73.4	7.1	69.0	8.1	81.9	5.9	12.2
FPD-HG + HRNet-W32-V1	96.1	1.5	92.5	3.2	82.9	6.2	77.0	8.2	82.9	7.7	74.6	7.8	70.1	9.0	83.0	6.1	10.9
4-HG-V1 + HRNet-W32-V2	95.7	1.7	92.1	2.9	81.9	6.3	75.2	8.2	80.6	8.6	73.8	7.9	69.4	8.9	81.9	6.2	11.9
4-HG-V2 + HRNet-W32-V2	95.5	1.6	91.9	3.1	81.7	6.0	74.9	7.9	81.6	8.3	74.1	8.0	70.4	9.4	82.1	6.2	11.7
FPD-HG + HRNet-W32-V2	95.9	1.7	92.5	3.0	83.0	6.0	76.7	8.8	82.4	8.4	76.1	8.0	72.1	9.6	83.3	6.3	10.4
4-HG-V1 + 4-HG-V2 + FPD-HG	95.6	1.2	91.7	2.5	82.0	4.7	74.6	5.6	81.7	7.1	73.7	6.2	69.2	6.1	81.9	4.7	13.4
4-HG-V1 + 4-HG-V2 + HRNet-W32-V1	95.1	1.1	90.5	2.2	78.5	4.3	71.7	5.2	77.6	6.5	69.7	5.2	65.3	4.9	79.1	4.2	16.7

Table 4.1: Experimental results on the test split for single neural networks and different diverse neural network ensembles, using the constants $c_{MPII} = 0.5$ and $c_R = 0.5$. Legend: C – percentage of correct detections; F – percentage of false detections; UC – percentage of uncertain results.

are performed with all individual neural networks (which have no method for reliable/unreliable classification), diverse neural network ensembles consisting of all potential pairs of neural networks, as well as selected combinations of three neural networks into one ensemble. Table 4.1 shows the results obtained from these experiments. It can be seen that all explored variants of the diverse neural network ensemble were able to significantly reduce the number of false results compared to the individual neural networks, while only using data available at inference time. Compared to the best individual neural network which is FPD-HG with only 11.0% false results, the worst diverse neural network ensemble consisting of 4-HG-V1 and 4-HG-V2 exposed only 6.7% false results, while the best of the examined diverse neural network ensembles consisting of 4-HG-V1, 4-HG-V2 and HRNet-W32-V1 reduced that number to 4.2%. Regarding the importance of diverse training and diverse architecture, the experiments showed that diversity induced through differences in training (randomization and other factors) is of high importance: Although 4-HG-V1 and 4-HG-V2 have the same architecture, the diverse neural network ensemble achieved an error reduction from 12.6% to 6.7%. Similar results were achieved when combining HRNet-W32-V1 with HRNet-W32-V2 in a diverse neural network ensemble. However, experimental results also suggest some benefits from architectural diversity: Diverse neural network ensembles consisting of two neural networks with the same or extremely similar architectures exposed between 6.5% and 6.7% false results, while only 5.9% to 6.3% false results for diverse neural network ensembles consisting of two neural networks with diverse architectures were observed. The two experiments with diverse neural network ensembles consisting of three diverse neural networks further highlight this effect: FPD-HG on its own performs significantly better than HRNet-W32-V1, with 89.0% correct and 11.0% false results compared to 85.4% correct and 14.6% false results. However, adding FPD-HG to a diverse neural network ensemble including 4-HG-V1 and 4-HG-V2 resulted in 4.7% false results, while adding HRNet-W32-V1 to the same ensemble instead resulted in 4.2% false results, despite the fact that HRNet-W32-V1 had weaker performance in the first place. Regarding the retention of correct results, all diverse neural network ensembles except for one achieved above 80% correct results, with the single outlier being close to that threshold with 79.1% correct results.



(a) Comparison on 4-HG-V1 **(b)** Comparison on HRNet-W32-V1

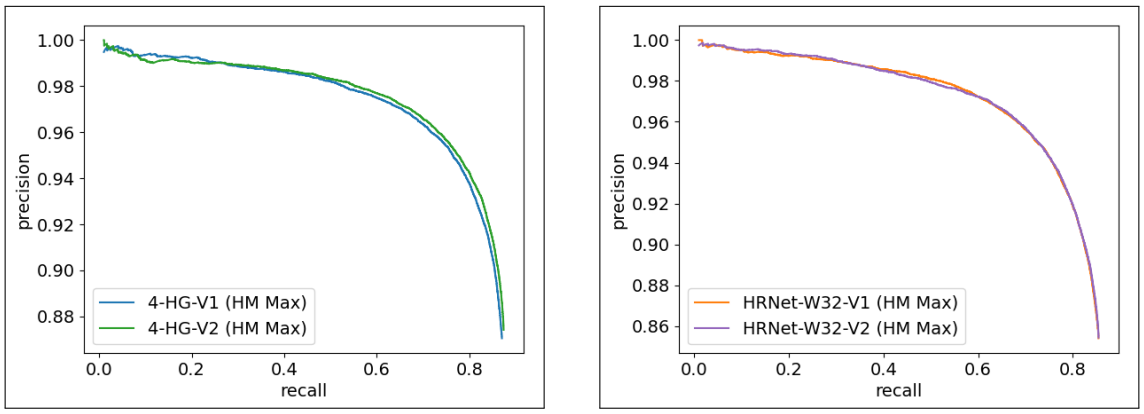
Figure 4.9: PRC curves comparing the effectiveness of the methods based on the heatmap maximum (HM Max) and MSE (HM MSE) for identifying incorrect results.

4.5.3 Evaluation of Heatmap-Based Approaches

Next, the evaluation of both heatmap-based approaches for error detection will be performed on the MPII Human Pose dataset. The previously introduced neural networks 4-HG-V1, 4-HG-V2, HRNet-W32-V1 and HRNet-W32-V2 will be employed to produce the respective heatmaps. As no specific way for calculating the necessary thresholds c_{max} and c_{mse} was introduced, the heatmap-based methods will be evaluated using the PRC curve to consider all their potential values. For plotting the PRC curve, precision and recall as introduced in Eq. (3.10) need to be calculated for different thresholds, which requires the number of true positives (#TP), false positives (#FP) and false negatives (#FN). Based on Eq. (4.1) and variable $v_{j,i}$ that indicates whether keypoint detection $\hat{\boldsymbol{y}}_{\boldsymbol{j},\boldsymbol{i}}$ has an annotated ground truth ($True$) or not, these amounts can be calculated as follows:

$$\begin{aligned} \#\text{TP} &= |\{\hat{\boldsymbol{y}}_{\boldsymbol{j},\boldsymbol{i}} | (\hat{\boldsymbol{y}}_{\boldsymbol{j},\boldsymbol{i}} \text{ is Correct}) \text{ and } (v_{j,i} = True)\}| \\ \#\text{FP} &= |\{\hat{\boldsymbol{y}}_{\boldsymbol{j},\boldsymbol{i}} | (\hat{\boldsymbol{y}}_{\boldsymbol{j},\boldsymbol{i}} \text{ is False}) \text{ and } (v_{j,i} = True)\}| \\ \#\text{FN} &= |\{v_{j,i} | v_{j,i} = True\}| - \#\text{TP} \end{aligned} \tag{4.16}$$

In simple terms, #TP is the number of predicted keypoints with annotations in the Correct category, #FP is the number of predicted keypoints with annotations in the False category, and #FN is the number of predicted keypoints with annotations that are not in the Correct category. Definition over $\hat{\boldsymbol{y}}_{\boldsymbol{j},\boldsymbol{i}}$ in the above formula is possible, as every annotated keypoint $\boldsymbol{y}_{\boldsymbol{j},\boldsymbol{i}}$ has only a single prediction $\hat{\boldsymbol{y}}_{\boldsymbol{j},\boldsymbol{i}}$ by the design of the 2D single-person human pose estimation problem. Using this definition, the PRC curve is calculated for both proposed methods, using $c_{MPII} = 0.5$ while exploring different potential thresholds for c_{max} and c_{mse} in the respective reliable/unreliable classifications (see Eq. (4.9) and (4.12)) on the test split. As these thresholds only govern how many results will be filtered out, the maximum achievable recall will not be 1.0, but will be limited to a smaller value due to $c_{MPII} = 0.5$ being fixed (the other factor that governs whether a result belongs to the Correct category). Experimental results are displayed in Figure 4.9 for 4-HG-V1 and HRnet-W32-V1. Looking at the right side of both curves, the precision rapidly improves for minor reductions in recall. This highlights the capability of both methods to reduce false results effectively. In both cases, the method building upon the heatmap maximum for thresholding outperforms the MSE-based method.



(a) Comparison on 4-HG

(b) Comparison on HRNet-W32-V1

Figure 4.10: PRC curves comparing the effect of training procedures V1 and V2 on the method using the heatmap maximum for incorrect result identification.

So far, all experiments with heatmaps have been performed on the V1 variant of the neural networks, where missing keypoint detections are incorporated during training in the form of all-zero heatmaps. For V2, these are excluded completely, with all ground truth heatmaps used for training having a distinctive peak. In theory, this could lead to the maximum heatmap value being less expressive, as the neural network is not explicitly trained to predict all-zero heatmaps in case of high uncertainty, which could impact the heatmap-based methods. Further experiments are conducted to find out if a notable impact exists, using the V1 and V2 variants of both neural networks for the previously superior method building on the heatmap maximum. Figure 4.10 shows the resulting PRC-curves. Minor differences exist for 4-HG-V1 and 4-HG-V2, while almost no differences are present for HRNet-W32-V1 and HRNet-W32-V2. Overall, it can be concluded that the choice between V1 and V2 has no major impact on the examined heatmap-based approach.

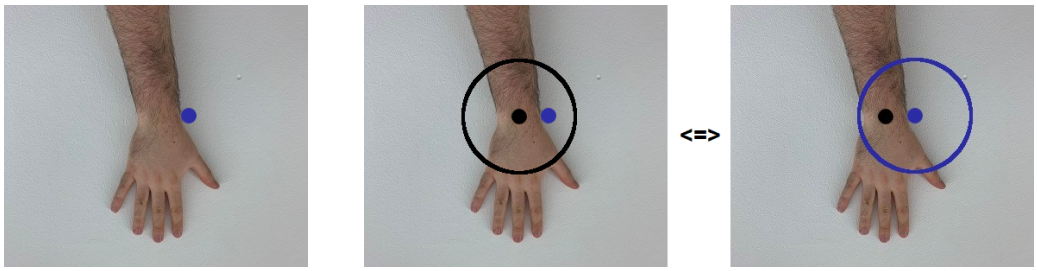
Last, a comparison between the diverse neural network ensemble and the best heatmap-based approach will be performed. For this comparison, a suitable, fixed value for c_{max} has to be selected. For a fair comparison of the capability to reduce false results, the diverse neural network ensemble will be evaluated first. Subsequently, c_{max} will be chosen for the heatmap-based method such that the percentage of correct results is the same. Hence the difference will be in the percentage of false results. An experiment on the test split will be performed using 4-HG-V1, HRNet-W32-V1, and a diverse neural network ensemble using both networks. The diverse neural network ensemble achieves a total of 81.7% correct results while still having 6.1% false results (as in Table 4.1). Using the heatmap-based approach that builds on the heatmap maximum and choosing c_{max} such that 81.7% correct results are achieved yields 6.2% false results for 4-HG-V1 and 8.4% false results for HRNet-W32-V1. Hence the diverse neural network ensemble has very similar performance to thresholding over the heatmap maximum when using 4-HG-V1, however, without reliance on an intermediate output like the heatmap.

In conclusion, all methods have shown to be valuable for reducing false keypoint detections. Thresholding over the maximum value of heatmaps has shown to be the best of the examined heatmap-based approach. The method showed comparable performance to the diverse neural network ensemble in direct comparison, with the latter being more broadly applicable (no heatmaps required). However, none of the investigated methods was able to fully eliminate false results while retaining a high level of correct results.

5 Measurement Error Estimation

In this chapter, the second of the four central points of this work will be discussed: *How can an upper bound for the distance between a keypoint detection and the actual keypoint location be determined?* In the previous chapter, it has already been discussed that a perfect localization of keypoint positions is unrealistic. This means that a certain degree of deviation between predicted and actual keypoint position will typically be present, which constitutes a measurement error. The presence of measurement errors is not prohibitive for the use in safety-critical applications like SSM, however, the presence of measurement errors needs to be factored in. Taking the protective separation distance definition of $S_p(t_0)$ (see Eq. (2.2)) from ISO/TS 15066 [51] as an example, the worst case measurement error has to be factored in through the additional surcharge Z_d. This means it is assumed that the human is Z_d closer to the robot than his measured position suggests. This makes it either necessary to have a global worst-case value for the measurement error or to obtain individual measurement-specific worst-case values that accompany each measurement. For the use of human pose estimation in safety-critical applications, this means that the worst-case measurement error of predicted keypoint positions must be available. Potential *upper bounds for the measurement error* specific to individual keypoint predictions conflict with the standard treatment of 2D single-person human pose estimation as a point estimation problem: the keypoint positions that are required as only output to solve the problem are not indicative of the measurement error. Alternatively, aiming for a global upper bound is also problematic, as large measurement errors are a common sight in human pose estimation [103]. Looking at the previous section, one could argue that $c_{MPII} \cdot s_{h,j}$ could be a suitable upper bound for the measurement error (specific to the j-th input) for correct keypoint predictions. However, $s_{h,j}$ is human annotated and not available at inference time, while $\hat{s}_{h,j}$ is no suitable substitute for an upper bound due to being more conservative (smaller). Overall, this highlights the need for methods to obtain upper bounds for the measurement error of predicted keypoints at inference time.

Again, investigations throughout this section will be limited to the fundamental task of 2D single-person human pose estimation. The primary goal is to find methods that produce high-quality upper bounds for the measurement error of keypoint predictions. Upper bounds that become very large are not useful for practical application, as they simply indicate that the keypoint could be anywhere. Hence a high-quality upper bound fits the true localization error closely. For practical application, it is also worth to revisit the correctness definition of keypoints in 2D single-person human pose estimation. In its current form, it lacks information at inference time about where the actual keypoint position could be relative to the predicted position. In the following, the formal problem definition and a new definition of keypoint correctness will be introduced first. Next, potential approaches to solve the problem of upper bound prediction will be discussed and specific methods will be proposed. Last, the proposed methods will be evaluated through experiments, assessing if they are able to predict meaningfully upper bounds for the measurement error of keypoint predictions at inference time. The contents of this chapter are primarily based upon previously published work of the author [109].



(a) No context **(b)** Correct prediction, equivalent interpretations

Figure 5.1: Illustration of what is known about the true location of a keypoint (black dot). No conclusion can be drawn about the true keypoint location at inference time, if only the predicted keypoint location (blue dot) is available without context as in (a). With an upper bound $\boldsymbol{r}$ for the measurement error, it is possible to define keypoint correctness as inclusion by a 1D sphere with radius $\boldsymbol{r}$, which can either be drawn around the true keypoint location ((b), left) or the predicted keypoint location ((b), right). If $\boldsymbol{r}$ is available at inference time, the latter defines the potential true keypoint locations if correct.

5.1 Problem Definition and Keypoint Correctness

To obtain a more detailed problem definition, the conceptual problem with results for the common 2D single-person human pose estimation problem will be discussed in greater detail first. As already highlighted, the common 2D single-person human pose estimation problem is solved by predicting a predefined set of keypoint positions $\hat{\boldsymbol{y}}$ only. For the i-th keypoint, this position is $(\hat{x}_i, \hat{y}_i) = \hat{\boldsymbol{y}}_{\boldsymbol{i}}$. No other outputs are required. Without further context, a predicted position holds no information about the true location of the keypoint and simply represents the position where the neural network believes the keypoint to be. Figure 5.1a illustrates this for the wrist keypoint. Especially, no upper bound for the measurement error of prediction $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ is available, which is crucial for safety applications, e. g., for the calculation of the protective separation distance as in Eq. (2.2). If context is added by assuming that the human pose estimator was evaluated on a representative dataset and that $X\%$ correct detections were achieved for the i-th keypoint with one of the common evaluation metrics from Section 3.1.3, it is still not possible to make a definite statement about $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$, except that it has a chance of $X\%$ to be correct. When the strong assumption is made that no incorrect results for the i-th keypoint exist (the ultimate goal of the previous section) while using the PCKh score (Eq. (4.13)) from MPII as exemplary evaluation metric, then it is possible to assume that a correct $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ lies within a 1D sphere (circle) with radius $c_{MPII} \cdot s_h$ around the actual keypoint position $\boldsymbol{y}_{\boldsymbol{i}}$. This can also be formulated the other way around, that $\boldsymbol{y}_{\boldsymbol{i}}$ lies within a 1D sphere with radius $c_{MPII} \cdot s_h$ around $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$, which would constitute an upper bound for the measurement error. See Figure 5.1b for an illustration. However, this upper bound has to be available at inference time to be applicable when the human pose estimator is deployed, which is not the case due to s_h being a human annotation. Thus, even such a strong assumption is not sufficient to obtain an upper bound for the measurement error of keypoint predictions with a common definition of keypoint correctness. This highlights the need for additional methods to obtain upper bounds for the measurement error at inference time, as well as the need to reformulation keypoint correctness to better suit the needs of safety applications.

A *reformulation of keypoint correctness* should stay close to existing definitions, while also resolving their shortcoming regarding the needs of safety applications. All metrics

from Section 3.1.3 that separate predictions into correct and incorrect do so by thresholding the Euclidean distance between prediction and annotation using thresholds obtained from human annotations. For safety, such thresholds must be available at inference time if they shall serve as upper bounds for the measurement error, therefore they shall be predicted as well. Let $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ and $\boldsymbol{y}_{\boldsymbol{i}}$ denote the predicted and annotated keypoint position of the i-th keypoint respectively, and $\hat{\boldsymbol{r}}_{\boldsymbol{i}}$ the predicted threshold. Then, function g_{rad} shall define the correctness of the predicted pair $(\hat{\boldsymbol{y}}_{\boldsymbol{i}}, \hat{\boldsymbol{r}}_{\boldsymbol{i}})$ as follows:

$$g_{rad}(\hat{\boldsymbol{y}}_{\boldsymbol{i}}, \boldsymbol{y}_{\boldsymbol{i}}, \hat{\boldsymbol{r}}_{\boldsymbol{i}}) = \delta(||\hat{\boldsymbol{y}}_{\boldsymbol{i}} - \boldsymbol{y}_{\boldsymbol{i}}||_2 \leq \hat{\boldsymbol{r}}_{\boldsymbol{i}}) \tag{5.1}$$

Predicting both $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ and $\hat{\boldsymbol{r}}_{\boldsymbol{i}}$ ensures that the threshold is available at inference time. For a correct prediction, $\hat{\boldsymbol{r}}_{\boldsymbol{i}}$ constitutes an upper bound for the measurement error and can be used together with $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ as seen in Figure 5.1b to indicate all potential true keypoint locations through a 1D sphere. However, building upon Eq. (5.1) alone has two shortcomings: There exists a trivial solution with $\hat{\boldsymbol{r}}_{\boldsymbol{i}} \to \infty$ that leads to meaningless upper bounds, and spatial vicinity between $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ of $\boldsymbol{y}_{\boldsymbol{i}}$ – the primary goal of metrics from Section 3.1.3 – is not explicitly required as a large enough $\hat{\boldsymbol{r}}_{\boldsymbol{i}}$ can counteract any degree of deviation. To address these shortcomings, *two additional quality metrics* will be introduced. A good prediction of $\hat{\boldsymbol{r}}_{\boldsymbol{i}}$ should be close to the actual deviation between $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ and $\boldsymbol{y}_{\boldsymbol{i}}$ and not unnecessarily large, which shall be assessed through quality metric Q_{rad} as follows:

$$Q_{rad}(\hat{\boldsymbol{y}}_{\boldsymbol{i}}, \boldsymbol{y}_{\boldsymbol{i}}, \hat{\boldsymbol{r}}_{\boldsymbol{i}}) = |\hat{\boldsymbol{r}}_{\boldsymbol{i}} - ||\hat{\boldsymbol{y}}_{\boldsymbol{i}} - \boldsymbol{y}_{\boldsymbol{i}}||_2| \tag{5.2}$$

Second, a good predicted keypoint position $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ should be close to the annotated keypoint position $\boldsymbol{y}_{\boldsymbol{i}}$, which will be assessed through quality metric Q_{pos} as follows:

$$Q_{pos}(\hat{\boldsymbol{y}}_{\boldsymbol{i}}, \boldsymbol{y}_{\boldsymbol{i}}) = ||\hat{\boldsymbol{y}}_{\boldsymbol{i}} - \boldsymbol{y}_{\boldsymbol{i}}||_2 \tag{5.3}$$

$Q_{rad}(\hat{\boldsymbol{y}}_{\boldsymbol{i}}, \boldsymbol{y}_{\boldsymbol{i}}, \hat{\boldsymbol{r}}_{\boldsymbol{i}}) \to 0$ as well as $Q_{pos}(\hat{\boldsymbol{y}}_{\boldsymbol{i}}, \boldsymbol{y}_{\boldsymbol{i}}) \to 0$ is desired for good quality. However, it can also be necessary to violate one or both quality metrics by an increased amount to achieve correctness with respect to g_{rad}, e. g., in case of strong occlusion where high positional uncertainty in the input data exists and all potential keypoint positions should be covered by $(\hat{\boldsymbol{y}}_{\boldsymbol{i}}, \hat{\boldsymbol{r}}_{\boldsymbol{i}})$. In the following, it will be the goal to find methods to predict values for $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ and $\hat{\boldsymbol{r}}_{\boldsymbol{i}}$ that achieve a high percentage of correct detections with respect to Eq. (5.1) while having decent quality.

5.2 Discussion and Selection of Approaches

To solve the 2D single-person human pose estimation problem under the reformulated correctness definition from Eq. (5.1) with good quality under the respective metrics Q_{rad} and Q_{pos}, two general ideas come to mind: The first idea is to add a method that over-approximates the human-annotated threshold used in traditional correctness definitions at inference time, as the reliance on human annotations prevents the use of this value at inference time in the first place as highlighted in Section 5.1. In case of MPII, this would require the prediction of an approximation $\hat{s}'_{h,j}$ for each input j, such that $c_{MPII} \cdot s_{h,j} \leq c_{MPII} \cdot \hat{s}'_{h,j} = \hat{\boldsymbol{r}}_{\boldsymbol{j,i}}$ for all keypoints i. The second idea is to directly predict individual values $\hat{\boldsymbol{r}}_{\boldsymbol{j,i}}$ for every keypoint i and input j. This could e. g., be done through jointly predicting $\hat{\boldsymbol{y}}_{\boldsymbol{j,i}}$ and $\hat{\boldsymbol{r}}_{\boldsymbol{j,i}}$ with a single neural network. In the following, both ideas will be discussed in greater detail and a selection of promising approaches will be made.

(a) Grasp bottom **(b)** Grasp front **(c)** Ambiguity

Figure 5.2: Illustration of ambiguity under occlusion. Images (a) and (b) are different: In (a), the left hand supports the box from the bottom, in (b) the box is pulled towards the body by grasping around the front of the box with the fingers of the left hand. The position of the corresponding left wrist keypoint is marked in both images (cyan dot), however, from image information alone the cases can not be distinguished. In (c), this ambiguity is highlighted, further showing that a 1D sphere with radius $c_{MPII} \cdot s_h$ and $c_{MPII} = 0.5$ (cyan circle) is unable to sufficiently capture the ambiguity.

Overapproximation of Evaluation Thresholds: Section 4.5.1 has shown that it is possible to obtain a suitable conservative approximation $\hat{s}_h$ of s_h in most cases. Therefore, an obvious idea would be to adjust the methodology there to calculate an over-approximation $\hat{s}'_h$ instead with the goal of $\hat{s}'_{h,j} \geq s_{h,j}$ for every input j. Then, $\hat{\boldsymbol{r}}_{\boldsymbol{j,i}} = c_{MPII} \cdot \hat{s}'_{h,j}$ could be used as threshold for every keypoint i of the j-th input. However, this idea suffers from two conceptual shortcomings. The first shortcoming is that the approximation procedure used for $\hat{s}_h$ naturally favors conservative approximation and not over-approximation. This is especially the case due to the foreshortening of body parts through the projection into 2D. While l_S and l_H are used in conjunction to limit the effect, both can be foreshortened to some degree at the same time, which leads to smaller approximation values. This effect is tolerable for a conservative approximation, but not an over-approximation. For both, the maximum degree of foreshortening before the other one takes over would have to be determined, and both values would always have to be increased respectively, e. g., by a certain percentage. Furthermore, a value $\overline{s}_{h,max}$ from the largest $s_{h,j}$ values on the dataset would have to be calculated as fallback value and counterpart of $\overline{s}_{h,min}$ for cases where l_S and l_H can not be calculated. This would most likely lead to frequent massive over-approximations of $s_{h,j}$, e. g., in cases with little to no foreshortening, which would mean poor quality regarding Q_{rad}. The same is true if $\hat{\boldsymbol{r}}_{\boldsymbol{j,i}} = c_{MPII} \cdot \overline{s}_{h,max}$ would be used as a single, global threshold. The second shortcoming is that using $s_{h,j}$ (or an approximation thereof) is also not ideal from a theoretical perspective. This value, derived from an annotated head bounding box, is neither expressive of the neural network's own positional uncertainty in its predicted position, nor is it expressive of the positional uncertainty inherent to a given input image, e. g., due to occlusion. The latter can even prevent that correct predictions are reliably produced when the positional uncertainty in an input image j exceeds $2 \cdot c_{MPII} \cdot s_{h,j}$, meaning that all potential positions for a keypoint can no longer be covered by 1D sphere with radius $s_{h,j}$. Figure 5.2 illustrates this problem.

Upper Bound Prediction: Values for $\hat{\boldsymbol{r}}_{j,i}$ could also be obtained by aiming at the prediction of upper bounds for the measurement error instead of approximating dataset thresholds. Two general strategies for obtaining upper bounds for the measurement error are: (i) obtaining them as a slight increase of the currently assumed measurement error, and (ii) obtaining them directly without considering the current measurement error itself. From the first strategy, good quality Q_{rad} can be expected when the upper bound is predicted as a slightly larger version of the current measurement error. However, this also means that smaller errors in the prediction process can already lead to incorrect results when the upper bound becomes smaller than the measurement error, in turn decreasing the percentage of correct detections regarding g_{rad}. Good quality regarding Q_{pos} could be ensured by e. g., extending an existing human pose estimation method that already aims at minimizing the distance between predicted and annotated keypoint positions. In practice, this could be realized by extending an existing neural network architecture through an additional branch or prediction head that predicts the current measurement error or a slightly increased version thereof to serve as the foundation for an upper bound calculation. An estimate of the measurement error could also be obtained from existing outputs of some networks like heatmaps. In the case of heatmaps, scores are already indicative of the assumed deviation between each pixel's position and the true keypoint location.

The second strategy of predicting an upper bound for the measurement error without directly incorporating a prediction of the measurement error itself could be based upon a change of the human pose estimation problem from point prediction to distribution prediction. This idea is supported by the fact that the prediction of distributions is already established in some areas, e. g., in uncertainty estimation, where Lakshminarayanan et al. [65] predicted a distribution's mean and variance through individual networks and refined the prediction with an ensemble of neural networks, using the resulting variance as uncertainty measure. Directly aimed at human pose estimation, Kreiss et al. [63] predicted parameters of a Laplace distribution as part of the loss function during training, while Gundavarapu et al. [40] predicted the covariance matrix of a multivariate Gaussian to obtain uncertainty measures of individual keypoint predictions. More specific to the problem at hand, Bertoni et al. [6] obtained confidence intervals for the 3D location of humans based on 2D keypoint predictions from a single image. This is done based on the prediction of parameters for Laplace distributions. In retrospect, this idea is now also supported by the work of Bramlage et al. [7] (simultaneously published to the author's work [109] on the topic), who predict uncertainty based on the prediction of the parameters of a Gaussian distribution combined together with Monte-Carlo dropout. Furthermore, confidence intervals were calculated from the resulting standard deviation, scaled by a learned calibration function to incorporate a certain percentage of results. Overall, these examples show that predicting distributions is useful for various tasks where uncertainty is involved, including tasks related to human pose estimation. One way to obtain upper bounds for the measurement error from predicted distributions would be to predict the parameters of Gaussian distributions and then apply the well-established 3-sigma rule to obtain the upper bounds $\hat{\boldsymbol{r}}_{j,i}$, which should encapsulate almost all measurement errors. For such an approach, a high number of correct results regarding g_{rad} can be expected, however, the resulting upper bounds are expected to become larger, thus achieving less quality Q_{rad}. The position quality Q_{pos} depends on how the position will be inferred.

Overall, approaches that aim at the over-approximation of existing evaluation thresholds do not seem very promising due to the aforementioned shortcomings. In contrast, the methods for predicting upper bounds look more promising: When the prediction of the

keypoint position and upper bound for the measurement error is realized by the same neural network, the network can express its positional uncertainty. A methodology for *obtaining upper bounds from heatmaps* is worth an investigation as their use is widespread. Methods that *additionally predict upper bounds*, be it directly based on the measurement error or not, are both worth an investigation due to the different properties that can be expected from them (better Q_{rad} vs. higher rate of correct predictions). In the following, concrete approaches for these strategies will be introduced and examined.

5.3 Heatmap-Based Measurement Error Prediction

The first proposed approach that will be introduced is the inference of upper bounds for the measurement error directly from predicted heatmaps $\hat{\boldsymbol{h}}$. Building the measurement error estimation upon heatmaps has the advantage that no architectural changes have to be made to the broad range of heatmap-based neural networks that already exist for the problem of 2D single-person human pose estimation. This opens up the question of how the measurement error or an upper bound for it can be inferred from a heatmap. To answer this question, the generation of ground truth heatmaps – the target a heatmap-based neural network is trained to predict – will be examined more closely. The typical procedure is the application of a 2D Gaussian, which comes down to the application of a formula like Eq. (4.15) at each pixel location (u, v). A more general form of this equation for the i-th keypoint under further consideration of a scaling factor $a > 0$, and with $(x'_i, y'_i) = \boldsymbol{y'_i}$ denoting the ground truth position of the i-keypoint in the heatmap would be:

$$\boldsymbol{h_i}[u, v] = a \cdot e^{-(\frac{(x'_i - u)^2}{2\sigma^2} + \frac{(y'_i - v)^2}{2\sigma^2})} \tag{5.4}$$

This equation clearly shows that the value of the heatmap $\boldsymbol{h}_i$ at each pixel position (u, v) depends on the individual distance of the ground truth keypoint at heatmap resolution to the pixel in both dimensions. Through several steps, the equation can be reformulated as follows:

$$\begin{aligned} \boldsymbol{h_i}[u, v] &= a \cdot e^{-(\frac{(x'_i - u)^2}{2\sigma^2} + \frac{(y'_i - v)^2}{2\sigma^2})} \\ \Leftrightarrow \; ln(\frac{\boldsymbol{h_i}[u, v]}{a}) &= -(\frac{(x'_i - u)^2}{2\sigma^2} + \frac{(y'_i - v)^2}{2\sigma^2}) \\ \Leftrightarrow \; -2\sigma^2 \cdot ln(\frac{\boldsymbol{h_i}[u, v]}{a}) &= (x'_i - u)^2 + (y'_i - v)^2 \end{aligned} \tag{5.5}$$

After the last transformation, the potential to obtain an equation for the Euclidean distance and thus the measurement error by simply applying the square root on both sides becomes obvious. For the square root to be applicable, the left side must be 0 or positive, which comes down to the natural logarithm being negative or zero (as $\sigma^2 > 0$). For the natural logarithm to be negative or zero, it must be applied to a value of the half-open interval $(0, 1]$. This is here the case, as $\boldsymbol{h_i}[u, v]$ can take values in the half-open interval $(0, a]$ by definition of Eq. (5.4), therefore the division by a leads to a value inside $(0, 1]$. This allows the application of the square root, leading to the following formula:

$$\sqrt{-2\sigma^2 \cdot ln(\frac{\boldsymbol{h_i}[u, v]}{a})} = \sqrt{(x'_i - u)^2 + (y'_i - v)^2} \tag{5.6}$$

This means that the Euclidean distance between an annotated keypoint position (x'_i, y'_i) at heatmap resolution and the pixel coordinates (u, v) can be directly calculated from the ground truth heatmap $\boldsymbol{h_i}$, as both a and σ are known, constant values.

At inference time, the ground truth heatmap $\boldsymbol{h_i}$ is not available, but only the predicted heatmap $\hat{\boldsymbol{h}}_i$. As heatmap-based neural networks are trained to produce predictions $\hat{\boldsymbol{h}}_i$ that equal $\boldsymbol{h_i}$, Eq. (5.6) will be applied to $\hat{\boldsymbol{h}}_i$. For this application, caution has to be taken. First, it has to be enforced that $0 < \hat{\boldsymbol{h}}_i[\boldsymbol{u}, \boldsymbol{v}] \leq a$ holds true for all pixel positions (u, v) of the heatmap. This might not be the case, as some neural networks do not enforce a value range for their produced heatmaps (as e. g., Newell et al. [86]). This makes post-processing necessary to enforce that heatmap values fall within $(0, a]$, which can e. g., be done by replacing values that are ≤ 0 with a very small $\epsilon > 0$ and values $> a$ with a itself. Second, even if $\hat{\boldsymbol{h}}_i$ is a good approximation of $\boldsymbol{h_i}$, some variation in heatmap values can be expected. This in turn impacts the calculated distances based on values from $\hat{\boldsymbol{h}}_i$. Hence, a constant b should be added to move the predicted distance values into the direction of an upper bound for the measurement error.

As final prediction for the i-th keypoint, the keypoint position $\hat{\boldsymbol{y}}_i$ will be obtained as the position of the maximum in the predicted heatmap $\hat{\boldsymbol{h}}_i$, backprojected into the original image through a transformation T^{-1}. To obtain $\hat{\boldsymbol{r}}_i$, Eq. (5.6) will be applied to the maximum value of the predicted heatmap. Then, a positive constant b is added and the resulting heatmap-related upper bound $\hat{\boldsymbol{r}}'_i$ is backprojected too. This is formalized as follows:

$$\begin{aligned} \hat{\boldsymbol{y}}'_i &= (\hat{x}'_i, \hat{y}'_i) = \underset{u,v}{\text{argmax}}(\hat{\boldsymbol{h}}_i[u, v]) \\ \hat{\boldsymbol{r}}'_i &= b + \sqrt{-2\sigma^2 \cdot ln(\frac{\hat{\boldsymbol{h}}_i[\hat{x}'_i, \hat{y}'_i]}{a})} \\ \hat{\boldsymbol{y}}_i &= T^{-1}(\hat{\boldsymbol{y}}'_i), \;\; \hat{\boldsymbol{r}}_i = T^{-1}(\hat{\boldsymbol{r}}'_i) \end{aligned} \tag{5.7}$$

Whenever $\hat{\boldsymbol{h}}_i$ is a good approximation of $\boldsymbol{h_i}$, this equation should result in a suitable upper bound $\hat{\boldsymbol{r}}_i$. This opens up the question of what will happen if $\hat{\boldsymbol{h}}_i$ is not a good approximation. One example of a bad approximation $\hat{\boldsymbol{h}}_i$ would be that the maximum of $\hat{\boldsymbol{h}}_i$ is far away from the maximum in $\boldsymbol{h_i}$, thus completely different keypoint positions would be indicated by both heatmaps. For keypoint predictions far away from the ground truth, experiments performed in Section 4.5.3 showed that the maximum values of $\hat{\boldsymbol{h}}_i$ are typically smaller than in cases where the keypoint prediction is close by, as thresholding over the maximum heatmap value showed to be an effective method for identifying incorrect results. In turn, smaller values for the heatmap maximum mean that larger upper bounds will be predicted through Eq. (5.7), which is the desired behavior in case of a larger distance from prediction to ground truth. Whether this is sufficient for the prediction of $\hat{\boldsymbol{r}}_i$ values such that $(\hat{\boldsymbol{y}}_i, \hat{\boldsymbol{r}}_i)$ pairs are correct in these cases will be assessed through experiments.

To realize this heatmap-based prediction strategy, a 4-stack hourglass model as in Section 4.5 will be used. The simple choice of $a = 1.0$ does not only simplify the previous equations, but also opens up the use of the sigmoid activation function to enforce that each heatmap value falls into the interval $(0, a = 1.0]$. Thus, instead of no activation function, the sigmoid activation function will be used to predict the final heatmaps in the 4-stack hourglass model. It will be called 1H-HG in the following. Training can still be performed with the mean squared error (see Eq. (3.3)) between predicted and ground truth heatmaps as loss function and will be called l_{MSE} here. The overall approach of this section will be called 1H-HG$_{\text{error}}$, as it uses the 1-HG model and is based on the measurement error.

5.4 Direct Measurement Error Prediction

The apparent shortcoming of using heatmap values to obtain upper bounds for the measurement error is that heatmaps were not designed for that task, but for the localization of keypoints instead. In the previous approach, they were used for both. In turn, this means that any adjustments to them that aim to improve the prediction of upper bounds would inevitably affect the localization of keypoints as well, and vice versa. Such a dependency should be omitted for better control over the prediction of positions and upper bounds. This could be achieved by having separate neural network outputs for each of the tasks, while the relation between the predicted position and upper bound for the measurement error can still be modeled, e. g., through the loss.

To obtain multiple outputs for different (sub)tasks from the same neural network, the established strategy of using multiple prediction heads can be employed. In this strategy, multiple short branches called prediction heads are added near the end of a neural network, each producing its own output. This approach was for example employed by Shi et al. [111], who use one prediction head to predict confidence scores and another one to predict keypoint positions, or Kreiss et al. [63], who use one head to predict Part Intensity Fields and another one to predict Part Association Fields. In this work, the hourglass model [86] shall be used as a foundation. Two different general strategies will be investigated. (i) The calculation of keypoint positions based on heatmaps shall be retained as is, and only upper bounds shall be optimized for the predicted positions. This means that the prediction of keypoint positions $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ is independent of the upper bound prediction, while the prediction of upper bounds $\hat{\boldsymbol{r}}_{\boldsymbol{i}}$ depends on the predicted keypoint positions. (ii) The calculation of predicted keypoint positions and upper bounds shall be performed based on a joint optimization of both, meaning that the prediction of keypoint positions depends on the prediction of upper bounds and the other way around. Both of these strategies will make adjustments to the hourglass model necessary. In the case of strategy (i), one prediction head will be added to predict values for the upper bound calculation, while two prediction heads will be added for strategy (ii) to enable joint optimization (details later). Both strategies will be pursued for an upper bound prediction that is *directly based on the measurement error* in this section. Therefore, the strategies will be called 2-HG$_{\text{error}}$ respectively 3H-HG$_{\text{error}}$, with details about the architectures, prediction of final results, and training being discussed in the following.

2-HG$_{\text{error}}$ Approach: For strategy 2-HG$_{\text{error}}$, the prediction of keypoint positions will be retained as is for the hourglass model, meaning that one prediction head is necessary that predicts heatmaps $\hat{\boldsymbol{h}}_{\boldsymbol{i}}$ for every keypoint i from which the predicted keypoint positions $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ will be inferred. For predicting the upper bounds for the measurement error $\hat{\boldsymbol{r}}_{\boldsymbol{i}}$, a second prediction head will be necessary. Following the paradigm of heatmaps to retain spatial dimensions, this can be done through the prediction of uncertainty maps $\hat{\boldsymbol{u}}_{\boldsymbol{i}}$ with the same spatial dimensions as $\hat{\boldsymbol{h}}_{\boldsymbol{i}}$. Furthermore, the feature dimension of $\hat{\boldsymbol{u}}_{\boldsymbol{i}}$ will be 1 in this approach. The value of $\hat{\boldsymbol{u}}_{\boldsymbol{i}}$ at position (u, v) shall correspond to an upper bound for the measurement error at heatmap resolution if (u, v) is the predicted keypoint position at heatmap resolution. This means that $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ and $\hat{\boldsymbol{r}}_{\boldsymbol{i}}$ are calculated as follows:

$$\begin{aligned} \hat{\boldsymbol{y}}'_{\boldsymbol{i}} &= (\hat{x}'_i, \hat{y}'_i) = \underset{u,v}{\text{argmax}}(\hat{\boldsymbol{h}}_{\boldsymbol{i}}[u, v]) \\ \hat{\boldsymbol{r}}'_{\boldsymbol{i}} &= \hat{\boldsymbol{u}}_{\boldsymbol{i}}[\hat{x}'_i, \hat{y}'_i] \\ \hat{\boldsymbol{y}}_{\boldsymbol{i}} &= T^{-1}(\hat{\boldsymbol{y}}'_{\boldsymbol{i}}), \quad \hat{\boldsymbol{r}}_{\boldsymbol{i}} = T^{-1}(\hat{\boldsymbol{r}}'_{\boldsymbol{i}}) \end{aligned} \tag{5.8}$$

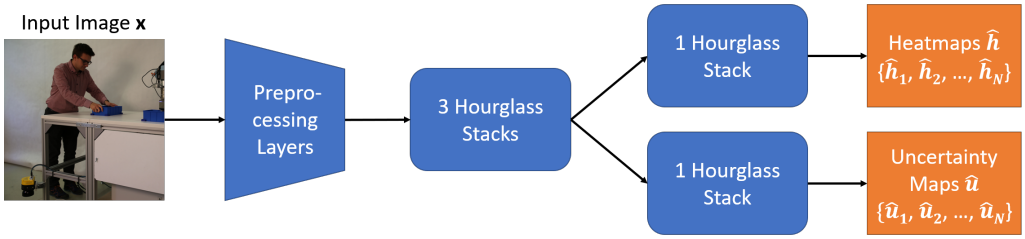


Figure 5.3: Architecture of the proposed two-head hourglass (2H-HG). It is similar to the standard hourglass model during the first three stacked hourglass blocks. Afterward, the network branches off, using an additional hourglass block in each of the branches before heatmaps and uncertainty maps are predicted. Based on author's figure from [109].

To realize this idea for the hourglass model, the architecture of a 4-stack hourglass network will be modified. Three hourglass stacks will be retained, and prediction heads will branch off afterward. Both branches will use an additional hourglass stack before predicting the heatmaps and the uncertainty maps respectively. An illustration of the proposed architecture can be found in Figure 5.3. It will be called 2H-HG in the following. For the prediction of the heatmaps, the sigmoid activation function will be used, while the softplus activation function will be used for the uncertainty maps to ensure that upper bounds $\hat{\boldsymbol{r}}_i' > 0.0$ without a fixed upper limit will be predicted at heatmap resolution.

The prediction of correct and high-quality upper bounds through Eq. (5.8) requires a suitable loss function for training. For the independent prediction of keypoint positions in strategy (i), the use of the mean squared error (MSE) between predicted and ground truth heatmaps can be continued, which will be called $l_{1,1} = l_{MSE}$ here. To ensure meaningful uncertainty maps $\hat{\boldsymbol{u}}_i$, a new loss function is required. This loss function should have several properties. When the predicted upper bound is smaller than the current measurement error, then the gradient of the loss should be the strongest, as obtaining correct predictions is the highest priority. If the upper bound is only marginally larger than the current measurement error, the neural network should further be pushed to predict a more significant upper bound such that small errors do not lead to an incorrect upper bound. Here, the gradient should still be strong, but not as strong as previously. For upper bounds that are significantly larger than the true measurement error, the neural network should be pushed towards predicting smaller upper bounds. The strength of the gradient in that case should be the weakest, as this only improves the quality Q_{rad} while obtaining correct results is of utmost importance. To design such a loss based on the measurement error, a two-part loss function is proposed, consisting of the parts $l_{1,2}$ and $l_{1,3}$. For better understanding, these losses will be defined as functions that calculate a single per-pixel loss for a single predicted uncertainty map $\hat{\boldsymbol{u}}_i$ – their calculation over all these maps and pixel positions for a single input will be discussed later. Let (x_i', y_i') denote the ground truth keypoint position at heatmap resolution for keypoint i, and let $c_{dir} > 0.0$ be a positive constant that indicates how much larger the upper bound should be compared to the actual measurement error at heatmap resolution. Then, the losses at pixel position (u, v) can be written as:

$$\begin{aligned} l_{1,2}(\hat{\boldsymbol{u}}_i, x_i', y_i', u, v) &= |\,(\hat{\boldsymbol{u}}_i[u, v] - ||\,(x_i', y_i') - (u, v)\,||_2\,|\,| \\ l_{1,3}(\hat{\boldsymbol{u}}_i, x_i', y_i', u, v) &= \max(0, -(\hat{\boldsymbol{u}}_i[u, v] - ||\,(x_i', y_i') - (u, v)\,||_2 - c_{dir}) \end{aligned} \tag{5.9}$$

The first part $l_{1,2}$ punishes a deviation of the predicted upper bound in $\hat{\boldsymbol{u}}_i$ from the measurement error for pixel (u, v) in both directions (smaller or larger) equally through the calculation of the absolute error between both. The second part $l_{1,3}$ is used to create

the desired behavior of the loss in combination with $l_{1,2}$. This combination will be performed by adding the losses using a weighting factor γ as follows: $l_{1,2} + \gamma \cdot l_{1,3}$. For $\hat{\boldsymbol{u}}_{\boldsymbol{i}}[u,v] \geq ||\,(x'_i, y'_i) - (u,v)\,||_2 + c_{dir}$, meaning that the predicted upper bound is significantly larger than the actual measurement error, the combined loss simply degrades to $l_{1,2}$. For $\hat{\boldsymbol{u}}_{\boldsymbol{i}}[u,v] < ||\,(x'_i, y'_i) - (u,v)\,||_2$ where the predicted upper bound becomes too small, the gradient of both $l_{1,2}$ and $l_{1,3}$ point into the same direction, hence they add up and produce a stronger gradient as in the previous case. For the middle case of $||\,(x'_i, y'_i) - (u,v)\,||_2 < \hat{\boldsymbol{u}}_{\boldsymbol{i}}[u,v] < ||\,(x'_i, y'_i) - (u,v)\,||_2 + c_{dir}$, the gradients of both loss functions point into different directions, hence weakening one another. To keep optimizing towards $||\,(x'_i, y'_i) - (u,v)\,||_2 + c_{dir}$, $\gamma > 1.0$ is required for this case. With $\gamma > 2.0$, the gradient for this case becomes larger than in case of $\hat{\boldsymbol{u}}_{\boldsymbol{i}}[u,v] \geq ||\,(x'_i, y'_i) - (u,v)\,||_2 + c_{dir}$, thus fulfilling the desired order of gradient strengths. To obtain the final loss function, $l_{1,1}$ will be combined with $l_{1,2}$ and $l_{1,3}$. Furthermore, masks $\boldsymbol{m_i}$ will be used to only consider pixels of the uncertainty maps in the vicinity to the true keypoint location during training – this is a procedure that was already used by Papandreou et al. [92] when results of an additional prediction head depended on the actual keypoint location. In this case, the vicinity will be defined as all positions (u,v) with $\boldsymbol{h_i}[u,v] > 0.02$, given that the ground truth heatmap $\boldsymbol{h_i}$ was calculated based on Eq. (5.4) with $a = 1.0$. In the case of a keypoint being not annotated, the respective uncertainty map will be ignored during training, as no measurement error can be calculated. Whether or not an annotation exists is indicated through variable v_i as previously. Last, the combined loss of $l_{1,2}$ and $l_{1,3}$ has to be calculated over all pixel positions of a heatmap with width W' and height H' for a total of N keypoints (in contrast, $l_{1,1}$ already is). Then, the final loss function with additional weighting factors α and β can be defined as follows (for a single input):

$$l_1 = \alpha \cdot l_{1,1} + \beta \cdot \frac{1}{N \cdot W' \cdot H'} \sum_{i=1}^{N} \sum_{u=1}^{W'} \sum_{v=1}^{H'} (\delta(v_i) \cdot \boldsymbol{m_i}[u,v] \cdot (l_{1,2}(\hat{\boldsymbol{u}}_{\boldsymbol{i}}, x'_i, y'_i, u, v) + \gamma \cdot l_{1,3}(\hat{\boldsymbol{u}}_{\boldsymbol{i}}, x'_i, y'_i, u, v))) \tag{5.10}$$

3-HG$_{\text{error}}$ Approach: For the 3-HG$_{\text{error}}$ strategy, the prediction of measurement-error-based upper bounds shall be jointly optimized together with the prediction of final keypoint positions. This goal poses a problem when the standard keypoint localization mechanism of the hourglass model is used – the heatmap. The heatmap does not directly solve a regression problem that delivers a keypoint position in the end, but only predicts per-pixel scores. Hence it is not possible to simply apply a loss to a position that indicates how it would have to change. A solution to this problem is to no longer rely on the heatmap and its maximum for the prediction of the final keypoint position, but to join the heatmap paradigm with regression itself. Respective approaches have e. g., been proposed by Papandreou et al. [92] and Kreiss et al. [63]. Both of them predict per-pixel regression vectors in addition to the heatmaps, with each regression vector pointing to where the respective keypoint for the map is expected. Such a procedure will be employed here, making a total of three prediction heads necessary: one for heatmaps $\hat{\boldsymbol{h}}_{\boldsymbol{i}}$, one for the prediction of regression vectors in form of regression maps $\hat{\boldsymbol{d}}_{\boldsymbol{i}}$, and one for the prediction of upper bounds in form of uncertainty maps $\hat{\boldsymbol{u}}_{\boldsymbol{i}}$. Each regression map $\hat{\boldsymbol{d}}_{\boldsymbol{i}}$ for keypoint i will have the same spatial resolution as the respective heatmap and uncertainty map, and a feature dimension of 2. At position (u,v), $\hat{\boldsymbol{d}}_{\boldsymbol{i}}$ will contain a displacement vector $(\Delta\hat{x}'_{i,u,v}, \Delta\hat{y}'_{i,u,v})$ relative to the pixels position that points towards the predicted position of the i-th keypoint, hence $\hat{\boldsymbol{d}}_{\boldsymbol{i}}[u,v] = (\Delta\hat{x}'_{i,u,v}, \Delta\hat{y}'_{i,u,v})$. This allows to infer the pre-

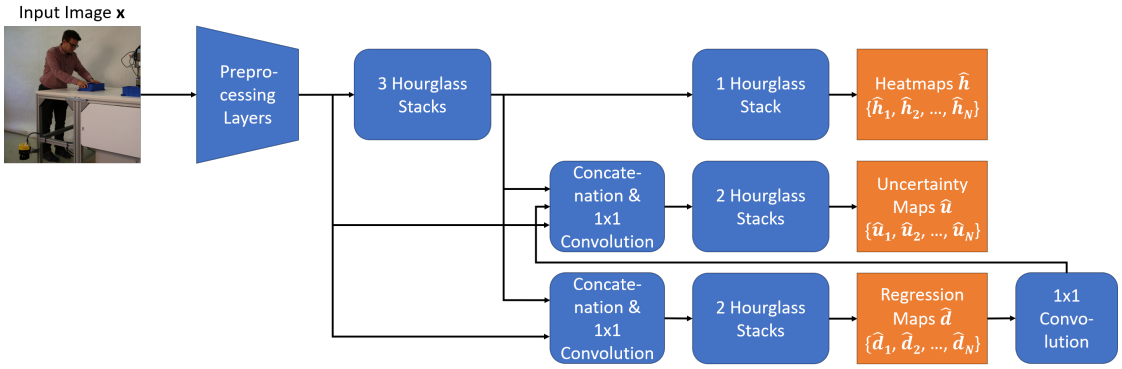


Figure 5.4: Architecture of the proposed three-head hourglass (3H-HG). The upper branch equals a 4-stack hourglass model. The lower (regression) branch aggregates features from preprocessing and the third hourglass stack through concatenation and a 1×1 convolution, to be processed by another two hourglass stacks to predict regression maps. The same input features of the regression branch, in addition to features obtained from the regression maps through a 1×1 convolution, are fed into the middle (uncertainty) branch with a similar structure to predict uncertainty maps. Based on author's figure from [109].

dicted position $(\hat{x}'_i, \hat{y}'_i)$ of the keypoint at heatmap resolution from a single position (u, v) as follows: $(\hat{x}'_i, \hat{y}'_i) = (u + \Delta\hat{x}'_{i,u,v}, v + \Delta\hat{y}'_{i,u,v})$. For the prediction of the measurement error, the uncertainty map $\hat{\boldsymbol{u}}_{\boldsymbol{i}}$ should contain the predicted measurement error for position $(\hat{x}'_i, \hat{y}'_i) = (u + \Delta\hat{x}'_{i,u,v}, v + \Delta\hat{y}'_{i,u,v})$ at position (u, v). This means $\hat{\boldsymbol{u}}_{\boldsymbol{i}}[u, v]$ corresponds to the predicted measurement error at position $(\hat{x}'_i, \hat{y}'_i)$. To obtain the final predicted keypoint position $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ and upper bound for the measurement error $\hat{\boldsymbol{r}}_{\boldsymbol{i}}$ for the i-th keypoint, a voting scheme based on the strength of heatmap activations will be employed. First, all positions (u, v) expected to be in the vicinity of the i-th keypoint will be determined through $\hat{\boldsymbol{h}}_{\boldsymbol{i}}[u, v] > 0.02$, assuming that Eq. (5.4) with $a = 1.0$ was used for ground truth heatmaps. Then, votes from these positions for $\hat{\boldsymbol{y}}'_{\boldsymbol{i}}$ and $\hat{\boldsymbol{r}}'_{\boldsymbol{i}}$ will be gathered and weighted based on the respective heatmap values, hereby avoiding a dependency on single values. For a heatmap with width W' and height H', this leads to:

$$
\begin{aligned}
\hat{\boldsymbol{r}}'_{\boldsymbol{i}} &= \frac{\sum_{u=1}^{W'} \sum_{v=1}^{H'} (\delta(\hat{\boldsymbol{h}}_{\boldsymbol{i}}[u, v] > 0.02) \cdot \hat{\boldsymbol{h}}_{\boldsymbol{i}}[u, v] \cdot \hat{\boldsymbol{u}}_{\boldsymbol{i}}[u, v])}{\sum_{u=1}^{W'} \sum_{v=1}^{H'} (\delta(\hat{\boldsymbol{h}}_{\boldsymbol{i}}[u, v] > 0.02) \cdot \hat{\boldsymbol{h}}_{\boldsymbol{i}}[u, v])} \\
\hat{\boldsymbol{y}}'_{\boldsymbol{i}} = (\hat{x}'_i, \hat{y}'_i) &= \frac{\sum_{u=1}^{W'} \sum_{v=1}^{H'} (\delta(\hat{\boldsymbol{h}}_{\boldsymbol{i}}[u, v] > 0.02) \cdot \hat{\boldsymbol{h}}_{\boldsymbol{i}}[u, v] \cdot ((u, v) + \hat{\boldsymbol{d}}_{\boldsymbol{i}}[u, v]))}{\sum_{u=1}^{W'} \sum_{v=1}^{H'} (\delta(\hat{\boldsymbol{h}}_{\boldsymbol{i}}[u, v] > 0.02) \cdot \hat{\boldsymbol{h}}_{\boldsymbol{i}}[u, v])} \\
\hat{\boldsymbol{y}}_{\boldsymbol{i}} &= T^{-1}(\hat{\boldsymbol{y}}'_{\boldsymbol{i}}), \quad \hat{\boldsymbol{r}}_{\boldsymbol{i}} = T^{-1}(\hat{\boldsymbol{r}}'_{\boldsymbol{i}})
\end{aligned}
\tag{5.11}
$$

The proposed architecture for predicting all three required maps will be called 3H-HG, as it employs three prediction heads, one for each map. Its architecture, based on the hourglass model, is depicted in Figure 5.4. Similar to 2H-HG, the top branch for predicting heatmaps corresponds to a 4-stack hourglass model, with three shared hourglass stacks and one more hourglass stack after branching off. Again, the sigmoid function is used for heatmap prediction. The bottom branch for predicting regression maps aggregates features from after the preprocessing layers and the three shared hourglass stacks through concatenation and a 1×1 convolution to bring the feature dimension down again to the standard value of 256 used by the hourglass stacks. Then, two more hourglass stacks are applied to predict the regression maps. No activation function is employed. Afterward, the regression maps are processed by a 1×1 convolution to make them available in the

form of features for the middle branch that predicts uncertainty maps. The rationale behind this procedure is that the uncertainty map no longer predicts the measurement error for each pixel location, but for each location after the application of the regression vectors from the regression map. Hence these should be made available to that branch. In the middle branch, these features are aggregated with the features from behind the preprocessing layers and the shared three hourglass stacks, again through concatenation and a 1×1 convolution to bring the feature dimension back to 256. Then, two hourglass stacks are applied to finally predict the uncertainty maps using the softmax activation function.

The last factor that has to be discussed is the loss definition for the 3-HG$_{\text{error}}$ approach. It should jointly tune the prediction of keypoint positions and upper bounds for the measurement error. Furthermore, the upper bounds shall still be based on the measurement error here. To achieve this, a modification of the loss functions proposed in Eq. (5.9) can be used, together with $l_{2,1} = l_{MSE}$ for the heatmaps (as in 2H-HG$_{\text{error}}$). The adaptation of $l_{1,2}$ and $l_{1,3}$ to new losses $l_{2,2}$ and $l_{2,3}$ can simply be done by rewriting the Euclidean distance between pixel position and ground truth in the equations to incorporate the predicted displacement vectors of the regression map $\hat{\boldsymbol{d}}_{\boldsymbol{i}}$, leading to $||\,(x'_i, y'_i) - ((u,v) + (\Delta\hat{x}'_{i,u,v}, \Delta\hat{y}'_{i,u,v}))\,||_2$ for the Euclidean distance. This formulation can be simplified such that the difference between ground truth displacement vectors $(\Delta x'_{i,u,v}, \Delta y'_{i,u,v})$ and predicted displacement vectors $(\Delta\hat{x}'_{i,u,v}, \Delta\hat{y}'_{i,u,v})$ is calculated:

$$\begin{aligned} & ||\,(x'_i, y'_i) - ((u,v) + (\Delta\hat{x}'_{i,u,v}, \Delta\hat{y}'_{i,u,v})\,||_2 \\ \Leftrightarrow & ||\,((u,v) + (\Delta x'_{i,u,v}, \Delta y'_{i,u,v})) - ((u,v) + (\Delta\hat{x}'_{i,u,v}, \Delta\hat{y}'_{i,u,v}))\,||_2 \\ \Leftrightarrow & ||\,(\Delta x'_{i,u,v}, \Delta y'_{i,u,v}) - (\Delta\hat{x}'_{i,u,v}, \Delta\hat{y}'_{i,u,v})\,||_2 \end{aligned} \tag{5.12}$$

Using this formulation, the keypoint regression can be incorporated into $l_{1,2}$ and $l_{1,3}$ to form $l_{2,2}$ and $l_{2,3}$. Together with another loss $l_{2,4}$ that favors the prediction of accurate per-pixel regression values, the following partial loss functions for 3-HG$_{\text{error}}$ can be derived, with $\boldsymbol{d_i}$ denoting a ground truth regression map such that $(\Delta x'_{i,u,v}, \Delta y'_{i,u,v}) = \boldsymbol{d_i}[u,v]$:

$$\begin{aligned} l_{2,2}(\hat{\boldsymbol{u}}_{\boldsymbol{i}}, \hat{\boldsymbol{d}}_{\boldsymbol{i}}, \boldsymbol{d_i}, u, v) &= |\,\hat{\boldsymbol{u}}_{\boldsymbol{i}}[u,v] - ||\,\boldsymbol{d_i}[u,v] - \hat{\boldsymbol{d}}_{\boldsymbol{i}}[u,v]\,||_2\,| \\ l_{2,3}(\hat{\boldsymbol{u}}_{\boldsymbol{i}}, \hat{\boldsymbol{d}}_{\boldsymbol{i}}, \boldsymbol{d_i}, u, v) &= \max(0, -(\hat{\boldsymbol{u}}_{\boldsymbol{i}}[u,v] - ||\,\boldsymbol{d_i}[u,v] - \hat{\boldsymbol{d}}_{\boldsymbol{i}}[u,v]\,||_2 - c_{dir}) \\ l_{2,4}(\hat{\boldsymbol{d}}_{\boldsymbol{i}}, \boldsymbol{d_i}, u, v) &= ||\,\boldsymbol{d_i}[u,v] - \hat{\boldsymbol{d}}_{\boldsymbol{i}}[u,v]\,||_2 \end{aligned} \tag{5.13}$$

These loss functions will again be combined together to form a final loss function l_2. From the above partial losses, $l_{2,2}$ and $l_{2,3}$ aim to achieve the same as $l_{1,2}$ and $l_{1,3}$. However, now these losses can be reduced by adjusting the predicted keypoint position as well as through changing the predicted upper bound, allowing to jointly optimize both of them. The loss function $l_{2,4}$ is here to ensure that predicted regression vectors stay close to the ground truth regression vectors. In turn, this aims at the prediction of keypoint positions close to the true keypoint locations. Combined together, these losses should ensure good quality with respect to both, Q_{rad} and Q_{pos}, with correct predictions being facilitated through the same mechanism of aiming at predicting bounds slightly above the true measurement error (by a factor of c_{dir}). The combined loss function for a single input will be written similarly to Eq. (5.10) as follows:

$$\begin{aligned} l_2 = \alpha \cdot l_{2,1} + \beta \cdot \frac{1}{N \cdot W' \cdot H'} \sum_{i=1}^{N} \sum_{u=1}^{W'} \sum_{v=1}^{H'} (\delta(v_i) \cdot \boldsymbol{m_i}[u,v] \cdot (l_{2,2}(\hat{\boldsymbol{u}}_{\boldsymbol{i}}, \hat{\boldsymbol{d}}_{\boldsymbol{i}}, \boldsymbol{d_i}, u, v) + \\ \gamma \cdot l_{2,3}(\hat{\boldsymbol{u}}_{\boldsymbol{i}}, \hat{\boldsymbol{d}}_{\boldsymbol{i}}, \boldsymbol{d_i}, u, v) + l_{2,4}(\hat{\boldsymbol{d}}_{\boldsymbol{i}}, \boldsymbol{d_i}, u, v))) \end{aligned} \tag{5.14}$$

5.5 Distribution-Based Upper Bound Prediction

In contrast to the previous section, which aimed at predicting upper bounds for the measurement error based on the measurement error itself, this section aims at *predicting upper bounds through the use of probability distributions*. By building upon distributions to predict upper bounds for the measurement error, it is no longer necessary to predict an increased version of the measurement error as upper bound. Apart from this detail of how the upper bounds for the measurement error are predicted, the investigated approaches in this section will be similar to 2H-HG$_{\text{error}}$ and 3H-HG$_{\text{error}}$. This is done for two reasons. First, it makes also sense for the distribution-based approaches to investigate a scenario where the position is independently optimized from the upper bound for the measurement error, as well as a scenario where this optimization is performed jointly. Second, by keeping the approaches as similar as possible, the influence of building the upper bound prediction on the measurement error itself versus building it upon probability distributions can be better examined. Therefore, the same 2H-HG and 3H-HG architectures will be used, except for minor necessary adjustments to the prediction of uncertainty maps $\hat{\boldsymbol{u}}_{\boldsymbol{i}}$.

For all approaches throughout this section, the prediction of the upper bounds will be performed based on the predicted standard deviation parameters $(\hat{\sigma}_x, \hat{\sigma}_y)$ of a *2D Gaussian* under the assumption of independent random variables to simplify the problem. This makes it necessary that the feature dimension of the uncertainty maps $\hat{\boldsymbol{u}}_{\boldsymbol{i}}$ is increased to 2. Subsequently, the predicted values for $(\hat{\sigma}_{x,i}, \hat{\sigma}_{y,i})$ allow the inference of position intervals along the x-axis and y-axis that shall contain all likely positions of the i-th keypoint with high confidence, hence being a suitable foundation for the calculation of an upper bound. In the following, two approaches that incorporate this idea will be investigated. Based on the use of Gaussian parameters and the 2H-HG as well as 3H-HG architectures, these will be called 2H-HG$_{\text{gauss}}$ and 3H-HG$_{\text{gauss}}$ respectively. In the following, both will be introduced in greater detail.

2H-HG$_{\text{gauss}}$ Approach: Similar to the 2H-HG$_{\text{error}}$ approach, the 2H-HG$_{\text{gauss}}$ approach employs the 2H-HG architecture to predict a heatmap $\hat{\boldsymbol{h}}_{\boldsymbol{i}}$ and an uncertainty map $\hat{\boldsymbol{u}}_{\boldsymbol{i}}$ for each keypoint i, with the difference that $\hat{\boldsymbol{u}}_{\boldsymbol{i}}$ contains two values $(\hat{\sigma}_{x,i,u,v}, \hat{\sigma}_{y,i,u,v})$ at each spatial position (u, v). Again, the prediction of final keypoint positions shall be performed without changes based on the position of the maximum in each predicted heatmap $\hat{\boldsymbol{h}}_{\boldsymbol{i}}$. In contrast, the prediction of each final upper bound $\hat{\boldsymbol{r}}_{\boldsymbol{i}}$ will be performed differently. Both predicted values $\hat{\sigma}_{x,i,u,v}$ and $\hat{\sigma}_{y,i,u,v}$ shall be the predicted standard deviations of a 2D Gaussian with independent random variables for which the position (u, v) serves as mean. In turn, the true keypoint location will be viewed as a sample from that distribution. This view is similar to how Lakshminarayanan et al. [65] view regression problems as a problem of predicting Gaussian distributions, however, adapted to the problem of heatmap-based human pose estimation. Now that the true keypoint location is viewed as a sample from the Gaussian distribution defined by (u, v) and $(\hat{\sigma}_{x,i,u,v}, \hat{\sigma}_{y,i,u,v})$, an upper bound for the measurement error can be calculated based on the 3-sigma-rule, as the true keypoint position should then be encapsulated in the resulting interval. As independent random variables are assumed, a deviation of 3σ has to be considered independently along the x- and y-axis. Thus, the resulting worst case of $3\hat{\sigma}_{x,i,u,v}$ and $3\hat{\sigma}_{y,i,u,v}$ has to be considered for the upper bound prediction, with $\hat{\boldsymbol{r}}'_{\boldsymbol{i,u,v}} = \sqrt{(3\hat{\sigma}_{x,i,u,v})^2 + (3\hat{\sigma}_{y,i,u,v})^2}$ capturing the predicted worst-case deviation in that case. Then, the final keypoint position $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ and predicted upper bound $\hat{\boldsymbol{r}}_{\boldsymbol{i}}$ for this position are calculated as follows:

$$\begin{aligned} \hat{\boldsymbol{y}}_{\boldsymbol{i}}' = (\hat{x}_i', \hat{y}_i') &= \underset{u,v}{\text{argmax}}(\hat{\boldsymbol{h}}_{\boldsymbol{i}}[u,v]) \\ (\hat{\sigma}_{x,i}, \hat{\sigma}_{y,i}) &= \hat{\boldsymbol{u}}_{\boldsymbol{i}}[\hat{x}_i', \hat{y}_i'] \\ \hat{\boldsymbol{r}}_{\boldsymbol{i}}' &= \sqrt{(3\hat{\sigma}_{x,i})^2 + (3\hat{\sigma}_{y,i})^2} \\ \hat{\boldsymbol{y}}_{\boldsymbol{i}} = T^{-1}(\hat{\boldsymbol{y}}_{\boldsymbol{i}}'), \;\; \hat{\boldsymbol{r}}_{\boldsymbol{i}} &= T^{-1}(\hat{\boldsymbol{r}}_{\boldsymbol{i}}') \end{aligned} \tag{5.15}$$

In order for this approach to work properly, the prediction of suitable standard deviations has to be ensured through the training procedure. The training of the heatmaps for the prediction of the keypoint positions – or the mean values of the standard deviations in this case – continues to use the mean squared error loss, thus adding the partial loss of $l_{3,1} = l_{MSE}$ for training. To obtain meaningful values $(\hat{\sigma}_{x,i}, \hat{\sigma}_{y,i})$ independent of which keypoint position is selected in the end, the values at each position (u, v) in the uncertainty map will be trained under the assumption that this position is the predicted mean. Optimization of the standard deviations can be performed through a loss function that is closely related to the more general proposed loss function of Lakshminarayanan et al. [65] and of Kübler [64]. The overall training goal is to maximize the value of a 2D Gaussian density function with independent random variables at each position (u, v), which is defined by the mean (u, v) as well as $\hat{\sigma}_{x,i,u,v}$ and $\hat{\sigma}_{y,i,u,v}$. It can be written as follows for the i-th keypoint with (x_i', y_i') denoting the keypoint's position at heatmap resolution:

$$\begin{aligned} &\frac{1}{2\pi\hat{\sigma}_{x,i,u,v}\hat{\sigma}_{y,i,u,v}} e^{-\frac{1}{2}\left(\frac{(x_i'-u)^2}{\hat{\sigma}_{x,i,u,v}^2} + \frac{(y_i'-v)^2}{\hat{\sigma}_{y,i,u,v}^2}\right)} \\ \Leftrightarrow e^{-\frac{1}{2}\left(2ln(2\pi) + 2ln(\hat{\sigma}_{x,i,u,v}) + \frac{(x_i'-u)^2}{\hat{\sigma}_{x,i,u,v}^2} + 2ln(\hat{\sigma}_{y,i,u,v}) + \frac{(y_i'-v)^2}{\hat{\sigma}_{y,i,u,v}^2}\right)} \end{aligned} \tag{5.16}$$

The second line rewrites the formula of the 2D Gaussian such that all variables become part of the exponent, making it easier to find the relevant parts for maximizing the equation. To maximize the equation, the exponent needs to be maximized, which comes down to minimizing the full term within the outer brackets. Removing the constant of $2ln(2\pi)$, this leads to a partial loss function $l_{3,2}$ for a single keypoint i and a single position (u, v), with $\hat{\sigma}_{x,i,u,v} = \hat{\boldsymbol{u}}_{\boldsymbol{i}}[u, v, 1]$ and $\hat{\sigma}_{y,i,u,v} = \hat{\boldsymbol{u}}_{\boldsymbol{i}}[u, v, 2]$:

$$l_{3,2}(\hat{\boldsymbol{u}}_{\boldsymbol{i}}, x_i', y_i', u, v) = 2ln(\hat{\boldsymbol{u}}_{\boldsymbol{i}}[u,v,1]) + 2ln(\hat{\boldsymbol{u}}_{\boldsymbol{i}}[u,v,2]) + \frac{(x_i'-u)^2}{(\hat{\boldsymbol{u}}_{\boldsymbol{i}}[u,v,1])^2} + \frac{(y_i'-v)^2}{(\hat{\boldsymbol{u}}_{\boldsymbol{i}}[u,v,2])^2} \tag{5.17}$$

This 2D loss function is highly similar to the general 1D version proposed by Lakshminarayanan et al. [65] and a straight extension into 2D with a fixed mean of the 1D variant proposed by Kübler [64]. During training, some practical adjustments have to be employed when using $l_{3,2}$. First, for values $\hat{\sigma}_{x,i,u,v}$ and $\hat{\sigma}_{y,i,u,v}$ that are very close to 0, the natural logarithm could become infinitely large. To avoid this, the values for both standard deviations will be capped to be $> \epsilon$, where $\epsilon > 0$ is a small positive constant. Second, the quadratic differences $(x_i' - u)^2$ and $(y_i' - v)^2$ will also be restricted to be larger or equal to a positive value of 0.025 to prevent them from vanishing in certain cases. To favor readability, these adjustments will not be included in the loss equations directly. By

combining $l_{3,1}$ and $l_{3,2}$, the final loss function l_3 can be defined for a single input by using the same additional variables introduced in Eq. (5.10) as follows:

$$l_3 = \alpha \cdot l_{3,1} + \beta \cdot \frac{1}{N \cdot W' \cdot H'} \sum_{i=1}^{N} \sum_{u=1}^{W'} \sum_{v=1}^{H'} (\delta(v_i) \cdot \boldsymbol{m_i}[u,v] \cdot l_{3,2}(\hat{\boldsymbol{u}}_{\boldsymbol{i}}, x'_i, y'_i, u, v)) \quad (5.18)$$

3H-HG$_{\text{gauss}}$ Approach: The 3H-HG$_{\text{gauss}}$ approach will be similar to the 3H-HG$_{\text{error}}$ approach, as the same 3H-HG architecture will be used, except that the predicted uncertainty maps $\hat{\boldsymbol{u}}_{\boldsymbol{i}}$ will contain two results at each position (u, v) instead of one – standard deviations $\hat{\sigma}_{x,i,u,v}$ and $\hat{\sigma}_{y,i,u,v}$, as in case of 2H-HG$_{\text{gauss}}$. Similar to the 3H-HG$_{\text{error}}$ approach, the regression map $\hat{\boldsymbol{d}}_{\boldsymbol{i}}$ will be used to obtain the final predicted keypoint position for the i-th keypoint, that also serves as the mean for the predicted 2D Gaussian distribution. The goal is to jointly optimize the predicted mean with the upper bound obtained from the standard deviations. Thus, for each position (u, v), the uncertainty map shall contain standard deviations $(\hat{\sigma}_{x,i,u,v}, \hat{\sigma}_{y,i,u,v}) = \hat{\boldsymbol{u}}_{\boldsymbol{i}}[u,v]$ for a 2D Gaussian with mean $(u + \Delta\hat{x}'_{i,u,v}, v + \Delta\hat{y}'_{i,u,v})$, where $(\Delta\hat{x}'_{i,u,v}, \Delta\hat{x}'_{i,u,v}) = \hat{\boldsymbol{d}}_{\boldsymbol{i}}[u,v]$. Then, the principle of using a voting scheme as in Eq. (5.11) can be employed, letting all keypoints that are suspected to be in the vicinity of the true keypoint position vote on the final location and standard deviations. Building upon this equation and assuming that the prediction of heatmaps was trained using ground truth heatmaps generated by Eq. (5.4) with $a = 1$, final results $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ and $\hat{\boldsymbol{r}}_{\boldsymbol{i}}$ are obtained as follows for the i-th keypoint:

$$\begin{aligned}
(\hat{\sigma}_{x,i}, \hat{\sigma}_{y,i}) &= \frac{\sum_{u=1}^{W'} \sum_{v=1}^{H'} (\delta(\hat{\boldsymbol{h}}_{\boldsymbol{i}}[u,v] > 0.02) \cdot \hat{\boldsymbol{h}}_{\boldsymbol{i}}[u,v] \cdot \hat{\boldsymbol{u}}_{\boldsymbol{i}}[u,v])}{\sum_{u=1}^{W'} \sum_{v=1}^{H'} (\delta(\hat{\boldsymbol{h}}_{\boldsymbol{i}}[u,v] > 0.02) \cdot \hat{\boldsymbol{h}}_{\boldsymbol{i}}[u,v]}) \\
\hat{\boldsymbol{r}}'_{\boldsymbol{i}} &= \sqrt{(3\hat{\sigma}_{x,i})^2 + (3\hat{\sigma}_{y,i})^2} \\
\hat{\boldsymbol{y}}'_{\boldsymbol{i}} = (\hat{x}'_i, \hat{y}'_i) &= \frac{\sum_{u=1}^{W'} \sum_{v=1}^{H'} \delta(\hat{\boldsymbol{h}}_{\boldsymbol{i}}[u,v] > 0.02) \cdot \hat{\boldsymbol{h}}_{\boldsymbol{i}}[u,v] \cdot ((u,v) + \hat{\boldsymbol{d}}_{\boldsymbol{i}}[u,v]))}{\sum_{u=1}^{W'} \sum_{v=1}^{H'} (\delta(\hat{\boldsymbol{h}}_{\boldsymbol{i}}[u,v] > 0.02) \cdot \hat{\boldsymbol{h}}_{\boldsymbol{i}}[u,v])} \\
\hat{\boldsymbol{y}}_{\boldsymbol{i}} &= T^{-1}(\hat{\boldsymbol{y}}'_{\boldsymbol{i}}), \quad \hat{\boldsymbol{r}}_{\boldsymbol{i}} = T^{-1}(\hat{\boldsymbol{r}}'_{\boldsymbol{i}})
\end{aligned} \quad (5.19)$$

Now, only the loss used during training has to be defined, such that meaningful Gaussian distributions with independent random variables are predicted. Furthermore, the distribution mean shall be jointly optimized with the standard deviations. This can simply be done by replacing the fixed mean defined in loss function $l_{3,2}$ with the predicted mean which can be calculated based on the current location (u, v) in the heatmap and the value of the regression map $\hat{\boldsymbol{d}}_{\boldsymbol{i}}$ at location (u, v). Replacing the respective part in $l_{3,2}$ to form $l_{4,2}$ (similar to the replacement from $l_{1,2}$ to $l_{2,2}$) leads to the following function:

$$\begin{aligned}
l_{4,2}(\hat{\boldsymbol{u}}_{\boldsymbol{i}}, \hat{\boldsymbol{d}}_{\boldsymbol{i}}, \boldsymbol{d_i}, u, v) =& 2ln(\hat{\boldsymbol{u}}_{\boldsymbol{i}}[u,v,1]) + 2ln(\hat{\boldsymbol{u}}_{\boldsymbol{i}}[u,v,2]) \\
&+ \frac{(\boldsymbol{d_i}[u,v,1] - \hat{\boldsymbol{d}}_{\boldsymbol{i}}[u,v,1])^2}{(\hat{\boldsymbol{u}}_{\boldsymbol{i}}[u,v,1])^2} + \frac{(\boldsymbol{d_i}[u,v,2] - \hat{\boldsymbol{d}}_{\boldsymbol{i}}[u,v,2])^2}{(\hat{\boldsymbol{u}}_{\boldsymbol{i}}[u,v,2])^2}
\end{aligned} \quad (5.20)$$

To obtain the final loss for 3H-HG$_{\text{gauss}}$, this loss has to be combined with the loss for the heatmaps $l_{4,1}$. As in every other approach, $l_{4,1} = l_{MSE}$ will be used. This leads to the final loss function l_4 (using the same variable names as in Eq. (5.14)):

$$l_4 = \alpha \cdot l_{4,1} + \beta \cdot \frac{1}{N \cdot W' \cdot H'} \sum_{i=1}^{N} \sum_{u=1}^{W'} \sum_{v=1}^{H'} (\delta(v_i) \cdot \boldsymbol{m_i}[u,v] \cdot l_{4,2}(\hat{\boldsymbol{u}}_{\boldsymbol{i}}, \hat{\boldsymbol{d}}_{\boldsymbol{i}}, \boldsymbol{d_i}, u, v)) \quad (5.21)$$

5.6 Experiments

Next, experiments will be performed to evaluate the capability of the proposed methods to produce correct upper bounds for the measurement error of their predicted keypoint positions. A detection will be considered correct in these experiments, if the function for assessing correctness g_{rad} (see Eq. (5.1)) yields 1. Whether the evaluation of this function is performed with all components being defined at heatmap resolution or at original image resolution does not matter. This is the case, as the function only compares two distances, the Euclidean distance between predicted and annotated keypoint, as well as the distance that is defined as upper bound $\hat{\boldsymbol{r}}_{\boldsymbol{i}}$. When transforming between original image and heatmaps, both of these distances are affected by the same percentual scaling factors, and remain unchanged for the other actions like image cropping. Looking at the quality metrics that will be employed during evaluation, Q_{rad} and Q_{pos}, evaluation at heatmap resolution will yield more meaningful results. This is the case, as input images for the hourglass model must have a fixed size of 256×256 pixels, which in turn means that image crops of individual persons are rescaled before being fed into the neural network. During backprojection into the original image, this scaling has to be reverted, meaning that the same predicted value for the upper bound at heatmap resolution ends up being different for differently-sized image crops. For larger image crops, the backprojected value of the upper bound will also be larger. This in turn leads to a larger difference when evaluating the absolute deviation between measurement error and upper bound in Q_{rad}. This effect can be omitted when performing evaluation at heatmap resolution which is the same for all inputs, which will thus be done in the following. All experiments will be performed on the MPII Human Pose dataset, using the same train/val/test splits introduced in Section 4.5.

With respect to the evaluation metrics, the *percentage of correct results* C-Rad regarding g_{rad} will be measured, as well as the *average errors* $\mathrm{E_{rad}}$ and $\mathrm{E_{pos}}$ which will be defined based on both quality metrics Q_{rad} and Q_{pos} at heatmap resolution. In addition, the *average size of the predicted upper bounds* $\hat{\boldsymbol{r}}'_{\boldsymbol{j,i}}$ at heatmap resolution will be assessed. Whenever a heatmap $\hat{\boldsymbol{h}}_{j,i}$ is predicted that does not contain a value above 0.02, the result is directly filtered out, which means that it will be considered not correct with respect to C-Rad and will not be included for the calculation of $\mathrm{E_{rad}}$, $\mathrm{E_{pos}}$ and the average upper bound value. This is done, as upper bound prediction was only learned for heatmap values larger than 0.02, meaning that no meaningful upper bound can be expected in this case. The resulting evaluation metrics are then defined as follows, using j to index a total of M inputs and i to index a total of N keypoints:

$$
\begin{aligned}
\text{C-Rad} &= 100 \cdot \frac{\sum_{j=1}^{M}\sum_{i=1}^{N}(\delta(v_{j,i})\delta(\max(\hat{\boldsymbol{h}}_{\boldsymbol{j,i}}) > 0.02)\delta(||\boldsymbol{y}'_{\boldsymbol{j,i}} - \hat{\boldsymbol{y}}'_{\boldsymbol{j,i}}||_2 \leq \hat{\boldsymbol{r}}'_{\boldsymbol{j,i}}))}{\sum_{j=1}^{M}\sum_{i=1}^{N}\delta(v_{j,i})} \\
\mathrm{E_{rad}} &= \frac{\sum_{j=1}^{M}\sum_{i=1}^{N}(\delta(v_{j,i})\delta(\max(\hat{\boldsymbol{h}}_{\boldsymbol{j,i}}) > 0.02)Q_{rad}(\hat{\boldsymbol{y}}'_{\boldsymbol{j,i}}, \boldsymbol{y}'_{\boldsymbol{j,i}}, \hat{\boldsymbol{r}}'_{\boldsymbol{j,i}}))}{\sum_{j=1}^{M}\sum_{i=1}^{N}(\delta(v_{j,i})\delta(\max(\hat{\boldsymbol{h}}_{\boldsymbol{j,i}}) > 0.02))} \\
\mathrm{E_{pos}} &= \frac{\sum_{j=1}^{M}\sum_{i=1}^{N}(\delta(v_{j,i})\delta(\max(\hat{\boldsymbol{h}}_{\boldsymbol{j,i}}) > 0.02)Q_{pos}(\hat{\boldsymbol{y}}'_{\boldsymbol{j,i}}, \boldsymbol{y}'_{\boldsymbol{j,i}}))}{\sum_{j=1}^{M}\sum_{i=1}^{N}(\delta(v_{j,i})\delta(\max(\hat{\boldsymbol{h}}_{\boldsymbol{j,i}}) > 0.02))} \\
\text{Avg. } \hat{\boldsymbol{r}}' &= \frac{\sum_{j=1}^{M}\sum_{i=1}^{N}\delta(v_{j,i})(\delta(max(\hat{\boldsymbol{h}}_{\boldsymbol{j,i}}) > 0.02)\hat{\boldsymbol{r}}'_{\boldsymbol{j,i}})}{\sum_{j=1}^{M}\sum_{i=1}^{N}(\delta(v_{j,i})\delta(\max(\hat{\boldsymbol{h}}_{\boldsymbol{j,i}}) > 0.02))}
\end{aligned}
\tag{5.22}
$$

Approach	C-Rad	E_{rad}	E_{pos}	Avg. $\hat{r}'$
1H-HG_{error}	73.8%	1.271	1.941	1.918
2H-HG_{error}	79.5%	1.417	2.043	2.367
3H-HG_{error}	76.4%	**1.234**	1.944	2.084
2H-HG_{gauss}	**94.7%**	3.017	1.946	4.417
3H-HG_{gauss}	89.5%	2.327	**1.931**	3.615

Table 5.1: Experimental results on the test split for all proposed methods. Previously published by the author [109], © 2023 IEEE, slightly altered.

Throughout all experiments, training of the previously introduced approaches will be performed using the RMS-prop optimizer [122] with a learning rate of $2.5e^{-4}$ and gradient clipping for 120 epochs on the train split. The same augmentations as in Section 4.5 will be performed and keypoints falling outside the cropped image will be treated as if the keypoint annotations were missing. The 1H-HG approach will be trained using the mean squared error between predicted and ground truth heatmaps as loss $\alpha \cdot l_{MSE}$, all other approaches will be trained using their previously introduced custom loss functions. With respect to weighting factors, $\beta = 1$ and $\gamma = 4$ will be employed for all approaches using these factors. For all 1H and 2H approaches $\alpha = 15$ will be employed, while $\alpha = 5$ will be used for all 3H approaches. This follows the rationale that good heatmaps are crucial for the success of all methods, however, in the case of the 3H approaches, the keypoint localization does not only depend on the heatmap but also on the regression map, hence the weighting of the heatmap is reduced. For the generation of ground truth heatmaps, a 2D Gaussian in accordance with Eq 5.4 will be employed, using $a = 1$ as required by most methods and $\sigma = 2$ with the goal of allowing more pixels to vote in the 3H approaches compared to using $\sigma = 1$. In case of a missing keypoint annotation, all heatmap values will be set to zero. No intermediate supervision will be used, and the layer for predicting intermediate results in each intermediate hourglass stack will be removed.

5.6.1 Evaluation of Approaches

The evaluation of the approaches 1H-HG_{error}, 2H-HG_{error}, 3H-HG_{error}, 2H-HG_{gauss} and 3H-HG_{gauss} is performed on the test split, using the settings and training procedures described previously. The experimental results with respect to the evaluation metrics from Eq. (5.22) are displayed in Table 5.1. The results show, that the Gaussian-based approaches 2H-HG_{gauss} and 3H-HG_{gauss} significantly outperform the measurement-error-based approaches when it comes to the percentage of correct results with respect to C-Rad. While the worst Gaussian-based approach achieves 89.5% correct results, the best measurement-error-based approach only achieves 79.5% correct results. However, this comes at a cost in quality, as the average deviation of the upper bound from the measurement error E_{rad} is significantly larger. The lowest value for E_{rad} achieved with the Gaussian-based approaches is about 2.3 pixels, while the highest value for E_{rad} from the measurement-error-based approaches is about 1.4 pixels. So far, this behavior can be expected, as the measurement-error-based methods aim by design at predicting upper bounds closer to the measurement error, with the potential cost of losing correct predictions. The value for E_{rad} is furthermore correlated with the average size of the predicted upper bound (Avg. $\hat{r}'$) in the presented results. This makes sense, as E_{rad} depends on two factors, (i) the error between the predicted keypoint position and annotated keypoint position and (ii) the size

of the predicted upper bound. With E_{pos} accounting for (i) and being similar for all approaches, a correlation between Avg. $\hat{\boldsymbol{r}}'$ and E_{rad} must exist when varying values for E_{rad} are observed. Avg. $\hat{\boldsymbol{r}}'$ values range from 1.9 to 4.4 pixels. To assess if these are reasonable sizes, a look at the annotated thresholds from the MPII Human Pose dataset for the calculation of the PCKh@0.5 score can be taken, where they describe the maximum allowed measurement error for correctness. When projecting the thresholds $0.5 \cdot s_{h,j}$ from the test split to heatmap resolution and calculating their average, a value of 3.2 pixels is obtained. Given that value, average upper bound sizes of 1.9 to 4.4 seem reasonable.

Apart from what the quantitative analysis through the evaluation metrics can capture, a good method for upper bound prediction should have additional properties. The magnitude of the predicted upper bound should grow with the size of the measurement error, which should be reflected by C-Rad, however, given the small average localization error E_{pos} for all methods, these cases will be rare and will thus not be well reflected. Furthermore, some cases where the predicted upper bounds overshoot the actual measurement error by far are acceptable, but such strong outliers may have a noticeable impact on the average E_{rad}. To get a better impression of how the approaches perform on a case-to-case basis, the relation between the predicted upper bounds $\hat{\boldsymbol{r}}'_{j,i}$ and the respective measurement errors (calculated using Q_{pos}) at heatmap resolution is displayed in Figure 5.5 for the left wrist, right hip, and top of the head keypoints on the test split. The y-axis represents the predicted upper bound for the measurement error and the x-axis the measurement error itself, both at heatmap resolution in pixels. All results above or on the dotted line in each plot are considered correct (upper bound $\geq$ measurement error). It can be seen that all measurement-error-based methods do not fare well, as their predicted upper bounds tend to cluster around a single value, frequently predicting small, incorrect upper bounds for larger measurement errors – an undesired behavior. On the other hand, the Gaussian-based methods perform very well regarding this aspect, regularly predicting large upper bounds for large measurement errors, which is desired. Some small predictions for large measurement errors exist, as well as some upper bounds that are slightly too small, however, these occur in far fewer cases. On the downside, upper bounds that are way larger than the measurement error occur more frequently, however, this behavior can be expected when the upper bound is designed to capture all potential positions for a keypoint.

Furthermore, qualitative results are supplied using two images that are not from the MPII dataset. In Figure 5.6a, the results of the 3H-HG_{gauss} approach for such an image supplied with MPII-style keypoint annotations are shown. Small upper bounds are predicted for the clearly visible keypoints of the upper body. Larger upper bounds are predicted for the lower body keypoints, which are harder to detect due to partial occlusions and monocolored loose jeans. Next, Figure 5.6b shows the result of the same method for the left wrist in the image that was previously used to illustrate high positional uncertainty in the image data. The predicted keypoint position may seem odd at first, but under the assumption that the neural network does not understand that both hands are required to carry the box, it makes perfect sense – then, occlusion by either the box or the human body would be possible, with the keypoint position being placed in-between. Compared to Figure 5.6a, the predicted upper bound is significantly larger, which is desired in the face of high positional uncertainty. On the downside, the prediction is not large enough to capture all potential positions for the keypoint (see Figure 5.2). Overall, this shows the potential of the method, but also highlights that further improvements are necessary. In Figure 5.6c, results for the left wrist are predicted using the 2H-HG_{gauss} approach. This method does not use a voting scheme for the final keypoint position as 3H-HG_{gauss} does,

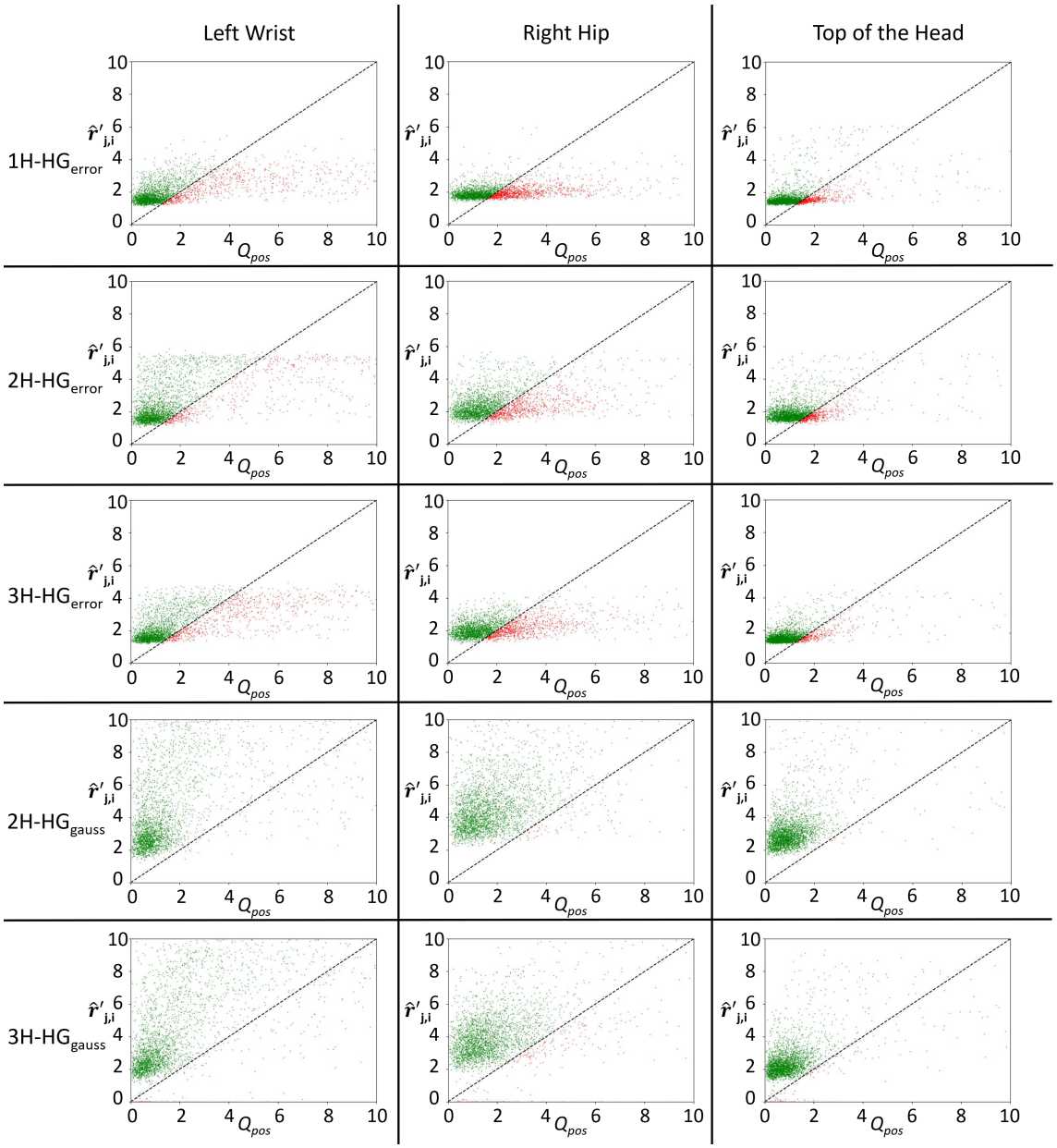


Figure 5.5: Plots comparing the size of the predicted upper bounds for the measurement error (y-axis) to the actual measurement errors (x-axis) on the test split. Results for the keypoints left wrist (left column), right hip (middle column), and top of the head (right column) are displayed for all investigated approaches. Results above or on the dotted line are considered correct (green), while those below are considered incorrect (red).

(a) 3H-HG$_{\text{gauss}}$ result **(b)** 3H-HG$_{\text{gauss}}$ result **(c)** 2H-HG$_{\text{gauss}}$ result

Figure 5.6: Predicted keypoint positions and upper bounds (green dots and circles) as well as annotated keypoint positions (red dots). In (b) and (c), only the predicted position and upper bound for the heavily occluded left wrist are displayed.

$\boldsymbol{m}$-threshold	C-Rad	E$_{\text{rad}}$	E$_{\text{pos}}$	Avg. $\hat{\boldsymbol{r}}'$
0.02	89.5	**2.327**	**1.931**	3.615
$3.5e^{-4}$	91.5	2.678	1.962	4.082
None	**97.7**	6.031	2.859	8.761

Table 5.2: Effect of training with a lower mask threshold or no mask on the 3H-HG$_{\text{gauss}}$ results. Previously published by the author [109], © 2023 IEEE, slightly altered.

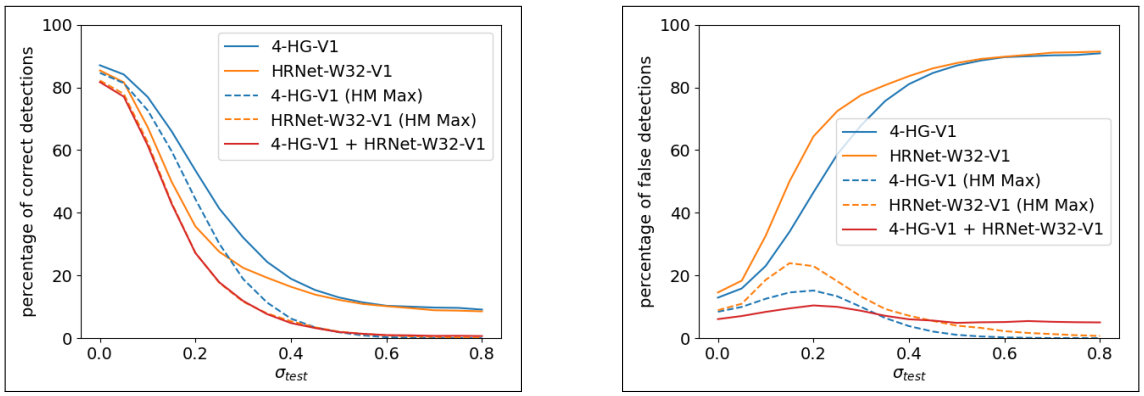
but uses the most likely position from the heatmap instead. In the given image, this was the location where the left wrist would probably be if the human supports the box with both hands from the bottom – a reasonable guess. Similar to the other approach, the predicted upper bound is large but not large enough to cover all potential keypoint locations.

Through the previous experiments, the potential of the Gaussian-based approaches for the prediction of upper bounds has been highlighted, which performed well in many cases. One further factor with a significant impact on the performance shall be investigated as part of this work: The choice to use a mask during training. This investigation will be performed for 3H-HG$_{\text{gauss}}$ only. During training, the mask $\boldsymbol{m_i}$ is used to limit the learning of regression vectors and standard deviations in the respective maps to pixel positions close to the annotated i-th keypoint. Training at far away pixel positions as well could be beneficial in theory, so that useful regression vectors and standard deviations are available across all pixels in the maps. In turn, the calculation of the final position and upper bound through the voting scheme in Eq. (5.19) could become less dependent on good heatmap predictions. Thus, the training of 3H-HG$_{\text{gauss}}$ will be repeated, once without mask, and once using a less restrictive threshold of $3.5e^{-4}$ for the mask. The results can be found in Table 5.2. While significantly more correct results were achieved without mask, the size of predicted upper bounds exploded to more than 8 pixels on average. 8 pixels already result in spheres with a diameter of 16 pixels, which is $1/4$ of the heatmap resolution of 64 pixels in each dimension – a massive upper bound for the measurement error that limits practical usability severely. On the contrary, applying the mask with threshold $3.5e^{-4}$ led to a slight increase in both, correct results and the average size of the upper bound, hinting that the threshold for the mask could be used to achieve a trade-off between correct results and upper bound size (and in turn quality with respect to Q_{rad}).

6 Impact and Handling of Noise

In this chapter, the third of the four central points of this work will be discussed: *How can noise occur and affect human pose estimation in safety-critical industrial robot applications, and how can its impact on human pose estimation be handled?* The presence of noise in input data can be very harmful to neural network performance, as it was e. g., highlighted by Geirhos et al. [38] and Zheng et al. [148] for the task of image classification with noise-corrupted input images. In the field of human pose estimation, the impact of noise was long neglected, with the work of Wang et al. [129] just recently bringing attention to the topic. They showed the negative impact of various kinds of noise on the performance of neural networks for human pose estimation, leading to a high number of keypoints being localized incorrectly. When it comes to safety-critical applications, such incorrect detections are unacceptable as previously outlined. To deal with them, methods from Chapter 4 could be used to detect or eliminate them, or methods from Chapter 5 could be capable of capturing the additional uncertainty induced by the noise through the prediction of larger upper bounds. Hence both methodologies could potentially reduce the amount of incorrect results that are induced by noise, but could also severely impact the usefulness of human pose estimation for SSM. When major amounts of incorrect results are simply filtered out, then there will be almost no correct results left that can be employed by SSM. When positional uncertainty from noise is captured through larger predicted upper bounds for the measurement error, then very large upper bounds can occur, which only indicate that the keypoint could basically be everywhere. Therefore, other methods are required to limit the impact of noise. For potential other methods, it has to be kept in mind that the environment of safety-critical industrial applications is highly regulated compared to applications in the wild. The safety standard ISO 12100 [47] mandates (i) that the limits of the machinery are determined and (ii) that hazards and risks are analyzed and appropriately reduced. The robot-specific safety standard ISO 10218-2 [49] is primarily in line with the requirements from ISO 12100. The determination of the limits of the machinery means, that a machine may only be operated under specific conditions. Such limitations can include that operation under direct sunlight is prohibited or that operation is restricted to a certain range of environmental temperatures [47]. In addition to these limits in environmental conditions, the mandatory analysis of hazards and risks makes it necessary to identify potential malfunctions of safety measures due to noise within these limits. Together, they affect if and how noise needs to be handled.

In the following, the negative impact of noise on 2D single-person human pose estimation and previously introduced methods will be highlighted through experiments, demonstrating the severity of its impact. Then, the problem of limiting this impact will be defined, under the special constraints and limitations that apply due to safety standards. Afterward, potential approaches for dealing with noise in human pose estimation will be discussed, with selected approaches being fleshed out further. Last, a large-scale evaluation of these selected approaches will be performed, showing how well they are able to handle noise, and how they perform in combination with the approaches from Chapter 4 and 5. The contents of this chapter are primarily based upon previous work of the author [108].



(a) Correct results

(b) False results

Figure 6.1: Experimental results for baseline models and error reduction methods from Chapter 4 on noisy data. The x-axis denotes the strength of the noise through the employed standard deviation σ_{test}. Correct results decrement for all methods, but all error detection methods are able to keep the amount of false results low.

6.1 Impact of Noise on Human Pose Estimation

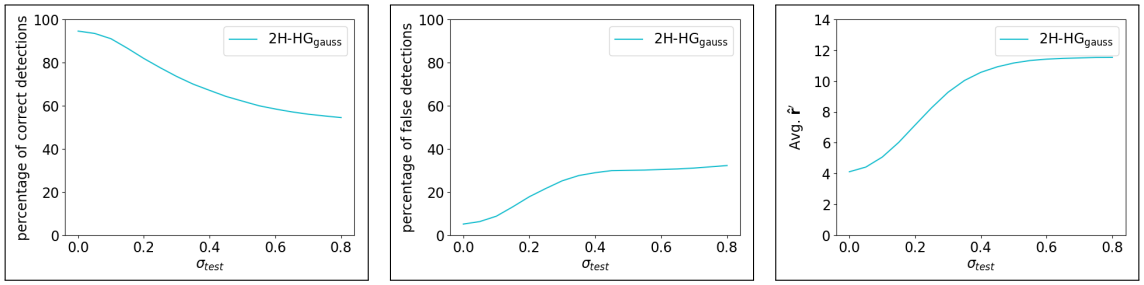
Throughout this section, the negative impact of noise on human pose estimation shall be highlighted, as well as its effect on the proposed methods from Chapter 4 and 5. To this end, 4-HG-V1 and HRNet-W32-V1 from Chapter 4 will be evaluated – once in their native form without adjustments, once with thresholding over the maximum heatmap values for the elimination of incorrect results, and once put together into a diverse neural network ensemble. When thresholding over the heatmap maximum, $c_{max} = 0.2$ will be used, and $c_R = 0.5$ is used for the diverse neural network ensemble. The same trained instances of these networks that were used in the experiments in Section 4.5 will be employed. Furthermore, the effects of noise on the 2H-HG$_{\text{gauss}}$ approach will be evaluated. For this evaluation, a newly trained instance (200 epochs) of the respective network will be used.

Experiments will be performed on the same test split employed in previous sections. To evaluate the impact of noise, Gaussian noise will be used as exemplary noise type. Each test image will be augmented with it. To do this, an individual, random noise value for each pixel and color channel of the input image will be drawn from a 1D Gaussian distribution $N(0, \sigma^2)$. The noise value will be added to the value of the respective pixel and color channel, and the result will be clipped to the valid value range of $[0.0, 1.0]$ to produce a valid image. To evaluate the relation between the severity of noise and neural network performance, 17 different values σ_{test} for σ will be examined, uniformly sampled from the interval [0.0,0.8]. For each σ_{test} value, the whole test split will be augmented and the performance of all methods will be assessed.

The results for the unaltered 4-HG-V1 and HRNet-W32-V1 as well as the respective error detection methods can be found in Figure 6.1. It can be seen that the percentage of correct results decreases fast for all methods, with less than 40% correct results for noise values $\sigma_{test} \geq 0.3$ remaining. The amount of false results grows heavily for the unaltered 4-HG-V1 and HRNet-W32-V1 networks, but all methods for incorrect result detection manage to keep the amount low. However, none of the methods can be used effectively in practice when strong noise occurs, as almost no correct results remain. A qualitative example of how noise affects human pose estimation is shown in Figure 6.2, using 4-HG-V1 under

(a) No noise **(b)** Medium noise ($\sigma = 0.3$) **(c)** Strong noise ($\sigma = 0.8$)

Figure 6.2: Qualitative results for 4-HG-V1 under no, medium and strong noise. While the human pose estimation works well without noise, many results turn incorrect for medium noise. Under strong noise, result do not make any sense at all.



(a) Correct results **(b)** False results **(c)** Upper bound size

Figure 6.3: Experimental results for the 2H-HG$_{\text{gauss}}$ method under noise. The percentage of correct results (here measured with C-Rad from Eq. (5.22)) decreases slower than for methods from Figure 6.1, while an increase in incorrect results can be witnessed. For the average predicted upper bound size, a strong increase occurs as noise increases.

no, medium, and strong amounts of noise. Keypoint positions are detected correctly when no noise is applied to the input. Under medium noise, some predicted positions remain correct (e. g., head and shoulders), however, many others become incorrect. Under strong noise, the results do not represent the pose of the human at all.

Results for 2H-HG$_{\text{gauss}}$ are displayed in Figure 6.3. In contrast to the other methods, this method is capable of retaining a significantly higher percentage of correct results with regards to its own correctness definition C-Rad. For example, for $\sigma_{test} = 0.8$, over 50% correct results are retained, while all methods displayed in Figure 6.1 fall below 20% correct results for $\sigma_{test} \geq 0.5$. However, this effect comes at the cost of a strong increase of the average predicted upper bound size, which almost tripled from about 4.1 pixels for no noise ($\sigma_{test} = 0.0$) to about 11.5 pixels under strong noise ($\sigma_{test} = 0.8$). High upper bounds indicate that the neural network is not very sure where the keypoint is located, with 11.5 pixels relating to a sphere with a diameter of 23 pixels for potential keypoint locations. This is a massive area and thus not very useful given the heatmap resolution of 64×64 pixels. Furthermore, a notable increase in false results can be witnessed. This is the case, as the 2H-HG$_{\text{gauss}}$ features no dedicated method against incorrect results. Only in case of the heatmap maximum being below 0.02, results are eliminated and thus not counted towards correct or false, as upper bound prediction was not trained for such values. Overall, the 2H-HG$_{\text{gauss}}$ method is also highly affected by noise in a negative way.

6.2 Problem Definition under Limitations from Safety Standards

Before a formal problem definition for human pose estimation under noise in safety-critical industrial applications will be given, the impact of safety standards on this problem will be discussed first. As first highlighted by Wang et al. [129] and confirmed by the experiments in Section 6.1, human pose estimation based on neural networks is highly prone to noise. Without additional measures, high amounts of false results are produced in the presence of medium to strong noise as shown in Figure 6.1. The incorrect localization of keypoints caused by noise is a threat to safety when they are used for SSM, as this can lead to the calculation of an incorrect distance between human and robot. Thus, the problem must be dealt with. Whether noise can occur or not depends on external, environmental factors. As safety standards highly regulate the operation environment of machines, they will affect both: the kinds of noise that can occur, and how they must be dealt with.

A major limitation imposed by safety standards is that a machine does not have to work without safety-critical malfunctions under all possible operation conditions. Instead, the potential (environmental) conditions that have to be considered are confined by the so-called limits of the machinery [47]. They explicitly outline under which conditions the machine may be operated, in turn defining the conditions under which safety has to be achieved. This also means that only a fraction of all potential noise types can be encountered as the environmental prerequisites for others will not be fulfilled. For example, if operation under direct sunlight is prohibited (an example of limits from ISO 12100 [47]) and strong light sources may also not be present, then the overexposure noise type should not occur. As another example, if the machine may only be operated indoors, noise from e. g., snow and rain should not occur. From all noise types that can occur, only a subset will be safety-critical and require handling. But is this subset of safety-critical noise types known beforehand? To this end, the safety standard for industrial robotics, ISO 10218-2 [49], mandates that a risk assessment in accordance with ISO 12100 [47] has to take place before the final robot system may be put into service. This includes the identification of potential additional hazards (with unacceptable risk) that arise from selected safety measures for risk reduction. With respect to hazards, ISO 12100 [47] mandates that all of them should be documented (together with cause and effect). In turn, this means that specific kinds of noise that can lead to a malfunction of safety measures and hence to hazards should be documented and are thus known for a specific industrial robot application. Looking at the S3000 safety laser scanner [113] as a practical example of safety measures, noise from reflection is specifically considered. It is documented that reflective objects can cause incorrect measurements or can prevent that measurements are obtained. Countermeasures include that retro-reflectors may not be placed within 1m of the protective field monitored by the laser scanner. If this is not possible, a surcharge of 0.2m has to be applied to the protective field to account for measurement errors due to noise [113].

In conclusion, it can be assumed that all safety-critical noise types that can occur in an industrial application are known beforehand. This enables the use of *noise-specific countermeasures* for safety-critical industrial applications. However, this statement builds on the definition of the limits of the machinery as well as the risk assessment (including the identification of hazards), which have different results for individual industrial applications. Thus, different safety-critical noise types can occur in different applications, making a selection of noise-specific countermeasures on a case-to-case basis necessary.

For safety-critical applications, a human pose estimator should work correctly whether noise is present or not. Resistance to noise in that context means that a correct keypoint detection for an input image is not turned into an incorrect one if noise is added to the image. Let g denote a function to assess whether a predicted keypoint position is correct (output 1) or not (output 0). Further, let f denote the function realized by a neural network for human pose estimation that predicts keypoint positions, $\boldsymbol{x}$ a single input image, and $\tilde{\boldsymbol{\epsilon}}$ the noise that corrupts the clean input image $\boldsymbol{x}$ to form a noise-corrupted input image $\tilde{\boldsymbol{x}} = \boldsymbol{x} + \tilde{\boldsymbol{\epsilon}}$. Last, $\boldsymbol{y_i}$ denotes the ground truth keypoint position for the i-th keypoint. Then, the goal of *resistance to noise* can be formalized as follows:

$$g(f(\boldsymbol{x})_i, \boldsymbol{y_i}) = 1 \implies g(f(\tilde{\boldsymbol{x}})_i, \boldsymbol{y}_i) = 1 \tag{6.1}$$

Of course, it will not be possible to achieve this goal for all possible input images $\boldsymbol{x}$ and for all potential kinds of noise $\tilde{\boldsymbol{\epsilon}}$. Assume e. g., the extreme case of an $\tilde{\boldsymbol{\epsilon}}$ with very high values that represents the effects of extreme overexposure and turns the whole image white. Then, any image $\boldsymbol{x}$ could have been the clean, uncorrupted image, making it impossible to achieve the above goal. However, the formula should be fulfilled as often as possible, such that the remaining cases in which it is not can be handled through additional methods like those from Chapter 4 without a significant negative impact for practical usability. Apart from keypoint correctness, additional quality criteria, if present, should also be maintained at comparable levels (like a rather small upper bound in case of methods from Chapter 5). The noise $\tilde{\boldsymbol{\epsilon}}$ that was used to compute $\tilde{\boldsymbol{x}}$ was not further specified in Eq. (6.1). Due to the limitations of safety-critical industrial applications, $\tilde{\boldsymbol{\epsilon}}$ can only come from a fixed, previously known set of noise types for a specific application. It can either be an instance of one of these noise types, or a combination of multiple ones. As safety-critical noise types can vary between applications, a choice must be made for further investigations.

To enable a large-scale comparison for different methods and experimental settings, Gaussian noise will be used as an exemplary noise type, as it is highly common in image-based applications (e. g., due to low environmental illumination). Throughout this work, two different kinds of Gaussian noise will be investigated. They will be called *channel-diverse Gaussian noise* G_{div} and *channel-identical Gaussian noise* G_{id}. In both cases, the foundation for sampled noise values will be a Gaussian distribution $N(0, \sigma^2)$, and noise values will be sampled from this distribution for each pixel location of an image independently. However, in case of G_{div}, three independent noise values will be sampled per pixel (one for each color channel), while only one noise value per pixel will be sampled and subsequently applied to all color channels in case of G_{id}. This means G_{id} can be seen as a special case of G_{div}, where the sampled noise values for the color channels are the same. This will allow the investigation of how minor changes in the noise type impact different measures against noise. In any case, only valid images will be examined in this work, which means that the image values (ranging from 0.0 to 1.0) will be clipped back to the valid image value range after noise has been applied. For a mathematical description, let $\tilde{\boldsymbol{\epsilon}}_{\boldsymbol{div,u,v}} = (\tilde{\boldsymbol{\epsilon}}_{\boldsymbol{div,u,v,1}}, \tilde{\boldsymbol{\epsilon}}_{\boldsymbol{div,u,v,2}}, \tilde{\boldsymbol{\epsilon}}_{\boldsymbol{div,u,v,3}}) \in \mathbb{R}^3$ denote a G_{div} sample for pixel location (u, v), and $\tilde{\boldsymbol{\epsilon}}_{\boldsymbol{id,u,v}} = (\tilde{\boldsymbol{\epsilon}}_{\boldsymbol{id,u,v,1}}, \tilde{\boldsymbol{\epsilon}}_{\boldsymbol{id,u,v,1}}, \tilde{\boldsymbol{\epsilon}}_{\boldsymbol{id,u,v,1}})$ with $\tilde{\boldsymbol{\epsilon}}_{\boldsymbol{id,u,v,1}} \in \mathbb{R}^1$ the same for G_{id}. Further, let clip denote a function that clips pixel values to the valid image value range. Then, the augmentation process can be written as follows:

$$\begin{aligned} \tilde{\boldsymbol{x}}_{\boldsymbol{div}}[u, v] &= \text{clip}(\boldsymbol{x}[u, v] + (\tilde{\boldsymbol{\epsilon}}_{\boldsymbol{div,u,v,1}}, \tilde{\boldsymbol{\epsilon}}_{\boldsymbol{div,u,v,2}}, \tilde{\boldsymbol{\epsilon}}_{\boldsymbol{div,u,v,3}})) \\ \tilde{\boldsymbol{x}}_{\boldsymbol{id}}[u, v] &= \text{clip}(\boldsymbol{x}[u, v] + (\tilde{\boldsymbol{\epsilon}}_{\boldsymbol{id,u,v,1}}, \tilde{\boldsymbol{\epsilon}}_{\boldsymbol{id,u,v,1}}, \tilde{\boldsymbol{\epsilon}}_{\boldsymbol{id,u,v,1}})) \end{aligned} \tag{6.2}$$

In the following, approaches against noise in human pose estimation will be defined and examined based on these two noise types and the described augmentation process.

6.3 Discussion of Potential Solutions

To deal with noise in human pose estimation, two general strategies will be discussed. The first will be to make the neural network for human pose estimation itself robust against noise. This would allow the use of the neural network in an environment with noisy data without the need for additional, external measures against noise. Second will be the use of a dedicated image denoiser against noise. By eliminating noise from the input image, the overall problem could potentially be solved without changes to the human pose estimator itself. In the following, both strategies will be discussed in greater detail.

Neural Network Robustness against Noise: The goal of this strategy is to make the neural network for human pose estimation itself robust against noise. Outside of human pose estimation, there are some works that discuss this problem for image classification. Approaches from that domain tend to focus on the training of the neural network to achieve robustness against noise. One common strategy is to add noise to the inputs during training, be it to increase robustness against artificial noise in the form of adversarial attacks targeted to make neural networks malfunction [39, 77] or to increase robustness against less targeted kinds of noise [38], which is what is required here. Other training-based strategies include the use of an altered loss function [148] or the use of model noise (amongst other strategies) [135] during training. In the field of human pose estimation, the work of Wang et al. [129] also employed a strategy for corrupting input images with noise during training based on an image augmentation strategy called AdvMix (for details see Section 3.1.4). This strategy aims at increasing robustness against previously unseen kinds of noise. For this work, following the common strategy of adding noise during training to achieve robustness against noise seems reasonable. First, it is broadly applicable, as no changes to neural network architectures or loss functions are necessary. Second, its typical points of criticism do only partially apply. They include (i) that training with one kind of noise does not necessarily increase robustness against other, unseen kinds of noise [38, 77, 129], and (ii) that performance on clean data can tank due to training with noisy input images [38, 129]. Safety-critical applications should not be affected by problem (i), as all kinds of noise that can occur and require handling are known and thus can be used during training. Only (ii) could be a problem, hence the clean data performance will have to be specifically considered when employing such a training strategy.

Image Denoising against Noise: The second strategy is the straightforward idea of resolving the problem of noise before the human pose estimator is applied by reconstructing the original, uncorrupted image through the use of a dedicated image denoiser. Image denoising itself is a broad research area on its own (see Tian et al. [120] for an overview), which features a variety of established methods to deal with noise. These existing methods can be leveraged in a pipeline with a human pose estimator to limit the impact of noise. A potential problem of this approach is that existing methods for image denoising do not reconstruct the original image perfectly and can e. g., introduce their own noise by eliminating details of the original image, resulting in a blur-like effect (see example images of Tian et al. [120]). If and how human pose estimation is affected by this has to be investigated, with the effect potentially prohibiting the direct use of a human pose estimator together with an image denoiser without retraining to the new, denoised images.

Both strategies show the potential to solve the problem of noise in safety-critical industrial applications. Making a human pose estimator itself robust against noise through *training* has the advantage that no additional components are required after the training is finished.

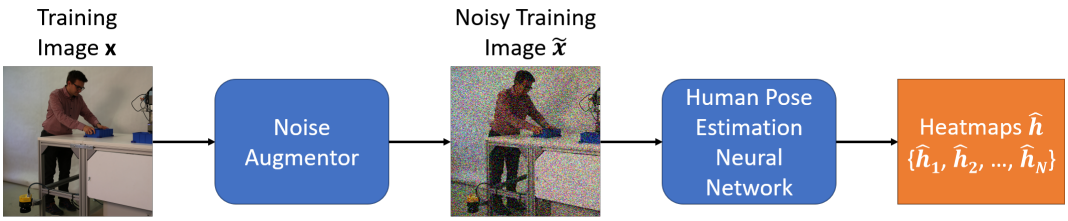


Figure 6.4: A pipeline that depicts the noise augmentation performed during training. Instead of being used directly, a clean input image is first corrupted with noise by a noise augmentor. Only then it is processed by the neural network for human pose estimation.

Downsides of this approach include that a loss of correct results on clean data is possible and that out-of-the-box use of existing human pose estimators is not possible as new training is required. Using a *dedicated denoiser* to deal with noise also has advantages and disadvantages. The strategy separates the task of dealing with noise from human pose estimation itself, allowing two separate methods to focus on only one of both problems. This could potentially allow the use of existing human pose estimators without retraining. For image denoising, existing methods from this field can be leveraged. On the downside, it is not guaranteed that human pose estimators will work on denoised images out-of-the-box due to imperfect reconstruction. Furthermore, additional computational resources are required at inference time for the image denoiser. In the following, realizations of both strategies will be introduced, and an extensive experimental evaluation will be performed.

6.4 Training Human Pose Estimators against Noise

The first discussed strategy was to train a neural network for human pose estimation to be robust against noise. This only requires changes to the training and allows the use of the neural network without any modifications during inference. The required changes for training are depicted in the pipeline from Figure 6.4. As pipeline input, a clean image is used. Such images can come from large-scale datasets like the MPII Human Pose dataset. Next, it is fed into a noise augmentor that corrupts it with noise, resulting in a noisy input image for the human pose estimation method. The neural network for human pose estimation predicts results based on this noisy image, which are heatmaps in the depicted pipeline. Then, neural network training can be performed based on this output with standard loss functions, like the mean squared error between predicted and ground truth heatmaps. At that point, there is no difference from training without noise.

If other elements are additionally used during training, like further image augmentations, then they remain unchanged. The only difference to standard training is the use of the noise augmentor, which is responsible for achieving robustness against noise. As experiments will be performed for the previously defined Gaussian noise variants G_{div} and G_{id}, the augmentation approach will be discussed for these noise types. One of them shall be used exclusively for each training, such that experiments can be performed for a single exemplary noise type. Every time an input image $\boldsymbol{x}$ is processed during training, an individual noise pattern $\tilde{\boldsymbol{\epsilon}}$ shall be sampled, such that a large variety of different noise patterns can be encountered during training. Each individual value from the noise pattern shall be drawn from a Gaussian distribution $N(0, \sigma^2_{train})$ – in case of G_{div} three values per pixel and in case of G_{id} only one. To increase the difference in observable noise patterns during

Figure 6.5: Depiction of the pipeline used after training when a dedicated image denoiser is employed. A (potentially) noisy test image is first denoised by the image denoiser. The resulting denoised image serves as input for the human pose estimation neural network.

training, σ_{train} will not always be the same. Every time an image $\boldsymbol{x}$ is processed during training, a new value for σ_{train} will be randomly determined from an interval σ_{range} of potential values to define $N(0, \sigma^2_{train})$. With this procedure, the strength of the noise will be the same during the augmentation of a single input image, but different otherwise.

The last factor to be considered is the potential decrease in clean data performance due to data augmentation with noise during training. To strengthen the clean data performance, only a certain percentage of the input data shall be augmented with noise, retaining some clean data to be part of the training – a strategy that successfully prevented performance loss on clean data in image classification [38]. To augment only a certain percentage of input data, a value a between 0.0 and 1.0 will be sampled from a continuous uniform distribution $U_{[0,1]}$ every time an input image $\boldsymbol{x}$ is encountered. If $a \leq c_{aug}$, then the image will be augmented. Mathematically, the whole augmentation process during training can be described as follows, with $\tilde{\boldsymbol{\epsilon}} \sim N(0, \sigma^2_{train})$ denoting that the individual noise values for a whole image were individually drawn from distribution $N(0, \sigma^2_{train})$ and $U_{\sigma_{range}}$ indicating the continuous uniform distribution for the value interval σ_{range}:

$$\begin{gathered} a \sim U_{[0,1]}, \quad \sigma_{train} \sim U_{\sigma_{range}}, \quad \tilde{\boldsymbol{\epsilon}} \sim N(0, \sigma^2_{train}) \\ \tilde{\boldsymbol{x}} = \text{clip}(\boldsymbol{x} + (\delta(a \leq c_{aug}) \cdot \tilde{\boldsymbol{\epsilon}})) \end{gathered} \tag{6.3}$$

How the values for $\tilde{\boldsymbol{\epsilon}}$ get sampled from $N(0, \sigma^2)$ depends on whether G_{div} or G_{id} is used. The above equation shows how each training sample in the form of image $\boldsymbol{x}$ is individually processed whenever it is encountered. How well this strategy works, e. g., for different values of c_{aug}, will be subject to the experimental section.

6.5 Human Pose Estimation with Denoisers

The second discussed strategy was to use a dedicated image denoiser together with a neural network for human pose estimation. This allows to solve the problems induced by noise separately from the actual human pose estimation. The resulting pipeline is depicted in Figure 6.5. Every input image is processed by an image denoiser first, attempting to remove noise from the image. This is always done whether the image is noisy or clean, as a different treatment of those would require an additional method to determine whether noise is present or not. Then, the denoised image is used as input for the human pose estimator which produces heatmaps in the depicted case. This straightforward procedure leaves two questions open: (i) which denoising approaches shall be used and (ii) how shall the training of the human pose estimator be pursued.

Regarding question (i), potentially any denoising approach can be applied, as long as its denoising capability is strong enough. Furthermore, the denoising approach should ideally not be too computationally expensive, as the denoiser is meant to be a supplement to the human pose estimator and not the primary source of computational cost. For this work, two different denoising approaches will be investigated: BM3D [17] and FFDNet [145]. BM3D is a traditional, well-established method for image denoising that does not rely on machine learning. Yet, it still achieves denoising results close to the quality of modern neural-network-based approaches on different benchmarks [120, 145]. Evaluating such a proven and still competitive method seems a good addition to modern machine-learning-based approaches. As a representative of modern machine-learning-based approaches, FFDNet will be explored. It is a small and fast neural network for image denoising with competitive performance [120, 145], which makes it an ideal choice for the proposed pipeline. Both BM3D and FFDNet have a parameter that controls the denoising strength of the respective approach. This parameter represents the expected standard deviation of the noise and will thus be called σ_{set} for both approaches. As the pipeline does not feature additional networks to determine the strength of the noise present in a given input, fixed values for σ_{set} independent of the current input will be explored during experiments.

This procedure directly ties into question (ii) of how training of the human pose estimator shall be pursued. An always active denoiser with fixed denoising strength means that alterations in the input image due to the denoising process will always occur, which e. g., means a loss of image details as shown by qualitative evaluations from Tian et al. [120]. In theory, a pretrained neural network for human pose estimation without previous exposure to denoised images could be used, however, the effects of image denoising could affect the neural network's performance. Alternatively, the neural network could be exposed to denoised images during training by applying the image denoiser to all training images. Both of these ideas will be pursued, however, during training, only clean training images will be denoised to expose the human pose estimator to the effects of the denoising process, but not to the noise itself. This has the goal of retaining a strong enough difference between the two strategies of training against noise and using a denoiser against noise.

6.6 Experiments

Throughout the experimental section, the effectiveness of the suggested strategies against noise in 2D single-person human pose estimation shall be evaluated. To this end, the following three questions shall be answered:

1. Are the proposed methods against specific kinds of noise capable of reducing the negative impact of noise on human pose estimation under the assumption that they are faced with a noise type they were designed against? If yes, to what extent can the negative impact of noise be reduced?
2. How well do the noise-specific countermeasures generalize to previously unknown, yet very similar noise types? While bad generalization for large differences in noise types is a common phenomenon [38, 129], small differences as e. g., between G_{div} and G_{id} are worth an investigation.
3. Section 6.1 highlighted the negative impact of noise on the proposed methods for error detection and measurement error upper bound prediction. Can this negative impact be reduced as well?

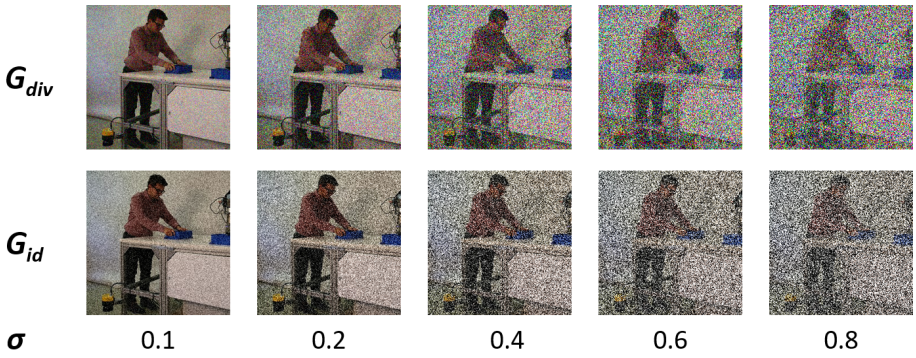


Figure 6.6: The effect of G_{div} and G_{id} on an image for different levels of noise that are defined by the magnitude of the standard deviation σ. Based on author's figure from [108].

First and foremost, the negative impact of noise that was demonstrated in Section 6.1 for standard human pose estimation and adaptations from this work was the loss of correct detections for all methods, and the increase of false detections and/or upper bounds for some of the methods. For standard human pose estimation, the percentage of false results is implicitly defined by the percentage of correct results. In this case, it is thus sufficient to only monitor the percentage of correct results to get a full picture. For all methods that can label a keypoint prediction as unreliable, it is necessary to monitor both, correct and false results, as outputs can also fall in the third Uncertain category. In the case of upper bound prediction, a third value that can be monitored to estimate the negative impact is the average size of the predicted upper bound values Avg. $\hat{\boldsymbol{r}}'$, as high values for this metric are always indicative of bad result quality (with respect to Q_{rad} and/or Q_{pos}).

Evaluations will be performed for a large number of experimental settings, each of them differing in the dimensions of how the test split is augmented, which countermeasures against noise are used, and how the parameters of the approaches are configured. In any case, the same test split for the MPII Human Pose dataset as in previous sections will serve as the foundation for all experiments. For (almost) each experimental setting, the test split will be augmented with noise. For the augmentation with noise, the following variables will be explored in different experimental settings:

1. Noise strength: $\sigma_{test} \in \{0.0, 0.05, 0.1, ..., 0.8\}$
2. Noise type: G_{div} or G_{id}

Specifically, this means that per experimental setup, only one noise type will be used to augment all test samples and that only one noise strength represented through σ_{test} will be employed to form the underlying Gaussian distribution $N(0, \sigma_{test}^2)$ from which individual noise samples are drawn. These settings allow to investigate how well different strategies fare for both, different noise types and strengths of the noise present in images. Figure 6.6 illustrates how both noise types affect images given different noise strengths.

Variable settings will not only be explored for the augmentation of the test split, but also for the countermeasures against noise. In the case of approaches relying on training procedures against noise, these variable factors include:

1. Noise type used during training: G_{div} or G_{id}
2. Chance for noise augmentation during training: $c_{aug} \in \{0.5, 1.0\}$ (50%/100%)
3. Range of noise strength during training: $\sigma_{range} = [0.0, 0.75]$

This means that training in one specific experimental setting will be performed using either G_{div} or G_{id} only, with either 50% or 100% of training samples being augmented. In every experimental setting, values for σ_{train} that are used to form the underlying Gaussian distribution $N(0, \sigma_{train}^2)$ for creating the training noise will be drawn from the same $\sigma_{range} = [0.0, 0.75]$ (individually for every training sample and epoch). This means that the neural networks are always exposed to different noise strengths within σ_{range}. The range ends at 0.75, such that a slightly stronger noise of $\sigma_{test} = 0.8$ can occur during tests, showing the performance for a previously unseen noise strength.

Not only do the training-based approaches feature variable settings, but the approaches that rely on a dedicated denoiser do so as well. For different experimental settings, the following variable choices will be explored:

1. Expected noise strength: $\sigma_{set} \in \{0.15, 0.3, 0.7\}$
2. Type of Denoiser: FFDNet or BM3D
3. Training data of human pose estimator: clean or clean-denoised (clean-den.)

This means that one specific denoiser with a single denoising strength σ_{set} will be used per experimental setting. A further difference is described by the last point: whether the human pose estimator was trained on the clean training images of the MPII Human Pose dataset or if these clean training images were passed through the denoiser before being used for training. In the latter case, the same σ_{set} is used during training and evaluation.

Independent from the specific experimental setting, a lightweight variant of the original hourglass model by Newell et al. [86] will be used. This variant will feature two consecutive hourglass blocks instead of the original eight, and will thus be called 2-HG. The reduction in model size is done to speed up the training and testing process, which is necessary due to the large number of combinations in experimental settings that are explored. The training of the 2-HG is pursued on the same training split as in previous sections and in a similar fashion to previous experiments where variants of the hourglass model were involved. The same data augmentation procedure proposed by Newell et al. [86] for training data is used, including image flipping, rotation, and image cropping with modified crop sizes. Inputs of the neural network are rescaled to the fixed input size of 256×256. Ground truth heatmaps are produced with the same procedure as Newell et al. did, and training of the neural network is pursued with intermediate supervision, using the mean squared error between predicted and ground truth heatmaps as the loss function. Batches of 16 samples each are processed, and RMSprop [122] is used with a learning rate of 0.001 and gradient clipping. Every training procedure is performed for 200 epochs. Depending on the specific experimental setup, additional alterations to the training process may take place (corruption of input images with noise/denoising of input images).

Apart from the human pose estimation method, the image denoiser FFDNet also requires configuration and training. In accordance with the suggestions from the authors [145], FFDNet will be configured to use 12 convolutional layers with a feature dimension of 96 for the initial 11 layers and a feature dimension of 12 for the last layer. In contrast to the original, training is performed on this work's training split for MPII, omitting other data sources that were used for training the original FFDNet. As for the training procedure, an altered, slightly simplified version is employed. Similar to the human pose estimator, FFDNet is trained with the annotated image crops that were rescaled to a size 256×256 pixels, however, no augmentations in the form of image flipping, rotation, or the adjustment of the crop size are performed. The only augmentation is the corruption of the input

with noise. Depending on whether FFDNet is trained to remove G_{div} or G_{id}, every training image is corrupted with the respective noise type. For every input image and epoch, an individual noise strength σ_{train} is drawn from the interval $[0.0, 0.75]$. This value is used to determine the underlying Gaussian distribution $N(0, \sigma^2_{train})$ from which the individual noise values for the image are drawn, and is also used to calculate the noise level map required by FFDNet (all value are set to σ_{train}). As loss function, the mean squared error between the original image (before adding noise) and the denoised image from FFDNet is used. Training is pursued for 50 epochs, using the RMSprop optimizer with a learning rate of 0.0001 and a batch size of 16. During tests, all values of the noise level map will be set to the respective value of σ_{set} in the specific experimental setting.

The image denoiser BM3D does not require training. Throughout the following experiments, the publicly available[1], GPU-accelerated version of the algorithm proposed by Honzátko and Kruliš [43] will be used. On a technical note, this implementation operates on image values ranging from 0 to 255, hence σ_{set}-values that are originally defined for an image value range of 0.0 to 1.0 have to be scaled accordingly. While this is what happens in practice, σ_{set} will always be defined for an image value range of 0.0 to 1.0 in this work, for the sake of simplicity and a coherent visualization across approaches.

6.6.1 Evaluation for Standard Human Pose Estimation

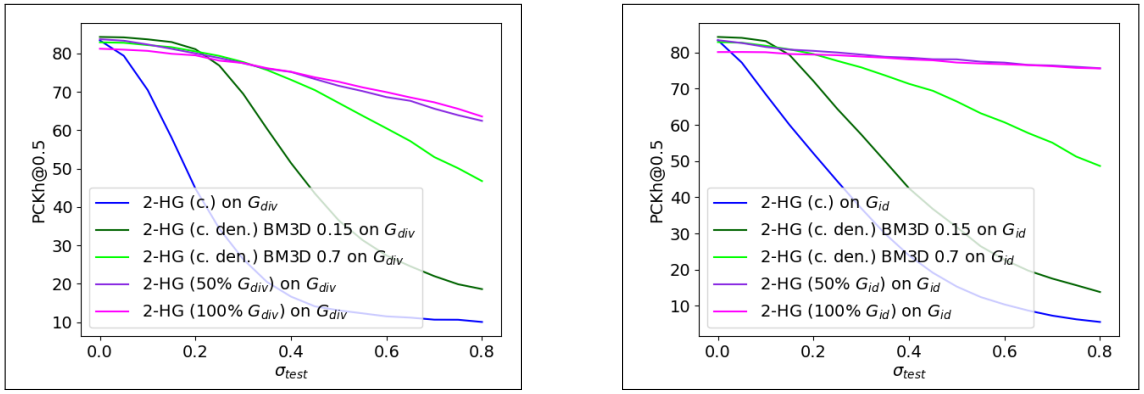
First, the impact of the proposed methods against noise will be evaluated for the standard 2D single-person human pose estimation problem. To this end, the percentage of correct detections will be evaluated on the test split of MPII (as defined in Section 4.5) using the standard PCKh@0.5 score. The 2-HG model will be used as the human pose estimator for all experiments. A large variety of experimental scenarios will be evaluated, consisting of different combinations of the variable settings described above. Different settings for both training- and denoiser-based approaches will be evaluated against the same target noise they were designed for, either G_{div} or G_{id}, for 17 variable levels of noise represented by σ_{test}. Values for σ_{test} start at 0.0 (meaning no noise) and are increased in steps of 0.05 up to 0.8 (strongest noise level). Whenever training takes place to deal specifically with G_{div} or G_{id} (training-based approaches and FFDNet denoiser), an evaluation against the other kind of noise is performed as well to highlight if the method generalizes well for small changes in the noise type. Overall, this leads to 782 experimental scenarios being evaluated. Table 6.1 depicts the evaluated scenarios together with the respective results (due to limited space, results for $\sigma_{test} \in \{0.65, 0.75\}$ are not displayed).

Looking at the experimental results in the table, even without going into detail, it becomes directly obvious that all investigated approaches lead to improvements when used against the noise type they were designed against (first and third block in the table). In these blocks, all methods improved the results for noise levels $\sigma_{test} \geq 0.2$ compared to the baseline model without countermeasures. With respect to performance on clean data and low noise settings, performance decreases of varying degrees can be observed for several methods. However, this is not true for all methods, as BM3D with $\sigma_{set} = 0.15/0.3$ as well as training with 50% noisy data either retain the performance (no. 37) on clean data or even slightly outperform (no. 11, 12, 14, 34 and 35) the 2-HG model without countermeasures on clean data. For the training-based approaches against noise, the decrease

[1] available at `https://github.com/DawyD/bm3d-gpu`

Experimental Setup						PCKh@0.5 Score for Noise Strength σ_{test}															
No.	Noise Type Test	Noise Type Train	2-HG Train Data	Denoiser	σ_{set}	σ_{test} 0.0	σ_{test} 0.05	σ_{test} 0.1	σ_{test} 0.15	σ_{test} 0.2	σ_{test} 0.25	σ_{test} 0.3	σ_{test} 0.35	σ_{test} 0.4	σ_{test} 0.45	σ_{test} 0.5	σ_{test} 0.55	σ_{test} 0.6	σ_{test} 0.7	σ_{test} 0.8
1		-	clean	-	-	83.4	79.4	70.5	58.1	44.8	34.4	26.4	20.5	16.7	14.2	13.1	12.3	11.5	10.7	10.1
2		G_{div}	clean	FFDNet	0.15	77.1	77.4	78.4	78.9	74.2	64.2	54.4	46.0	39.0	34.2	29.6	26.2	23.2	18.1	15.1
3		G_{div}	clean	FFDNet	0.3	70.7	71.1	71.6	72.3	73.0	74.0	73.0	69.3	62.0	53.4	45.1	39.0	34.1	26.3	21.0
4		G_{div}	clean	FFDNet	0.7	58.7	58.4	58.2	58.0	58.4	58.3	58.7	59.3	59.6	59.8	59.4	59.2	57.5	54.2	49.8
5		G_{div}	clean-den.	FFDNet	0.15	81.7	81.4	81.2	79.1	68.6	56.2	46.4	38.8	32.4	27.9	23.2	20.8	18.4	15.2	12.7
6		G_{div}	clean-den.	FFDNet	0.3	80.2	80.1	80.0	79.2	78.3	76.8	71.9	63.0	50.3	39.1	31.6	26.1	22.4	17.2	14.5
7		G_{div}	clean-den.	FFDNet	0.7	75.5	75.0	74.5	73.9	73.1	72.5	71.8	70.7	70.0	68.1	66.6	64.7	62.3	55.5	48.6
8	G_{div}	-	clean	BM3D	0.15	82.8	82.6	82.1	81.7	80.1	76.1	68.5	60.2	51.9	44.5	38.7	33.3	29.2	23.4	19.2
9		-	clean	BM3D	0.3	82.2	82.0	81.5	81.0	80.1	79.2	77.7	76.7	74.3	71.7	68.6	65.8	62.0	52.9	44.1
10		-	clean	BM3D	0.7	81.2	81.1	80.4	80.1	79.2	78.2	76.8	75.6	73.6	71.7	69.3	67.3	64.8	59.7	54.6
11		-	clean-den.	BM3D	0.15	**84.3**	**84.2**	**83.6**	**82.9**	81.1	76.8	69.5	60.3	51.5	43.5	36.5	31.4	27.4	22.0	18.6
12		-	clean-den.	BM3D	0.3	83.8	83.5	83.2	82.7	**81.6**	**80.5**	**79.5**	**77.9**	**75.3**	72.3	68.4	64.6	60.8	52.3	43.9
13		-	clean-den.	BM3D	0.7	82.9	82.8	82.2	81.6	80.5	79.3	77.7	75.7	73.1	70.4	67.1	63.7	60.5	53.0	46.8
14		G_{div}	50% noisy	-	-	83.7	83.3	82.3	81.2	80.0	78.8	77.4	76.0	75.2	73.3	71.6	70.2	68.6	65.6	62.4
15		G_{div}	100% noisy	-	-	81.2	81.0	80.6	79.9	79.5	78.1	77.5	76.1	75.2	**73.8**	**72.6**	**71.1**	**69.9**	**67.3**	**63.6**
1		-	clean	-	-	83.4	79.4	70.5	58.1	44.8	34.4	26.4	20.5	16.7	14.2	13.1	12.3	11.5	10.7	10.1
16		G_{id}	clean	FFDNet	0.15	76.9	76.6	**73.9**	**67.4**	**55.8**	**44.6**	35.9	28.7	23.8	20.6	17.7	15.8	14.3	12.6	11.8
17		G_{id}	clean	FFDNet	0.3	72.2	72.2	68.4	61.4	53.2	44.3	**37.7**	**31.1**	**25.2**	**20.8**	**18.4**	**16.1**	**14.8**	**12.9**	**11.9**
18		G_{id}	clean	FFDNet	0.7	63.3	61.6	58.2	50.8	43.8	37.2	31.4	26.7	22.7	19.6	17.0	15.7	13.9	12.1	10.9
19	G_{div}	G_{id}	clean-den.	FFDNet	0.15	81.3	79.2	71.8	60.3	45.0	32.2	22.6	16.2	12.5	10.4	8.9	7.9	7.7	7.0	6.7
20		G_{id}	clean-den.	FFDNet	0.3	80.3	77.3	66.8	52.0	36.6	24.9	17.0	12.6	10.1	8.7	7.7	7.4	6.7	6.4	6.0
21		G_{id}	clean-den.	FFDNet	0.7	78.7	75.8	64.8	51.3	38.8	29.0	22.7	18.7	16.2	14.3	12.9	12.0	11.4	10.8	10.4
22		G_{id}	50% noisy	-	-	**83.4**	**81.5**	73.4	58.0	39.4	23.2	13.9	9.8	7.6	6.8	6.2	6.2	5.9	5.8	5.4
23		G_{id}	100% noisy	-	-	80.1	76.4	64.7	48.2	32.7	22.3	15.8	12.3	10.8	9.8	8.7	8.2	8.0	7.4	6.9
24		-	clean	-	-	83.4	77.2	68.5	60.0	52.1	44.4	37.0	29.9	23.9	19.2	15.3	12.5	10.4	7.3	5.6
25		G_{id}	clean	FFDNet	0.15	76.9	77.4	78.0	78.7	72.9	60.3	49.5	41.7	34.6	29.5	24.8	21.1	18.4	14.0	11.6
26		G_{id}	clean	FFDNet	0.3	72.2	72.7	73.4	73.8	74.5	75.2	75.0	72.8	66.7	57.0	47.7	38.3	31.7	22.1	16.1
27		G_{id}	clean	FFDNet	0.7	63.3	64.1	65.0	66.0	66.9	67.6	67.8	68.8	68.5	68.8	68.4	69.0	68.3	67.5	66.2
28		G_{id}	clean-den.	FFDNet	0.15	81.3	81.3	81.2	79.0	67.1	52.9	42.4	34.3	28.0	22.7	18.9	16.1	14.4	11.9	10.7
29		G_{id}	clean-den.	FFDNet	0.3	80.3	80.3	80.4	80.3	80.2	79.3	77.4	71.8	59.7	45.2	33.7	26.2	21.8	16.7	14.3
30		G_{id}	clean-den.	FFDNet	0.7	78.7	78.7	78.4	78.2	78.0	78.0	77.6	77.0	76.9	76.5	76.2	75.6	75.5	73.8	72.1
31	G_{id}	-	clean	BM3D	0.15	82.7	82.6	81.8	78.1	69.5	60.7	52.8	45.7	39.3	33.1	28.0	23.9	20.4	15.6	12.5
32		-	clean	BM3D	0.3	82.2	82.1	81.2	80.5	78.9	76.9	73.5	66.9	58.7	50.2	42.6	35.9	30.7	23.2	18.1
33		-	clean	BM3D	0.7	81.2	81.0	80.2	79.6	78.2	76.5	75.1	73.1	70.5	68.5	66.0	63.1	60.6	54.3	48.7
34		-	clean-den.	BM3D	0.15	**84.3**	**84.0**	**83.1**	79.3	72.1	64.4	57.3	49.9	42.5	36.8	31.8	26.5	22.8	17.5	13.8
35		-	clean-den.	BM3D	0.3	83.8	83.6	82.9	**81.8**	**80.4**	78.0	74.2	68.7	63.1	57.0	52.0	47.4	43.0	35.7	30.0
36		-	clean-den.	BM3D	0.7	82.9	82.7	81.8	80.8	79.6	77.7	75.9	73.6	71.3	69.4	66.5	63.2	60.7	55.1	48.6
37		G_{id}	50% noisy	-	-	83.4	82.6	81.5	80.8	**80.4**	**80.0**	**79.4**	**78.8**	**78.5**	**78.1**	**78.1**	**77.4**	**77.2**	**76.4**	**75.6**
38		G_{id}	100% noisy	-	-	80.1	80.1	80.1	79.6	79.4	79.2	78.9	78.5	78.1	77.8	77.2	76.9	76.7	76.2	**75.6**
24		-	clean	-	-	83.4	77.2	68.5	60.0	52.1	44.4	37.0	29.9	23.9	19.2	15.3	12.5	10.4	7.3	5.6
39		G_{div}	clean	FFDNet	0.15	77.1	77.9	78.7	67.6	55.8	46.1	37.9	30.5	24.6	19.8	16.1	13.7	11.8	8.9	7.4
40		G_{div}	clean	FFDNet	0.3	70.7	71.2	72.2	73.5	71.2	58.7	46.3	38.1	29.7	24.1	19.1	15.5	13.2	9.4	7.5
41		G_{div}	clean	FFDNet	0.7	58.7	58.4	58.1	57.9	58.6	59.2	58.6	55.7	51.2	44.8	37.8	30.2	23.8	13.9	9.2
42	G_{id}	G_{div}	clean-den.	FFDNet	0.15	81.7	81.2	77.3	62.1	51.5	43.5	36.5	30.4	25.1	20.7	18.0	15.5	13.5	10.7	8.8
43		G_{div}	clean-den.	FFDNet	0.3	80.2	80.1	79.3	76.5	65.1	46.1	33.7	26.1	20.7	16.4	13.2	11.3	10.2	9.0	8.1
44		G_{div}	clean-den.	FFDNet	0.7	75.5	75.1	74.3	73.3	71.7	68.7	62.6	51.6	39.1	28.5	21.1	15.5	12.1	9.6	8.6
45		G_{div}	50% noisy	-	-	**83.7**	**82.0**	79.3	75.4	71.1	66.3	60.4	55.4	49.9	44.5	38.8	33.3	28.5	20.5	15.3
46		G_{div}	100% noisy	-	-	81.2	80.8	**79.5**	**77.5**	**74.3**	**70.3**	**66.1**	**61.3**	**55.9**	**50.6**	**45.1**	**40.0**	**35.2**	**27.4**	**21.5**

Table 6.1: Experimental results together with the respective experimental setup for 15 out of 17 investigated noise strengths. PCKh@0.5 is used as evaluation metric. Results are grouped in four blocks: the first and second display experiments where tests are pursued using G_{div}. The first block displays experiments where training (if applicable) was performed on the same noise type, while the second block includes experiments where training was pursued with the other noise type G_{id} to show how well the training transfers to tests on G_{div}. Blocks three and four are similarly structured, this time with G_{id} as the noise type for tests. Results that are better than the respective baseline in grey (2-HG without countermeasures against noise) are marked in green, and worse results are marked in red. The best result in each block for each noise level σ_{test} is highlighted in bold numbers. Previously published by the author [108], © 2023 IEEE, slightly altered.

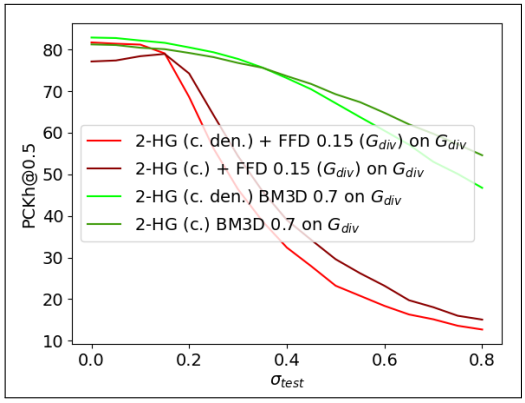


(a) Train G_{div}, test G_{div} **(b)** Train G_{id}, test G_{id}

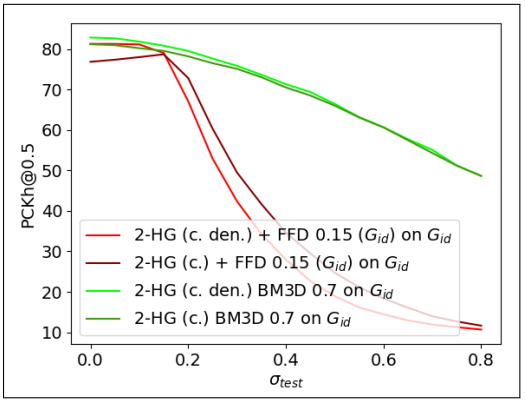
Figure 6.7: Plots of experimental results for methods against noise while testing on the same noise they were designed against. The respective experiment numbers in Table 6.1 are 1, 11, 13, 14 and 15 for the left as well as 24, 34, 36, 37 and 38 for the right plot. The first bracket in a method's name indicates on which data the 2-HG model was trained (c. for clean, c. den. for clean-denoised and X% G_{div}/G_{id} for an augmentation of X% of the training data with the specified noise type). The number after the name of a denoiser indicates which value for σ_{set} was used.

in clean data performance reported by Wang et al. [129] for training the neural network with the noise type that shall be handled during inference was confirmed when all training samples are augmented with noise (no. 15 and 38), however, the opposite is true when 50% noisy data is used (no. 22 and 37). This shows that not the training with the test noise type leads to a performance decrease on clean data by itself, but that it depends on the used training strategy. From the experimental results, it can be concluded that it is important to retain a percentage of clean training data when training with noise. Depending on the noise level, denoiser-based approaches where the human pose estimator was trained on clean-denoised images and training-based approaches fared best. Figure 6.7 shows a performance comparison for some of the best methods. Although all displayed methods highly improve the performance under noise, it can be seen that the denoiser-based approaches tend to tank for high levels of noise in comparison to the training-based approaches. The effect is more dramatic for low σ_{set} values, although these achieve better performance for scenarios with low levels of noise, making the choice of σ_{set} a trade-off.

The next question that can be answered based on the experimental results from the first and third block is whether neural networks for human pose estimation can be used together with a dedicated denoiser without (re)training them to adapt to the new, altered inputs. It can be seen from experiments no. 2-13 as well as 25-36, that the use of a human pose estimator trained on clean data is possible together with a dedicated denoiser to improve the results under noise, however, at a loss of clean data performance. In direct comparison, training the neural network based on image denoiser outputs for clean images leads to better performance for no and low levels of noise, but with an increasing noise level, the use of a human pose estimator trained on clean data becomes better in most cases. Figure 6.8 illustrates the effect. Although this behavior under high levels of noise seems surprising, it could potentially be explained through the fact that the neural network training with denoiser outputs was only performed for images without noise which were processed by the denoiser. Hence, the neural network was never tuned towards the denoiser behavior at high noise levels.



(a) Train G_{div}, test G_{div}



(b) Train G_{id}, test G_{id}

Figure 6.8: Plots of experimental results for the denoiser-based strategy that illustrate the effect of training the 2-HG model on clean denoised images from the denoiser vs. training it on clean images. With respect to Table 6.1, experiments 2, 5, 10, and 13 are depicted on the left as well as 25, 28, 33, and 36 on the right. The naming of methods in the plots is mostly the same as in Figure 6.7, with the difference that the bracket after FFDNet indicates which noise type was used to train this denoiser.

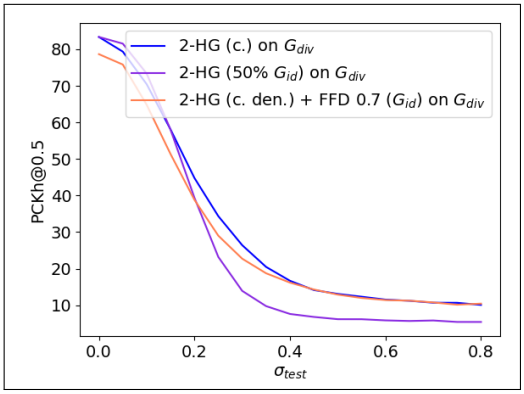
(a) Noisy, G_{div}, $\sigma = 0.1$

(b) Denoised, BM3D, $\sigma_{set} = 0.3$

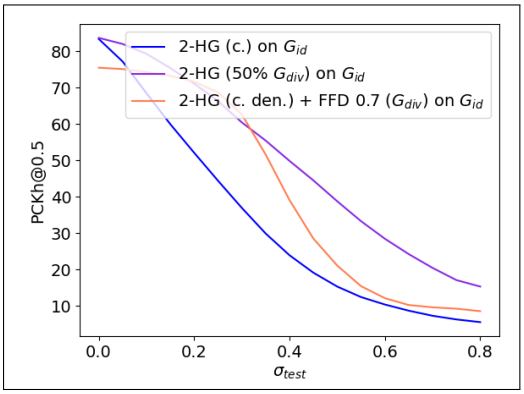
(c) Denoised, FFDNet, $\sigma_{set} = 0.3$

Figure 6.9: Visual comparison of the denoising results from BM3D and the custom-trained FFDNet with $\sigma_{set} = 0.3$ for an image corrupted with G_{div} of strength $\sigma = 0.1$.

Another effect that can be observed is that FFDNet performed significantly worse than BM3D in the first and third block, despite the original paper reporting comparable performance to BM3D [145]. This behavior must originate from the altered training procedure that is used in this work, as a simple visual inspection shows that FFDNet outputs are not near the quality of BM3D outputs in the experiments (see Figure 6.9). However, having a denoiser with a high loss in image details is not bad for the experiments: it allows to investigate how the denoiser-based strategy fairs when a lot of image details are lost in the denoising process. The effect is most severe for the highest denoiser strength $\sigma_{set} = 0.7$, and without adjusting the human pose estimator to the effects, large performance decreases can be seen when no or only low amounts of noise are present (no. 4 and 27). However, when the human pose estimator is trained with the denoised images, the negative effect can be limited (no. 7 and 30). This shows that the strategy can still work at a decent, although not ideal level, even if the denoiser eliminates many image details.



(a) Train G_{id}, test G_{div}



(b) Train G_{div}, test G_{id}

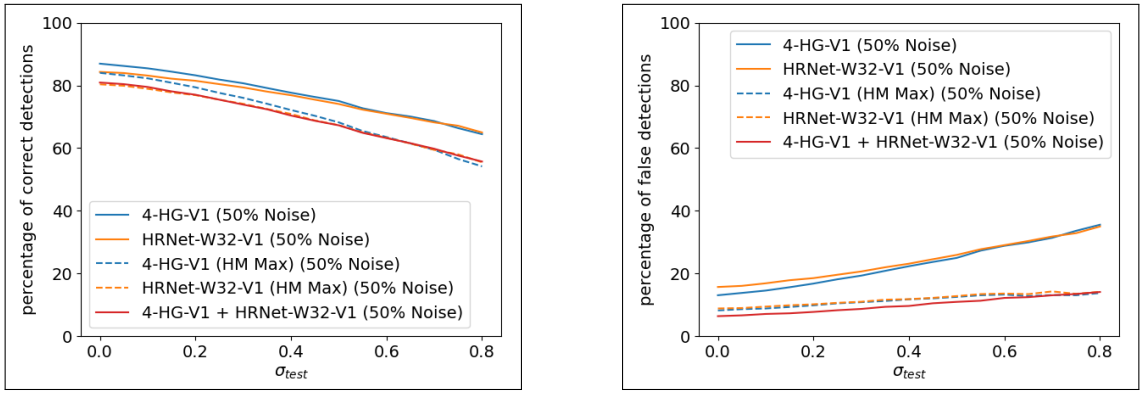
Figure 6.10: Plots of experimental results (left: 1, 21, 22, right: 24, 44, 45) that highlight the effect of changing the noise type between training and testing using G_{div} and G_{id}, compared to the baseline model. The methods do not generalize well.

The last question that can be answered from the results is how well the strategies generalize between G_{div} and G_{id}. This was evaluated in the second and fourth block for methods that relied on either G_{div} or G_{id} for training (including the FFDNet denoiser). Generalization is not good in both cases, with the performance for training on G_{id} and testing on G_{div} (second block) even tanking below the baseline without countermeasures in many cases. Figure 6.10 shows the generalization from G_{id} to G_{div} and vice versa for two methods. Performance losses are significant in all cases, especially compared to the same noise performance (Figure 6.7 and 6.8). This shows that generalization is not even guaranteed between similar noise types. In turn, hazardous noise types have to be precisely defined, e. g., during the risk assessment, when noise-specific countermeasures shall be used.

6.6.2 Impact on Further Methods

After the positive impact on standard human pose estimation has been verified for the proposed methods, the last remaining question is how they interact with the methods for error detection and measurement error upper bound prediction. To this end, the experiments from Section 6.1 will be repeated using the training-based strategy with 50% noisy data, as it was highly effective against noise while retaining or even improving clean data performance. Both training and testing are done with G_{div}, the same noise type used in Section 6.1. To apply the strategy of training with 50% noisy data, all networks will be trained from scratch. Apart from the addition of noisy data, the training remains the same.

First, the results of the error detection methods will be discussed. Figure 6.11 shows these results for the base methods 4-HG-V1 and HRNet-W32-V1, as well as for the respective error detection methods which either leverage the strategy of thresholding over the heatmap maximum or employ the diverse neural network ensemble. Compared to the results without countermeasure against noise (previously shown in Figure 6.1), significantly more correct results are retained for increasing levels of noise. This includes both, the baseline methods and all methods for error detection. The number of false results is significantly reduced for the baseline methods. All methods for error reduction experience a higher number of false results for higher levels of noise. However, the total amount of false results stays at a low level for the error detection methods, significantly lower than



(a) Correct results **(b)** False results

Figure 6.11: Evaluation of baseline models and error detection methods, when all of them are trained with 50% noisy data as a countermeasure against noise. Noise type G_{div} is used during training and evaluation.

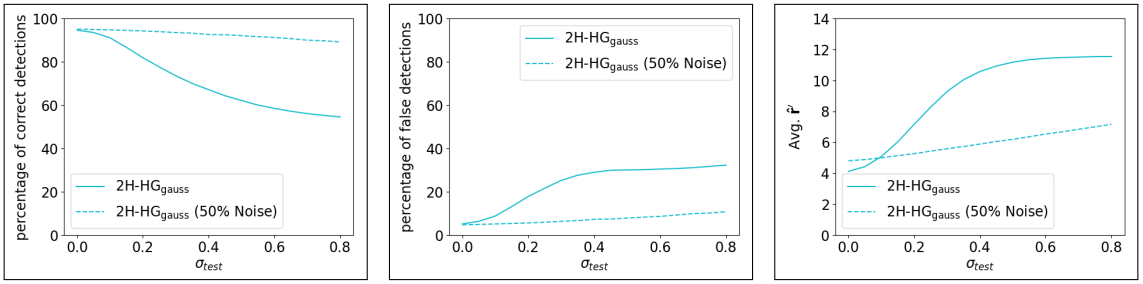
(a) No noise **(b)** Medium noise ($\sigma = 0.3$) **(c)** Strong noise ($\sigma = 0.8$)

Figure 6.12: Predicted keypoint positions and resulting pose for an input image with no/medium/strong noise. Keypoint predictions were produced by the 4-HG-V1 model, trained with 50% noisy data. Noise type G_{div} is used for training and testing.

without them. Furthermore, the lower amount of false results for high levels of noise that was observed without training on noisy data (see Figure 6.1) was not due to a selective elimination of false results, but due to the methods simply labeling almost all keypoints as unreliable. This means that the methods were basically unusable in practice for higher amounts of noise when no countermeasures were used.

Section 6.1 also featured a qualitative comparison of human poses predicted by 4-HG-V1 under different levels of noise. To give an impression of how the predictions changed on a qualitative level, the experiment is repeated, with 4-HG-V1 employing the strategy of training with 50% noisy data (G_{div} is used for both training and testing). The results can be found in Figure 6.12. Between no noise and medium noise, there is almost no difference in predicted keypoint positions. Even under strong noise, a subjectively good pose estimate is produced, with only the right wrist keypoint being a bit further away from where it should be located. However, locating this keypoint correctly is also challenging for humans due to the strong noise.

Next, the effect of training with 50% noisy data as a countermeasure against noise is investigated for measurement error upper bound prediction. Thus, the respective experiment from Section 6.1 is repeated with the addition of 50% noisy data (G_{div}) in the training



(a) Correct results **(b)** False results **(c)** Upper bound size

Figure 6.13: Comparison of experimental results for measurement error upper bound prediction, once without countermeasures against noise and once using training with 50% noisy data as a countermeasure. Noise type G_{div} is used for training and testing.

process. The results can be found in Figure 6.13. They can be described as almost completely positive: Correct results are only slightly reduced, even for high levels of noise, while false results also only slightly increase. In both cases, significant improvements are achieved for every noise level in comparison to not using a countermeasure. When it comes to the average upper bound size as an indicator of result quality, slightly worse results can be observed for no and low noise, however, for increasing levels of noise, significant improvements are achieved. Instead of growing similar to a logistic function, the growth in upper bound size is now more linearly correlated with the increasing noise level. Furthermore, the overall increase is significantly smaller, from approximately 4.8 pixels at $\sigma_{test} = 0.0$ to about 7.2 at $\sigma_{test} = 0.8$.

Overall, it can be said that training with 50% noisy data as a countermeasure against noise is capable of limiting the negative impact of noise on methods for error detection and measurement error upper bound prediction when training and testing noise types are the same. In the case of error detection, training with noisy data resolved the problem that almost all results were labeled as unreliable, which meant that almost no keypoint predictions remained under higher levels of noise for further use. On the downside, training with noisy data also resulted in (undetected) false results slightly increasing for higher levels of noise. Without countermeasure, this was not the case for high noise levels with $\sigma_{test} \geq 0.4$. This can probably be explained by human pose estimation results becoming completely non-sensical for large amounts of noise when no countermeasures are used (as in Figure 6.2), which in turn should make their detection and hence elimination easier. The fact that almost no detections (neither correct nor false) remained in that case when error detection methods were applied supports this claim. For measurement error upper bound prediction, no long discussion is needed: apart from a minor increase in upper bound size on clean data and for very low levels of noise, effects of training with 50% noisy data were positive through and through, making it a valuable addition for the method without significant drawbacks.

7 Human Pose Estimation and Distance Calculation in Hard Real-Time

In this chapter, the fourth central point of this thesis shall be examined: *How can hard real-time capability be ensured for human-robot distance monitoring in SSM based on human pose estimation results?* This question shall be discussed assuming that current human pose estimation methods are not hard real-time capable themselves. Concerning safety through SSM, hard real-time capability is the last major missing piece for safe human detection, when a sufficiently low probability for dangerous errors can be guaranteed and correct upper bounds for measurement errors are available. The requirement for a hard real-time capable human detection mechanism arises from the calculation of a minimum safety distance that has to be maintained between human and robot, which is the *minimum distance* S in ISO 13855 [46] and the *protective separation distance* $S_p(t_0)$ in ISO/TS 15066 [51] (see Section 2.2.3 and 2.2.4). As part of these distances, the (worst case) movement of the human towards the robot has to be considered for the maximum time it takes to detect the human, evaluate the distance, and then fully stop the robot. This means that both, the human detection and the human-robot distance calculation must be hard real-time capable, so that the respective maximum times for both tasks exist. Guaranteeing these maximum times for both tasks is crucial, as both S and $S_p(t_0)$ are designed to ensure that the robot is fully stopped before a human reaches it. Hence, missing deadlines imposed by the hard real-time requirement can lead to a collision between human and moving robot, which is a critical safety violation. To prevent such violations by the human detection process, two methods are required: one to ensure that a valid detection for every (required) keypoint is available with a fixed, guaranteed frequency, and another one to ensure that the keypoint-based human-robot distance calculation and evaluation against a minimum safety distance are performed within a maximum amount of time.

These challenges will be addressed for 3D human pose estimation instead of 2D, as (i) the calculation of a minimum safety distance relies on the distance in 3D space, and (ii) the full potential of human-robot distance calculations based on keypoints can only be leveraged in 3D. For an isolated investigation of hard real-time capability, a 3D human pose estimation system not capable of hard real-time will be assumed first, which is error-free through previously introduced measures. Next, the calculation of the protective separation distance will be introduced in greater detail for a formal problem definition and to highlight alternative paths to its fulfillment. Potential solutions to the problem are discussed afterward. Selected approaches are pursued further, creating a solution to the problem based on bridging the time in-between human pose estimation results, together with distance calculation based on a human volume model. The viability of the proposed solution is shown through a comparative theoretical analysis. Overall, the contents of this chapter are primarily based on previously published work of the author [107].

7.1 Assumptions for an Isolated Examination

First, the foundation for the research in this chapter has to be created by formalizing the properties of the required 3D human pose estimation system. The properties of the system should be formulated in a way that allows to separate the problem of achieving hard real-time capability from the other problem of achieving a low enough error rate for safety. This separation allows the application of different methods to solve these problems (mostly) independently from one another, although potential interactions will have to be considered. In the following, it will be assumed that the 3D human pose estimation system already has methods in place that reduce errors sufficiently. This means that the problem of achieving hard real-time capability can be pursued without need for further error reduction. However, employed methodology must not introduce new errors.

As no such 3D human pose estimation system exists to date, assumptions have to be made about how such a system could realistically look like. To ensure applicability in safety-critical scenarios at inference time, it will be assumed that the 3D human pose estimation system follows the proposed reformulation of the human pose estimation problem from Chapter 5, altered to the 3D case. This means it predicts 3D keypoint positions together with upper bounds for the measurement error, and that a result is correct if the predicted upper bound is larger or equal to the distance between predicted and ground truth position. As experiments from Section 5.6 never achieved 100% correct results, it would be unfair to assume that this methodology alone will be sufficient to avoid incorrect results completely. Thus it will be assumed that the 3D human pose estimation system employs a second method to eliminate incorrect results, e. g., like those from Chapter 4. It is assumed that this leads to a system without incorrect results, but with less than 100% correct results due to the elimination of individual, (potentially) incorrect results. In turn, not every keypoint will have a valid detection in every human pose estimation output. Furthermore, the 3D human pose estimation system will be assumed to be incapable of hard real-time, which is the problem that shall be solved. Summarized, the *assumed 3D human pose estimation system* will have the following properties:

1. Pairs of keypoint positions and upper bounds for the measurement error are predicted in 3D space, resulting in a uniform keypoint sphere per keypoint.
2. No incorrect results are produced by the human pose estimation method.
3. Not every output of the 3D human pose estimation system has a valid position and upper bound for each keypoint, as incorrect results are filtered out.
4. The 3D human pose estimation system is not capable of hard real-time.

The assumption of missing hard real-time capability is made as the focus of human pose estimation research does not lie on hard real-time capability, but on something that rather resembles *soft* real-time capability: the processing of a certain number of inputs (e. g., images) per second on average [11, 23, 61, 81, 82]. The difference to *hard* real-time capability is that no upper time limit for the availability of an individual result is guaranteed. With this differing understanding of real-time, it is unlikely that the considerable efforts necessary in both hardware and software for a hard real-time capable 3D human pose estimation system will be realized soon. Furthermore, calculating results in hard real-time would not be sufficient for the assumed error-free human pose estimation system, as incorrect results are filtered out and no guarantee can be given when the next valid result for a keypoint arrives, even if results are calculated within a fixed time span.

7.2 Problem Definition

Throughout this section, a formal definition for the problem of making the *human-robot distance calculation* based on human pose estimation *hard real-time capable* shall be given. To this end, let f_{hpe} denote the function that is realized by the 3D human pose estimator from Section 7.1, taking an input $\boldsymbol{x}$ and producing outputs $\hat{\boldsymbol{y}}, \hat{\boldsymbol{r}}$ that denote the predicted 3D keypoint positions and upper bounds for the measurement error of all valid keypoint detections. Furthermore, let f_{dist} denote the function for human-robot distance calculation, $\boldsymbol{D_{hpe}}$ the human pose estimation data that is supplied to that function, and $\boldsymbol{D_{dist}}$ other required data for the distance calculation (e. g., data reflecting the robot's position). The calculated human-robot distance is called $d_{h\text{-}r}$ and is calculated as follows:

$$\begin{aligned} \hat{\boldsymbol{y}}, \hat{\boldsymbol{r}} &= f_{hpe}(\boldsymbol{x}) \\ d_{h\text{-}r} &= f_{dist}(\boldsymbol{D_{hpe}}, \boldsymbol{D_{dist}}) \end{aligned} \tag{7.1}$$

For hard real-time capability, the distance calculation f_{dist} itself as well as the acquisition of required data $\boldsymbol{D_{hpe}}$ and $\boldsymbol{D_{dist}}$ must be hard real-time capable. This is the reason why the calculation of $\boldsymbol{D_{hpe}}$ is not defined closer. The simplest solution would be to use $(\hat{\boldsymbol{y}}, \hat{\boldsymbol{r}})$ directly as $\boldsymbol{D_{hpe}}$, however, this would require f_{hpe} to be hard real-time capable. Instead, $\boldsymbol{D_{hpe}}$ could also be obtained by other means, e. g., an adaptation mechanism that safely adapts previous human pose estimation results, hereby shifting the need for hard real-time capability from f_{hpe} to the adaptation mechanism. Thus, the given formulation opens up more potential solutions to the problem.

As an additional stipulation, the calculated distance $d_{h\text{-}r}$ shall consider all *measurement- and evaluation-related factors* from the protective separation distance (which is used as realization of a minimum safety distance) that account for the fact that the human can be closer to the robot than the measured distance. This change allows the use of new and potentially better solutions for SSM. Detailed reasoning will be given in the following.

The formula for the calculation of the protective separation distance $S_p(t_0)$ in accordance with ISO/TS 15066 [51] (see Eq. (2.2)) considers multiple factors: First is the (worst case) distance the human can traverse towards the robot before it is fully stopped (S_h) and the (worst case) distance the robot can move into the direction of the human (split into two parts, the distance S_s traversed while the robot is in the process of stopping and the distance S_r traversed before the stopping is initiated). Further factors include the distance C the human can be closer to the robot due to certain body parts being potentially not detected, and distance surcharges for the worst case measurement error in the human (Z_d) and robot (Z_r) position. All of these factors account for how much closer the human could be to the robot at the time the robot has been safely stopped compared to the measured human-robot distance if the measured distance takes none of these factors into account. It is further possible to split these factors or parts of them into two groups: Those that are relevant to the measurement process and subsequent evaluation of the minimum safety-distance, leading to the decision whether or not the robot has to be stopped, and those that are relevant to the stopping process afterward. The worst case time required for the measurement and evaluation will be called T_r', while the equivalent for the stopping will be T_s'. Their definition slightly differs from those of T_r and T_s in ISO/TS 15066, where the time required to activate the safety stop (excluding the actual time to stop) is also counted towards T_r (in contrast to T_r', where it is part of T_s'). Instead, it is more in line with time spans used by ISO 13855 [46]. Furthermore, using T_r' and T_s' instead of T_r and T_s is not

problematic, as $T'_r + T'_s = T_r + T_s$, thus the same overall time is covered. However, some minor changes to the individual factors used for the calculation of the protective separation distance are necessary. The following original definitions are affected [51]:

$$S_h = \int_{t_0}^{t_0+T_r+T_s} v_h(t)dt, \;\; S_r = \int_{t_0}^{t_0+T_r} v_r(t)dt, \;\; S_s = \int_{t_0+T_r}^{t_0+T_r+T_s} v_s(t)dt \tag{7.2}$$

Hereby, v_h denotes a time-dependent function for the human movement speed, v_r denotes a time-dependent function for the robot speed, and v_s denotes a time-dependent function for the robot speed while the robot is trying to stop. These speeds are defined directed towards the robot and the human respectively, thus the integral indicates the change in distance between human and robot [51]. With the use of T'_r and T'_s, new formulas for distances S'_h, S'_r and S'_s can be defined, such that $S'_h = S_h$ and $S'_r + S'_s = S_r + S_s$:

$$\begin{aligned} S'_h = S'_{h,0} + S'_{h,1}, \;\; S'_{h,0} = \int_{t_0}^{t_0+T'_r} v_h(t)dt, \;\; S'_{h,1} = \int_{t_0+T'_r}^{t_0+T'_r+T'_s} v_h(t)dt \\ S'_r = \int_{t_0}^{t_0+T'_r} v_r(t)dt, \;\; S'_s = \int_{t_0+T'_r}^{t_0+T_r} v_r(t)dt + \int_{t_0+T_r}^{t_0+T_r+T_s} v_s(t)dt \end{aligned} \tag{7.3}$$

To explain the change for S'_s, it has to be considered that T_r includes the required time to initiate the stopping procedure of the robot in contrast to T'_r. This means that $T_r > T'_r$, and between $t_0 + T'_r$ and $t_0 + T_r$, the robot will keep operating with a speed determined by v_r instead of v_s. Thus v_r is used between $t_0 + T'_r$ and $t_0 + T_r$ in the equation for S'_s. With these new definitions, the protective separation distance can be reformulated as follows:

$$S_p(t_0) = S_h + S_r + S_s + C + Z_d + Z_r = S'_{h,0} + S'_{h,1} + S'_r + S'_s + C + Z_d + Z_r \tag{7.4}$$

With this reformulation, it is now possible to separate the different parts of the protective separation distance into the aforementioned groups: Those that account for how much closer the human could be to the robot than the measured human-robot distance $d'_{h\text{-}r}$ due to the measurement process and evaluation of the protective separation distance, and those that account for potential further distance decreases during to the stopping procedure of the robot. The first group of factors includes $S'_{h,0}$, S'_r, as well as C, Z_d and Z_r, as they directly relate to the measurement of the human and robot position. The latter set includes $S'_{h,1}$ as well as S'_s.

With these groups of factors, the evaluation of the protective separation distance can be reformulated as well. The unaltered version checks whether a measured distance $d'_{h\text{-}r}$ that incorporates none of the protective separation distance factors violates the protective separation distance $S_p(t_0)$, i. e., a safety stop of the robot has to be activated if it is smaller:

$$d'_{h\text{-}r} < S_p(t_0) = S'_{h,0} + S'_{h,1} + S'_r + S'_s + C + Z_d + Z_r \tag{7.5}$$

While this version is designed to work with the measured distance $d'_{h\text{-}r}$ that does not consider the worst-case reduction of the distance between human and robot during its determination and evaluation against $S_p(t_0)$, this can be changed by incorporating the respective protective separation distance factors from the first group into the calculation of the human-robot distance. This yields a new calculated human-robot-distance $d''_{h\text{-}r}$ and an altered protective separation distance $S'_p(t_0)$ through the following formula:

$$d''_{h\text{-}r} = d'_{h\text{-}r} - S'_{h,0} - S'_r - C - Z_d - Z_r < S'_{h,1} + S'_s = S'_p(t_0) \tag{7.6}$$

In this formulation the newly defined human-robot distance $d''_{h\text{-}r}$ is now compared to the altered version of the protective separation distance $S'_p(t_0)$ to assess whether or not the robot has to be stopped. In function, Eq. (7.6) is identical to Eq. (7.5). However, considering the measurement- and evaluation-related factors that might lead to a lower human-robot distance than the measured distance $d'_{h\text{-}r}$ as part of $d''_{h\text{-}r}$ opens up new ways of human-robot distance calculation that might lead to less conservative outcomes (robot has to be stopped later), while still ensuring safety.

To highlight the implications of Eq. (7.6), it shall be assumed that multiple distances to the robot are measured, and that the smallest of all measured distances is used as $d'_{h\text{-}r}$. This could e. g., be the case, if multiple people are present, or if the distance for a single person is calculated based on multiple measurements (as it could be the case in the presence of multiple keypoint detections). Let i denote the i-th of N measured distances $d'_{h\text{-}r,i}$, with $d'_{h\text{-}r} = \min(d'_{h\text{-}r,1}, d'_{h\text{-}r,2}, ..., d'_{h\text{-}r,N})$. Further assume that each measured distance $d'_{h\text{-}r,i}$ has its own, associated upper bound for the measurement error $Z_{d,i}$, with $Z_d = \max(Z_{d,1}, Z_{d,2}, ..., Z_{d,N})$. The other elements shall remain as is for this example, without further changes. Following the paradigm of the protective separation distance to operate on worst-case assumptions, the following adaptation of Eq. (7.6) would have to be evaluated for N distance measures with custom upper bounds for the measurement error:

$$\begin{aligned} d''_{h\text{-}r} = &\min(d'_{h\text{-}r,1}, d'_{h\text{-}r,2}, ..., d'_{h\text{-}r,N}) \\ &- \max(Z_{d,1}, Z_{d,2}, ..., -Z_{d,N}) \\ &- S'_{h,0} - S'_r - C - Z_r < S'_p(t_0) \end{aligned} \tag{7.7}$$

However, in case of $\text{argmin}_i(d'_{h\text{-}r,i}) \neq \text{argmax}_i(Z_{d,i})$, the calculated distance $d''_{h\text{-}r}$ becomes smaller than it has to be for safety, as the correlation between $Z_{d,i}$ and $d'_{h\text{-}r,i}$ is not modeled. Without putting safety at risk, this correlation can be factored into the calculation of the equation, modeling it as follows:

$$\begin{aligned} &\min(d'_{h\text{-}r,1}, d'_{h\text{-}r,2}, ...d'_{h\text{-}r,N}) - \max(Z_{d,1}, Z_{d,2}, ..., -Z_{d,N}) \leq \\ &\min(d'_{h\text{-}r,1} - Z_{d,1}, d'_{h\text{-}r,2} - Z_{d,2}, ..., d'_{h\text{-}r,N} - Z_{d,N}) \end{aligned} \tag{7.8}$$

The right side of the Eq. (7.8) can be used as a substitute for the left side without a risk to safety, as it models the existing dependency between individual distance measurements and their associated upper bound for the measurement error, instead of simply taking the uncorrelated worst case of both. When this substitution is used in Eq. (7.7), then it leads to equal or larger values for $d''_{h\text{-}r}$. In turn, larger values for $d''_{h\text{-}r}$ mean that the violation of the altered protective separation distance (which is the case if $d''_{h\text{-}r} < S'_p(t_0)$) becomes less likely, which in turn means that potentially fewer safety stops will be required.

The previous example showed that incorporating measurement- and evaluation-related factors from the protective separation distance into the calculation of the human-robot distance can lead to less conservative results when evaluating the protective separation distance, or in that case, its altered variant $S'_p(t_0)$ (less violations of the altered protective separation distance). This can be achieved by modeling the interconnection between different elements of the protective separation distance in higher detail, instead of using the worst-case value for each individual part. This is the reason why the stipulation was made to include them into the calculation of $d_{h\text{-}r}$, so that $d_{h\text{-}r}$ only needs to be evaluated against $S'_p(t_0)$. This stipulation does not restrict solutions that want to rely on the original definition of the protective separation distance, as the equivalent formulation in Eq. (7.6) can be applied. Instead, it opens up new ways to implement the distance calculation of $d_{h\text{-}r}$ in hard real-time while retaining safety.

7.3 Discussion of Potential Solutions

To solve the problem of hard real-time capable distance calculation between human and robot, two separate problems have to be tackled: First, the distance calculation itself has to be hard real-time capable, and second, availability of data for the distance calculation from human pose estimation must be ensured. Both factors will be discussed in the following.

Distance Calculation in Hard Real-Time: A distance calculation in hard real-time requires an upper limit for the amount of operations that have to be performed, as well as hard real-time capable hardware and software for their execution. The last two points can easily be addressed by restricting the necessary distance calculations to a simple set of operations that can be executed on a real-time operating system and device without the need for highly specialized libraries and hardware which would e. g., be required for deep learning.

A simple way to measure the human-robot distance could be to calculate the distance between human keypoints and further points representing the robot, as e. g., done by Svarny et al. [119]. This would lead to several calculations of the Euclidean distance between pairs of 3D points, with each calculation being performed through a fixed, small number of simple operations. The overall number of calculations can be limited by using a fixed number of (key)points for humans and robots while guaranteeing an upper limit for the total number of people and robots. Such a limit could e. g., be enforced by stopping the robot(s), as soon as too many people enter the shared human-robot workspace. To adjust the distance to include selected factors from the protective separation distance (see left side of Eq. (7.6)) as proposed in Section 7.2, the calculation of C, Z_d, Z_r, $S'_{h,0}$ and S'_r (or their alternative incorporation) is necessary. The variable C encompasses constant values, which are known beforehand, thus hard real-time capability is no problem here. Z_d and Z_r can also be represented through constant values, or they could be determined as part of the human or robot position acquisition, as it is e. g., done with the upper bounds for the measurement error of human keypoints in the proposed 3D human pose estimator. In turn, they pose no greater challenge to the hard real-time capability than the position acquisition itself. With $S'_{h,0}$ and S'_r, it becomes potentially more challenging, as the formulas for their calculation (see Eq. (7.3)) encapsulate a time-dependent function for the human and robot movement speed respectively. If the time dependence shall be leveraged, human and robot movement during T'_r must be determined safely, e. g., by a safe simulation. Such a safe and detailed simulation can be expected to be very complex, thus its hard real-time capability could be questioned, if such a simulation would be available. As an alternative, constant movement speeds could be used in both cases. This procedure is also suggested by ISO/TS 15066 [51]. For human movement, 1.6m/s is suggested by ISO/TS 15066 [51] as worst case speed (up to 2m/s in ISO 13855 [46], depending on the use case), while the active upper speed limit for the robot can serve as (worst case) speed for the robot.

Instead of measuring the human-robot distance based on individual points from both of them, distance calculations could also be performed based on volume models. The factors C, Z_d, Z_r, $S'_{h,0}$, and S'_r could then be directly incorporated into the volume model to be part of the distance calculation. In any case, determining the respective volume models as well as the distance calculation between them would have to be hard real-time capable. A suitable foundation for such volume modeling could e. g., be the work of Tornero et al. [124], who introduced a volume modeling and fast distance calculation approach based on spheres, which are also the outputs of the proposed 3D human pose estimator.

Making Human Pose Estimation Hard Real-Time Capable: While the proposed 3D human pose estimator in the form of a neural network was already declared to be incapable of hard real-time in the assumptions from Section 7.1, it shall nevertheless be quickly discussed whether making it hard real-time capable would be sufficient to supply the data needed for the hard real-time capable distance calculation. In short, it is not, at least not on its own. Making it hard real-time capable would mean that predictions of the 3D human pose estimator would be available within a fixed amount of time. However, it was assumed that not all of these results would always be valid. Through the removal of invalid data, it is impossible to define the maximum amount of time required to obtain a valid result for a keypoint, which is required for further distance calculations.

Bridging the Time In-Between Human Pose Estimation Results: To sufficiently supply a distance calculation method with results, two potential problems under the given assumptions from Section 7.1 have to be tackled. These are that (i) no upper time limit can be guaranteed until the human pose estimation method produces an output and (ii) that some keypoints may lack a valid result in that output. A potential solution to both of these problems would be to bridge the time in-between detections for individual keypoints from the human pose estimation method. After an initialization period, where at least one detection per keypoint has to be gathered, adjustments to these keypoint detections could be made to ensure that they remain valid with respect to safety. Such adaptations could include e. g., an increase of the upper bound for the measurement error based on the time that passed since the keypoint position was obtained, as well as adjustments to the position of the keypoint detection. For these adjustments, human movement speeds as outlined in ISO 13855 [46] and ISO/TS 15066 [51] could be leveraged. The necessary initialization period could e. g., be covered by more traditional safety measures, switching to SSM based on human pose estimation as soon as the initialization finished, or the robot may only be allowed to start operating after the initialization was performed.

Pursuing a solution that safely bridges the time in-between human pose estimation results shows especially potential when combined with a volume model that is derived from human pose estimation results. Whenever the distance calculation requires data, the volume model can be adapted based on the previous human pose estimation results and the time that passed since the last valid result for each keypoint arrived. These adaptations could be designed to account for both: the missing hard real-time capability of the human pose estimation method, as well as the human-specific factors from the protective separation distance that the distance calculation shall account for. The robot could be modeled similarly, with the robot-specific factors being incorporated into the robot model. Furthermore, additional desired properties could be realized through such models, like a continuous function for the distance calculation through continuous surface changes of the involved volume models. Such a property would be desirable when the robot speed is adapted based on the human-robot distance in the SSM application.

Conclusion: From the discussed potential solutions, an approach that builds upon a *volume model* built from human pose estimation results seems most promising. The construction of the model shall include a *bridging method* for safely bridging the time in-between human pose estimation results to resolve the missing hard real-time capability of the 3D human pose estimation method, making it applicable for safety applications like SSM. When a similar model is used for the robot, all factors from the protective separation distance that shall now be considered during the distance calculation could be incorporated into the respective models. Then the models would cover the whole area where parts of the human and robot could be when the human-robot distance is calculated and evaluated.

7.4 Human Volume Modeling and Pipeline Design

First, it is necessary to define a volume model for the human. This model should be used to (i) resolve the missing hard real-time capability of the human pose estimation method, (ii) incorporate human-specific factors from the protective separation distance that shall now be considered during the human-robot distance calculation, and (iii) enable human-robot distance calculations in hard real-time. The foundation for the model's calculation will be the results from the proposed 3D human pose estimator, uniform keypoint spheres consisting of the 3D keypoint positions as centers and upper bounds of the measurement error as radii. The previously mentioned modeling approach of Tornero et al. [124] is a promising choice under this condition, as it leverages sweeping volumes from spheres with one or more degrees of freedom for volume modeling. To be applicable, a suitable way to model the human with the methods of Tornero et al. must be found, and hard real-time capable distance calculation must be ensured with the resulting volume model.

From the sweeping volumes introduced and examined by Tornero et al. [124], the spherical cone seems most suited for human volume modeling. Its sweeping volume is defined by a sphere with one degree of freedom and bounded movement. Mathematically, it is described as a linear combination of two spheres. Let $\boldsymbol{p_{i,1}}$ and $\boldsymbol{p_{i,2}}$ denote the centers of these two spheres, while $\boldsymbol{r_{i,1}}$ and $\boldsymbol{r_{i,2}}$ denote the radius values. Then, the functions $p_i(\lambda_i)$ and $r_i(\lambda_i)$ with $\lambda_i \in [0, 1]$ define the i-th spherical cone as follows [124]:

$$\begin{aligned} p_i(\lambda_i) &= \boldsymbol{p_{i,1}} + \lambda_i(\boldsymbol{p_{i,2}} - \boldsymbol{p_{i,1}}) \\ r_i(\lambda_i) &= \boldsymbol{r_{i,1}} + \lambda_i(\boldsymbol{r_{i,2}} - \boldsymbol{r_{i,1}}) \end{aligned} \tag{7.9}$$

This formulation is especially useful for the assumed 3D human pose estimation method, as keypoints are already detected in form of keypoint sphere and thus pairs of keypoint spheres can be employed to define spherical cones. Taking into account that keypoints are often defined based on the position of human body joints [3, 70], suitable pairs of such keypoints can be employed to model the human limbs connecting these joints, e. g., wrist and elbow keypoints for the lower arms. For the torso and the head, other suitable keypoints pairs can be chosen, if available e. g., the upper neck keypoint and top of the head keypoint for the head and the thorax keypoint and pelvis keypoint for the torso (although one more intermediate keypoint would be preferable here to account for spine flexibility). Then, the human volume is defined by the union of all these spherical cones. Required adaptations to the volume model to account for the missing hard real-time capability of the human pose estimator and for the respective human-specific factors from the protective separation distance that shall be incorporated in the distance calculation can then be performed by adapting the individual spheres, as each spherical cone is defined by the linear combination of two existing spheres. It is important to note that individual spheres do now no longer only account for the position of the keypoint itself, but also for the position of associated body limbs – this must be considered, when adaptations are performed. An illustration of how spherical cones can be used to create a volume model of the human can be found in Figure 7.1, simplified from 3D to 2D and only for one arm of the human.

After answering how the human can be modeled using spherical cones, the question of hard real-time capable distance calculations arises, especially when considering that a sweeping volume as defined by Eq. (7.9) consist of an infinite number of spheres. For the calculation of the distance between a pair of spherical cones, Tornero et al. [124] offer an analytical solution based on the definition from Eq. (7.9) that consists of a few simple

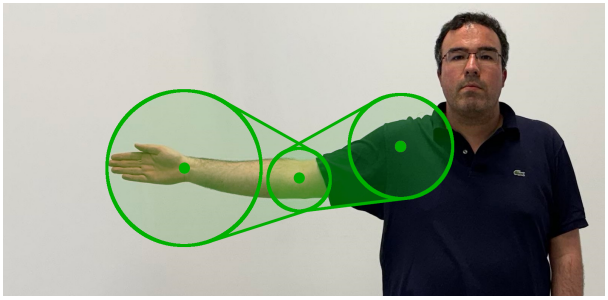


Figure 7.1: 2D illustration of how the lower arm (including the hand) and the upper arm are modeled using two spherical cones, each derived from a pair of keypoint spheres. Based on author's figure from [107].

steps only (like additions, subtractions, division, (inverse) trigonometric functions, and square roots). This could be implemented in hard real-time without the need for complex software libraries, thus a hard real-time capable solution for the distance calculation between pairs of spherical cones is available. To calculate the human-robot distance in hard real-time with this solution, two more things are required. First, the robot must also be modeled using spherical cones, so that the human-robot distance can be calculated as the distance between spherical cones. This can be done similar to human modeling with spherical cones. The joints and base of the robot can serve as the centers of spheres with suitable radius values, and robot links as well as the connection from the base to the first joint can be modeled as spherical cones. Other factors could also be incorporated into the robot volume model, like the robot-specific factors from the protective separation distance that shall now be considered during the distance calculation. Second, the amount of distance calculations must be limited. This is the case if (i) the amount of spherical cones is limited for individual humans and robots, and (ii) the total number of humans and robots is limited. Defining a limited number of keypoints or points of the robot for which spheres and spherical cones are calculated resolves issue (i), as the number of keypoints per human is already limited. To resolve issue (ii), additional measures could be employed to enforce a maximum allowed number of people, e. g., shutting off the robot if the number is exceeded, while the amount of robots is limited by the design of the robot work cell in a controlled industrial environment. Then, only a limited amount of distance calculations from all spherical cones of the humans to those of the robots has to be performed to find the smallest human-robot distance, enabling distance calculation in hard real-time.

The resulting distance calculation pipeline is displayed in Figure 7.2. Without the need for hard real-time capability, a 3D human pose estimation system supplies 3D keypoint spheres, each consisting of the predicted keypoint position and upper bound for the measurement error. For some keypoints, corresponding keypoint spheres can miss. They serve as input for the hard real-time capable module for human volume modeling, which adapts previously received keypoint spheres to safely model the human volume through spherical cones. With a fixed frequency, it supplies adapted keypoint spheres defining spherical cones to the distance calculation module. Adapted spheres for the robot are also passed to the distance calculation module for the calculation of the human-robot distance. In the figure, they are calculated in similar fashion to the adapted keypoint spheres, however, any hard real-time capable solution is fine that ensures that the robot is safely modeled through spherical cones derived from these spheres and that they include the robot-specific factors from the protective separation distance that have to be considered when the calculated human-robot distance shall be compared against $S'_p(t_0)$ from Eq. (7.6).

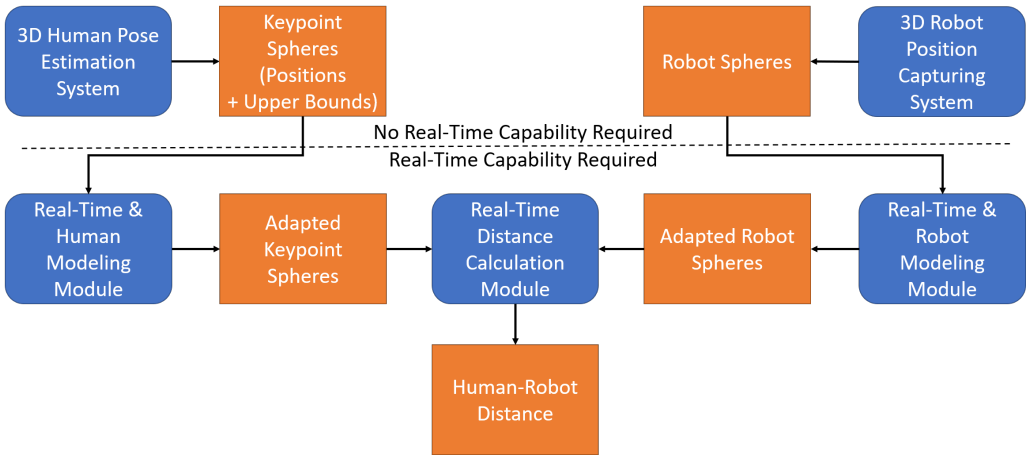


Figure 7.2: Illustration of the proposed pipeline for distance calculation between human and robot, highlighting the parts that require hard real-time capability and those that do not. In the pipeline, it is assumed that adapted robot spheres are acquired with comparable steps to those used for obtaining adapted keypoint spheres. However, any hard real-time capable and safe procedure that supplies suitable spheres for the robot will suffice.

7.5 Volume Model Adaptations

The previous section introduced the concept of modeling the human using spherical cones, which are generated based on the keypoint spheres produced by a 3D human pose estimation system under the assumptions from Section 7.1. However, these keypoint spheres only lay the foundation for the volume model and require further adjustments to account for the missing hard real-time capability of the human pose estimator, as well as for the additional human-specific factors from the protective separation distance that shall be considered during the distance calculation. In the following, these two aspects will be discussed separately by (i) introducing an algorithm for bridging the time in-between human pose estimation results and (ii) introducing adaptations to account for the aforementioned human-related factors of the protective separation distance. As spherical cones are indirectly defined by pairs of keypoint spheres, (i) and (ii) will define necessary adaptations to individual keypoint spheres. These adaptations must ensure that safety through SSM can be achieved for any arbitrary point in time t_0. This includes that the adaptations themselves must be executable within a maximum amount of time so that the potential worst-case movement of the human towards the robot during their calculation can be factored in. Thus, the hard real-time capability of the adaptation methods must be shown.

7.5.1 Adaptations for Hard Real-Time Capability

To bridge the time in-between human pose estimation results it will be assumed that the 3D human pose estimator sends new results to the bridging method whenever they are available. These results are keypoint spheres, each consisting of the predicted keypoint position $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ and the upper bound for the measurement error $\hat{\boldsymbol{r}}_{\boldsymbol{i}}$, with i denoting the i-th keypoint. When the human pose estimation process (including data acquisition) that led to the calculation of the latest result for the i-keypoint was started at $t_{1,i}$ and a current human pose estimation result is requested at t_2, it is not possible to define a maximum period of time that is guaranteed to include $t_2 - t_{1,i}$. This is the case as the assumed

3D human pose estimator is neither hard real-time capable (results not guaranteed to be available within a fixed time limit) nor is the availability of a valid keypoint sphere for the i-th keypoint in each result guaranteed. In practice, an upper limit for $t_2 - t_{1,i}$ would be required to account for the worst-case distance the human could have moved towards the robot during the human pose estimation process and up to t_2. The simplest way to account for this distance without an upper time limit would be to extend the individual keypoint spheres based on the maximum human movement speed from safety standards (considering worst-case movement of every keypoint into any direction) and the time that actually passed between t_2 and each $t_{1,i}$. Then, only an upper limit for the time necessary to adapt the individual keypoint spheres would be necessary to resolve the missing hard real-time capability. This time limit will be called Δt_b. Let r_g denote a function that adapts the radius of the keypoint sphere and p_g denote a function that adapts the keypoint position. Then, when human pose estimation results are requested at t_2, the functions can be formalized with human movement speed v_h (1.6m/s [51] or up to 2m/s [46]) as follows:

$$\begin{aligned} r_g(t_2, t_{1,i}, \hat{\boldsymbol{r}}_{\boldsymbol{i}}) &= \hat{\boldsymbol{r}}_{\boldsymbol{i}} + v_h(t_2 - t_{1,i}) \\ p_g(t_2, \hat{\boldsymbol{y}}_{\boldsymbol{i}}) &= \hat{\boldsymbol{y}}_{\boldsymbol{i}} \end{aligned} \tag{7.10}$$

In simple terms, r_g adapts the radius $\hat{\boldsymbol{r}}_{\boldsymbol{i}}$ of the latest detected i-th keypoint sphere by assuming worst case human movement into all directions with v_h from $t_{1,i}$ to t_2, while the sphere's position remains unaltered through p_g. This equation safely bridges the time between the start of the human pose estimation process and the actual time human pose estimation results are requested. With a simple, limited number of operations per keypoint sphere, it can be implemented in hard real-time for a limited number of spheres. However, it exposes undesired behavior when a new keypoint sphere $(\hat{\boldsymbol{y}}'_{\boldsymbol{i}}, \hat{\boldsymbol{r}}'_{\boldsymbol{i}})$ with new associated time $t'_{1,i} > t_{1,i}$ is predicted and received for the i-th keypoint. Then, discontinuous changes in the volume model's surface can be observed over time. The function r_g is continuous with respect to continuous changes in time t_2 while $t_{1,i}$ and $\hat{\boldsymbol{r}}_{\boldsymbol{i}}$ remain constant. However, a noncontinuous change of the output occurs when the inputs $t_{1,i}$ and $\hat{\boldsymbol{r}}_{\boldsymbol{i}}$ are replaced with their newly arrived counterparts $t'_{1,i}$ and $\hat{\boldsymbol{r}}'_{\boldsymbol{i}}$, a variable change that is non-continuous itself. Furthermore, the overall position of the keypoint sphere also changes in a non-continuous way due to the sudden substitution also taking place in p_g. This leads to a non-continuous change of the human volume model's surface if this sudden substitution happens between two points in time $t_{2,j}$ and $t_{2,j+1}$ where results are requested, potentially causing a non-continuous change of the human-robot distance, which is e. g., problematic if the robot speed shall also be controlled based on the distance (jump in robot speed).

This discontinuity is caused by the replacement of the old adapted keypoint sphere with the newly arriving one. A first step to resolve this issue is to retain and further adapt the contributions of the previous keypoint sphere to the human volume instead of instantly deleting it. Then, the only remaining cause of a discontinuous volume change is if the new keypoint sphere is not fully encapsulated by the old one upon arrival, hence directly contributing to the volume. Let $(\hat{\boldsymbol{y}}'_{\boldsymbol{i}}, \hat{\boldsymbol{r}}'_{\boldsymbol{i}})$ denote the position and radius of a newly arriving keypoint sphere, $t'_{1,i}$ the time when the associated human pose estimation process started and $t_{r,i}$ the time when it was received by the bridging method. Then, $\boldsymbol{r}'_{\boldsymbol{g,i,t_{r,i}}} = r_g(t_{r,i}, t'_{1,i}, \hat{\boldsymbol{r}}'_{\boldsymbol{i}})$ will denote the adapted radius of the new keypoint sphere at $t_{r,i}$, and $\boldsymbol{p}'_{\boldsymbol{g,i,t_{r,i}}} = p_g(t_{r,i}, \hat{\boldsymbol{y}}'_{\boldsymbol{i}})$ the adapted position. Furthermore, let $\boldsymbol{r}_{\boldsymbol{g,i,t_{r,i}}} = r_g(t_{r,i}, t_{1,i}, \hat{\boldsymbol{r}}_{\boldsymbol{i}})$ and $\boldsymbol{p}_{\boldsymbol{g,i,t_{r,i}}} = p_g(t_{r,i}, \hat{\boldsymbol{y}}_{\boldsymbol{i}})$ denote the respective adapted values for the previous keypoint sphere at $t_{r,i}$. Then, the newly arrived keypoint sphere is fully encapsulated by the previous one, if:

$$\boldsymbol{r}'_{\boldsymbol{g,i,t_{r,i}}} + ||\boldsymbol{p}'_{\boldsymbol{g,i,t_{r,i}}} - \boldsymbol{p}_{\boldsymbol{g,i,t_{r,i}}}||_2 \leq \boldsymbol{r}_{\boldsymbol{g,i,t_{r,i}}} \tag{7.11}$$

From the center $\boldsymbol{p'_{g,i,t_{r,i}}}$ of the new sphere, all points on its surface have a distance of $\boldsymbol{r'_{g,i,t_{r,i}}}$. If the radius of the previous sphere is larger than the radius of the new sphere plus the distance between both sphere centers, then all keypoints on the surface of the new sphere will be encapsulated by the previous sphere. The same is also true for all spherical cones that could be created from $(\boldsymbol{p'_{g,i,t_{r,i}}}, \boldsymbol{r'_{g,i,t_{r,i}}})$ due to the spherical cone definition from Eq. (7.9). Thus, retaining the old keypoint sphere and only accepting a new keypoint sphere if it fully falls inside the old one resolves the problem of discontinuity. However, this opens up a new question: how shall the two or more keypoint spheres for the same keypoint after the arrival and acceptance of a new one be adapted? This is an especially interesting question, as the presence of more than one keypoint sphere per keypoint can endanger both, the hard real-time capability of the bridging method as well as the distance calculation, if no maximum number of spheres can be defined.

To ensure that all potential positions for keypoint i are well reflected when requested at t_2, the adaptation of its newest keypoint sphere needs to be pursued with Eq. (7.10), as it was done previously. The question of how an old keypoint sphere shall be adapted after the arrival of a newer one at $t_{r,i}$ remains to be answered. As an old keypoint sphere indicates only where the keypoint could have been before a newer keypoint sphere arrived and was accepted, it is now no longer needed. However, it is still contributing to the human volume model at $t_{r,i}$, hence its instant removal can lead to a discontinuous volume change, which shall be avoided. Hence, instead of instantly removing the old keypoint sphere, it would be logical to shrink its size with speed v_d instead of growing it, up until it can be removed without causing a discontinuous volume change. This point is either reached when the radius of the keypoint sphere becomes zero or when it is fully encapsulated by another keypoint sphere for the same keypoint, as the encapsulation means that it no longer contributes to the volumes of spherical cones that use the keypoint. Let the updated keypoint sphere radius in this new adaptation be $\boldsymbol{r_{b,i,t_2}}$ when an update is requested at t_2, and let $t_{r,i}$ denote the point in time where a newer keypoint sphere for keypoint i arrived and was accepted. Then, $\boldsymbol{r_{b,i,t_2}}$ can be defined as follows:

$$\boldsymbol{r_{b,i,t_2}} = \begin{cases} r_g(t_2, t_{1,i}, \hat{\boldsymbol{r}}_{\boldsymbol{i}}), \text{ if no newer keypoint sphere was accepted} \\ r_g(t_{r,i}, t_{1,i}, \hat{\boldsymbol{r}}_{\boldsymbol{i}}) - v_d \cdot (t_2 - t_{r,i}), \text{ else} \end{cases} \tag{7.12}$$

While this procedure ensures a continuous change of the volume surface, it is not sufficient for hard real-time capable keypoint sphere adaptations and distance calculations based on spherical cones. This is the case, as the amount of keypoint spheres and spherical cones that have to be considered per human is no longer limited. To prove this statement, assume that one sphere is currently present to represent the i-th keypoint. Upon the arrival and acceptance of a new keypoint sphere, the new keypoint sphere is fully inside the previous one, meaning that only the previous one contributes to the volume at that time. However, through the process of shrinking the old one and growing the new one, it can happen that both contribute to the human volume model at the same time. This is the case when neither of them fully encapsulates the other, which can make it necessary to consider spherical cones derived from both of them. With the arrival and acceptance of yet another new keypoint sphere, this process can repeat itself all over again. This can lead to an unknown number of keypoint spheres and spherical cones contributing to the human volume model, hereby preventing hard real-time capability. Figure 7.3 illustrates the problem. To achieve hard real-time capability, the sphere adaptation process must be changed to ensure that only a limited number of keypoint spheres (and thus spherical cones) has to be considered per keypoint.

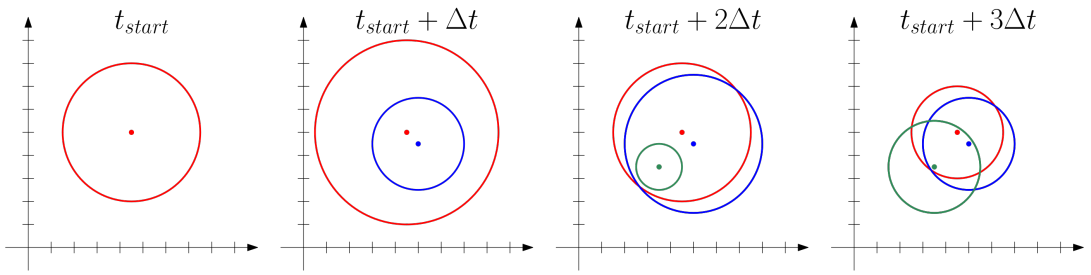


Figure 7.3: Illustration of the problem that multiple keypoint spheres for the same keypoint can contribute to the overall volume when the radius of the latest keypoint sphere is grown with v_h and the radii of all previous ones are shrunk with v_d ($v_d = v_h$ here).

The underlying problem of the previously described scenario is that the growing keypoint sphere $(\boldsymbol{p}'_{\boldsymbol{g},i,\boldsymbol{t_2}}, \boldsymbol{r}'_{\boldsymbol{b},i,\boldsymbol{t_2}})$ is no longer fully encapsulated by the shrinking keypoint sphere $(\boldsymbol{p}_{\boldsymbol{g},i,\boldsymbol{t_2}}, \boldsymbol{r}_{\boldsymbol{b},i,\boldsymbol{t_2}})$ when results are requested at t_2, all while the shrinking keypoint sphere still exists. This is the case when the following three conditions are fulfilled:

$$\begin{aligned} &\text{Condition 1: } \boldsymbol{r}'_{\boldsymbol{b},i,\boldsymbol{t_2}} + ||\boldsymbol{p}'_{\boldsymbol{g},i,\boldsymbol{t_2}} - \boldsymbol{p}_{\boldsymbol{g},i,\boldsymbol{t_2}}||_2 > \boldsymbol{r}_{\boldsymbol{b},i,\boldsymbol{t_2}} \\ &\text{Condition 2: } \boldsymbol{r}_{\boldsymbol{b},i,\boldsymbol{t_2}} + ||\boldsymbol{p}'_{\boldsymbol{g},i,\boldsymbol{t_2}} - \boldsymbol{p}_{\boldsymbol{g},i,\boldsymbol{t_2}}||_2 > \boldsymbol{r}'_{\boldsymbol{b},i,\boldsymbol{t_2}} \\ &\text{Condition 3: } \boldsymbol{r}_{\boldsymbol{b},i,\boldsymbol{t_2}} > 0 \end{aligned} \tag{7.13}$$

Condition 1 & 2 ensure that neither of both keypoint spheres is fully encapsulated by the other, meaning that both contribute to the volume. Condition 2 & 3 ensure that the shrinking keypoint sphere still exists due to not being fully encapsulated and having a radius larger than zero. If at least one of the conditions is always violated, distance calculations will have to consider only one keypoint sphere per keypoint: if the first condition is violated, the growing keypoint sphere is fully encapsulated by the shrinking one, meaning only the latter contributes to the volume. If condition 2 or 3 is violated, then the shrinking keypoint sphere is deleted and only the growing one remains. Furthermore, the constant violation of at least one condition ensures that a maximum of two keypoint spheres per keypoint have to be retained for further adaptations. To prove this, consider the arrival and acceptance of yet another keypoint sphere $(\boldsymbol{p}''_{\boldsymbol{g},i,\boldsymbol{t}'_{r,i}}, \boldsymbol{r}''_{\boldsymbol{b},i,\boldsymbol{t}'_{r,i}})$ for the i-th keypoint at $t'_{r,i}$. If condition 2 or 3 is violated at $t'_{r,i}$, then there is only only one other keypoint sphere, as the condition for deleting the shrinking one is met, bringing the total number of keypoint spheres for the i-th keypoint to two. If the first condition is violated instead then $(\boldsymbol{p}'_{\boldsymbol{g},i,\boldsymbol{t}'_{r,i}}, \boldsymbol{r}'_{\boldsymbol{b},i,\boldsymbol{t}'_{r,i}})$ is fully encapsulated by $(\boldsymbol{p}_{\boldsymbol{g},i,\boldsymbol{t}'_{r,i}}, \boldsymbol{r}_{\boldsymbol{b},i,\boldsymbol{t}'_{r,i}})$. This means $(\boldsymbol{p}'_{\boldsymbol{g},i,\boldsymbol{t}'_{r,i}}, \boldsymbol{r}'_{\boldsymbol{b},i,\boldsymbol{t}'_{r,i}})$ can directly be deleted instead of being transformed into a shrinking keypoint sphere, as it has currently no contribution to the volume and won't have one in the future, as $(\boldsymbol{p}''_{\boldsymbol{g},i,\boldsymbol{t}'_{r,i}}, \boldsymbol{r}''_{\boldsymbol{b},i,\boldsymbol{t}'_{r,i}})$ is now the growing keypoint sphere. Again, two keypoint spheres remain. Thus, violating at least one condition from Eq. (7.13) limits both, the number of keypoint spheres and spherical cones that have to be considered per keypoint.

Now, the violation of at least one condition from Eq. (7.13) must be ensured, which can be achieved by adapting the shrinking keypoint sphere's center in a different way. Let $(\boldsymbol{p}_{\boldsymbol{g},i,\boldsymbol{t}_{r,i}}, \boldsymbol{r}_{\boldsymbol{b},i,\boldsymbol{t}_{r,i}})$ be a keypoint sphere for the i-th keypoint upon the arrival and acceptance of a newer keypoint sphere $(\boldsymbol{p}'_{\boldsymbol{g},i,\boldsymbol{t}_{r,i}}, \boldsymbol{r}'_{\boldsymbol{b},i,\boldsymbol{t}_{r,i}})$ at $t_{r,i}$. From that point in time on, $(\boldsymbol{p}_{\boldsymbol{g},i,\boldsymbol{t}_{r,i}}, \boldsymbol{r}_{\boldsymbol{b},i,\boldsymbol{t}_{r,i}})$ will shrink with v_d and $(\boldsymbol{p}'_{\boldsymbol{g},i,\boldsymbol{t}_{r,i}}, \boldsymbol{r}'_{\boldsymbol{b},i,\boldsymbol{t}_{r,i}})$ will grow with v_h. The minimum distance between the sphere surfaces at $t_{r,i}$ can be calculated as follows:

$$\boldsymbol{r}_{\boldsymbol{b},i,\boldsymbol{t}_{r,i}} - (\boldsymbol{r}'_{\boldsymbol{b},i,\boldsymbol{t}_{r,i}} + ||\boldsymbol{p}_{\boldsymbol{g},i,\boldsymbol{t}_{r,i}} - \boldsymbol{p}'_{\boldsymbol{g},i,\boldsymbol{t}_{r,i}}||_2) \tag{7.14}$$

It is possible to keep this distance constant despite the growing and shrinking of the keypoint spheres for a limited period of time $\Delta t_c = ||\boldsymbol{p'_{g,i,t_{r,i}}} - \boldsymbol{p_{g,i,t_{r,i}}}||_2/(v_h + v_d)$ when moving the center of the shrinking keypoint sphere with speed $v_h + v_d$ towards the center of the growing keypoint sphere until both positions are the same. This is possible without a risk to safety as the volume defined by the old keypoint sphere is already outdated. Furthermore, movement with fixed speed doe not endanger the continuous volume change over time. The retention of the same minimum distance between the surface of both keypoint spheres while moving can be shown for $0 \leq \Delta t \leq \Delta t_c$ as follows, with $d_{p',p} = ||\boldsymbol{p'_{g,i,t_{r,i}}} - \boldsymbol{p_{g,i,t_{r,i}}}||_2$ denoting the Euclidean distance between both sphere centers:

$$
\begin{aligned}
&(\boldsymbol{r_{b,i,t_{r,i}}} - v_d\Delta t) - ((\boldsymbol{r'_{b,i,t_{r,i}}} + v_h\Delta t) \\
&\quad + ||\boldsymbol{p_{g,i,t_{r,i}}} + \frac{\boldsymbol{p'_{g,i,t_{r,i}}} - \boldsymbol{p_{g,i,t_{r,i}}}}{d_{p',p}}(v_d + v_h)\Delta t - \boldsymbol{p'_{g,i,t_{r,i}}}||_2) \\
=&(\boldsymbol{r_{b,i,t_{r,i}}} - v_d\Delta t) - (\boldsymbol{r'_{b,i,t_{r,i}}} + v_h\Delta t) \\
&\quad - ||\frac{\boldsymbol{p_{g,i,t_{r,i}}} d_{p',p} + (\boldsymbol{p'_{g,i,t_{r,i}}} - \boldsymbol{p_{g,i,t_{r,i}}})(v_d + v_h)\Delta t - \boldsymbol{p'_{g,i,t_{r,i}}} d_{p',p}}{d_{p',p}}||_2 \\
=&(\boldsymbol{r_{b,i,t_{r,i}}} - v_d\Delta t) - (\boldsymbol{r'_{b,i,t_{r,i}}} + v_h\Delta t) \\
&\quad - ||\frac{\boldsymbol{p_{g,i,t_{r,i}}}(d_{p',p} - (v_d + v_h)\Delta t) - \boldsymbol{p'_{g,i,t_{r,i}}}(d_{p',p} - (v_d + v_h)\Delta t)}{d_{p',p}}||_2 \\
=&(\boldsymbol{r_{b,i,t_{r,i}}} - v_d\Delta t) - (\boldsymbol{r'_{b,i,t_{r,i}}} + v_h\Delta t) \\
&\quad - |d_{p',p} - (v_d + v_h)\Delta t| \cdot ||\frac{\boldsymbol{p_{g,i,t_{r,i}}} - \boldsymbol{p'_{g,i,t_{r,i}}}}{d_{p',p}}||_2 \\
=&(\boldsymbol{r_{b,i,t_{r,i}}} - v_d\Delta t) - (\boldsymbol{r'_{b,i,t_{r,i}}} + v_h\Delta t) - (d_{p',p} - (v_d + v_h)\Delta t) \\
=&\boldsymbol{r_{b,i,t_{r,i}}} - (\boldsymbol{r'_{b,i,t_{r,i}}} + d_{p',p})
\end{aligned}
\tag{7.15}
$$

The penultimate transformation in this equation is possible due to $\Delta t \leq \Delta t_c$, hence the calculation of the absolute value can be removed as the value is positive anyway. This leads to equivalence with Eq. (7.14) after the final transformation, showing that the minimum distance between the sphere surface of both spheres is retained during the movement. Recalling that a new keypoint sphere is only accepted if it lies within the previous sphere's volume, this means that the new keypoint sphere will remain fully inside the old keypoint sphere during this movement. In turn, the first condition from Eq. (7.13) is violated during the movement. For time t_2 with $t_2 \geq t_{r,i} + \Delta t_c$, the center of the old and new keypoint sphere are the same. Then, it is sufficient to check whether $\boldsymbol{r_{b,i,t_2}} \leq \boldsymbol{r'_{b,i,t_2}}$ to decide if the shrinking keypoint sphere can be deleted (i. e., is fully encapsulated). In turn, this means that as soon as the first condition from equation 7.13 becomes true (which degenerates to $\boldsymbol{r'_{b,i,t_2}} > \boldsymbol{r_{b,i,t_2}}$ for equal sphere centers), the second condition is no longer fulfilled (which degenerates to $\boldsymbol{r_{b,i,t_2}} > \boldsymbol{r'_{b,i,t_2}}$ for equal sphere centers), as their fulfillment is mutually exclusive for equal sphere centers.

To describe the movement of a shrinking keypoint sphere's center mathematically, its updated position will be called $\boldsymbol{p_{b,i,t_2}}$ when requested at t_2. Over the lifetime of a keypoint sphere, multiple newer keypoint spheres can be received and accepted. They will be enumerated using index k for K newer keypoint spheres, with $t_{r,i,k}$ denoting the time they were received ($t_{r,i,k} < t_{r,i,k+1}$). Their respective keypoint sphere centers will be called $\boldsymbol{p'_{b,i,t_{r,i,k},k}}$. Then, $\boldsymbol{p_{b,i,t_2}}$ at t_2 can be described as follows:

$$\boldsymbol{p_{b,i,t_2}} = \begin{cases} p_g(t_2, \boldsymbol{y_i}) \text{ , if no newer keypoint sphere was accepted} \\ \\ p_g(t_{r,i,1}, \boldsymbol{y_i}) + d_{min}(\boldsymbol{p'_{b,i,t_{r,i,K},K}} - \boldsymbol{p_{b,i,t_{r,i,K}}}, (v_h + v_d) \\ \cdot (t_2 - t_{r,i,K}) \cdot \dfrac{\boldsymbol{p'_{b,i,t_{r,i,K},K}} - \boldsymbol{p_{b,i,t_{r,i,K}}}}{||\boldsymbol{p'_{b,i,t_{r,i,K},K}} - \boldsymbol{p_{b,i,t_{r,i,K}}}||_2}) \\ + \sum_{k=1}^{K-1} d_{min}(\boldsymbol{p'_{b,i,t_{r,i,k},k}} - \boldsymbol{p_{b,i,t_{r,i,k}}}, (v_h + v_d) \\ \cdot (t_{r,i,k+1} - t_{r,i,k}) \cdot \dfrac{\boldsymbol{p'_{b,i,t_{r,i,k},k}} - \boldsymbol{p_{b,i,t_{r,i,k}}}}{||\boldsymbol{p'_{b,i,t_{r,i,k},k}} - \boldsymbol{p_{b,i,t_{r,i,k}}}||_2} \text{ , else} \end{cases} \tag{7.16}$$

$$d_{min}(\boldsymbol{v_1}, \boldsymbol{v_2}) = \begin{cases} \boldsymbol{v_1}, & \text{if } ||\boldsymbol{v_1}||_2 < ||\boldsymbol{v_2}||_2 \\ \boldsymbol{v_2}, & \text{else} \end{cases}$$

While looking complex at first, this formula can be described in simple words. As long as no newer keypoint detection for keypoint i exists, the center of the keypoint sphere is not moved. Then, when a newer keypoint sphere arrives and is accepted, the old keypoint sphere is moved towards the center of the newest accepted keypoint sphere for keypoint i. In the equation, $d_{min}(\boldsymbol{p'_{b,i,t_{r,i,K},K}} - \boldsymbol{p_{b,i,t_{r,i,K}}}, (v_h + v_d) \cdot (t_2 - t_{r,i,K}) \cdot (\boldsymbol{p'_{b,i,t_{r,i,K},K}} - \boldsymbol{p_{b,i,t_{r,i,K}}})/||\boldsymbol{p'_{b,i,t_{r,i,K},K}} - \boldsymbol{p_{b,i,t_{r,i,K}}}||_2)$ is the movement that has been performed by the old keypoint sphere towards the newest keypoint up to t_2, while $\sum_{k=1}^{K-1} d_{min}(\boldsymbol{p'_{b,i,t_{r,i,k},k}} - \boldsymbol{p_{b,i,t_{r,i,k}}}, (v_h+v_d)\cdot(t_{r,i,k+1}-t_{r,i,k})\cdot(\boldsymbol{p'_{b,i,t_{r,i,k},k}}-\boldsymbol{p_{b,i,t_{r,i,k}}})/||\boldsymbol{p'_{b,i,t_{r,i,k},k}}-\boldsymbol{p_{b,i,t_{r,i,k}}}||_2)$ denotes the combined movement that was performed previously by the old keypoint sphere towards new keypoint spheres, except for the newest one. The formula for $\boldsymbol{p_{b,i,t_2}}$ can also be calculated in hard real-time, although K is not limited in theory. This problem can be resolved by an incremental calculation. Whenever a new keypoint sphere arrives at $t_{r,i,k}$ and is accepted, the shrinking keypoint sphere's position can be updated and saved, so that further calculations build upon $\boldsymbol{p_{b,i,t_{r,i,k}}}$. This way, the sum over $K-1$ previous position updates in Eq. (7.16) can be omitted, reducing the calculation to a limited number of simple operations. Furthermore, a minimum amount of time lies between the arrival of new keypoint spheres as the 3D human pose estimator cannot calculate them arbitrarily fast, which can be expected to take longer than the position update. Thus, a hard real-time capable adaptation can be achieved in practice. Figure 7.4 shows the adaptation process.

The final keypoint sphere adaptation mechanism works as follows: Per keypoint, a maximum number of up to two keypoint spheres are tracked. This always includes one growing keypoint sphere and potentially one shrinking keypoint sphere. The adaptation of keypoint spheres over time is performed as highlighted in Eq. (7.12) and (7.16). If a shrinking keypoint sphere exists, a new keypoint sphere is accepted if it is fully inside the shrinking one, replacing the growing one. If only a growing keypoint sphere exists, a new keypoint sphere is accepted if it is fully inside the growing one, turning that sphere into a shrinking one while becoming the growing sphere itself. A shrinking keypoint sphere is deleted, as soon as it is found to be fully encapsulated by the growing (due to the positional adaptation, the radius can not become zero before that). Adaptations to a keypoint sphere's radius and position are calculated whenever a new keypoint sphere for the i-th keypoint arrives (and is checked for acceptance), and when human pose estimation results are periodically requested at points in time ($t_{2,j}$, $t_{2,j+1}$, $t_{2,j+2}$,...). Performing adaptations incrementally instead of calculating Eq. (7.16) in full limits the number of operations that

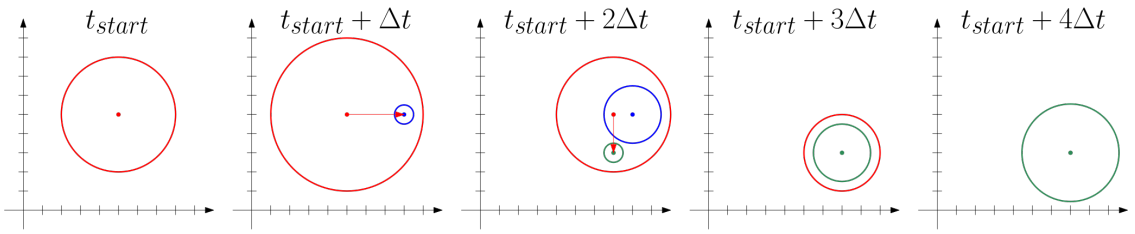


Figure 7.4: 2D illustration of keypoint sphere growth, shrinking, and movement in accordance with Eq. (7.14) and (7.16) using $v_h = v_d$. The red keypoint sphere is grown until $t_{start} + \Delta t$ when a new keypoint sphere arrives and is accepted. Then, it starts to shrink and move towards the new one. At $t_{start} + 2\Delta t$, an even newer keypoint sphere arrives and is accepted. The previously new sphere (blue) is deleted, and movement is presumed towards the newer one, until the sphere centers are the same at $t_{start}+3\Delta t$. The old sphere is finally deleted at $t_{start} + 3.5\Delta t$, with only the newest one remaining at $t_{start} + 4\Delta t$.

have to be performed per incremental update, as only the movement towards a single new keypoint sphere has to be considered. With the frequency at which updates are necessary being limited and a maximum of two keypoint spheres per keypoint that require updates through a limited number of simple operations, hard real-time capable updates can be achieved for a limited number of people and thus keypoints. When adapted keypoint spheres are requested for distance calculations, only one has to be returned, either the currently shrinking one if it exists (as it fully encapsulates the other) or the growing one otherwise. Thus, keeping track of two keypoint spheres per keypoint does not influence the number of spherical cones for the distance calculation. In the following, the keypoint sphere returned for a request from the distance calculation at t_2 is called $(\boldsymbol{p_{b,i,t_2}}, \boldsymbol{r_{b,i,t_2}})$.

7.5.2 Adaptations for Protective Separation Distance Factors

After a bridging method for resolving the missing hard real-time capability of the 3D human pose estimator has been introduced, further adjustments are necessary to incorporate the human-related factors from the left side of Eq. (7.6) into the adapted keypoint spheres and thus the human volume model. Together with a robot volume model that incorporates the respective factors for the robot, the calculated distance can be used to evaluate the altered protective separation distance $S'_p(t_0)$ from Eq. (7.6). So far, the potential movement of the i-th keypoint towards the robot between the start of the human pose estimation process at $t_{1,i}$ up to the time the respective keypoint sphere was requested at t_2 is already considered as part of $(\boldsymbol{p_{b,i,t_2}}, \boldsymbol{r_{b,i,t_2}})$. The same is true for the measurement error factor Z_d through the inclusion of individual predicted upper bounds for the measurement error. Remaining factors that have to be considered are the potential human movement after t_2 (that should be covered by $S'_{h,0}$) as well as elements of C that account for parts and dimension of the human body not being detected by the human pose estimation process.

First, the human-specific parts of C shall be discussed. The assumed 3D human pose estimator detects keypoint spheres that account for the keypoints positions and the worst-case measurement errors. However, the predicted keypoint positions typically fall inside the body and do not account for actual body dimensions, which is neither done by the radius of the predicted keypoint spheres so far. Hence, individual factors C_i that can be added to the radius of spheres are necessary to account for the human body dimensions. These surcharges to keypoint spheres must be sufficiently large to ensure that associated

spherical cones fully encapsulate the body parts they model. This shall be achieved for a perfect keypoint prediction with no measurement error (as measurement errors are already handled through other means). The wrist and elbow keypoints and the associated spherical cone will be used as an example. They will be denoted as keypoint i' and i'' respectively. Then, $C_{i'}$ must ensure that the whole wrist itself and the hand attached to it is covered (assuming no hand keypoint exists), and $C_{i''}$ must ensure that the whole elbow is covered. Further, the choice of $C_{i'}$ and $C_{i''}$ must ensure that the whole forearm is covered by the associated spherical cone. Each C_i value can be chosen as worst-case constant to achieve this, which can easily be added in hard real-time to the radius of keypoint spheres. They could e. g., be obtained based on worst-case measurements from all involved individuals.

Last, potential human movement after t_2 has to be considered, such that the altered protective separation distance remains valid for any potential point t_0 in time. After adapted keypoint spheres to account for missing hard real-time capability have been requested at t_2, the bridging method must be executed. The maximum time it takes from t_2 until the adapted keypoint spheres are available will be called Δt_b. Then, further adaptations for including the human-specific factors from the protective separation distance that shall be included in the distance calculation are necessary. The maximum time required for these further adaptations will be called Δt_a. Last, the distance calculation itself has to be performed. The maximum time necessary after the further adaptations up until the distance calculation has been performed and the evaluation against $S'_p(t_0)$ has finished will be called Δt_d. After the evaluation has finished, the human can still move towards the robot. Up until the altered protective separation distance is evaluated the next time, it has to remain valid, as the robot could otherwise not necessarily be stopped in time. If the evaluation is performed with a fixed frequency f_e, then further movement during $1/f_e$ has to be considered. During all these periods of time, worst-case human movement will be assumed, using a human movement speed v_h that is in line with safety standards like ISO 13855 [46] and ISO/TS 15066 [51]. This leads to the following further adaptations that have to be applied during Δt_a to the radius $\boldsymbol{r_{b,i,t_2}}$ of an adapted keypoint sphere $(\boldsymbol{p_{b,i,t_2}}, \boldsymbol{r_{b,i,t_2}})$ requested from the bridging method at t_2, while the position remains untouched:

$$\boldsymbol{r_{b,i,t_2}} + v_h \cdot (\Delta t_b + \Delta t_a + \Delta t_d + \frac{1}{f_e}) + C_i \tag{7.17}$$

It is known from the previous section that the maximum times Δt_b and Δt_d exist for a limited number of keypoint spheres and spherical cones, which is the case for a limited number of people and robots. With the above equation, the same is obviously true for Δt_a, the time during which the simple alterations of this equation have to be performed for the radius of each adapted keypoint sphere. Hence, all adaptations and the distance calculation can be performed in hard real-time for a limited number of people and robots.

After applying Eq. (7.17), the missing hard real-time capability is resolved safely for a limited number of people and human-specific factors for a comparison against $S'_p(t_0)$ are integrated. The volume model resulting from the respective spherical cones still exposes continuous surface change, as constant values are added only. Together with a safe robot volume model based on spherical cones that incorporates the robot-specific factors and exposes a continuous volume change, a continuous and hard real-time capable human-robot distance calculation of $d''_{h\text{-}r}$ can be realized for a limited number of people and robots. This shows how missing hard real-time capability of the assumed 3D human pose estimator can be resolved and how human pose estimation can be used in SSM.

7.6 Theoretical Analysis

Last, a theoretical comparison between the proposed approach for human-robot distance calculation based on human pose estimation and the use of a safety laser scanner for SSM will be performed. It shall be determined how well the proposed approach would be suited for human detection in an SSM application. Apart from having to fulfill necessary requirements for safety as hard real-time capability and low enough error rate, the suitability of a human detection method primarily depends on how often safety stops become necessary. A good value for assessing this is $d''_{h\text{-}r}$ from Eq. (7.6) – the larger $d''_{h\text{-}r}$ is in the same scenario the less likely it is that it is exceeded by $S'_p(t_0)$, which would necessitate a safety stop. Across different methods for human detection, $S'_p(t_0)$ has similar values with the only potential difference being deviations in its element $S'_{h,1}$ if different human movement speeds have to be assumed by human detection methods. However, these differences are not major, as safety-standard based human movement speeds range only from 1.6m/s to 2.0m/s [46]. Hence, comparing the performance of different human detection methods by assessing how large their $d''_{h\text{-}r}$ values are in the same scenario is suitable to assess whether they are roughly on the same level, or if one is significantly outperforming the other. In the following, let d''_{hpe} denote the value of $d''_{h\text{-}r}$ for the human pose estimation based approach, and let d''_{sls} be the respective value for the safety laser scanner.

The calculation of $d''_{h\text{-}r}$ depends on both, human- and robot-specific factors. From the robot-specific factors, the respective parts of C as well as the maximum amount of time that is required to obtain current robot data for the distance calculation are independent of the human detection method. Dependence exists for the maximum time needed for the distance calculation (depends on the representation of the human position), as well as its frequency (relevant for $S'_{h,0}$ and S'_r). By always using the same method to supply robot data, most major differences can be eliminated. To simplify the comparison of d''_{hpe} and d''_{sls}, the dependent robot-specific factors will be ignored, so that only the measured distance $d'_{h\text{-}r}$ and human-specific factors $S'_{h,0}$, C and Z_d have to be considered. For a rough comparison, this can be done, as the contribution of the time required for the distance calculation to $d''_{h\text{-}r}$ is small and neglectable (shown later). Furthermore, the difference in contribution to $d''_{h\text{-}r}$ for high frequencies is only a few centimeter for speeds around 1.6m/s and 2.0m/s. For the robot, this will be first ignored, but discussed later.

To calculate the human-specific factors, additional assumptions to those from Section 7.1 have to be made for both, the proposed approach and the safety laser scanner. Afterward, d''_{hpe} and d''_{sls} will be compared in two different scenarios, where the underlying setting will always be the interaction of one human with one robot featuring six axis.

Assumptions for the Proposed Approach Based on Human Pose Estimation: For the human modeling and distance calculation approach based on human pose estimation, the human and robot need to be modeled using spherical cones. Taking keypoints defined by the MPII Human Pose dataset [3] as foundation, the following keypoint pairs can be used to model different body parts: ankle and knee for the lower legs, knee and hip for the upper legs, wrist and elbow for the lower arm, elbow and shoulder for the upper arm, upper neck and top of the head for the combination of neck and head, as well as thorax and pelvis for the torso. This results in 10 spherical cones and 16 unique keypoints total that are required to create the human volume model. The six-axis robot will be modeled through 6 spherical cones total – one between each pair of axes that is connected by a link, and one for the base to the first axis.

Further assumptions have to be made for the calculation of $\boldsymbol{r_{b,i,t_2}}$, as well as for the human-specific factors from the protective separation distance that shall be included in the human volume model. This is made harder by the fact that the 3D human pose estimation system assumed in Section 7.1 is hypothetical in nature, thus an argument has to be made to obtain realistic estimates for required values. Whenever no (reasonable) worst-case scenario can be determined for a required value, the average or most likely case will be used, with the impact of deviations on d''_{hpe} being discussed later. To obtain a realistic value for $\boldsymbol{r_{b,i,t_2}}$, an average case for the predicted upper bound of the measurement error (that accounts for Z_d) is required first. For a reasonable estimate, experiments from Section 5.6.1 are used as a foundation. The best experiment achieved 94.7% correct results (meaning 5.3% have to be filtered out) with an average radius of 4.4 pixels at a heatmap resolution of 64×64. Assuming the human was not cropped too tightly, a human height of 50 pixels at heatmap resolution is reasonable. Assuming 180cm average body height, this translates to an average measurement error of $15,84$cm, with the rounded value of 16cm being used in the following. To model the time that passed between the start of the human pose estimation process at $t_{1,i}$ and the request for a updated keypoint sphere at t_2 it will be assumed that the 3D human pose estimator operates with a frequency of $f_{hpe} = 30$Hz, including the transfer to the bridging method – a reasonable assumption, given the fact that powerful human pose estimation methods like OpenPose are already able to achieve 22 outputs per second [11] (on older hardware than what is available today). Furthermore, it will be assumed that data acquisition also operates on 30Hz, including the transfer of data to the 3D human pose estimator. This means that the time between the start of the human pose estimation process and the arrival of a new result for the i-th keypoint at the bridging method is $t_{r,i} - t_{1,i} = 2 \cdot 1/f_{hpe}$ on average if a valid results for the keypoint exists. If not, that time is extended by $1/f_{hpe}$ every time, a case that is discussed later. With respect to the j-th request of updated keypoint spheres at $t_{2,j}$, the worst case scenario is that the previous request was made directly before $t_{r,i}$ arrived, thus $t_{2,j-1} < t_{r,i} < t_{2,j-1} + \epsilon$ for a small ϵ. Then, potential human movement during the whole time until the next request at $t_{2,j}$ has to be considered, which is about $1/f_d$ for a fixed frequency f_d at which the distance calculation is performed and thus requests keypoint spheres. Human movement with v_h has to be considered for all this time. As hands are included in the human volume model, $v_h = 2$m/s will be assumed based on ISO 13855 [46] (see table 2.1).

Apart from the calculation of $\boldsymbol{r_{b,i,t_2}}$, several fixed factors have to be defined. First, this includes the fixed frequency f_d at which the distance calculation operates in hard real-time. To achieve hard real-time in practice, f_d must be large enough that the bridging method, further keypoint sphere adaptations and the distance calculation itself can actually be performed within $1/f_d$. Of all these parts, the distance calculation is most complex. For ten spherical cones representing the human and six representing the robot, a total of 60 distance calculations between pairs of spherical cones have to be performed to find the minimum distance. In their work, Tornero et al. [124] evaluated the time necessary for a distance calculation between a pair of spherical cones, with the worst value observed being 1.6ms. However, computations where performed on a SUN SPARCStation 1, only featuring a CPU speed of 20 MHz [5]. With today's computational power of a single CPU core being about 150 to 300 times higher, it can realistically be assumed that a hard real-time capable implementation of the distance calculation between two spherical cones is able to calculate the distance for 60 of them in under 1ms. Bridging method and keypoint sphere adaptations should be even faster, as they only have to update the keypoint spheres with a few simple operations. The contribution of the maximum time required for all

three activities to d''_{hpe} is neglectable, as a duration of 1ms contributes only 0.2cm with $v_h = 2$m/s. For the frequency at which distance calculations are performed, $f_d = 50$Hz is chosen, which is reasonable due to $1/f_d >> 1$ms. The last required fixed factor are the surcharges for the body dimensions that are not detected by the human pose estimation, represented through C_i for the i-th keypoint. For the following scenarios, those values will be required for three keypoints: the thorax, the pelvis, and the wrist. Based on a measured worst-case torso width of 40cm among members of the lab, a radius surcharge C_i of 25cm will be chosen for the thorax and pelvis keypoint respectively, such that the whole torso is fully encapsulated by the associated spherical cone (note that spine flexibility might require more keypoints for a better approximation, which is ignored for this simple comparison). For the wrist keypoint, a worst-case distance of 20cm was measured to the tip of the hand, hence a C_i value of 22cm will be chosen for the wrist to fully encapsulate the hand (and forearm) with the resulting spherical cone between wrist and elbow.

With the assumptions made, reasonable values can be calculated for $\boldsymbol{r_{b,i,t_2}}$ (includes Z_d), $S'_{h,0}$ and C_i. As an adapted keypoint sphere reaches it's largest extent while growing, it will be assumed that $\boldsymbol{r_{b,i,t_2}}$ is growing. Then, necessary adaptations to a keypoint sphere for the distance calculation as described in Eq. (7.17) can be approximated as follows:

$$\begin{aligned}
\boldsymbol{r_{b,i,t_2}} =& v_h \cdot (2 \cdot \frac{1}{f_{hpe}} + \frac{1}{f_d}) + Z_d \\
=& 200\frac{\text{cm}}{\text{s}} \cdot (2 \cdot \frac{1}{30}\text{s} + \frac{1}{50}\text{s}) + 16\text{cm} \approx 33\text{cm} \\
S'_{h,0} = S'_{h,0,hpe} =& v_h \cdot (\Delta t_b + \Delta t_a + \Delta t_d + \frac{1}{f_d}) \\
\approx& 200\frac{\text{cm}}{\text{s}} \cdot (\frac{1}{50}\text{s}) = 4\text{cm} \\
C_i =& C_i
\end{aligned} \tag{7.18}$$

In this formula, Δt_b, Δt_a and Δt_d are removed due to their neglectable impact on the results. C_i remains dependent on the keypoint, while average, keypoint-independent values partially based on worst-case assumptions are calculated otherwise.

Assumptions for the Safety Laser Scanner: For the safety laser scanner, a representative device in form of the S3000 safety laser scanner [113] will be chosen to supply the human-specific values. Assuming that reflections are not present, the required surcharge Z_d for the measurement error of this device is 10cm [113]. For the human-specific C, the manual of the S3000 [113] refers to the formulas of ISO 13855 for calculation, meaning that C can be calculated depending on the installation height H_D as follows[46, 113]: $C = \min(120\text{cm} - 0.4 \cdot H_D, 85\text{cm})$. This means that mounting the safety laser scanner higher can reduce C down to a minimum of 85cm. However, mounting the safety laser scanner higher than 30cm makes additional safety measures against crawling beneath the scan plane necessary [46]. Thus, it will be assumed that the safety laser scanner will be mounted at 30cm height, leading to $C = 108$cm. For T'_r that is required to calculate $S'_{h,0}$, the laser scanner's basic response time of 60ms [113] will be used. For the distance calculation, it will be assumed that the laser scanner determines the distance from the intersection of the human with the scan plane to a 2D projection of the robot volume model onto the scan plane. Furthermore, the simplifying assumption will be made that the basic response time stays the same for this distance calculation and evaluation. Then, using the assumed human movement speed of 1.6m/s [46, 113] the last human-specific factor is $S'_{h,0} = S'_{h,0,sls} = 160\text{cm/s} \cdot 0.06\text{s} = 9,6\text{cm}$.

Scenario 1: For both human detection methods, the same robot volume model including identical robot-specific factors is used by prior assumptions. Thus, let d'_{hpe} denote the measured distance between robot volume model and the human based on human pose estimation, and let d'_{sls} denote the measured distance of the laser scanner between a 2D projection of the robot volume model into the scan plane and the human, both excluding human-specific surcharges. The first scenario is designed to evaluate a situation in which the actual distance of the robot volume model to the human d_1 equals the measured distance of the laser scanner, thus $d'_{sls} = d_1$. The distance is measured based on where the outerior of the leg intersects the scan plane at the height of 30cm. In 3D, it is assumed that the same closest distance exists between (the outerior of) the human torso and the 3D robot volume model. For this to be true, the human is assumed to stand straight up with arms at his side, so that no other points of the human body get closer to the robot. Based on keypoint positions only, d'_{hpe} measures the distance to the line connecting two keypoints, which would be thorax and pelvis representing the torso in this case. Assuming a torso-depth of 30cm (largest value of people in the lab) and that the line runs through the middle of the torso leads to $d'_{hpe} = d' + 15cm$ (line is 15cm inside the torso).

Next, d''_{sls} can be calculated by directly using the values defined in the assumptions. The calculation of d''_{hpe} is more complex due to keypoint-specific values being used that define the radius of keypoint spheres. They affect the distance calculation, as well as which spherical cone is closest to the robot. Given that the final adapted radius of each keypoint sphere is calculated based on the same average values except for C_i in this analysis, it can be assumed that the spherical cone for the torso is closest to the robot, as the encapsulated thorax is the closest human body part and the individual assumed C_i values are largest for thorax and pelvis which define the torso spherical cone. As thorax and pelvis have the same assumed C_i value and all other elements contributing to the radius are also assumed to be the same for this comparison, the torso spherical cone consists of an infinitely large amount of spheres with identical radius (equal to thorax and pelvis sphere radius). Hence, the shortest distance from every point on the surface of the spherical cone to the line connecting the sphere positions equals this radius. Hence, the measured distance d'_{hpe} is foreshortened by $\boldsymbol{r_{b,i,t_2}} + S'_{h,0} + C_i$ to form d''_{hpe}. Using the assumed values for elements of this equation and $C_i = 25$cm for thorax and pelvis, d''_{hpe} and d''_{sls} can be compared:

$$\begin{aligned} d''_{hpe} - d''_{sls} =& (d'_{hpe} - \boldsymbol{r_{b,i,t_2}} - S'_{h,0,hpe} - C_i) - (d'_{sls} - S'_{h,0,sls} - C - Z_d) \\ =& ((d_1 + 15\text{cm}) - 33\text{cm} - 4\text{cm} - 25\text{cm}) \\ & - (d_1 - 9,6\text{cm} - 108\text{cm} - 10\text{cm}) \\ =& -47\text{cm} + 127.6\text{cm} = 80.6\text{cm} \end{aligned} \tag{7.19}$$

This result means that the proposed approach based on human pose estimation strongly outperforms the use of the safety laser scanner, as the latter requires a safety stop 80.6cm earlier for the same $S'_p(t_0)$. The difference is hereby strong enough that it excuses inaccuracies of the assumptions and simplifications made, with a few centimeters in difference not impacting the interpretation of the result. If e. g., the (average) measurement error would be twice as large for the proposed approach (32cm instead of 16cm), a notable difference of 64.6cm would yet remain. Whenever a valid keypoint sphere is missing, d''_{hpe} would be further decreased by $v_h \cdot 1/f_{hpe} = 6.67$cm, which can happen 12 times in a row before the advantage of 80.6cm is used up. This leads to the conclusion that the proposed approach is better suited for the first scenario.

Scenario 2: The major advantage of the proposed method in the first scenario can be explained by the necessary surcharge of $C = 108\text{cm}$ for the laser scanner required to account for all potential upper body positions. As the smallest distance to the upper body was the same as the distance to the leg, this was a massive over-approximation. Hence, the second scenario shall incorporate a situation with less over-approximation where the upper body is significantly closer to the robot than the position of the legs by fully stretching out one arm towards the robot.

Except for one arm being stretched out towards the robot, the second scenario will be the same as the first one. This means that nothing changes for the calculation of d''_{sls}. However, changes to d'_{hpe} and d''_{hpe} occur. Compared to the first scenario, where the minimum distance between 3D robot volume model and human was the distance to the thorax, it is now assumed that this distance is foreshortened by a full arm's length when the arm is fully stretched out towards the robot. To determine what a full arm's length is, it will be assumed that the human is 180cm tall. Then, the human body model employed by Winter [132] can be used to determine the arm's length based on the person's height, leading to 79.2cm for the full arm (including the hand) and 19.44cm from the wrist to the tip of the hand. As the distance to the thorax was d_1, the distance to the tip of the hand will now be $d_1 - 79.2\text{cm}$. For the measured distance d'_{hpe}, the arm being fully stretched out means that the closest detected part of the human is the wrist keypoint, 19.44cm further away from the robot than the tip of the fingers, thus $d'_{hpe} = d_1 - 79.2\text{cm} + 19.44\text{cm} = d_1 - 59.76\text{cm}$. For the scenario with the tip of the hand being closest to the robot, it is reasonable to assume that the spherical cone modeling the lower arm is closest to the robot, with the keypoint sphere around the wrist being the closest part towards the robot volume model. Using $C_i = 22\text{cm}$ for the wrist, comparison between d''_{hpe} and d''_{sls} can be performed as in Eq. (7.19) using the new values for the approach based on human pose estimation:

$$\begin{aligned} d''_{hpe} - d''_{sls} =& ((d_1 - 59.76\text{cm}) - 33\text{cm} - 4\text{cm} - 22\text{cm}) \\ & - (d_1 - 9,6\text{cm} - 108\text{cm} - 10\text{cm}) \\ =& -118,76\text{cm} + 127,6\text{cm} = 8,84\text{cm} \end{aligned} \tag{7.20}$$

The performance difference for scenario two is no longer sufficient to determine if one method is significantly better than another, given the inaccuracies due to simplifications and assumptions. It can be said that both approaches are expected to perform on roughly the same level. For example, a 10cm higher predicted upper bound for the measurement error or two missing detections in a row would already turn the result in favor of the laser scanner here. The human leaning further towards the robot would have the same effect.

Summary: The theoretical comparison under the given simplifications and assumptions has shown that the proposed approach for hard real-time capable human modeling and human-robot distance calculation based on human pose estimation can be beneficial compared to a safety laser scanner when employed in an SSM scenario. Especially when the human keeps the arms close to the body and does not fully stretch them out, a large potential advantage can be observed due to large surcharges required by the laser scanner to account for all potential upper body positions. When the hand was fully stretched out towards the robot, both methods exposed comparable performance. For the many practical tasks where the arms do not need to be fully stretched out, the method shows great potential over the laser scanner. How large the advantage will be can be determined as soon as a system similar to the hypothesized one in this theoretical comparison is developed and put into service, so that the proposed approach can be applied in practice.

8 Discussion, Outlook and Conclusion

Throughout this work, methods have been proposed and evaluated that aim at improving shortcomings of human pose estimation methods that prohibit their use in safety-critical applications like SSM. These shortcomings include an error rate that is too high, as well as missing hard real-time capability. Throughout this chapter, the contributions of this work towards resolving these shortcomings will be recapitulated and discussed, including the identification of necessary future work for their full resolution.

8.1 Discussion

This work aimed to answer four central research questions revolving around the ultimate goal of enabling the use of human pose estimation in safety-critical applications. In essence, these questions aimed at the reduction of previously undetected errors, the prediction of upper bounds for measurement errors, the potential impact and handling of noise, and resolving the lack of hard real-time capability in human pose estimation. In the following, the contributions to these points will be recapped and discussed.

Error Reduction: A low enough rate of dangerous errors (e. g., below 10^{-6} dangerous errors per hour for robotic applications [48, 53]) is a crucial property that any system employed in a safety-critical scenario must possess. Current human pose estimation methods do not. To reduce the number of undetected errors experienced in 2D single-person human pose estimation, a diverse neural network ensemble was proposed, which employs two or more diverse neural networks (different architecture/training) and compares their results for the same input to decide whether or not a result has to be filtered out, thereby creating an implementation of the essential safety concept of diversity for human pose estimation. The comparison was performed based on individual thresholds derived from human pose estimation results of the diverse neural networks for each input. As a means of comparison, two methods for error detection were used that relied on the thresholding of heatmap values, thus requiring a heatmap-based human pose estimation method.

Methods were evaluated on the MPII Human Pose dataset [3]. All of them reduced the amount of previously undetected incorrect results significantly while retaining a high amount of correct results. In contrast to the heatmap-based methods, the diverse neural network ensemble makes no assumptions about the kind of human pose estimation method used, making it generally applicable. However, none of the methods could completely reduce the number of incorrect results while retaining a high number of correct ones. Furthermore, evaluations were performed on the correctness definition used by MPII, which is not particularly suitable for safety – an issue that was resolved later. Although the diverse neural network ensemble could not eliminate all incorrect detections

on its own, it shows great potential for further development due to its ability to eliminate a significant amount of incorrect detections with only small losses to correct ones. The potential is further highlighted by the fact that the original work of the author on this topic [106] was picked up by the International Organization for Standardization (ISO) in their recent publication ISO/IEC TR 5469 on 'Functional safety and AI systems' [54].

Measurement Error Estimation: In terms of safety, the shortcomings of typical correctness definitions for 2D single-person human pose estimation based on thresholding the distance between annotated and detected keypoint positions using a threshold derived from human annotations were highlighted. Such thresholds are (i) not available at inference time and (ii) not a suitable representation of a result's positional uncertainty and thus the potential measurement error. Therefore, a reformulation of the human pose estimation problem was proposed with a new correctness definition that aims at the specific needs of safety applications: together with each predicted keypoint position, an upper bound for the measurement error of this position shall be predicted. A prediction is considered correct if the distance to the annotated keypoint position is lower than the predicted upper bound. In addition to being available at inference time, this allows the prediction of upper bounds that are actually tailored towards the positional uncertainty of each keypoint prediction. Additional quality criteria are used to ensure meaningful results.

To tackle the reformulated 2D single-person human pose estimation problem, several approaches and neural network architectures were designed that either tried to solve the problem based on a direct prediction of the measurement error or by predicting the parameters of a Gaussian distribution and inferring upper bounds based on the standard deviations. While methods based on direct measurement error prediction did not fare well, Gaussian-based methods exposed desired behavior, with the best method predicting correct upper bounds in almost 95% of all cases while exposing decent quality regarding the introduced quality metrics. While 100% correct results would have been desirable, separate methods could be employed to deal with the remaining incorrect results, like those previously introduced. A realization of this combination constitutes potential future work. Overall, the Gaussian-based methods and networks proved to be suitable solutions for the prediction of upper bounds for the measurement error in human pose estimation, while necessary improvements entail the removal of the remaining incorrect results and optional improvements include the quality of predicted upper bounds.

Impact and Handling of Noise: Proper functioning of devices and methods employed in safety-critical applications has to be ensured not only under ideal conditions but under all foreseeable, likely enough conditions that can occur within the limits of the machinery. This includes the occurrence of noise, a common threat to the proper functioning of neural networks, with the negative effect on 2D single-person human pose estimation being verified through experiments. An argument was made that all dangerous noise types for a specific realization of a safety-critical industrial application can be identified due to restrictions from the limits of the machinery and the mandatory risk assessment that has to take place. This led to the conclusion that noise-specific countermeasures can be employed to deal with noise in safety-critical applications.

To deal with specific noise types, two general strategies were explored: training against noise, as well as using a dedicated denoiser. Existing denoisers and training strategies from outside the domain of human pose estimation were applied to the problem in a large variety of experimental settings, using Gaussian noise and a slightly altered version thereof as exemplary specific noise types. The experiments were able to show that all

methods are capable of reducing the negative impact of the specific noise type they were designed against. However, in some cases, specific precautions (like the retention of some amounts of clean training data) had to be taken to ensure that no loss of performance on clean data occurred. Sometimes, it was even possible to slightly improve clean data performance, while also improving performance under noise. The impact of small amounts of noise was fully negated by some of the methods, while this was no longer possible for increasingly large amounts of noise. Whether these amounts can occur in a specific safety-critical application is up for debate and would have to be discussed on a case-to-case basis. Nevertheless, significant improvements can be achieved with the examined approaches. One potential problem that was identified during the experiments is that slight alterations in the noise type can already cause malfunctions of the training-based countermeasures. While not invalidating the proposed approaches, this calls for a clear identification and definition of noise types that can occur, with further research into countermeasures that are resistant to slight noise-type alterations being desirable.

Hard Real-Time Capability: While other parts of this work were more or less directly targeted towards correctness and error reduction in (2D single-person) human pose estimation, further contributions were made towards the second major factor that prohibits the application of human pose estimation in safety-critical scenarios: the missing hard real-time capability. The problem was addressed in the context of an SSM application, which requires an (almost) error-free and hard real-time capable method for human detection. In contrast to other contributions, the problem had to be approached for 3D human pose estimation. For an isolated investigation on hard real-time capability, a 3D human pose estimation system was assumed that is not hard real-time capable and employs approaches from other parts of this work to become error-free, meaning that it predicts upper bounds for the measurement error alongside keypoint positions and filters out potentially incorrect results in every output. Hereby, the combination of keypoint position and associated upper bound constitutes a uniform sphere in 3D space, which is called a keypoint sphere.

The problem of missing hard real-time capability was resolved through a combination of two things: a bridging method and a human volume model derived from human pose estimation results. The bridging method is designed to adapt previous human pose estimation results for a keypoint, such that a safely adapted keypoint sphere can be returned whenever it is necessary for distance calculations of the SSM method. The adaptations are designed to account for the worst-case movement of the human from the start of the human pose estimation process up until results are required, while they also ensure that the human volume model will have a continuous volume change. The human volume model is created by using multiple spherical cones which are derived from pairs of keypoint spheres. Several factors from the protective separation distance are also incorporated into the volume model by adjusting the respective keypoint spheres and thus the resulting spherical cones. For calculating the distance between a single pair of spherical cones, a hard real-time capable analytical solution already exists. To calculate distances in an SSM application in hard real-time, the robot(s) must also be modeled with spherical cones, and the number of humans and robots must be limited. Then, the distance calculation is capable of hard real-time. The same is true for the bridging method and further adaptations – as long as the amount of people and thus keypoints is limited, they can be calculated in hard real-time. A theoretical comparison against a safety laser scanner showed that the method has especially high potential for practical application in scenarios where the arms are close to the body, where it significantly outperforms the safety laser scanner. Overall, the approach can be used to achieve hard real-time capability for human pose estimation,

even if the human pose estimator itself is not hard real-time capable, and is promising for practical application in SSM. However, two potential downsides remain: First is the required limit for the number of humans and robots to achieve hard real-time capability. While robots can be limited by the design of the robot work cell, additional measures are necessary to limit the number of people. Second, there is the initialization and shutdown problem: For initialization, the proposed bridging method requires one detection for every relevant keypoint before it starts. On the other hand, once bridging starts, it must be determined when the bridging has to stop (e. g., when people are leaving the work cell). Both issues could potentially be resolved through a combination with established safety measures, which could be used for safe operation until the initialization is finished and to determine when a person left and no further bridging is required.

8.2 Outlook

This work has explored how human pose estimation methods can be brought closer to an application in safety-critical scenarios by addressing and improving upon the major problems that prohibit their use in such scenarios. While previously undetected errors were significantly reduced and the human pose estimation problem itself was altered to better suit the needs of safety, further reductions in the error rate are necessary for compliance with safety standards. Furthermore, most methods were discussed for fundamental 2D single-person human pose estimation, while safety-critical applications would require 3D human pose estimation, ideally for multiple people. This lays out several topics that can or have to be assessed through future research.

Further Error Reduction: The methods for error reduction presented in this work were able to significantly reduce the amount of undetected errors in human pose estimation, however, not to a level required by safety standards. This makes further improvements a necessity. Future research could improve upon the proposed diverse neural network ensemble by increasing the diversity of employed methods or by training the neural networks in a way that favors the prediction of significantly different results among them in case of existing uncertainty. In the spirit of further leveraging the safety concept of diversity, a second, functionally completely different method for error detection could be developed and deployed in combination with the diverse neural network ensemble.

Combination of Methods: Throughout the work, the diverse neural network ensemble has been proposed for error reduction, and upper bound prediction for the measurement error was pursued through other methods. However, both have not been combined so far, which could e. g., be done to improve predicted upper bounds and/or to eliminate incorrect ones. To truly leverage the concept of diversity in this context, two functionally different ways to calculate the upper bounds would be required. However, only approaches employing a Gaussian-based strategy were successful for upper-bound prediction in this work. Therefore, further research for the development of a second successful strategy would be required, before the concept of a diverse neural network ensemble could be altered and applied.

From 2D to 3D: Methods for error reduction and measurement error upper bound prediction were focused on the fundamental problem of 2D single-person human pose estimation in this work. However, to e. g., calculate the human-robot distance in 3D space, 3D keypoint predictions are necessary. A potential solution to this problem could be pursued

by keeping the application of the developed methods at the 2D level, with 3D results being obtained by projecting keypoint positions and measurement errors into 3D through a stereo vision approach. Additional errors resulting from this procedure would have to be determined and incorporated. Another potential solution would be the adaptation of these approaches to 3D human pose estimation methods, as neither the diverse neural network ensemble nor the upper bound prediction is strictly limited to 2D human pose estimation. However, they would have to be realized in slightly different ways.

Single- to Multi-Person: A potential future improvement would be a step from single-person human pose estimation to multi-person human pose estimation. While a robot work cell could be designed for a single human with additional measures being in place to ensure that only a single human is present, this would mean a restriction to the kinds of applications that can be realized. Hence, enabling multi-person human pose estimation for a limited number of people could be a desirable goal. However, pursuing safe multi-person human pose estimation introduces new problems to safety. For example, keypoint detection could be associated with the incorrect person in bottom-up methods, or human instances could be completely missed or incorrectly located in top-down methods. The safe resolution of these problems poses material for future research.

Incorporation of Sequence Data: Further potential improvements for the error reduction could be achieved by leveraging the fact that data for human pose estimation arrives most likely in the form of a data sequence in a practical application, instead of the single, uncorrelated data points that are used in standard 2D human pose estimation. Consistency of new detections with previous ones could be assessed, or potential human movement could be forecast to detect unlikely and thus potentially incorrect results. However, such approaches do also have the risk that errors propagate forward, with errors in the past impacting the capability to detect errors in the present. Appropriate measures to deal with this problem would have to be put in place.

8.3 Conclusion

In this work, the primary problems that prohibit the use of human pose estimation in safety-critical industrial applications were addressed. The goal was to bring human pose estimation closer to being applied in such scenarios by reducing the amount of undetected errors and resolving the missing hard real-time capability. To these goals, the following primary contributions were made:

Diverse Neural Network Ensembles for Error Reduction: The use of the safety concept of diversity for error reduction was realized through the proposed diverse neural network ensemble, which can detect incorrect results through a result comparison from functionally different neural networks. During experiments on the MPII Human Pose dataset, the lowest error rate achieved was 4.2%. This is a significant reduction, as the error rates of the individual neural networks involved in this ensemble were 12.6%, 12.9%, and 14.6%. Furthermore, the ensemble achieved a correct detection rate of 79%.

A Safety-Focused Reformulation of the Human Pose Estimation Problem: The standard 2D single-person human pose estimation problem only requires the prediction of keypoint positions and relies on human annotations to define thresholds that represent the acceptable deviation from prediction to annotation. These thresholds define upper bounds

for the allowed measurement error. However, annotations are not available at inference time, where an upper bound for the measurement error is required to ensure safety. Therefore, a reformulation of the human pose estimation problem was proposed that requires the prediction of upper bounds for the measurement error alongside keypoint positions, thereby making them available at inference time. Further quality measures were proposed to ensure meaningful results.

The Prediction of Upper Bounds for the Measurement Error: To solve the reformulated human pose estimation problem, different methods for the prediction of upper bounds were proposed. Approaching human pose estimation as a distribution prediction problem where the keypoint position is the mean of a Gaussian distribution and the upper bound for the measurement error is obtained based on three times the standard deviations showed the best success. It was possible to determine a correct upper bound in 94.7% of all cases while having decent quality.

Large-Scale Evaluation of Countermeasures against Noise: A large-scale investigation on human pose estimation under noise was conducted, as noise poses a severe risk to correct neural network functioning and is thus a potential source of errors. Based on restrictions from safety standards, it was argued that noise-specific countermeasures can be used, and experiments were performed using Gaussian noise and a slightly changed form of it as exemplary specific noise types. The impact of noise on human pose estimation as well as the effectiveness of a variety of countermeasures was evaluated. Results showed that it was possible to negate the impact of small amounts of noise completely without a negative impact on the clean data performance. Furthermore, a significant performance decrease was observed for training-based countermeasures when slight changes to the noise type were performed, which highlighted the need for a close definition of noise types that can occur in a specific application.

Hard Real-Time Capability through Bridging and Volume Modeling: A method to resolve the missing hard real-time capability of human pose estimation was proposed. It builds upon the combination of a bridging algorithm, that safely adapts human pose estimation results under consideration of the time that passed since their prediction process started, and a volume model, that incorporates other safety-relevant factors. Both, the bridging algorithm and the model can be calculated in hard real-time for a limited number of people, and the volume model allows distance calculations in hard real-time for a limited number of persons and robots through existing methods. Together, they not only resolve the problem of missing hard real-time capability in human pose estimation but also resolve that problem for necessary distance calculations. In a theoretical comparison, the potential advantage of the proposed approach over a safety laser scanner in an SSM application was highlighted, showing that fewer violations of the protective separation distance can be expected from the proposed approach.

Overall, this work has achieved various improvements regarding the problems that prohibit the use of human pose estimation in safety-critical industrial applications. The human pose estimation problem was adapted in a way that makes it applicable in safety-critical scenarios and errors were significantly reduced, although not to the level required by safety standards yet. Hard real-time capability has been achieved for a limited number of people and robots through the proposed bridging method. Through all of this, a strong foundation for further work is provided. Together with future research on error-reduction methods for human pose estimation, this work can guide the way to the safe use of human pose estimation in safety-critical industrial applications.

Bibliography

[1] Alexander Amini, Wilko Schwarting, Ava Soleimany, and Daniela Rus. Deep evidential regression. In *Advances in Neural Information Processing Systems*, volume 33, pages 14927–14937. Curran Associates, Inc., 2020.

[2] António Amorim, Diana Guimares, Tiago Mendona, Pedro Neto, Paulo Costa, and António Paulo Moreira. Robust human position estimation in cooperative robotic cells. *Robotics and Computer-Integrated Manufacturing*, 67:102035, 2021.

[3] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2D human pose estimation: New benchmark and state of the art analysis. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3686–3693. IEEE, 2014.

[4] Mykhaylo Andriluka, Umar Iqbal, Eldar Insafutdinov, Leonid Pishchulin, Anton Milan, Juergen Gall, and Bernt Schiele. PoseTrack: A benchmark for human pose estimation and tracking. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5167–5176. IEEE, 2018.

[5] Andreas V Bechtolsheim and Edward H Frank. Sun's SPARCstation 1: A workstation for the 1990s. In *Digest of Papers Compcon Spring '90. Thirty-Fifth IEEE Computer Society International Conference on Intellectual Leverage*, pages 184–188. IEEE, 1990.

[6] Lorenzo Bertoni, Sven Kreiss, and Alexandre Alahi. MonoLoco: Monocular 3D pedestrian localization and uncertainty estimation. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6860–6870. IEEE, 2019.

[7] Lennart Bramlage, Michelle Karg, and Cristóbal Curio. Plausible uncertainties for human pose regression. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 15087–15096. IEEE, 2023.

[8] Adrian Bulat, Jean Kossaifi, Georgios Tzimiropoulos, and Maja Pantic. Toward fast and accurate human pose estimation via soft-gated skip connections. In *2020 15th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2020)*, pages 8–15. IEEE, 2020.

[9] Yuanhao Cai, Zhicheng Wang, Zhengxiong Luo, Binyi Yin, Angang Du, Haoqian Wang, Xiangyu Zhang, Xinyu Zhou, Erjin Zhou, and Jian Sun. Learning delicate local representations for multi-person pose estimation. In *Computer Vision – ECCV 2020*, pages 455–472. Springer International Publishing, 2020.

[10] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2D pose estimation using part affinity fields. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1302–1310. IEEE, 2017.

[11] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. OpenPose: Realtime multi-person 2D pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(1):172–186, 2019.

[12] João Carreira, Pulkit Agrawal, Katerina Fragkiadaki, and Jitendra Malik. Human pose estimation with iterative error feedback. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4733–4742. IEEE, 2016.

[13] Yilun Chen, Zhicheng Wang, Yuxiang Peng, Zhiqiang Zhang, Gang Yu, and Jian Sun. Cascaded pyramid network for multi-person pose estimation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7103–7112. IEEE, 2018.

[14] Bowen Cheng, Bin Xiao, Jingdong Wang, Honghui Shi, Thomas S Huang, and Lei Zhang. HigherHRNet: Scale-aware representation learning for bottom-up human pose estimation. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5385–5394. IEEE, 2020.

[15] Chia-Jung Chou, Jui-Ting Chien, and Hwann-Tzong Chen. Self adversarial training for human pose estimation. In *2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 17–30. IEEE, 2018.

[16] Xiao Chu, Wei Yang, Wanli Ouyang, Cheng Ma, Alan L Yuille, and Xiaogang Wang. Multi-context attention for human pose estimation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5669–5678. IEEE, 2017.

[17] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Image denoising by sparse 3-D transform-domain collaborative filtering. *IEEE Transactions on Image Processing*, 16(8):2080–2095, 2007.

[18] Qi Dang, Jianqin Yin, Bin Wang, and Wenqing Zheng. Deep learning based 2D human pose estimation: A survey. *Tsinghua Science and Technology*, 24(6):663–676, 2019.

[19] Terrance DeVries and Graham W Taylor. Learning confidence for out-of-distribution detection in neural networks. *arXiv preprint arXiv:1802.04865*, 2018.

[20] Directorate-General for Communication. Types of legislation. `https://european-union.europa.eu/institutions-law-budget/law/types-legislation_en`. Accessed: 2024-01-22.

[21] Directorate-General for Internal Market, Industry, Entrepreneurship and SMEs. Harmonised standards. `https://single-market-economy.ec.europa.eu/single-market/european-standards/harmonised-standards_en`. Accessed: 2024-01-23.

[22] Directorate-General for Internal Market, Industry, Entrepreneurship and SMEs. Summary of references of harmonised standards published in the Official Journal – Directive 2006/42/EC of the European Parliament and of the Council of 17 May 2006 on machinery, and amending Directive 95/16/EC. `https://ec.europa.eu/docsroom/documents/55576`, 2023. Accessed: 2024-01-23.

[23] Junting Dong, Wen Jiang, Qixing Huang, Hujun Bao, and Xiaowei Zhou. Fast and robust multi-person 3D pose estimation from multiple views. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7784–7793. IEEE, 2019.

[24] Marcin Eichner, Manuel Marin-Jimenez, Andrew Zisserman, and Vittorio Ferrari. 2D articulated human pose estimation and retrieval in (almost) unconstrained still images. *International Journal of Computer Vision*, 99:190–214, 2012.

[25] European Union. Directive 2001/95/EC of the European Parliament and of the Council of 3 December 2001 on general product safety. *Official Journal of the European Union (OJ)*, L 11:4–17, 2002.

[26] European Union. Directive 2006/42/EC of the European Parliament and of the Council of 17 May 2006 on machinery, and amending Directive 95/16/EC (recast). *Official Journal of the European Union (OJ)*, L 157:24–86, 2006.

[27] European Union. Directive 2009/48/EC of the European Parliament and of the Council of 18 June 2009 on the safety of toys. *Official Journal of the European Union (OJ)*, L 170:1–37, 2009.

[28] European Union. Directive 2014/35/EU of the European Parliament and of the Council of 26 February 2014 on the harmonisation of the laws of the member states relating to the making available on the market of electrical equipment designed for use within certain voltage limits. *Official Journal of the European Union (OJ)*, L 96:357–374, 2014.

[29] European Union. Consolidated version of the Treaty on the Functioning of the European Union: Part six - institutional and financial provisions: Title I - institutional provisions: Chapter 2 - legal acts of the Union, adoption procedures and other provisions: Section 1 - the legal acts of the Union: Article 288 (ex article 249 TEC). *Official Journal of the European Union (OJ)*, C 202:171–172, 2016.

[30] Hao-Shu Fang, Shuqin Xie, Yu-Wing Tai, and Cewu Lu. RMPE: Regional multi-person pose estimation. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2353–2362. IEEE, 2017.

[31] Federal Ministry of Justice and Federal Office of Justice. Gesetz über die Bereitstellung von Produkten auf dem Markt (Produktsicherheitsgesetz - ProdSG). (German) [Act on making products available on the market (product safety act - ProdSG)]. `https://www.gesetze-im-internet.de/prodsg_2021/ProdSG.pdf`, 2021. Accessed: 2024-01-23.

[32] Federal Ministry of Justice and Federal Office of Justice. Neunte Verordnung zum Produktsicherheitsgesetz (Maschinenverordnung) (9. ProdSV). (German) [Ninth ordinance on the product safety act (machinery regulation) (9. ProdSV)]. `https://www.gesetze-im-internet.de/gsgv_9/9._ProdSV.pdf`, 2021. Accessed: 2024-01-23.

[33] Fabrizio Flacco, Torsten Kröger, Alessandro De Luca, and Oussama Khatib. A depth space approach to human-robot collision avoidance. In *2012 IEEE International Conference on Robotics and Automation*, pages 338–345. IEEE, 2012.

[34] Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 1050–1059. PMLR, 2016.

[35] Miniar Ben Gamra and Moulay A Akhloufi. A review of deep learning techniques for 2D and 3D human pose estimation. *Image and Vision Computing*, 114:104282, 2021.

[36] Jakob Gawlikowski, Cedrique Rovile Njieutcheu Tassi, Mohsin Ali, Jongseok Lee, Matthias Humt, Jianxiang Feng, Anna Kruspe, Rudolph Triebel, Peter Jung, Ribana Roscher, et al. A survey of uncertainty in deep neural networks. *Artificial Intelligence Review*, 56(Suppl 1):1513–1589, 2023.

[37] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231*, 2018.

[38] Robert Geirhos, Carlos RM Temme, Jonas Rauber, Heiko H Schütt, Matthias Bethge, and Felix A Wichmann. Generalisation in humans and deep neural networks. In *Advances in Neural Information Processing Systems*, volume 31, pages 7538–7550. Curran Associates, Inc., 2018.

[39] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[40] Nitesh B. Gundavarapu, Divyansh Srivastava, Rahul Mitra, Abhishek Sharma, and Arjun Jain. Structured aleatoric uncertainty in human pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, volume 2, pages 50–53, 2019.

[41] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778. IEEE, 2016.

[42] Emily J. Herron, Steven R. Young, and Thomas E. Potok. Ensembles of networks produced from neural architecture search. In *High Performance Computing*, pages 223–234. Springer International Publishing, 2020.

[43] David Honzátko and Martin Kruliš. Accelerating block-matching and 3D filtering method for image denoising on GPUs. *Journal of Real-Time Image Processing*, 16 (6):2273–2287, 2019.

[44] Eyke Hüllermeier and Willem Waegeman. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine Learning*, 110(3):457–506, 2021.

[45] International Electrotechnical Commission. Functional safety of electrical/electronic/programmable electronic safety-related systems – part 7: Overview of techniques and measures (IEC 61508-7:2010); German version EN 61508-7:2010. Standard IEC 61508-7:2010, 2011.

[46] International Organization for Standardization. Safety of machinery - positioning of safeguards with respect to the approach speeds of parts of the human body; German version EN ISO 13855:2010. Standard ISO 13855:2010, 2010.

[47] International Organization for Standardization. Safety of machinery - general principles for design - risk assessment and risk reduction; German version EN ISO 12100:2010. Standard ISO 12100:2010, 2011.

[48] International Organization for Standardization. Robots and robotic devices – safety requirements for industrial robots – part 1: Robots; German version EN ISO 10218-1:2011. Standard ISO 10218-1:2011, 2012.

[49] International Organization for Standardization. Robots and robotic devices - safety requirements for industrial robots - part 2: Robot systems and integration; German version EN ISO 10218-2:2011. Standard ISO 10218-2:2011, 2012.

[50] International Organization for Standardization. Safety of machinery - safety-related parts of control systems - part 2: Validation; German version EN ISO 13849-2:2012. Standard ISO 13849-2:2012, 2013.

[51] International Organization for Standardization. Robots and robotic devices - collaborative robots (ISO/TS 15066:2016). Technical Specification ISO/TS 15066:2016, 2017.

[52] International Organization for Standardization. Safety of machinery - positioning of safeguards with respect to the approach speeds of parts of the human body; German and English version prEN ISO 13855:2022. Standard ISO 13855:2022, 2022.

[53] International Organization for Standardization. Safety of machinery - safety-related parts of control systems - part 1: General principles for design; German version EN ISO 13849-1:2023. Standard ISO 13849-1:2023, 2023.

[54] International Organization for Standardization and International Electrotechnical Commission. Artificial intelligence — functional safety and AI systems. Technical Report ISO/IEC TR 5469:2024, 2024.

[55] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6M: Large scale datasets and predictive methods for 3D human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1325–1339, 2013.

[56] Arjun Jain, Jonathan Tompson, Yann LeCun, and Christoph Bregler. MoDeep: A deep learning framework using motion features for human pose estimation. In *Computer Vision – ACCV 2014*, pages 302–315. Springer International Publishing, 2015.

[57] Sheng Jin, Lumin Xu, Jin Xu, Can Wang, Wentao Liu, Chen Qian, Wanli Ouyang, and Ping Luo. Whole-body human pose estimation in the wild. In *Computer Vision – ECCV 2020*, pages 196–214. Springer International Publishing, 2020.

[58] Sam Johnson and Mark Everingham. Clustered pose and nonlinear appearance models for human pose estimation. In *Proceedings of the British Machine Vision Conference*, pages 12.1–12.11. BMVA Press, 2010.

[59] Sam Johnson and Mark Everingham. Learning effective human pose estimation from inaccurate annotation. In *CVPR 2011*, pages 1465–1472. IEEE, 2011.

[60] Angjoo Kanazawa, Michael J. Black, David W. Jacobs, and Jitendra Malik. End-to-end recovery of human shape and pose. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7122–7131. IEEE, 2018.

[61] Muhammed Kocabas, Salih Karagoz, and Emre Akbas. MultiPoseNet: Fast multi-person pose estimation using pose residual network. In *Computer Vision – ECCV 2018*, pages 437–453. Springer International Publishing, 2018.

[62] Nikos Kolotouros, Georgios Pavlakos, and Kostas Daniilidis. Convolutional mesh regression for single-image human shape reconstruction. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4496–4505. IEEE, 2019.

[63] Sven Kreiss, Lorenzo Bertoni, and Alexandre Alahi. PifPaf: Composite fields for human pose estimation. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11969–11978. IEEE, 2019.

[64] Robert Kübler. Get uncertainty estimates in regression neural networks for free. `https://towardsdatascience.com/get-uncertainty-estimates-in-neural-networks-for-free-48f2edb82c8f`. Accessed: 2023-02-15.

[65] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, volume 30, pages 6402–6413. Curran Associates, Inc., 2017.

[66] Przemyslaw A. Lasota, Gregory F. Rossano, and Julie A. Shah. Toward safe close-proximity human-robot interaction with standard industrial robots. In *2014 IEEE International Conference on Automation Science and Engineering (CASE)*, pages 339–344. IEEE, 2014.

[67] Wenbo Li, Zhicheng Wang, Binyi Yin, Qixiang Peng, Yuming Du, Tianzi Xiao, Gang Yu, Hongtao Lu, Yichen Wei, and Jian Sun. Rethinking on multi-stage networks for human pose estimation. *arXiv preprint arXiv:1901.00148*, 2019.

[68] Yanjie Li, Shoukui Zhang, Zhicheng Wang, Sen Yang, Wankou Yang, Shu-Tao Xia, and Erjin Zhou. TokenPose: Learning keypoint tokens for human pose estimation. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 11293–11302. IEEE, 2021.

[69] Xiaodan Liang, Ke Gong, Xiaohui Shen, and Liang Lin. Look into Person: Joint body parsing & pose estimation network and a new benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(4):871–885, 2018.

[70] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *Computer Vision – ECCV 2014*, pages 740–755. Springer International Publishing, 2014.

[71] Hongyan Liu, Daokui Qu, Fang Xu, Zhenjun Du, Kai Jia, Jilai Song, and Mingmin Liu. Real-time and efficient collision avoidance planning approach for safe human-robot interaction. *Journal of Intelligent & Robotic Systems*, 105(4):93, 2022.

[72] Hongyi Liu, Yuquan Wang, Wei Ji, and Lihui Wang. A context-aware safety system for human-robot collaboration. *Procedia Manufacturing*, 17:238–245, 2018.

[73] Quan Liu, Zhihao Liu, Bo Xiong, Wenjun Xu, and Yang Liu. Deep reinforcement learning-based safe interaction for industrial human-robot collaboration using intrinsic reward function. *Advanced Engineering Informatics*, 49:101360, 2021.

[74] Zhenguang Liu, Haoming Chen, Runyang Feng, Shuang Wu, Shouling Ji, Bailin Yang, and Xun Wang. Deep dual consecutive network for human pose estimation. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 525–534. IEEE, 2021.

[75] Zhenguang Liu, Runyang Feng, Haoming Chen, Shuang Wu, Yixing Gao, Yunjun Gao, and Xiang Wang. Temporal feature alignment and mutual information maximization for video-based human pose estimation. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10996–11006. IEEE, 2022.

[76] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. SMPL: A skinned multi-person linear model. *ACM Transactions on Graphics*, 34(6):248, 2015.

[77] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[78] Weian Mao, Yongtao Ge, Chunhua Shen, Zhi Tian, Xinlong Wang, and Zhibin Wang. TFPose: Direct human pose estimation with transformers. *arXiv preprint arXiv:2103.15320*, 2021.

[79] Weian Mao, Yongtao Ge, Chunhua Shen, Zhi Tian, Xinlong Wang, Zhibin Wang, and Anton van den Hengel. Poseur: Direct human pose regression with transformers. In *Computer Vision – ECCV 2022*, pages 72–88. Springer Nature Switzerland, 2022.

[80] Julieta Martinez, Rayat Hossain, Javier Romero, and James J Little. A simple yet effective baseline for 3D human pose estimation. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2659–2668. IEEE, 2017.

[81] Angel Martínez-González, Michael Villamizar, Olivier Canévet, and Jean-Marc Odobez. Real-time convolutional networks for depth-based human pose estimation. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 41–47. IEEE, 2018.

[82] Dushyant Mehta, Srinath Sridhar, Oleksandr Sotnychenko, Helge Rhodin, Mohammad Shafiei, Hans-Peter Seidel, Weipeng Xu, Dan Casas, and Christian Theobalt. VNect: Real-time 3D human pose estimation with a single RGB camera. *ACM Transactions on Graphics*, 36(4):44, 2017.

[83] Kelly Merckaert, Bryan Convens, Chi-ju Wu, Alessandro Roncone, Marco M Nicotra, and Bram Vanderborght. Real-time motion control of robotic manipulators for safe human–robot coexistence. *Robotics and Computer-Integrated Manufacturing*, 73:102223, 2022.

[84] Gyeongsik Moon, Ju Yong Chang, and Kyoung Mu Lee. Camera distance-aware top-down approach for 3D multi-person pose estimation from a single RGB image. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10132–10141. IEEE, 2019.

[85] Christopher Neff, Aneri Sheth, Steven Furgurson, John Middleton, and Hamed Tabkhi. EfficientHRNet: Efficient and scalable high-resolution networks for real-time multi-person 2D human pose estimation. *Journal of Real-Time Image Processing*, 18(4):1037–1049, 2021.

[86] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *Computer Vision – ECCV 2016*, pages 483–499. Springer International Publishing, 2016.

[87] Alejandro Newell, Zhiao Huang, and Jia Deng. Associative embedding: End-to-end learning for joint detection and grouping. In *Advances in Neural Information Processing Systems*, volume 30, pages 2277–2287. Curran Associates, Inc., 2017.

[88] Dong Hai Phuong Nguyen, Matej Hoffmann, Alessandro Roncone, Ugo Pattacini, and Giorgio Metta. Compact real-time avoidance on a humanoid robot for human-robot interaction. In *2018 13th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 416–424, 2018.

[89] Xuecheng Nie, Jiashi Feng, Jianfeng Zhang, and Shuicheng Yan. Single-stage multi-person pose machines. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6950–6959. IEEE, 2019.

[90] Luis Oala, Cosmas Heiß, Jan Macdonald, Maximilian März, Wojciech Samek, and Gitta Kutyniok. Interval neural networks: Uncertainty scores. *arXiv preprint arXiv:2003.11566*, 2020.

[91] Michael Otte and Emilio Frazzoli. RRTX: Asymptotically optimal single-query sampling-based motion planning with quick replanning. *The International Journal of Robotics Research*, 35(7):797–822, 2016.

[92] George Papandreou, Tyler Zhu, Nori Kanazawa, Alexander Toshev, Jonathan Tompson, Chris Bregler, and Kevin Murphy. Towards accurate multi-person pose estimation in the wild. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3711–3719. IEEE, 2017.

[93] Dario Pavllo, Christoph Feichtenhofer, David Grangier, and Michael Auli. 3D human pose estimation in video with temporal convolutions and semi-supervised

training. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7745–7754. IEEE, 2019.

[94] Tim Pearce, Alexandra Brintrup, Mohamed Zaki, and Andy Neely. High-quality prediction intervals for deep learning: A distribution-free, ensembled approach. In *Proceedings of the 35th International Conference on Machine Learning*, pages 4075–4084. PMLR, 2018.

[95] Chao Peng, Tete Xiao, Zeming Li, Yuning Jiang, Xiangyu Zhang, Kai Jia, Gang Yu, and Jian Sun. MegDet: A large mini-batch object detector. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6181–6189. IEEE, 2018.

[96] Tomas Pfister, James Charles, and Andrew Zisserman. Flowing ConvNets for human pose estimation in videos. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1913–1921. IEEE, 2015.

[97] Pilz GmbH & Co. KG. The first safe camera system SafetyEYE opens up new horizons for safety & security. `https://www.automate.org/robotics/news/the-first-safe-camera-system-safetyeye-opens-up-new-horizons-for-safety-and-security`. Accessed: 2024-08-27.

[98] Leonid Pishchulin, Eldar Insafutdinov, Siyu Tang, Bjoern Andres, Mykhaylo Andriluka, Peter V. Gehler, and Bernt Schiele. DeepCut: Joint subset partition and labeling for multi person pose estimation. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4929–4937. IEEE, 2016.

[99] Maithra Raghu, Katy Blumer, Rory Sayres, Ziad Obermeyer, Bobby Kleinberg, Sendhil Mullainathan, and Jon Kleinberg. Direct uncertainty prediction for medical second opinions. In *Proceedings of the 36th International Conference on Machine Learning*, pages 5281–5290. PMLR, 2019.

[100] Christopher Reardon, Huan Tan, Balajee Kannan, and Lynn Derose. Towards safe robot-human collaboration systems using human pose detection. In *2015 IEEE International Conference on Technologies for Practical Robot Applications (TePRA)*, pages 1–6. IEEE, 2015.

[101] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, volume 28, pages 91–99. Curran Associates, Inc., 2015.

[102] Martin J Rosenstrauch, Tessa J Pannen, and Jörg Krüger. Human robot collaboration-using Kinect v2 for ISO/TS 15066 speed and separation monitoring. *Procedia CIRP*, 76:183–186, 2018.

[103] Matteo Ruggero Ronchi and Pietro Perona. Benchmarking and error diagnosis in multi-instance pose estimation. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 369–378. IEEE, 2017.

[104] Ben Sapp and Ben Taskar. MODEC: Multimodal decomposable models for human pose estimation. In *2013 IEEE Conference On Computer Vision and Pattern Recognition*, pages 3674–3681. IEEE, 2013.

[105] Matteo Saveriano and Dongheui Lee. Distance based dynamical system modulation for reactive avoidance of moving obstacles. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5618–5623. IEEE, 2014.

[106] Patrick Schlosser and Christoph Ledermann. Using diverse neural networks for safer human pose estimation: Towards making neural networks know when they don't know. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10305–10312. IEEE, 2020.

[107] Patrick Schlosser and Christoph Ledermann. Achieving hard real-time capability for 3D human pose estimation systems. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3772–3778. IEEE, 2021.

[108] Patrick Schlosser and Christoph Ledermann. Robust human pose estimation under Gaussian noise. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10504–10510. IEEE, 2023.

[109] Patrick Schlosser, Christoph Ledermann, and Tamim Asfour. Upper bounds for localization errors in 2D human pose estimation. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5934–5941. IEEE, 2023.

[110] Divya Shanmugam, Davis Blalock, Guha Balakrishnan, and John Guttag. Better aggregation in test-time augmentation. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1194–1203. IEEE, 2021.

[111] Dahu Shi, Xing Wei, Liangqi Li, Ye Ren, and Wenming Tan. End-to-end multi-person pose estimation with transformers. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11059–11068. IEEE, 2022.

[112] Dong Won Shin and Hyung Il Koo. Confidence estimation method for regression neural networks. *Electronics Letters*, 57(13):523–525, 2021.

[113] SICK AG. Operating instructions S3000. `https://cdn.sick.com/media/docs/3/63/863/operating_instructions_s3000_safety_laser_scanner_en_im0011863.pdf`, . Accessed: 2024-06-17.

[114] SICK AG. Operating instructions safeRS. `https://cdn.sick.com/media/docs/8/58/558/operating_instructions_safers_en_im0086558.pdf`, . Accessed: 2024-08-27.

[115] Leonid Sigal. *Human Pose Estimation*, pages 573–592. Springer International Publishing, 2021.

[116] Zhihui Su, Ming Ye, Guohui Zhang, Lei Dai, and Jianda Sheng. Cascade feature aggregation for human pose estimation. *arXiv preprint arXiv:1902.07837*, 2019.

[117] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5686–5696. IEEE, 2019.

[118] Xiao Sun, Jiaxiang Shang, Shuang Liang, and Yichen Wei. Compositional human pose regression. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2621–2630. IEEE, 2017.

[119] Petr Švarnỳ, Zdenek Straka, and Matej Hoffmann. Toward safe separation distance monitoring from RGB-D sensors in human-robot interaction. *arXiv preprint arXiv:1810.04953*, 2018.

[120] Chunwei Tian, Lunke Fei, Wenxian Zheng, Yong Xu, Wangmeng Zuo, and Chia-Wen Lin. Deep learning on image denoising: An overview. *Neural Networks*, 131: 251–275, 2020.

[121] Zhi Tian, Hao Chen, and Chunhua Shen. Directpose: Direct end-to-end multi-person pose estimation. *arXiv preprint arXiv:1911.07451*, 2019.

[122] Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning*, 4(2):26–31, 2012.

[123] Jonathan J. Tompson, Arjun Jain, Yann LeCun, and Christoph Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In *Advances in Neural Information Processing Systems*, volume 27, pages 1799–1807. Curran Associates, Inc., 2014.

[124] J. Tornero, J. Hamlin, and R. B. Kelley. Spherical-object representation and fast distance computation for robotic applications. In *Proceedings. 1991 IEEE International Conference on Robotics and Automation*, pages 1602–1608. IEEE Computer Society, 1991.

[125] Alexander Toshev and Christian Szegedy. DeepPose: Human pose estimation via deep neural networks. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1653–1660. IEEE, 2014.

[126] Veo Robotics Inc. Veo FreeMove® - 3D safeguarding for safe, dynamic human-robot collaboration (product website). `https://www.veobot.com/freemove`. Accessed: 2024-08-27.

[127] Timo Von Marcard, Roberto Henschel, Michael J Black, Bodo Rosenhahn, and Gerard Pons-Moll. Recovering accurate 3D human pose in the wild using IMUs and a moving camera. In *Computer Vision – ECCV 2018*, pages 614–631. Springer International Publishing, 2018.

[128] Guotai Wang, Wenqi Li, Michael Aertsen, Jan Deprest, Sébastien Ourselin, and Tom Vercauteren. Aleatoric uncertainty estimation with test-time augmentation for medical image segmentation with convolutional neural networks. *Neurocomputing*, 338:34–45, 2019.

[129] Jiahang Wang, Sheng Jin, Wentao Liu, Weizhong Liu, Chen Qian, and Ping Luo. When human pose estimation meets robustness: Adversarial algorithms and benchmarks. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11850–11859. IEEE, 2021.

[130] Jinbao Wang, Shujie Tan, Xiantong Zhen, Shuo Xu, Feng Zheng, Zhenyu He, and Ling Shao. Deep 3D human pose estimation: A review. *Computer Vision and Image Understanding*, 210:103225, 2021.

[131] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4724–4732. IEEE, 2016.

[132] David A Winter. *Biomechanics and Motor Control of Human Movement*. John Wiley & Sons, 2009.

[133] Jiahong Wu, He Zheng, Bo Zhao, Yixin Li, Baoming Yan, Rui Liang, Wenjia Wang, Shipei Zhou, Guosen Lin, Yanwei Fu, et al. AI Challenger: A large-scale dataset for going deeper in image understanding. *arXiv preprint arXiv:1711.06475*, 2017.

[134] Bin Xiao, Haiping Wu, and Yichen Wei. Simple baselines for human pose estimation and tracking. In *Computer Vision – ECCV 2018*, pages 472–487. Springer International Publishing, 2018.

[135] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with noisy student improves ImageNet classification. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695. IEEE, 2020.

[136] Chengjun Xu, Xinyi Yu, Zhengan Wang, and Linlin Ou. Multi-view human pose estimation in human-robot interaction. In *IECON 2020 The 46th Annual Conference of the IEEE Industrial Electronics Society*, pages 4769–4775. IEEE, 2020.

[137] Sen Yang, Zhibin Quan, Mu Nie, and Wankou Yang. TransPose: Keypoint localization via transformer. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 11782–11792. IEEE, 2021.

[138] Wei Yang, Shuang Li, Wanli Ouyang, Hongsheng Li, and Xiaogang Wang. Learning feature pyramids for human pose estimation. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 1290–1299. IEEE, 2017.

[139] Yi Yang and Deva Ramanan. Articulated human detection with flexible mixtures of parts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(12): 2878–2890, 2012.

[140] Changqian Yu, Bin Xiao, Changxin Gao, Lu Yuan, Lei Zhang, Nong Sang, and Jingdong Wang. Lite-HRNet: A lightweight high-resolution network. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10435–10445. IEEE, 2021.

[141] Cuihong Yu, Cheng Han, Qi Zhang, and Chao Zhang. MsF-HigherHRNet: Multiscale feature fusion for human pose estimation in crowded scenes. In *Computer-Aided Design and Computer Graphics*, pages 16–29. Springer Nature Singapore, 2024.

[142] Andrea Maria Zanchettin, Nicola Maria Ceriani, Paolo Rocco, Hao Ding, and Björn Matthias. Safety in human-robot collaborative manufacturing environments: Metrics and control. *IEEE Transactions on Automation Science and Engineering*, 13 (2):882–893, 2015.

[143] Feng Zhang, Xiatian Zhu, and Mao Ye. Fast human pose estimation. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3512–3521. IEEE, 2019.

[144] Jianfeng Zhang, Yujun Cai, Shuicheng Yan, Jiashi Feng, et al. Direct multi-view multi-person 3D pose estimation. In *Advances in Neural Information Processing Systems*, volume 34, pages 13153–13164. Curran Associates, Inc., 2021.

[145] Kai Zhang, Wangmeng Zuo, and Lei Zhang. FFDNet: Toward a fast and flexible solution for CNN-based image denoising. *IEEE Transactions on Image Processing*, 27(9):4608–4622, 2018.

[146] Jianan Zhen, Qi Fang, Jiaming Sun, Wentao Liu, Wei Jiang, Hujun Bao, and Xiaowei Zhou. SMAP: Single-shot multi-person absolute 3D pose estimation. In *Computer Vision – ECCV 2020*, pages 550–566. Springer International Publishing, 2020.

[147] Ce Zheng, Wenhan Wu, Chen Chen, Taojiannan Yang, Sijie Zhu, Ju Shen, Nasser Kehtarnavaz, and Mubarak Shah. Deep learning-based human pose estimation: A survey. *ACM Computing Surveys*, 56(1):11, 2023.

[148] Stephan Zheng, Yang Song, Thomas Leung, and Ian Goodfellow. Improving the robustness of deep neural networks via stability training. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4480–4488. IEEE, 2016.

[149] Yufan Zhou, Haiwei Dong, and Abdulmotaleb El Saddik. Learning to estimate 3D human pose from point cloud. *IEEE Sensors Journal*, 20(20):12334–12342, 2020.

List of Figures

List of Tables

List of Author's Publications

Relevant Publications for this Thesis

Patrick Schlosser and Christoph Ledermann. Using diverse neural networks for safer human pose estimation: Towards making neural networks know when they don't know. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10305–10312. IEEE, 2020.

Patrick Schlosser and Christoph Ledermann. Achieving hard real-time capability for 3D human pose estimation systems. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3772–3778. IEEE, 2021.

Patrick Schlosser and Christoph Ledermann. Robust human pose estimation under Gaussian noise. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10504–10510. IEEE, 2023.

Patrick Schlosser, Christoph Ledermann, and Tamim Asfour. Upper bounds for localization errors in 2d human pose estimation. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5934–5941. IEEE, 2023.

Further Publications

Patrick Schlosser, David Munch, and Michael Arens. Investigation on combining 3D convolution of image data and optical flow to generate temporal action proposals. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2019.

Xi Huang, Tom P. Huck, Patrick Schlosser, Woo-Jeong Baek, and Christoph Ledermann. The RoboShield demonstrator: From static to flexible safety measures in industrial human robot collaboration. In *Forum Safety & Security 2020 (Proceedings)*, 2020.

Matthias Budde, Jan Felix Rohe, Lina Hirschoff, Patrick Schlosser, Michael Beigl, Jussi Holopainen, and Andrea Schankin. SpaceMaze: incentivizing correct mobile crowdsourced sensing behaviour with a sensified minigame. *Behaviour & Information Technology*, 40(15):1627–1642, 2021.

Daniel Hillen, Tom P. Huck, Nishanth Laxman, Christoph Ledermann, Jan Reich, Patrick Schlosser, Andreas Schmidt, Daniel Schneider, and Denis Uecker. Plug-and-produce... safely! End-to-end model-based safety assurance for reconfigurable industry 4.0. In *International Symposium on Model-Based Safety and Assessment*, pages 83–97. Springer International Publishing, 2022.