sensors

MDPI

Article

Hybrid Anomaly Detection in Time Series by Combining Kalman Filters and Machine Learning Models

Andreas Puder [1,†], Moritz Zink [2,†], Luca Seidel [2,†] and Eric Sax [2,*,†]

[1] Embedded Systems, Getinge AB, 76437 Rastatt, Germany; andreas.puder@getinge.com
[2] Institute for Information Processing Technologies (ITIV), Karlsruhe Institute of Technology (KIT), 76131 Karlsruhe, Germany; moritz.zink@kit.edu (M.Z.); luca.seidel@kit.edu (L.S.)
* Correspondence: eric.sax@kit.edu
† These authors contributed equally to this work.

Abstract: Due to connectivity and automation trends, the medical device industry is experiencing increased demand for safety and security mechanisms. Anomaly detection has proven to be a valuable approach for ensuring safety and security in other industries, such as automotive or IT. Medical devices must operate across a wide range of values due to variations in patient anthropometric data, making anomaly detection based on a simple threshold for signal deviations impractical. For example, surgical robots directly contacting the patient's tissue require precise sensor data. However, since the deformation of the patient's body during interaction or movement is highly dependent on body mass, it is impossible to define a single threshold for implausible sensor data that applies to all patients. This also involves statistical methods, such as Z-score, that consider standard deviation. Even pure machine learning algorithms cannot be expected to provide the required accuracy simply due to the lack of available training data. This paper proposes using hybrid filters by combining dynamic system models based on expert knowledge and data-based models for anomaly detection in an operating room scenario. This approach can improve detection performance and explainability while reducing the computing resources needed on embedded devices, enabling a distributed approach to anomaly detection.

Keywords: medical; machine learning; deep learning; anomaly detection; sensor fusion; time series analysis; Kalman filter; service-oriented architecture (SOA); ROS2; simulation

Citation: Puder, A.; Zink, M.; Seidel, L.; Sax, E. Hybrid Anomaly Detection in Time Series by Combining Kalman Filters and Machine Learning Models. *Sensors* **2024**, *24*, 2895. https://doi.org/10.3390/s24092895

Academic Editor: Weizhi Meng

Received: 21 March 2024
Revised: 19 April 2024
Accepted: 30 April 2024
Published: 1 May 2024

1. Introduction

Challenges such as the aging population [1] and the increasing prevalence of chronic diseases [2] are putting pressure on healthcare systems worldwide. A strong correlation between an aging population and an increase in the incidence and mortality of chronic diseases like cardiovascular diseases, tumors, and diabetes [3] exists. Thus, there is a need to increase efficiency in all areas of healthcare, including Operating Room (OR). Medical devices that control physical objects while constantly interacting with other connected devices are considered to facilitate effective and affordable healthcare [4]. Consequently, connectivity and automation in the OR promise to contribute to solving these challenges, for instance, by the increased application of robotic medical devices. In addition, Service-oriented Device Connectivity (SDC) [5], which is a family of medical device communication standards, intends to enable data exchange and manufacturer-independent interoperability of medical devices in the OR to improve patient care.

Nonetheless, this increased connectivity also increases the risk of potential cyberattacks on these devices [6]. Historically, medical devices have operated in isolation without permanent connections to external networks, like the Internet [7]. These devices have been typically self-contained, embedded systems. Therefore, security requirements were not in scope during development, and manufacturers focused on safety. By opening these

devices up to connectivity with other devices, new safety risks to patient health arise from security threats.

Operating Room Tables (OR tables) are robotic medical devices that will leverage the potential of connectivity in the OR as they are central components. Nevertheless, positional plausibility for OR tables must be ensured regarding safety and security [6], especially if other devices rely on the provided information. Examples are angiography systems, which enable image-guided surgery, or surgical robots. Both rely on the correctness of the determined OR table positions to avoid, e.g., collisions and thus severe patient harm. This, however, bears some unique challenges, as OR tables deform under a load of heavy patients, whereby the maximum patient load for a middle-class OR table is ∼250 kg (e.g., Getinge Maquet Meera [8]). The plausibility check in the sense of detecting anomalies in the measurement of sensor data, e.g., for position or load, using mathematical models for non-linear deformation is a challenge. In addition, the body, especially of heavy patients, deforms due to gravitational forces during movements of the OR table. This makes it also challenging to determine the actual physical values based on sensor measurements and detect possible errors in the system or its sensors without a high false alarm rate.

Problem: Anomaly detection in security systems primarily emphasizes routing-based attacks, with only a limited number of systems tackling physical signals like sensor data [9]. In addition, the mathematical model of a rigid body's dynamic behavior is straightforward, but including the previously described deformation in such a model is not. One approach to overcome this obstacle is to use machine learning algorithms that can learn the nonlinear behavior of the deformation from training data. Let us assume that the system behavior and all occurring states can be written as a set $\mathfrak{S} = \{\mathfrak{s}_0, \mathfrak{s}_1, ..., \mathfrak{s}_n\}$. The conventional states to be modeled via a physical system description can be written as $\mathcal{S} = \{\int_0, \int_1, ..., \int_n\}$. The following, therefore, applies $\mathcal{S} \subset \mathfrak{S}$. If we now transfer this relationship to a singular state $\mathfrak{s}_t \in \mathfrak{S}$ in the specific moment t, we can rewrite the relation through an addition of a disturbance d_t of our modeled system:

$$\mathfrak{s}_t = \int_t + d_t \tag{1}$$

However, machine learning-based approaches bear the challenge of data acquisition in OR and healthcare settings, since conditions for proper data collection are not given [10]. Data-based models for the state estimation of sensor data may result in inadequate quality, particularly as sensor noise increases [11]. Integrating these approaches with conventional model-based estimation is a more promising area for research.

Contribution: The manipulation of deformable objects is a growing research problem in robotics, especially in healthcare applications such as surgical procedures [12]. Meanwhile, models representing the characteristics of the surgical robot and the human body can improve the safety of operations [13], for example, when checking the plausibility of measurements such as joint positions. To improve the sensing of deformable objects, Zhu et al. propose a hierarchical sensing model. It combines a simple model with a more sophisticated nonlinear model, such as a linear model as a base layer combined with a deep neural network that learns the rest of the behavior [12]. Such models can be built for OR tables to check the plausibility of position data. Based on intelligent sensor systems such as a load detection system [14] and a corresponding fault detection system [15], we suggest an anomaly detection approach for similar medical devices to contribute to a safer OR.

First, we propose a generic approach that uses and combines different anomaly detection techniques. Furthermore, we provide an example of a distributed software architecture to detect anomalies in an embedded environment. Our proposed system for anomaly detection involves a combined modeling approach. We use a Kalman Filter (KF) (Figure 1) to model the deductive system and patient behavior, while a data-based algorithm learns any indeterminable or non-linear behavior. *The advantage of this approach is that the algorithm does not have to learn and know the behavior. Instead, it only learns the misbehavior of the manual model, which is then used for anomaly detection.* This ensures that our system can detect anomalies that may not be accounted for by manual modeling.

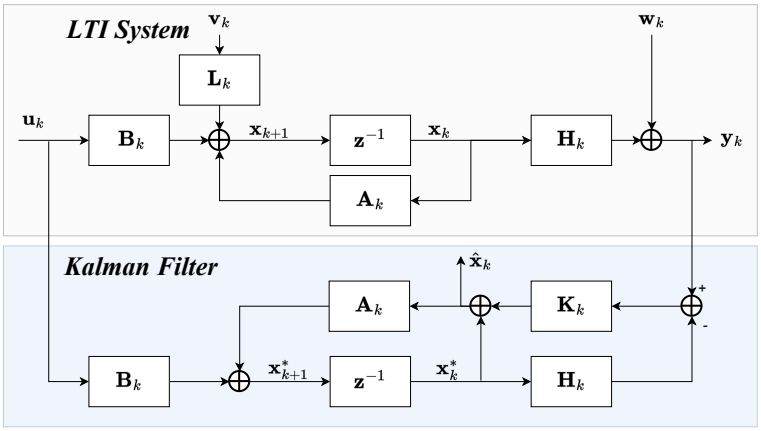

Figure 1. Signal processing for a stochastic disturbance-affected observed system using a KF [16] in a Time-Discrete Linear Time Invariant (LTI) system [17] with measurable input variables.

Outline: Section 2 provides an overview of anomaly detection approaches using KFs, classical statistical methods Section 1, and machine learning methods. Section 3 will review related work in anomaly detection. The concept of applying hybrid filters using KFs as dynamic models in combination with data-based models for anomaly detection is presented in Section 4, followed by an evaluation of a prototypical implementation in Section 5. A discussion of the results and the effectiveness of the approach is conducted in Section 6. Finally, Section 7 provides a summary and suggestions for future work in this area.

2. Background in Anomaly Detection

Outlier data points can be described as observations that are so different from the other observations that it is suspected that they were generated by some other mechanism [18]. Anomaly detection techniques identify data points as such [19]. In this application, the working range of values of the fusion method used must be monitored to ensure that trustworthy values are generated. We have adapted the notation of formula symbols and equations to the standard notation used in the literature for algorithm notation.

2.1. Kalman Filter (KF)

Since our goal is to combine the KFs, which we generally refer to here as dynamic models, with machine learning-based algorithms, we focus on the comparison of the presented linear KF, Extended Kalman filter (EKF) and Unscented Kalman filter (UKF), which assume a Gaussian distribution. Other KF-based state estimators, such as the Particle KF and the Ensembled KF, which are used for non-Gaussian distributions [20], are not considered in this paper.

The KF [21] is a mathematical method that uses a process model in state space and a method for iterative state or signal estimation of time-independent signals in transient random processes [16]. In such processes, the mean and variance moments depend on time. This tool has been widely used in various fields, including aerospace, robotics, and finance, due to its ability to accurately and efficiently estimate the state of a system in the presence of noise and uncertainty. An iteration consists of a prediction step for the state $\mathbf{x}$ of dimension m (Table 1) that uses the state space model and a correction step that improves the prediction with a measurement $\mathbf{y}$ of dimension n (Figure 2). The KF is only suitable for a few variables in a linear system model but has a low computational cost [20]. When constructing the difference vector of the predictions or estimations of the KF with the measurements, a deviation score can be generated using l_k norms. If the deviation exceeds a defined threshold value for an iteration k, the corresponding state or measurement vector of that iteration is classified as an anomaly. Thus, the KF is also a valuable approach to

anomaly detection, e.g., when using the l_1 norm, which is also called the Mean Absolute Error (MAE):

$$MAE_{state} = \frac{1}{n}\sum_{i=1}^{n}|\hat{\mathbf{x}}_i - \mathbf{y}_i| \tag{2}$$

The traditional KF is unsuitable for dealing with processes or measurements with nonlinearity. The EKF is used to overcome this limitation, approximating the nonlinear system by linearizing it around the current mean and covariance [22]. To estimate the current state of a system, the nonlinear state transition function $f(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1}, 0)$ is used to predict the state. Similarly, the nonlinear observation function $h(\mathbf{x}_k^*, 0)$ is used to project the estimated state $\mathbf{x}^*$ onto the observation $\mathbf{y}$ (Figure 3). Although the EKF can be used for non-linear system models and has low computational costs, divergences may occur due to the linearization [20].

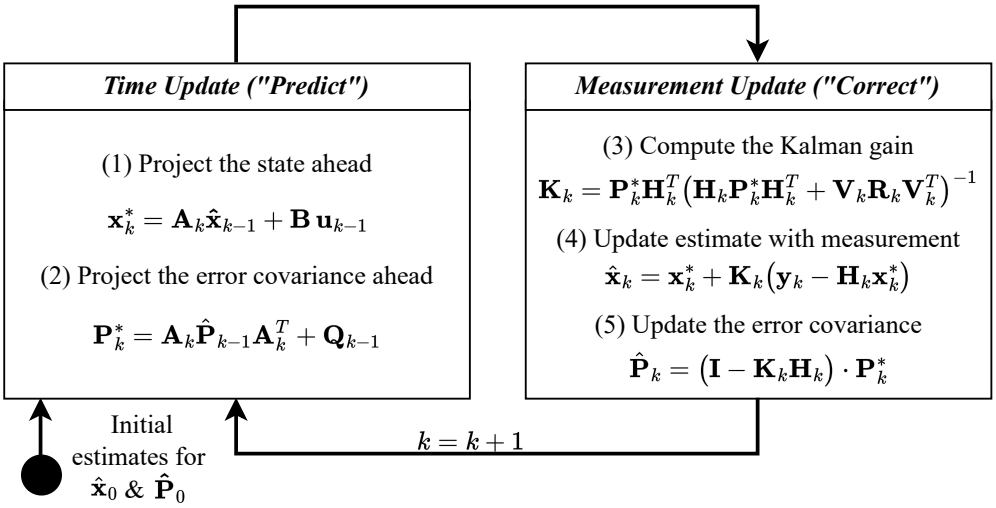

Figure 2. KF operation for each iteration k [22] (Table 1).

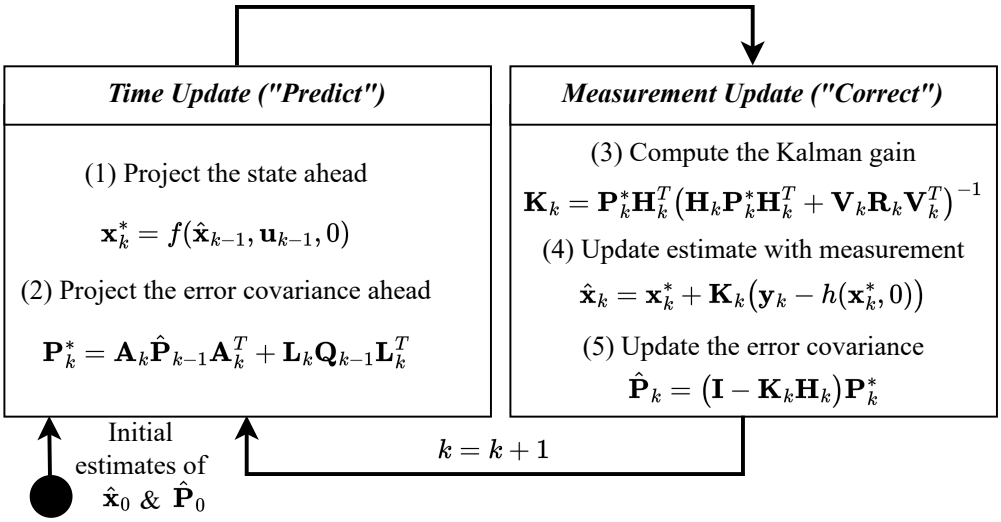

Figure 3. EKF operation for each iteration k [22] (Table 1).

Table 1. KF variables [16]. Since **v** is expected to be white noise, **L** can be neglected [23].

Variable	Description
$\mathbf{x}^{*(m)}$	Predicted system state vector
$\hat{\mathbf{x}}^{(m)}$	Estimated system state vector after new observation **y**
$\mathbf{u}^{(l)}$	System input vector
$\mathbf{y}^{(n)}$	New observation/measurement vector
$\mathbf{B}^{(m\times l)}$	Dynamics of the system input **u** and projection on system vector **x**
$\mathbf{K}^{(m\times n)}$	Kalman-Gain-Matrix to project the residuals onto the correction of the system state
$\mathbf{A}^{(m\times m)}$	Transition-Matrix to propagate the system state to next time point
$\mathbf{P}^{*\,(m\times m)}$	A-priori covariance matrix of the predicted system state before new observation **y**
$\hat{\mathbf{P}}^{(m\times m)}$	A-posteriori covariance matrix of the estimated system state after new observation **y**
$\mathbf{Q}^{(m\times m)}$	Process noise covariance matrix for uncertainties from modeling errors or changing boundaries
$\mathbf{H}^{(n\times m)}$	Observation matrix projecting the system states on the measurement
$\mathbf{R}^{(n\times n)}$	Covariance matrix of the measurement noise
L	Input matrix projecting the input noise **v** on the system state $\mathbf{x}^*$

For highly nonlinear problems, the UKF (Figure 4) [24] is generally more accurate than the EKF [25]. It uses the probability distribution of states by mapping so-called weighted sigma points through the nonlinear function and calculating the new mean and variance from this [20]. The UKF avoids the problems that can occur due to the linearization divergences of the EKF but requires more computational resources and performance degrades as the number of state variables increases [20].

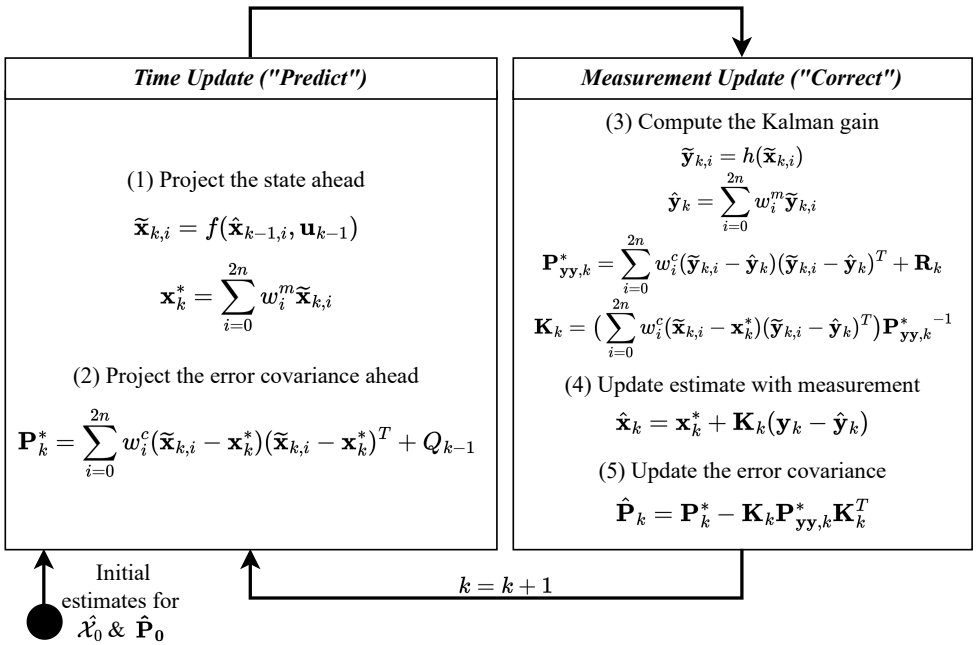

Figure 4. UKF operation for each iteration k [25] (Tables 1 and 2).

Table 2. UKF variable overview addendum to Table 1.

Variable	Description
$\widetilde{\mathbf{x}}_i$	i-th sigma point of the predicted state
$\mathcal{X}$	Set of i sigma points $\widetilde{\mathbf{x}}_i$ of the predicted state
$\widetilde{\mathbf{y}}_i$	i-th sigma point of the predicted measurement
$\hat{\mathbf{y}}_k$	Predicted measurement
$\mathbf{w}_i^m$ & $\mathbf{w}_i^c$	Sigma point weights for mean and covariance calculation

The sigma points $\mathcal{X}$ for the state are calculated by a sigma-function $s(\hat{\mathbf{x}}, \hat{\mathbf{P}})$. For the calculation of sigma points, the Van der Merwe Scaled Sigma Points are mainly used in industry and research as they represent a balanced compromise between accuracy and performance [25].

2.2. Statistical Methods

To check whether sensor measurements $\mathbf{y}$ values or the state estimation $\hat{\mathbf{x}}$ of a KF are within the intended working range and plausible, statistical models can be used even in the most trivial case. This includes the Z-score, which has shown promising results in other applications [26]. The Z-score is defined as follows:

$$Z = \frac{(x - \mu)}{\sigma} \tag{3}$$

Here, x is a single data point and a scalar, but for general validity we write $\mathbf{x}_i$ as it is also possible to find multivariate inputs here. μ is the mean and σ is the standard deviation of $\mathbf{x}_i$. The threshold value at which an anomaly is present can then be selected in the interval $\mathbb{W}_Z = [2, 3.5]$ depending on the application and data distribution. If data that are approximately normally distributed are available, the Z-score can be used as a relative metric. If this is not the case, the Inter Quartile Range (IQR) can instead be calculated as an absolute metric, which is also applied in outlier detection frameworks [27]. First, the IQR is calculated:

$$IQR = x_{0.75} - x_{0.25} \tag{4}$$

Afterward, we define the interval of our common value distribution:

$$I_{dist} = [(x_{0.25} - 1.5 \cdot IQR), (x_{0.75} + 1.5 \cdot IQR)] \tag{5}$$

The definition of the outlier set can then be written as $\mathcal{X}_{out} \forall x \notin I_{dist}$. The disadvantage becomes clear when the variables under consideration are no longer scalars, but vectors or even tensors: While there may still be strategies for some low-dimensional $\mathbf{x}$, with $\mathbf{x}_i \in \mathbb{R}^d$ using Boolean algebra (and $\wedge$/or $\vee$), this is hardly practical for images, audio signals or large sensor arrays where $\mathbf{x}_i \in \mathbb{R}^D$ with $d << D$.

2.3. Machine Learning Based Methods

In addition to classical statistical methods, machine learning models can identify outliers. The advantage here is that the algorithms' models draw their conclusions from the data provided and thus generalize better than, for example, purely statistical methods. This is particularly beneficial in real-world scenarios, where accurate knowledge of the data distribution is critical to the application of statistical methods. The Z-score, for example, requires normally distributed data, which is a condition that is rarely met in practice.

Unsupervised Learning

Since it is possible that there are no labels available to identify when the measurements or estimations of the KF are insufficient in the intended feature space, unsupervised learning is attractive here. A potentially suitable algorithm that meets our requirements is the Isolation Forest (IF), as it requires little memory, has a linear time complexity, and is not computationally expensive because no inter-distances and densities are calculated [28]. The conceptual idea behind this algorithm is that for any given $\mathbf{x}_i$, a traversal is made from the leaves to the tree's root, where the number of edges traversed is described by $l(\mathbf{x}_i)$. A recursive call is assumed to pass through a partition of the data points, where the length $l(\mathbf{x}_i)$ is a measure of the normality of the respective data point. Since this distribution is randomized between $\mathbf{x}_{max} \in \mathcal{X}$ and $\mathbf{x}_{min} \in \mathcal{X}$ where $\mathcal{X} = \{\mathbf{x}_i \in \mathbb{R} \mid 0 < i < \infty, i \in \mathbb{N}\}$, it can be assumed that extremely large or small values have shorter lengths. To determine a quantitative value of the anomaly, the use of normalization is desired, but this is not possible due to the different growth possibilities of the maximum length $l_{max}(\mathbf{x}) = |\mathcal{X}|$ and

the average length of $\bar{l}_{avg} = log(|\mathcal{X}|)$. However, since the isolation tree has an equivalent structure to the binary search tree, the average path length can be derived from it:

$$c(|\mathcal{X}|) = 2H(|\mathcal{X}| - 1) - (2(|\mathcal{X}| - 1)/|\mathcal{X}|) \tag{6}$$

Here, $H(i)$ represents the harmonic number, which is uniquely described by the elements i, i.e., $|\mathcal{X}|$, via the mathematical series of the harmonic sequence. The final anomaly score can be derived from this, as we can now use $c(|\mathcal{X}|)$ to normalize the path length $l(\mathbf{x}_i)$.

$$s(|\mathcal{X}|, \mathbf{x}_i) = 2^{-\frac{\mathbb{E}(l(\mathbf{x}))}{c(|\mathcal{X}|)}} \tag{7}$$

Here, $\mathbb{E}$ is the expectation value of a collection of isolation trees. The following anomaly values follow for the respective contexts [28]:

- $\mathbb{E}(l(\mathbf{x})) \to c(|\mathcal{X}|), s(|\mathcal{X}|, \mathbf{x}) \to 0.5$
- $\mathbb{E}(l(\mathbf{x})) \to 0, s(|\mathcal{X}|, \mathbf{x}) \to 1$
- $\mathbb{E}(l(\mathbf{x})) \to |\mathcal{X}| - 1, s(|\mathcal{X}|, \mathbf{x}) \to 0$

Three possible scenarios can now be derived from this [28]:

- Data points very close to 1 are most likely anomalies.
- If all points are close to 0.5, the sample data set probably has no anomalies.
- If the values are clearly below 0.5, they are probably data points that can be classified as normal.

2.4. Deep Learning Based Approach

In principle, the use of neural networks is also suitable for anomaly detection, although a distinction must be made here between different model architectures. Nevertheless, the basic functionality of neural networks should be briefly outlined here. A neural network maps an input variable tensor to an output, which can be either a vector space or a discrete set. In general, the following function can be written for a neural network without restriction:

$$\mathcal{N}_\Theta : \mathcal{X} \to \hat{\mathcal{Y}} \tag{8}$$

Thus, it can be written: $\mathcal{N}_\Theta : \mathbf{x}_i \to \hat{y}$, where $\mathbf{x}_i \in \mathcal{X} \in \mathbb{R}^n$ and $\hat{y} \in \hat{\mathcal{Y}}$. This is to be understood abstractly, as $\mathcal{Y}$ can be a set, a subset, or a continuous value. $\mathcal{N}_\Theta$ is parameterized via a set Θ. Precisely, these parameters can be found during training, which can be described as follows:

$$\Delta\theta_{i,j} = -\alpha \frac{\partial \mathcal{L}(\Theta, \mathbf{x}_i, y_i, \hat{y}_i)}{\partial \theta_{i,j}}. \tag{9}$$

Therefore, it is necessary first to run a forward pass through the model to obtain the output $\hat{y}_i$, calculate an abstract cost function $\mathcal{L}$ using the true value, the label, y_i and then find out how much the individual weights $\theta_{i,j}$ contribute to the change in the inference. The amount of the updating of the weights is determined by the learning rate α, e.g., to prevent overshooting and to get as close to the global minimum as possible. The aim is, therefore, to adjust the parameters Θ in such a way that the transmission function (see Equation (10)) of the model results in $\hat{y} \approx y$. This is the case if: $\Theta = \Theta^*$.

$$\hat{y}_i = \Gamma(\Theta^* \mathbf{x}_i + \mathbf{b}) \tag{10}$$

Here, Γ is a (mainly non-linear) activation function that allows the approximation of any continuous transfer function [29]. Figure 5 illustrates the principle and shows the transfer from one neuron in a layer to the next one [30]. Although deep learning has a high computational cost in the training phase, this approach will be analyzed in more detail here due to its power and adaptability.

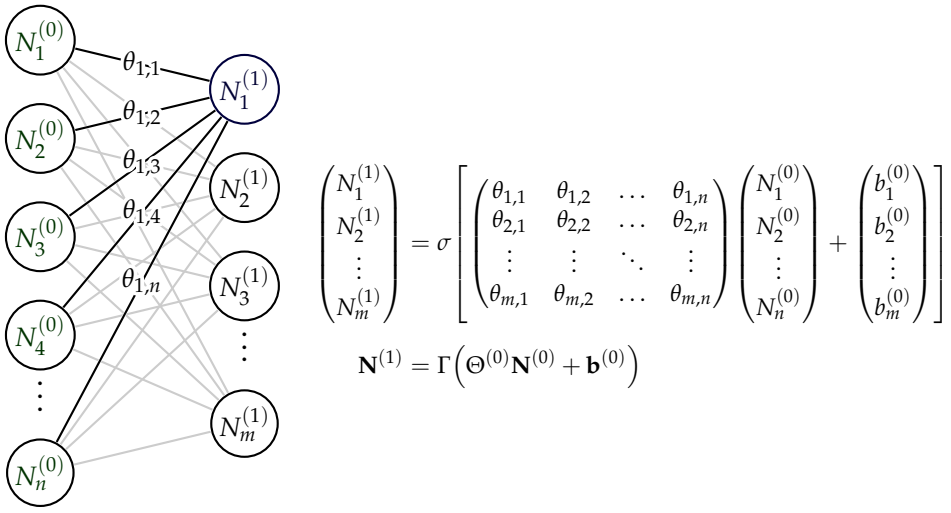

Figure 5. Basic neural network functionality with transmission function.

2.4.1. Autoencoder (AE)

Anomaly detection using Autoencoders (AEs) is sufficiently well researched, and now state of the art [31]. Here, the input features are mapped to themselves after the inference so that $f(\mathbf{x}) \rightarrow \hat{\mathbf{x}}$. The advantage of using deep learning-based methods is that they work with data of any dimension and with unstructured data such as images and audio signals. Figure 6 schematically outlines the concept of an AE.

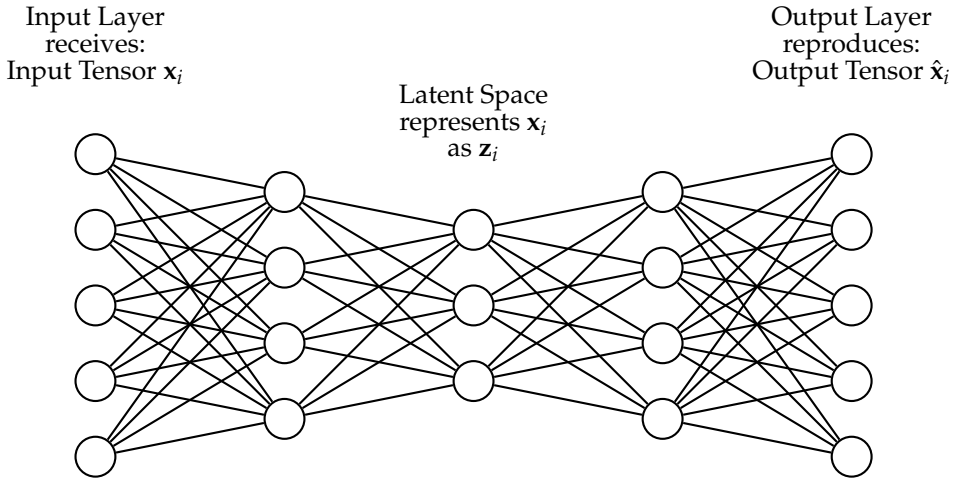

Figure 6. Functional principle of an AE.

Here, the model consists of two parts. First, the input vector $\mathbf{x}_i$ of a dimension n is compressed to a latent variable $\mathbf{z}_i$ of dimension c, which is part of the latent space $\mathcal{Z}$ (see Equation (11)).

$$\mathcal{E}_{\Theta_E} : \mathcal{X} \rightarrow \mathcal{Z} \tag{11}$$

This intermediate representation of the input is now used to build a reconstruction $\hat{\mathbf{x}}_i$ of the original variable $\mathbf{x}_i$ with dimensionality n (see Equation (12)), where $\mathbb{R}^c < \mathbb{R}^n$.

$$\mathcal{D}_{\Theta_D} : \mathcal{Z} \rightarrow \hat{\mathcal{X}} \tag{12}$$

Both are parameterized over different sets: $(\Theta_E, \Theta_D) \subseteq \Theta$. Instead of generating a representation model, such as in an IF, and comparing the occurring data over it, the data points are reduced to a smaller dimension c, where $\mathbb{R}^d < \mathbb{R}^D$ holds. The assumption is that anomalies behave differently after this compression than non-anomalous data. Sub-

sequently, a reconstruction of the data to the original input $\mathbb{R}^n$ dimension is performed. From this, the reconstruction error (see Equation (13)) can be calculated, which is small for known examples and high for anomalous samples due to inconsistencies [32]:

$$\mathcal{L}_R = \sum_{j=0}^{n} (\mathbf{x}_{i_j} - \hat{\mathbf{x}}_{i_j})^2 \tag{13}$$

The practical challenge is to find a suitable threshold to declare a reconstruction anomaly. This can be conducted by defining thresholds via the training data, e.g., via a desired percentile. This would involve looking at which values are below the 99th percentile [33]. The threshold can then be set above this. Alternatively, there are also approaches in which the most significant error occurring during training is saved and then set as a threshold [34]. The following function is used: a value is declared as an anomaly M depending on its reconstruction error $\mathcal{L}_R$ and the set threshold T: 14).

$$f(\mathcal{L}_R) = \begin{cases} M, & \mathcal{L}_R < T \\ M, \text{ else} \end{cases} \tag{14}$$

2.4.2. Long Short Term Memory (LSTM) Networks

A Long Short Term Memory (LSTM) is a Recurrent Neural Network (RNN) whose structure is shown in Figure 7. To solve the problem of the vanishing gradient, which can occur with such networks that are capable of processing transient time series data, the cell must store an internal state, the cell state c_t [35]. The hidden state h_t also exists. While the internal state c_t is stored, the gates can adjust it successively. These gates, in turn, receive the state h_{t-1} and the training data input. The forget gate decides which information is no longer needed in the cell state. To conduct this, the value $\mathbf{f}_t$ is first calculated:

$$\mathbf{f}_t = \sigma(\Theta_f \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_f) \tag{15}$$

The forget gate activation function uses the sigmoid function to ensure that the output vector of the gate produces continuous values between 1 (for retention) and 0 (for forgetting).

$$\mathbf{c}_f = \mathbf{c}_{t-1} \circ \mathbf{f}_t \tag{16}$$

Next, the input gate is run through. Two functional units are are accommodated here. First, the direction of the adjustment is determined by hyperbolic tangent ($\mathbb{W} = [-1, 1]$), and then the amount of the adjustment is determined by a logical sigmoid operation, which is then used to calculate the influence on the internal state.

$$\tilde{\mathbf{c}}_t = \tanh(\Theta_c \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_c) \tag{17}$$

$$\mathbf{m}_t = \sigma(\Theta_m \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_m) \tag{18}$$

The two values that have just been created are now processed together to update the internal state.

$$\mathbf{c}_t = \mathbf{c}_f + \mathbf{m}_t \circ \tilde{\mathbf{c}}_t \tag{19}$$

Finally, the internal state is influenced by the output gate. Relevant information from the current and previous output is filtered here. The next output $\mathbf{h}_t$ is then predicted. This is either the network output or serves as input for the next LSTM layer [36]:

$$\mathbf{o}_t = \sigma(\Theta_o \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_o) \tag{20}$$

$$\mathbf{h}_t = \mathbf{o}_t \circ \tanh(\mathbf{c}_t) \tag{21}$$

Such cells have a large number of parameters and are, therefore, challenging to train.

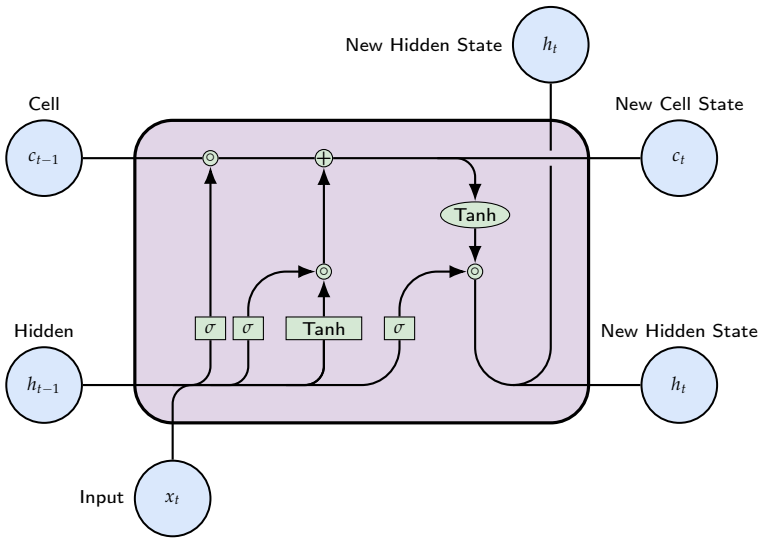

Figure 7. Structure of an LSTM-Cell.

3. Related Work

Historically, most anomaly detection approaches have been model-based [37]. An anomaly detection process decides whether a value is considered an anomaly based on the deviation between the estimated and the measured result. Dynamic Bayesian Networks (DBNs) are well-established for real-time anomaly detection in streaming data. These networks are dynamically expanded by incrementally adding new variables at each time step. New measurements are added according to a predefined template that outlines the conditional dependencies between features and their relationships with the existing network. The KF is an exemplary realization of a DBN in this context (Section 2.1). Hill et al. have effectively applied the KF to identify anomalies in low-resolution time series data [38,39]. However, such anomaly detection is highly dependent on the model's accuracy. Model tuning is time-consuming and depends on prior knowledge of the system. Due to the limitations of model-based approaches, data-based approaches have received increasing interest.

A computationally inexpensive method is to check statistically whether a data point deviates from historical data. Statistical control charts [40], principal component analysis [41], and other statistical methods have been used.

With additional sensors and appropriate methods such as Fourier or Wavelet transforms, misbehavior can be detected, as shown in [42]. Due to the nature of classical signal analysis, only one-dimensional signals can be considered. Besides this limitation, additional sensors are only an option for some scenarios where space and cost must not be considered.

Machine learning-based methods are gaining attention due to promising results. In anomaly detection, a distinction can be made between supervised and unsupervised methods. Supervised methods, as shown in [43], are only suitable if a sufficient amount of labeled data is available.

In other domains, such as automotive in-vehicle communications, hybrid combinations of statistical, model-based, and machine learning-based anomaly detection have been explored to overcome the individual drawbacks of each method, as shown in [44]. Due to the underlying model, time series data can be processed to consider the temporal dependencies of anomalies. Another anomaly detection in the automotive domain is presented by Weber with the automotive observer [45]. This approach, which was initially developed for Controller Area Network (CAN) and later extended for Ethernet messages in automotive contexts in [46], have been adapted for general data analysis based on [44].

They use AEs and Lightweight On-line Detector of Anomalies (LODA) algorithms to detect signal anomalies, which have been modified to work with a sliding window for improved accuracy. The analysis is performed according to the ISO26262 standard, which ensures the reliability and consistency of the results. However, the authors do not incorporate dynamic models, they focus on single signals, and neglect the correlation of different signals. The authors use the *hybrid* term for the combination of specification-based anomaly detection and machine learning algorithms.

The error determination system presented in the patent [15] describes how a KF can determine sensor errors for a load recognition system for OR tables. Based on the prediction of the expected movement trajectory of the patient's measured Center of Gravity (CoG) (Figure 8), the measurements are checked for plausibility. Nevertheless, the inventors do not recommend improvements in combining the KF with data-based models.

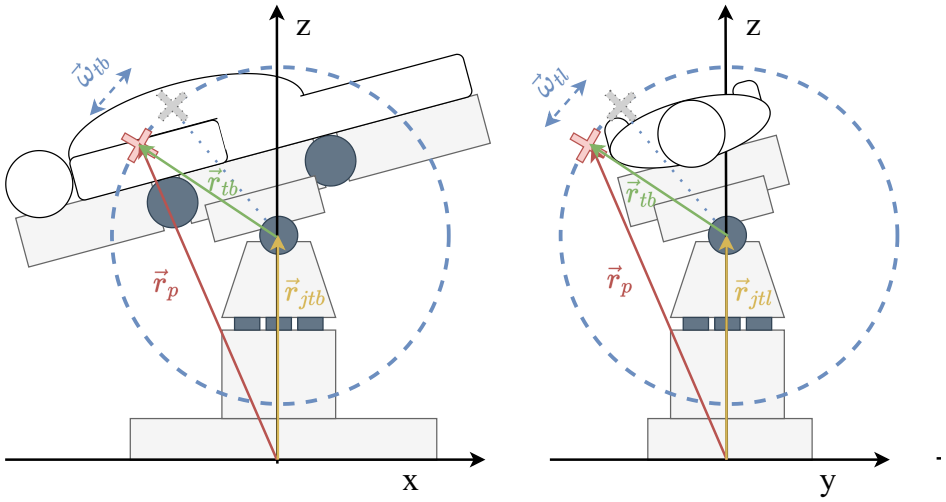

Figure 8. Overview of whole body movements Trendelenburg (**left**) and Tilt **(right)** used for the dynamic model (Table 3).

Table 3. Overview of variables for whole body movements.

Variable	Description
$\vec{r}_p$	Position of the patient's CoG in world coordinates
$\vec{v}_p$	Velocity vector of the patient's CoG
$\vec{\omega}_{tb}$	Angular velocity vector of the rotational Trendelenburg joint in world coordinates
$\vec{\omega}_{tl}$	Angular velocity vector of the rotational tilt joint in world coordinates
$\vec{r}_{tl}$	Distance vector between the tilt joint origin and the CoG of the patient ($\vec{r}_p - \vec{r}_{jtl}$)
$\vec{r}_{tb}$	Distance vector between the trend joint origin and the CoG of the patient ($\vec{r}_p - \vec{r}_{jtb}$)

Huang et al. present a novel anomaly detection approach [47] specifically tailored for anomaly detection in time series data. The basic concept of their approach is to use the state estimation capabilities of the KF to improve the anomaly detection performance of an AE. This distinguishes their KalmanAE from traditional Kalman-based techniques, as it optimizes the KF using the embeddings produced by the AE. However, this method does not directly model the relationships between channels in multivariate time series data, an area that our approach aims to address.

In their study, Sanchez et al. [48] examine the robotic handling and detection of flexible objects and propose a novel categorization of these objects that takes into account not only their geometric form, but also their physical properties. In their analysis, they note that there is a significant gap in the field due to the lack of methods that provide universal

solutions for different types of objects and tasks. In particular, they point out that the dynamic handling of flexible objects is still at an early stage of development.

In their paper, Cheng et al. present an unsupervised fault detection system for industrial robots based on a Gaussian mixture model using current signals [42]. They argue that the effectiveness of unsupervised machine learning algorithms in fault detection depends on the quality of the input fault features. They further propose that a better understanding of robotic systems through physics-based system analysis can help extract more effective fault features. The framework they propose involves pre-processing the signal and accurately distinguishing between different robot motion states in a measured current signal. Unique feature extraction is then used to derive robust fault features. These features are designed to be sensitive to faults but not to the robot's movements. However, they do not address problems such as deformation.

The field of research into combining model-based and data-based methods, commonly called hybrid filters, has gained momentum in recent years. Jin et al. [11] have noted that this approach is beneficial when a system becomes challenging to model accurately and a promising way to enhance a model's accuracy. One study by Liu et al. [49] demonstrated the effectiveness of this approach. They improved the predicted outcome of a model-based filter by training an LSTM model to estimate the discrepancy between the predicted and reference trajectories. By leveraging model-based and data-based approaches, hybrid filters can provide better results than using either method alone. However, they do not use their approaches to detect anomalies, a gap we target in this paper (Section 1).

4. Concept

Applying statistical methods (Section 2.2) that assume normal distribution to non-normal patient data will result in unwanted outliers since humans vary heavily in body proportions. The shoulder width, for example, varied from 1.5 to 3.25 head lengths for male subjects as examined by Kilgore [50]. This means that the standard deviation for the Z-score must be set so high to avoid false positives that the detection of true positives is unlikely. Even non-normally based methods such as IQR assume a probability distribution that does not adapt to individual patients. In addition, the static interpretability and generalization ability of, e.g., multivariate Z-scores become challenging, especially for high-dimensional data. In addition, statistical and information-theoretical methods are not suitable for modeling sequential data [51].

Based on the automotive observer by Weber et al., which uses different types of plausibility checks to monitor anomalies (Section 3), we present a generic approach for anomaly detection systems using static, dynamic, and learning checks (Figure 9). Dynamic checks use dynamic models (Section 2.1) and learning checks use data-based models (Sections 2.3 and 2.4). This paper focuses on combining the dynamic and learning checks since the dynamic checks are our proposed extensions, which are expected to reduce the number of features required (curse of dimensionality [30]) and still reduce the False Positive Rate (FPR). By using expert knowledge for the dynamic tests, the explainability given for a KF is also given for the hybrid variant. This means that only the part of the data-based learning check cannot be easily explained.

Accordingly, we have chosen a hybrid filter approach: The first part of the equation is represented by a Kalman filter KF to predict physically explainable system behavior. Here, the filter is parameterized through $\mathbf{A}, \mathbf{B}, \mathbf{Q}, \mathbf{H}, \mathbf{R}, \mathbf{L}$, (see Table 1) and receives an input tuple $\mathcal{U}, \mathcal{Y}$ (see Table 1) to predict the state.

$$KF_{(\mathbf{A},\mathbf{B},\mathbf{Q},\mathbf{H},\mathbf{R},\mathbf{L})} : (\mathcal{U}, \mathcal{Y}) \rightarrow \hat{\mathcal{X}} \tag{22}$$

In the case of UKF/EKF the functions h and f replace $\mathbf{A}, \mathbf{H}$, and for UKF the sigma function s is an additional parameter. The machine learning model $\mathcal{N}$ only has to model d_t from $\mathcal{D}$ in order to describe the system behavior that is physically difficult to model:

$$\mathcal{N}_\Theta : (\hat{\mathcal{X}} - \mathcal{Y}) \rightarrow \mathcal{D} \tag{23}$$

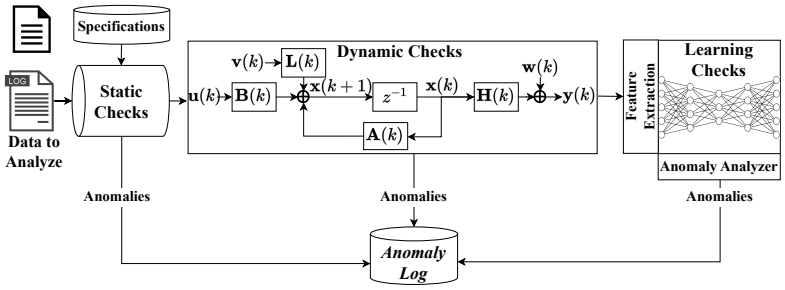

Figure 9. Extended anomaly observer concept based on [44].

Static checks are derivable from the system specification, leading to rules, e.g., that a velocity signal needs to be in a specified value range. Dynamic Checks are manually created physical models that are mathematically described as proposed here for the partial body movements of a patient (Section 4.1) or for whole body motions, as proposed in [15] (Section 3). Learning checks are data-based models such as AE, LSTM model (Section 2.4) or IF (Section 2.3) trained on data collected for the system. Each check can be combined and can provide input to the following checks at the same time. If one of the checks detects an anomaly, it is logged for further processing, such as user notification.

4.1. Dynamic Check for Partial Movements of the Patient's Body

Given an OR table with a load determination system (Section 1), the overall CoG of the load can be determined. Therefore, the idea is to check the trajectories of the CoG during a movement for plausibility as described in [15] including movements of the patient's body parts (Figure 10). Nevertheless, as a simplification, the dynamic check assumes that the patient's body mass distribution and flexible elements are neglected, so rigid body dynamics can be assumed (Section 1).

In addition, for a proof of concept, it is sufficient to sum individual body parts into two big parts: the upper and lower body of the patient. This simplification allows a more straightforward calculation of the patient's CoG. Specifically, for rigid body dynamics with homogeneous mass distribution, the patient's upper body CoG position $\vec{r}_{ub}$ as well as the patient's lower body CoG $\vec{r}_{lb}$ can be used to calculate the patient's CoG $\vec{r}_p$.

To calculate the patient's CoG, it is necessary to use a proportion λ_{ub} for the moved upper body part with the back joint and a proportion λ_{lb} for the impacted upper body part with the leg joint.

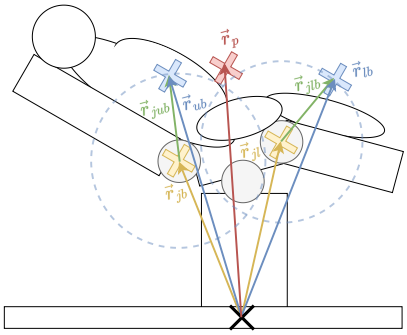

Figure 10. Patient position $\vec{r}_p$ in dependency to upper body distance $\vec{r}_{ub}$, lower body distance $\vec{r}_{lb}$, back joint distance $\vec{r}_{jb}$ and leg joints distance $\vec{r}_{jl}$ as well as the resulting distances $\vec{r}_{jlb}$ and $\vec{r}_{jub}$ describing the rotational distances (Table 4).

Table 4. Overview of variables for partial body movements.

Variable	Description
$\vec{r}_p$	Position of the patient's CoG in world coordinates
$\vec{v}_p$	Velocity vector of the patient's CoG
$\vec{r}_{ub}$	Position of the patient's upper body in world coordinates
$\vec{r}_{lb}$	Position of the patient's lower body in world coordinates
$\vec{r}_{jb}$	Position of the back joint origin
$\vec{r}_{jl}$	Position of the leg joint origin
$\vec{\omega}_{jb}$	Angular velocity vector of the rotational back joint in world coordinates
$\vec{\omega}_{jl}$	Angular velocity vector of the rotational leg joint in world coordinates
$\vec{r}_{jub}$	Distance vector between the back joint origin and the CoG of the patient's upper body ($\vec{r}_{ub} - \vec{r}_{jb}$)
$\vec{r}_{jlb}$	Distance vector between the leg joint origin and the CoG of the patient's lower body ($\vec{r}_{lb} - \vec{r}_{jl}$)

Neglecting the inhomogeneous mass distribution and deformation of the human body, the dynamics for rigid bodies can be applied to describe the position and velocity of the patient's CoG:

$$\vec{r}_p = \lambda_{ub}\vec{r}_{ub} + \lambda_{lb}\vec{r}_{lb} \tag{24}$$

$$\dot{\vec{r}}_p = \vec{v}_p = \lambda_{ub}\vec{v}_{ub} + \lambda_{lb}\vec{v}_{lb} = \lambda_{ub}\vec{\omega}_{jb} \times \vec{r}_{jub} + \lambda_{lb}\vec{\omega}_{jl} \times \vec{r}_{jlb} \tag{25}$$

To simplify matters, no tilting of the patient is considered, so only the rotation around the y-axis is considered for the leg and back joint, resulting in the y-axis value being irrelevant.

$$\begin{bmatrix} r_{px} \\ r_{py} \\ r_{pz} \end{bmatrix} = \begin{bmatrix} 0 \\ \lambda_{ub}\omega_{jb} \\ 0 \end{bmatrix} \times \begin{bmatrix} r_{jubx} \\ r_{juby} \\ r_{jubz} \end{bmatrix} + \begin{bmatrix} 0 \\ \lambda_{lb}\omega_{jl} \\ 0 \end{bmatrix} \times \begin{bmatrix} r_{jlbx} \\ r_{jlby} \\ r_{jlbz} \end{bmatrix} \tag{26}$$

$$\begin{bmatrix} \dot{r_{ub,x}} \\ r_{ub,z} \\ r_{lb,x} \\ r_{lb,z} \\ \omega_{jb} \\ \omega_{jl} \end{bmatrix} = \begin{bmatrix} \omega_{jb}(r_{ub,z} - r_{jb,z}) \\ -\omega_{jb}(r_{ub,x} - r_{jb,x}) \\ \omega_{jl}(r_{lb,z} - r_{jl,z}) \\ -\omega_{jl}(r_{lb,x} - r_{jl,x}) \\ 0 \\ 0 \end{bmatrix} \tag{27}$$

Other joints of the OR table are neglected in this scenario so that the positions of the joints $\vec{r}_{jb}$ and $\vec{r}_{jl}$ can be assumed as constants. Thus, they do not need to be measured and do not need to be contained in the state vector $\vec{\mathbf{x}}$. Another simplification to reduce the dimension of the state is achieved by using a constant velocity model rather than a constant acceleration model ($\dot{r}_{ub,x} = v_{ub,x} = C$). The non-linear measurement function $h(\mathbf{x})$ can be calculated as follows:

$$h(\mathbf{x}) = \begin{bmatrix} \lambda_{ub}r_{ub,x} + \lambda_{lb}r_{lb,x} \\ \lambda_{ub}r_{ub,z} + \lambda_{lb}r_{lb,z} \\ \omega_{jb} \\ \omega_{jl} \end{bmatrix} \tag{28}$$

4.2. Learning and Hybrid Filter Checks

Since the dynamic model is based on the assumptions of the standardized patient and a rigid OR table model with Gaussian process and measurement noise, the system will detect measurements with patients outside these norms as anomalies. Therefore, classical methods are expected to fail in these situations.

Thus, AE, LSTM model (Section 2.4), and IF (Section 2.3) are considered as standalone variants as well as hybrid variants in combination with the dynamic check for partial patient body movements (Section 4.1).

It is common practice to use 1D Convolutional Neural Network (1DCNN) AEs (Figure 11) for analyzing time series data [52]. This design is also considered suitable for the scenario examined here. The encoder consists of two convolutional layers: a max-pooling layer and a dense layer. On the other hand, the decoder has a structure that is the mirror

image of the encoder, which includes a dense layer followed by an unpooling layer and two deconvolutional layers. Finally, the decoder ends with another convolutional layer. The activation functions used are Rectified Linear Unit (ReLU), Leaky Rectified Linear Unit (LReLU), and Sigmoid.

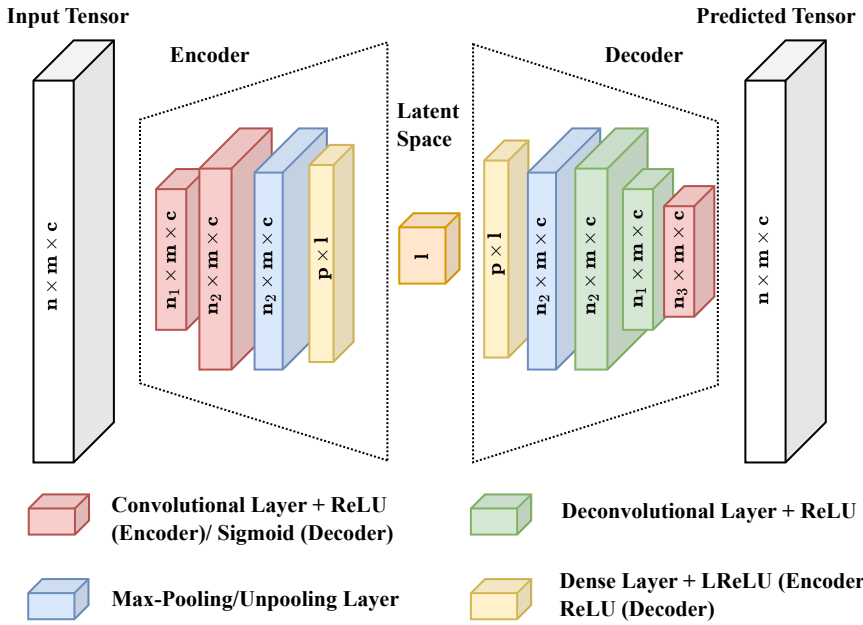

Figure 11. AE architecture (n: Number of windows, m: window size, c: Number of channels/features, n_1: Neuron number of first conv. and second deconv. layer, n_2: Neuron number of second conv. and first deconv. layer, n_3: Neuron number of third conv. layer, p: pooling layer size, l: latent space dimension).

The LSTM network model used in this study consists of three layers of LSTM units, with a dropout layer following each LSTM layer to prevent overfitting (Figure 12). The activation functions can be adjusted accordingly: Each LSTM layer contains a ReLU activation function. In contrast, a linear activation function should be used in the dense layer at the output. Also, other activation functions are usable, such as Swish activation instead of ReLU in the LSTM layers. However, the first approach was selected to predict the absolute expected error in this case.

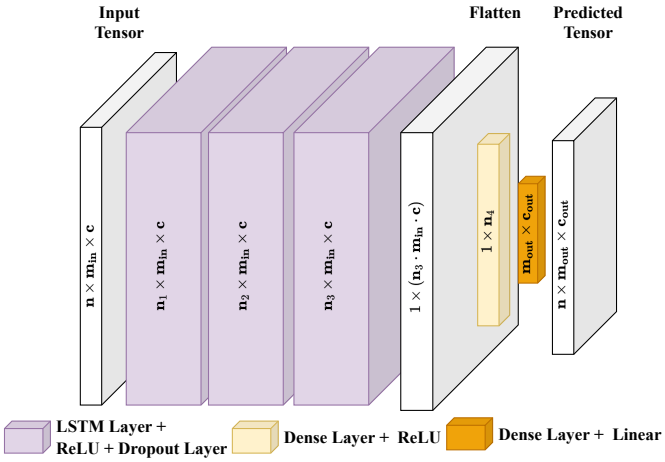

Figure 12. Implemented LSTM architecture with n: number of windows, m_{in}: window size, c: number of channels/features of input, n_i: neurons number of layer i, m_{out}: window size of predicted output tensor, c_{out}: number of channels/features of output.

5. Prototypical Implementation and Evaluation

In the following, we apply the generic approach (Section 4) and the safety system for detecting errors in load recognition systems of OR tables (Section 3) to monitor the influence of the back movement of an OR table on the patient's CoG.

5.1. Data synthesis Using Digital Twins

In the context of medical robotics, simulation models are becoming increasingly prevalent for developing and testing surgical procedures and devices. A digital twin of the Gazebo OR table (Figure 13) is employed with Robot Operating System (ROS) 2 nodes for generating physical data, allowing the evaluation of algorithms under various scenarios. Simulation models offer a cost-effective and efficient approach to system development, minimizing the risk associated with physical prototypes and providing accurate representations of the system's behavior and dynamics.

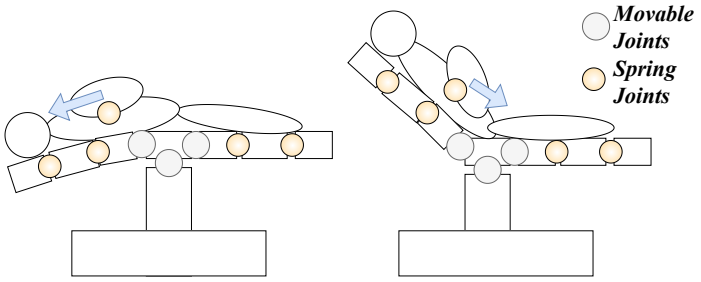

Figure 13. Patient and OR table models during a back movement.

Flexible elements such as muscles and tissues need to be accurately represented to enhance the accuracy and reliability of patient models. Spring joints are applied here to model these elements, allowing a more realistic simulation of the patient's body movement and deformation (Figure 13). The use of spring joints also enables the representation of the flexibility of the OR table's components, which is crucial for accurate simulations of the system's dynamics. These deformations are not designed into the dynamic model (Section 4.1) and, therefore, are expected to impact its estimation negatively.

The patient model is created in Unified Robot Description Format (URDF) based on relevant standards and regulations in medical technology. Here, the IEC 60601-1 [53] is used as a foundation (Figure 14). Flexible elements are introduced at the abdomen and upper legs for heavy patients to represent the patient's movement and deformation.

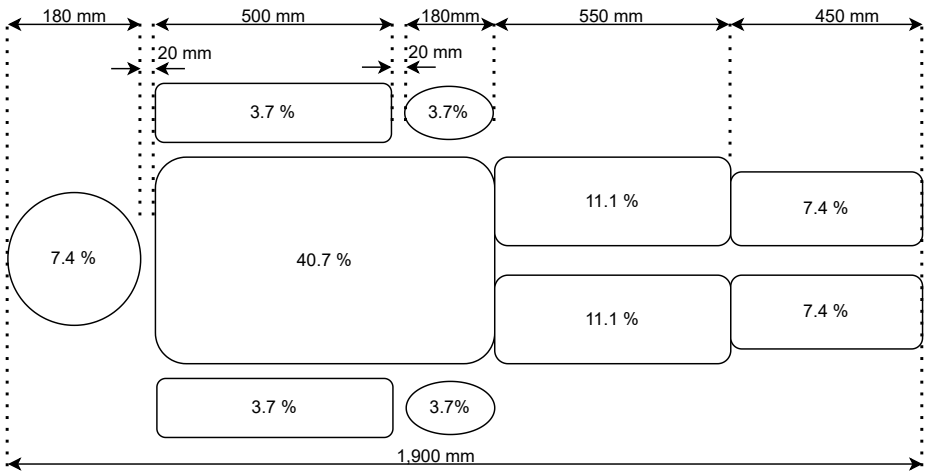

Figure 14. Patient mass distribution according to IEC60601-1 [53].

5.2. Distributed Anomaly Detection

Our architecture for hybrid anomaly detection consists of the OR table and a backend system (Figure 15). This server-client separation allows the computationally intensive data-based models to be processed in a cloud. In addition, flexibility is created as models can be exchanged or retrained without interfering with the embedded system of the OR table. The OR table uses a service-oriented architecture with Data Distribution Service (DDS) as middleware through the use of ROS2. The test setup consists of a controller service that represents the user interaction of the OR table. During simulation time, the OR table is moved to different target positions at random times.

Based on the Gazebo proxy data, a dynamic check of the patient CoG estimate is performed every 100 ms. By transferring the estimate and the measurement data to the backend system, the estimate is checked there for anomalies using data-based models. The results are logged in a Comma-Separated Values (CSV) file. The dynamic check of the CoG estimate is realized as an ROS2 node (see Section 5.4). Together with the learning checks (see Sections 5.5–5.7), the system forms the hybrid filter. In the case of a connection failure, the system performs a simplified independent check based on the ROS2 node alone.

The connection between the backend system and the OR table is established over a REpresentational State Transfer (REST) interface. Several REST services provide the interaction needed for the dynamic checks and the simulation environment. At startup, the trained models are selected over a REST interface based on the currently determined patient weight and preferred data-based model (LSTM, IF, or AE) and loaded into the learning check. With another ROS2 node, random positions are commanded to the Gazebo OR table simulation that executes the desired joint movement. This is then sent to the *Patient CoG Estimation* dynamic check using the ROS1/ROS2 Bridge.

The dynamic check estimates each patient's CoG, which is then sent to the backend system for evaluation. The inference is sent back to the node for the patient estimation. If the estimated value is an anomaly, the UI raises an alarm triggered by the dynamic check.

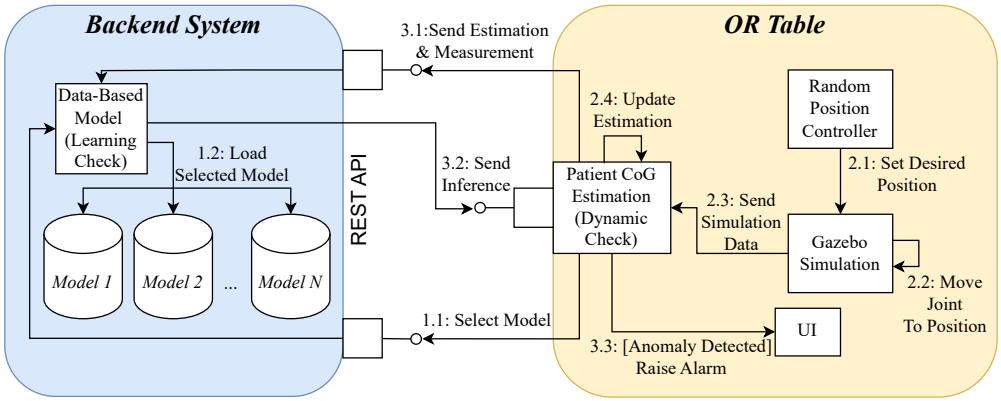

Figure 15. Communication diagram for distributed hybrid anomaly detection implementation.

In a real-world OR scenario, we consider the backend system either as part of the hospital IT, as a dedicated cloud server provided by the device manufacturer, or as a supervision system directly in the OR, as proposed in [54]. In the latter scenario, the OR table would send the estimation data to the OR monitoring system via a SDC network, which has more computational resources than the embedded system of the OR table. The dynamic check is considered to be executable on a performant embedded system and thus, for example, as part of a dedicated communication gateway of the OR table, comparable to the architecture proposed in [7].

5.3. Classification of Deviations as Anomaly

As mentioned in Section 2.4, a threshold must be found for the AE and the LSTM network used to ensure a clean categorization of the anomaly definition. These include:

- The reconstruction error of the AE variants
- The LSTM prediction of the difference value of the KF output to the measurement
- The absolute difference of the pure LSTM position prediction to the measurements

When using the IF, the grouping is based on the path length and thus implicitly finds its threshold (Section 2.3).

Similar to [33], we have decided to use the 99th percentile. There are several reasons for this method over a manual threshold, such as the ability to adapt to new data and the creation of objective comparability between the approaches in the evaluation. The choice of percentile can also be changed. However, a low FPR is vital for our medical application as false alarms in hospitals [55] lead to alarm fatigue, causing critical alarms to be missed [56], and reducing the quality of care [57]. Accordingly, we have selected a correspondingly high percentile to meet this target. In addition, each signal, e.g., CoG in x and z direction, is evaluated separately using a dedicated threshold. Therefore, an input tensor is classified as an anomaly when it exceeds the threshold value in one of these signals.

Four types of anomalies, each in the x and z position (Figure 16), are used for the evaluation of the different algorithms based on the overview of anomalies by Weber [45] (Table 5).

Table 5. Position anomaly scenarios (Figure 16).

Anomaly	X-axis	Z-axis	Potential Scenario
Positive/Negative Step Plateau	(1) jump to (x = −0.3 m) between 100 and 103 s (3 s)	(2) jump to $z = 1.2$ m between 140 and 143 s (3 s)	Indicative of a sensor defect or a potential attack where specific positions are set.
Plateau	(4) x stuck at a value between 245 and 250 s (5 s)	(3) z stuck at a value between 163 and 168 s (5 s)	A defect or manipulation attempt that keeps the current position value stuck at the current value while recovering the actual value after the anomaly.
Positive/Negative Ramp Plateau	(5) x increase from 300–310 s (10 s), decrease from 320 to 330 s with ≈ 1.33 cm/s	(6) z increase between 400 and 410 s (10 s), decrease from 410 to 420 s with ≈ 1.33 cm/s	Indicates a possible sensor error or an attempt by an attacker to manipulate the position to a desired value gradually.
Positive/Negative Ramp with Jump Back	(7) x increase between 500 and 510 s (10 s) with ≈ 1.33 cm/s	(8) z increase between 550 and 560 s (10 s) with ≈ 1.33 cm/s	Indicates a possible sensor error or an attempt by an attacker to manipulate the position to a desired value gradually, ending with a jump to the actual value.

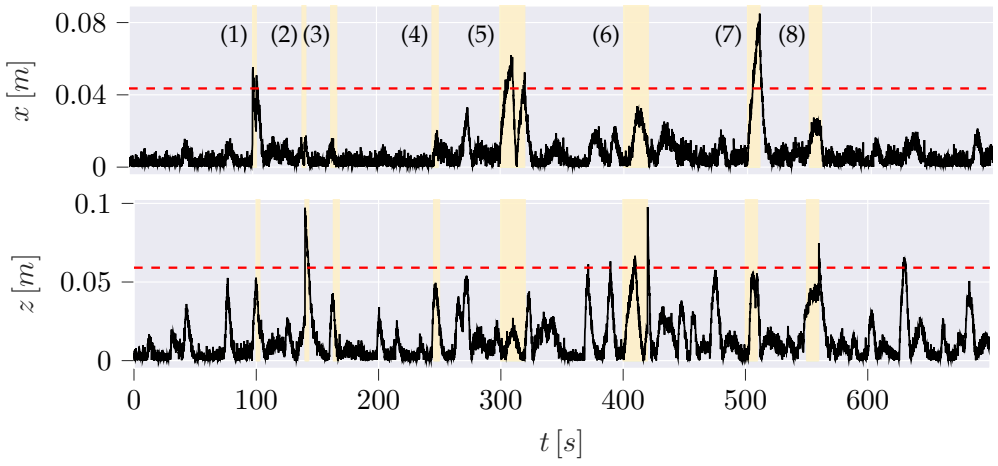

Figure 16. Absolute error of the estimated CoG by the UKF to the measurement data including anomalies (yellow highlighted area, Table 5).

5.4. Partial Body Movement UKF

As a dynamic check for partial body movements, the UKF and EKF algorithms have been implemented. These implementations are structured as ROS2 nodes, which subscribe to the positions and velocities of the CoG derived from the Gazebo simulation. The estimated CoG positions and OR table joint velocities are logged into a CSV file for comprehensive analysis and training of the data-based models. A dedicated internal class, Gazebo Proxy, facilitates communication between the Gazebo environment and the dynamic checks. The estimations are updated every 100 milliseconds to balance real-time responsiveness and computational efficiency. However, as the results are similar for the simulated scenario, only the UKF is discussed here for further examination because the EKF is expected to create similar results as its hybrid variant.

With the previously mentioned simplifications (Section 4.1), the patient's body is only divided in the lower and upper body. Thus, according to the standard patient of IEC60601-1 [53], the upper body is considered to be $\lambda_{ub} = 0.63$, and the lower body is considered to be $\lambda_{lb} = 0.37$. When a patient is seated in a beach chair position (Figure 10), which matches the scenario under examination, their upper body makes up around 63% of their overall mass.

5.5. Isolation Forest (IF)

The Python implementation of our IF utilizes the Scikit-learn [58] library and its dependencies. Our implementation can take dynamic model estimations, sensor measurements, or their difference as inputs. The standalone variant is trained on the position measurements (Figure 17). On the other hand, the hybrid variant of the implementation uses a two-dimensional input of the difference between the measurement and the UKF estimation in the x and z CoG direction (Figure 18).

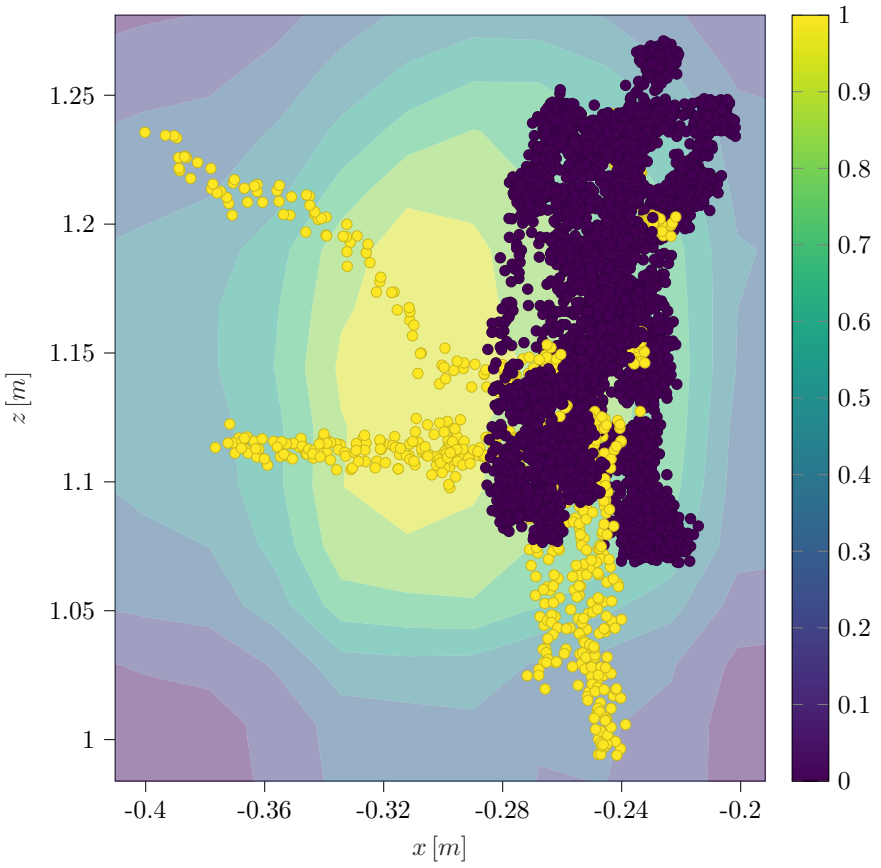

Figure 17. Anomalies (yellow) detected by the pure IF in position measurement x and z.

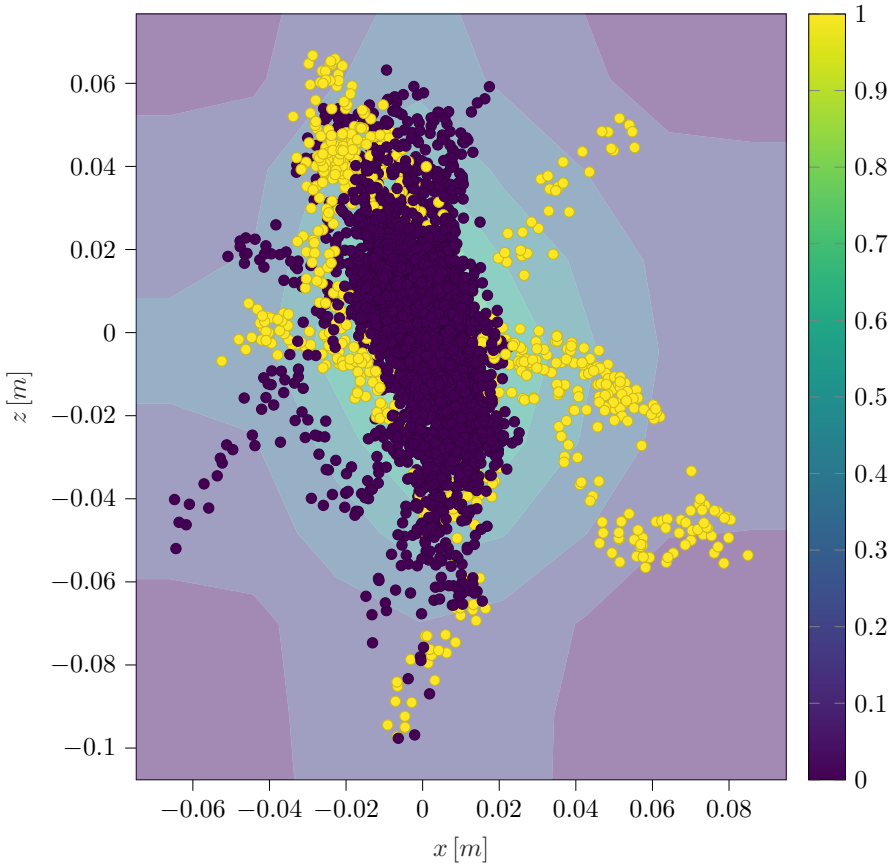

Figure 18. Anomalies (yellow) detected by the hybrid IF in difference of UKF estimation to the measurement of positions.

We do not select any time window to ensure the IF implementation is minimal, and evaluate the low-resource variant of the hybrid filter. As the IF does not require a manually predefined threshold for classification into anomalies (Section 2.3), the training data primarily affect the evaluation metrics such as FPR and True Positive Rate (TPR), and there is limited manual influence for subsequent fine-tuning in inference time.

In our result, the pure IF (Figure 17) is not capable of determining meaningful thresholds, leading to high FPR (∼40%) and False Negative Rate (FNR) (∼55.5%). At the same time, the hybrid variant centers the normal data in between the thresholds so that the FPR (∼13%) as well as the FNR (∼18.8%) can be improved significantly.

Since the trajectories of anomalies can be guessed from the hybrid plot, a disadvantage of the hybrid variant or at least of the classification of anomalies is revealed: The anomalies are classified based on the manipulation of the measurement data. It is not considered that the dynamic model, which is the UKF here, may need time to recover from a sudden change of the measurement after an anomaly back to normal data. Especially in jump scenarios, the difference between measurement and estimation will remain high for a short period until the dynamic model has recovered. These data points are not classified as anomalies here, which will also increase the FNR of all examined learning checks. However, it is noticeable in the time series of the pure IF (Figure 19) that most of the detected anomalies do not correlate with the anomalies in contrast to the hybrid variant (Figure 20).

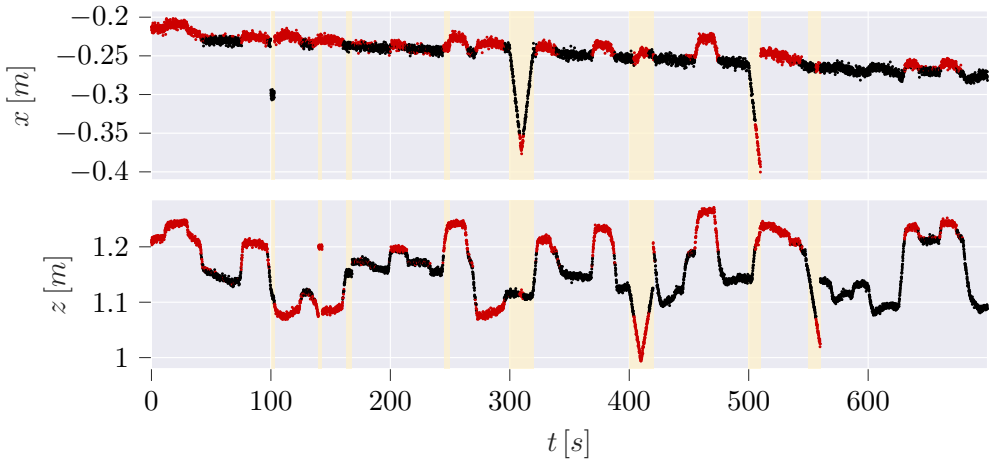

Figure 19. Detected anomalies (red) in time series for IF.

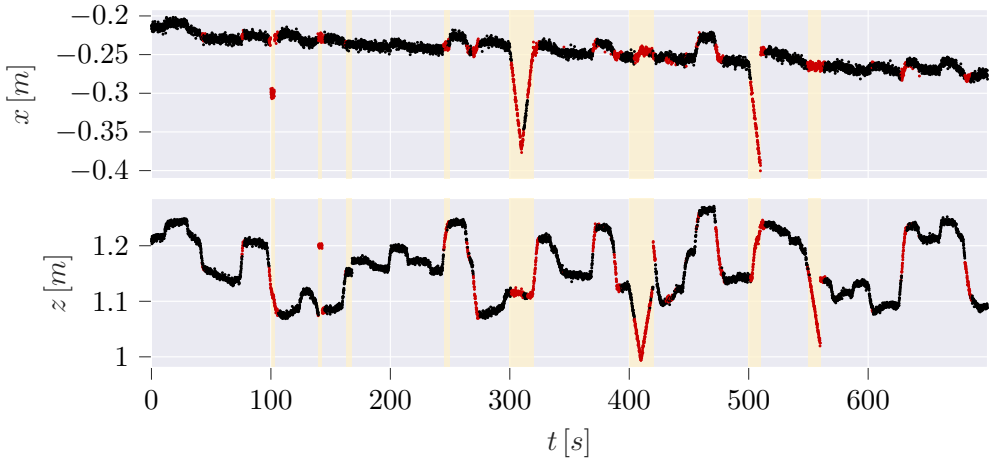

Figure 20. Detected anomalies (red) in time series for hybrid IF.

5.6. LSTM

The LSTM network is realized in Python using the TensorFlow library [59]. It is suitable for various feature combinations for predicting the output based on the position and joint velocity measurements and the estimated state of the KFs (Section 4.1), which can be arbitrarily combined, including a variable time series.

As the LSTM predicts the next value (Figure 21), we consider the pure variant as a deep learning variant to the dynamic model. Therefore, we chose all position and velocity measurements as input for a time window of 30 values, which corresponds to 3 s. However, the LSTM does have a disadvantage compared to the UKF, which is not iteratively updated based on all previous measurements. Instead, it uses an adequate time window, which needs to be exploratively determined.

The hybrid LSTM variant is designed to predict the estimated deviation of the following UKF iteration based on a time window of 20 values corresponding to 2 s. The error of the UKF estimation compared to the measurement (blue) (Figure 22) is, therefore, directly predicted by an LSTM (orange). The absolute difference is then used as the basis for the anomaly determination (black).

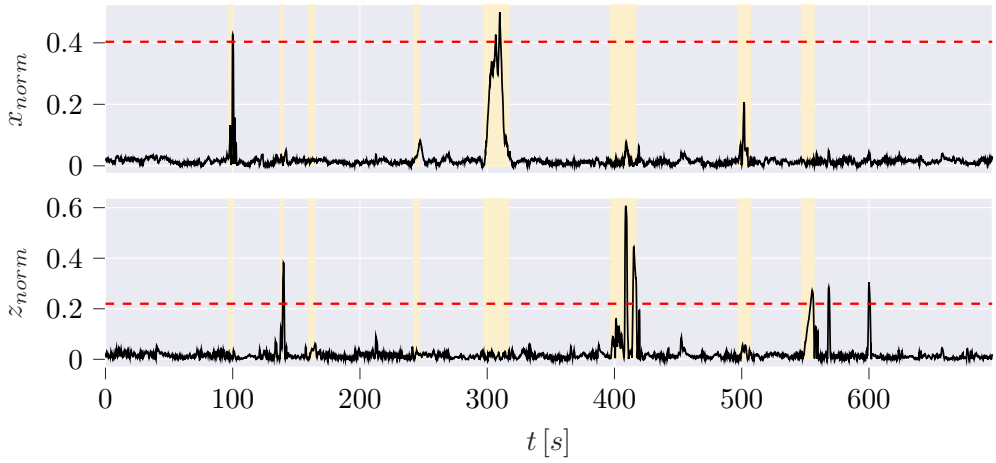

Figure 21. Absolute scaled difference between estimation and measurement to forecast at LSTM.

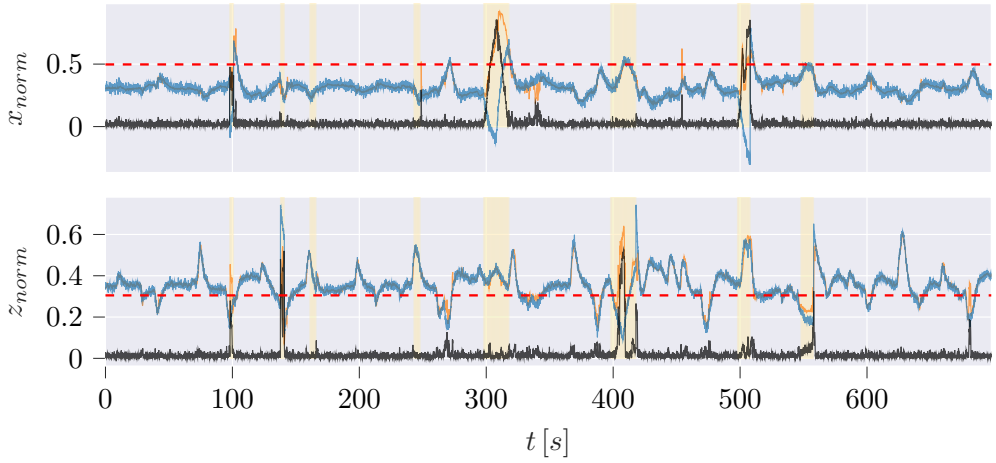

Figure 22. Difference (black) of the absolute error (difference of estimation and measurement for CoG position x and z) of the UKF (blue) predicted by an LSTM (orange).

The best results are achieved with the hybrid LSTM in terms of FPR as it decreases the FPR of the standalone LSTM by a factor of 30 (∼0.6% to ∼0.02%), while also decreasing the FNR by nearly 14%. In both cases, this leads to high precision. Furthermore, it is conspicuous that both LSTM variants have a high threshold based on the 99th percentile compared to the average non-anomaly data or even compared to peaks in the non-anomaly data. Other architectures of recurrent neural networks, such as GRU, will behave in a similar way to LSTM, which is why a closer look can be omitted at this point. They can also lead to serious problems such as vanishing gradients (vanilla LSTM) and are, therefore, considered obsolete.

5.7. Autoencoder (AE)

The PyTorch library in Python is used to implement the AE, which is used to reconstruct a desired time series of estimations and measurements [60]. The reconstructed CoG x and z values in a time window of 30 values (3 s) are generated by the basic AE, (Figure 23). The first hybrid variant, on the other hand, employs the difference between UKF estimation and measurement using the same window size (Figure 24).

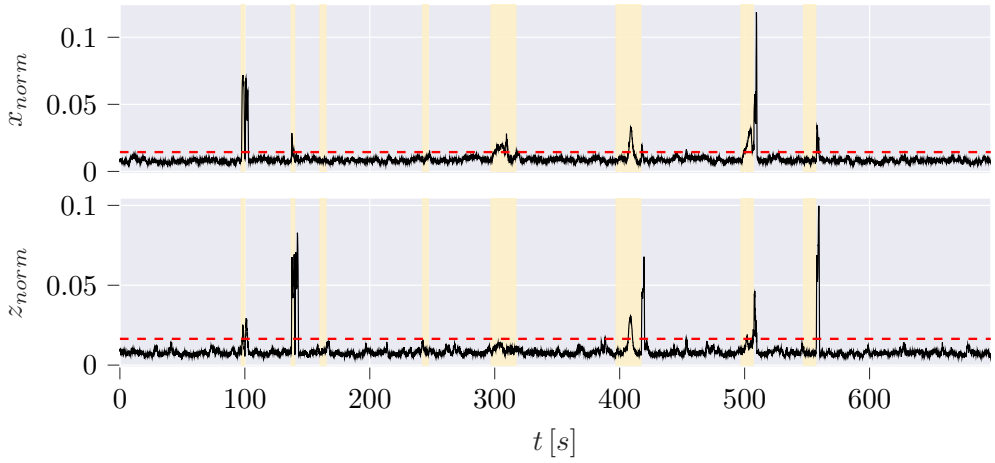

Figure 23. AE reconstruction MAE for position difference of estimation to measurement x and z.

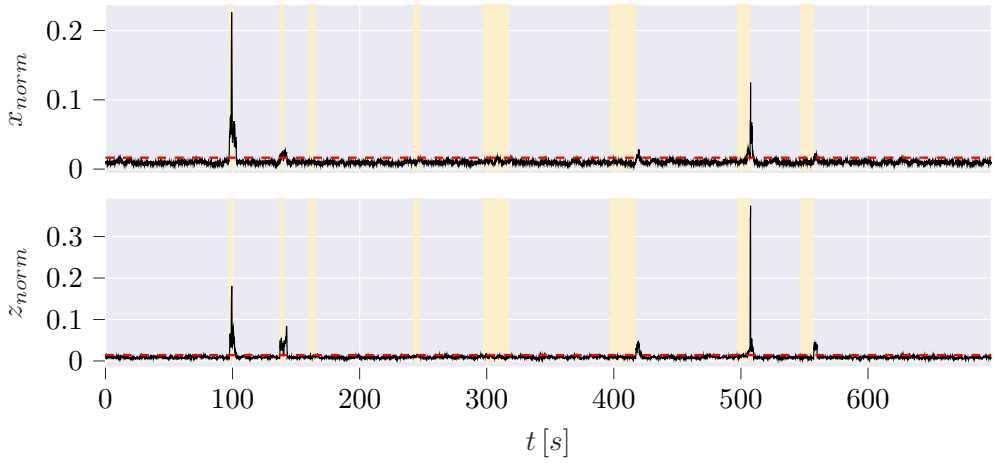

Figure 24. First hybrid AE variant reconstruction MAE for position difference of estimation to measurement x and z.

We calculate the reconstruction error for all variants as MAE, meaning that the average of all reconstructed values compared to the measurement is calculated as follows:

$$MAE_{window} = \frac{1}{n}\sum_{i=1}^{n}|\hat{\mathbf{x}}_i - \mathbf{x}_i| \tag{29}$$

where

- n: Number of data points in the window.
- $\hat{\mathbf{x}}_i$: Reconstructed value for the i-th data point.
- $\mathbf{x}_i$: Measured value for the i-th data point.

While the AE has a higher FPR (2.3%) and FNR (65.5%) with smaller window sizes, another approach to the hybrid AE reveals improved performance compared to the difference between dynamic model estimation and measurement (FPR: 2.7%; FNR: 87.9%): When using the measurements and estimations as separate values, a time window of four results in a lower FPR and FNR (Figure 25), while increasing the window size worsens the results. In addition, the latent space can be reduced from size 8 to 2 and improve the performance at the same time.

The hybrid AE shows that although the FPR and FNR cannot be lowered for the examined scenario, the second approach shows that training effort and window size can be decreased, which makes this approach more interesting for higher dimensional tasks. It slightly increases the FPR by 1%, but decreases the FNR by 16.3% compared to the UKF.

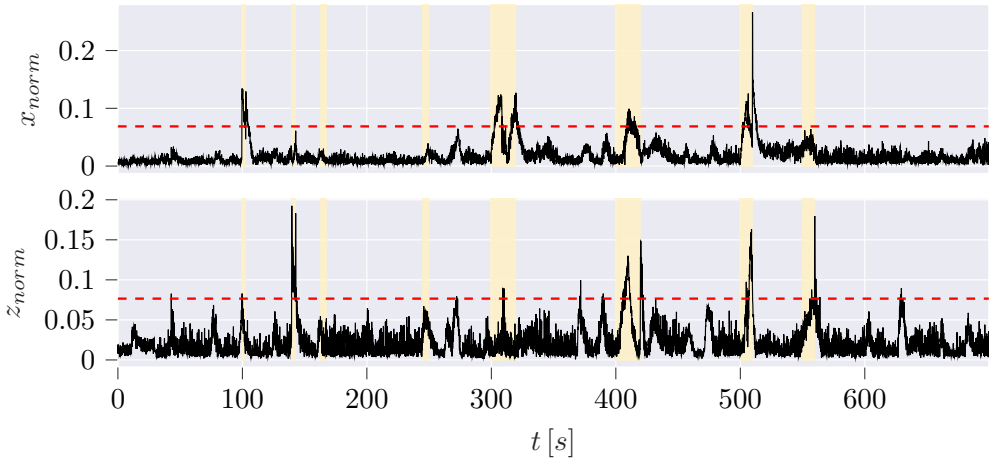

Figure 25. Second hybrid AE variant reconstruction MAE of measurements with window size 4 reconstructing UKF estimations and measurements of positions.

6. Discussion

6.1. Discussion on Pure Learning & Dynamic Checks

Table 6 compares the confusion matrices of all data-based algorithms with the UKF. The pure LSTM network performs best for true-negative and false-positive, closely followed by the UKF. Although the pure IF performs best on false-negative and true-negative data, the difference between correctly and incorrectly classified cases is so insignificant that the data point only has a circa 58.4% chance of being classified correctly (accuracy). Hence, the practical relevance of this approach is questionable.

Table 6. Confusion matrices of pure data-based and dynamic models.

Model	**AE**		**IF**		**LSTM**		**UKF**	
	False	True	False	True	False	True	False	True
False	6064	141	3746	2488	**6167**	**37**	6169	65
True	502	264	**425**	**341**	674	92	594	172

Table 7 compares the metrics of all examined data-based algorithms and the UKF as representative for the dynamic checks. While the pure LSTM fulfills the aim of a low false-alarm rate the most (FPR of 0.6%), the other methods outperform it in the other metrics. The UKF has the highest precision (72.6%), the AE has the highest accuracy (90.8%), the highest F1 Score (45.1%), and the highest ROC AUC Score (66.1%). Therefore, for a balanced anomaly detection performance, the AE is the best choice for a pure dynamic/learning check. As previously stated, the practical relevance of the pure IF performance is questionable, although it has the highest recall (44.5%) and the lowest FNR.

Table 7. Metrics of all pure learning and dynamic checks.

Metric	**AE**	**IF**	**LSTM**	**UKF**
Accuracy	**0.908**	0.584	0.898	0.906
Precision	0.652	0.121	0.713	**0.726**
Recall	0.344	**0.445**	0.120	0.225
F 1 Score	**0.451**	0.190	0.205	0.343
ROC AUC Score	**0.661**	0.523	0.557	0.607
False Positive Rate	0.023	0.400	**0.006**	0.010
False Negative Rate	0.655	**0.555**	0.880	0.775

6.2. Discussion on Hybrid Filters

Table 8 compares the confusion matrices of the different hybrid filters. Here, the hybrid LSTM network performs best in the case of true-negatives and false-positives, while

the hybrid IF performs best in the case of false-negatives and true-negatives. The hybrid AEs offers a solution between the performance of the others, while the second variant is significantly more promising.

Table 8. Confusion matrices of hybrid filters in comparison.

Hybrid Model	**AE (Variant 1 & 2)**		**IF**		**LSTM**	
	False	True	False	True	False	True
False	6040/6107	165/124	5424	810	**6213**	**1**
True	673/469	93/297	**144**	**622**	571	195

Table 9 compares the metrics of all examined hybrid filters. Overall, the hybrid LSTM network fulfills the aim of low false alarms (Section 5.3) best since it has the highest accuracy, precision, and lowest FPR. However, severe anomalies might not be identified with the high FNR and low recall. Here, AE and IF in both model variants perform better.

Table 9. Hybrid filter metrics in comparison in all anomaly scenarios and normal behavior data.

Metric	**Hybrid AE**	**Hybrid IF**	**Hybrid LSTM**
Accuracy	0.880/0.915	0.864	**0.918**
Precision	0.360/0.705	0.434	**0.995**
Recall	0.121/0.388	**0.812**	0.255
F 1 Score	0.182/0.500	**0.565**	0.405
ROC AUC Score	0.547/0.684	**0.841**	0.627
False Positive Rate	0.027/0.020	0.130	**0.0002**
False Negative Rate	0.879/0.612	**0.188**	0.745

Also, in the case of the hybrid IF, the FPR and the FNR can both be significantly improved by a factor of approximately 3. Still, it is much higher than the LSTM models. Nevertheless, the hybrid IF is the best choice if a balanced FPR and FNR is desired with high recall and F1 Score. All algorithms except for the hybrid IF, which is not evaluated based on the 99th percentile, have a high FNR of more than 50%. *Thus, we assume that the chosen anomalies represent hard-to-find ones* due to partially minor deviations in the measurement compared to the true values that are additionally covered by noise. Therefore, we do not reach the F1 value of 0.7 typically targeted in the literature, and in addition, the 99th percentile is a very conservative threshold. However, the jump anomalies (1 & 2) are detected by all checks. In comparison, the stuck-at anomalies (3 & 4) are only discovered in the x (3) direction by the AEs, but only shortly above the threshold.

Therefore, it is likely that an ideal anomaly detection system, as proposed in Section 4, uses a mixture of checks to cover as not all seem to be discoverable with all algorithms. In addition, the FPR could be reduced further by a voting system of several checks, e.g., an anomaly must be discovered in at least two checks. Furthermore, only the LSTM variants outperform the dynamic check implemented as UKF in terms of the FPR. Nevertheless, all data-based and hybrid models outperform the UKF in terms of FNR and recall.

7. Conclusions and Future Work

Regarding feature reduction and possible data combination, whether the pure database models can keep their performance is questionable. More features may result in less distinguishable anomalies (distance in multidimensional space). Therefore, reducing necessary features and a smaller window size of the hybrid filters may outperform the entire setup for position monitoring of the patient. Furthermore, in all cases, the hybrid variant improves performance. In addition, for the AE and LSTM cases, the feature set, including the necessary window size, can be reduced compared to pure data-based models.

An application-specific advantage of the proposed check is that the position of the patient represented as CoG results from the entire kinematic chain and thus the entire

OR table joint positions. Checking the patient's CoG is sufficient to check all other joint positions for plausibility. If one of these positions is determined incorrectly, this leads to deviations in the movements. For example, suppose the patient is tilted around their longitudinal axis. In that case, there will also be a change in the y-axis during a movement of the upper body, which is not predicted by the dynamic model (Section 4.1) and would, therefore, lead to deviations.

The scenario chosen for evaluation is only a small subset of the capabilities of the OR table and the possibilities of patient positioning and anthropometry. The data collected via a simulation of 5 h of driving the back joint in a static scenario is enough to yield results on the performance of several algorithms. It is questionable if all the scenarios not examined here can be covered with the data collected in the field. While in the automotive industry manufacturers can reach out to the data of millions of cars increasing year by year, in the medical device industry, such as for the OR table, manufacturers can only use the data of few hundred to a thousand each year. In addition, not all of these OR table are used for all patient types, positions, and surgical procedures [7].

The primary focus to improve the presented hybrid anomaly detection lies in selecting an appropriate data-based model and design, as this choice substantially impacts the results. Following this, optimizing the dynamic model becomes the next priority. The lower priority is the enhancement or expansion of the training dataset, serving as the final adjustment to further refine overall performance.

As the hybrid AEs, in general, needs smaller windows and especially the IF could be improved significantly using the UKF estimation, we assume that most of the historical information is stored in the state of the KF. Therefore, also other internal states of the KF that result from sensor fusion and are not measured, e.g., $\vec{r}_{ub}$ or $\vec{r}_{lb}$, can additionally improve the results. Thus, we expect the performance of the hybrid filters to be heavily dependent on the design of the dynamic model.

As this paper concentrates on a single device, the approach can be applied to interoperable medical devices in an SDC network (compare with the approach presented in [54]). The different devices in the network may observe each other, creating a more robust system of systems. Each medical device in the OR is supervised by backend features, and compromising a single heavily connected device might be more challenging than a standalone system with minimal connectivity interfaces. However, this also requires analyses of response times, especially when hard real-time is needed.

To enhance the outcome, it is possible to assess other KFs that utilize Interacting Multiple Model (IMM) algorithms to adjust to varying patient positions and types. Moreover, since the second variation of the hybrid AE employs a small window, a DenseAutoencoder may be more appropriate than the currently selected 1DCNN AE.

The current estimation could be corrected to the likely value as another possible next step to recover from anomalies. This can be conducted with the values predicted from the LSTM or the AE's reconstructed window. This could also increase the availability of the medical device in case of anomalies, vastly improving safety.

In the case of a safety or security incident, it is mostly not important which data point was an anomaly, but only the fact that an anomaly has occurred for several seconds or even minutes and with high confidence (small FPR). Therefore, the uncertainty should be calculated and logged, e.g., using Monte Carlo Sampling.

Finally, a comprehensive runtime and resource consumption evaluation must be carried out in the form of a benchmark for the various combinations of algorithms, also in comparison to their pure variants. This also includes the runtime evaluation for distributed approaches using a backend system. However, this requires a more extensive prototype implementation in a distributed architecture under hospital network conditions.

Author Contributions: Conceptualization, A.P.; methodology, A.P., M.Z. and L.S.; software, A.P., M.Z. and L.S.; validation, A.P., M.Z. and L.S.; writing—original draft preparation, A.P., M.Z. and L.S.; writing—review and editing, A.P., M.Z., L.S. and E.S.; visualization, A.P., M.Z. and L.S.; super-

vision, E.S.; funding acquisition, A.P. All authors have read and agreed to the published version of the manuscript.

Funding: The APC was funded by KIT-Publication Fund of the Karlsruhe Institute of Technology grant number 3293.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data is contained within the article.

Acknowledgments: We acknowledge support by the KIT-Publication Fund of the Karlsruhe Institute of Technology. In addition, we acknowledge support by Laurin Koch for the implementation of the LSTM network.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

1DCNN	1D Convolutional Neural Network
AE	Autoencoder
CAN	Controller Area Network
CSV	Comma-Separated Values
CoG	Center of Gravity
DBN	Dynamic Bayesian Network
DDS	Data Distribution Service
EKF	Extended Kalman filter
FNR	False Negative Rate
FPR	False Positive Rate
IF	Isolation Forest
IMM	Interacting Multiple Model
IQR	Inter Quartile Range
KF	Kalman Filter
LODA	Lightweight On-line Detector of Anomalies
LReLU	Leaky Rectified Linear Unit
LSTM	Long Short Term Memory
LTI	Linear Time Invariant
MAE	Mean Absolute Error
OR Table	Operating Room Table
OR	Operating Room
REST	REpresentational State Transfer
RNN	Recurrent Neural Network
ROS	Robot Operating System
ReLU	Rectified Linear Unit
SDC	Service-oriented Device Connectivity
TPR	True Positive Rate
UKF	Unscented Kalman filter
URDF	Unified Robot Description Format

References

1. Christensen, K.; Doblhammer, G.; Rau, R.; Vaupel, J.W. Ageing populations: The challenges ahead. *Lancet* **2009**, *374*, 1196–1208. [CrossRef]
2. Prince, M.J.; Wu, F.; Guo, Y.; Gutierrez Robledo, L.M.; O'Donnell, M.; Sullivan, R.; Yusuf, S. The burden of disease in older people and implications for health policy and practice. *Lancet* **2015**, *385*, 549–562. [CrossRef] [PubMed]
3. Qiao, M.; Guo, J.; Wang, R.; Chen, C.; Li, J.; Lyu, J. Research progress on population aging and chronic diseases. *Meds Public Health Prev. Med.* **2023**, *3*, 28–35. [CrossRef]
4. Sangiovanni-Vincentelli, A.; Damm, W.; Passerone, R. Taming Dr. Frankenstein: Contract-Based Design for Cyber-Physical Systems. *Eur. J. Control.* **2012**, *18*, 217–238. [CrossRef]

5. Pfeiffer, J.H.; Dingler, M.E.; Dietz, C.; Lueth, T.C. Requirements and architecture design for open real-time communication in the operating room. In Proceedings of the 2015 IEEE International Conference on Robotics and Biomimetics (ROBIO), Zhuhai, China, 6–9 December 2015; IEEE: New, York, NY, USA, 2015; pp. 458–463. [CrossRef]
6. Puder, A.; Henle, J.; Sax, E. Threat Assessment and Risk Analysis (TARA) for Interoperable Medical Devices in the Operating Room Inspired by the Automotive Industry. *Healthcare* **2023**, *11*, 872. [CrossRef] [PubMed]
7. Puder, A.; Henle, J.; Rumez, M.; Vetter, A. A Mixed E/E-Architecture for Interconnected Operating Tables Inspired by the Automotive Industry. In Proceedings of the International Symposium on Medical Robotics, Atlanta, GA, USA, 13–15 April 2022; IEEE: New, York, NY, USA, 2022.
8. Getinge. Maquet Meera. Available online: https://www.getinge.com/de/produkte/maquet-meera/?tab=1 (accessed on 1 October 2023).
9. Rbah, Y.; Mahfoudi, M.; Balboul, Y.; Fattah, M.; Mazer, S.; Elbekkali, M.; Bernoussi, B. Machine Learning and Deep Learning Methods for Intrusion Detection Systems in IoMT: A survey. In Proceedings of the 2022 second International Conference on Innovative Research in Applied Science, Engineering and Technology (IRASET), Meknes, Morocco, 3–4 March 2022; IEEE: New, York, NY, USA, 2022; pp. 1–9. [CrossRef]
10. Teber, D.; Engels, C.; Maier-Hein, L.; Ayala, L.; Onogur, S.; Seitel, A.; März, K. Wie weit ist Chirugie 4.0? *Urologe Ausg. A* **2020**, *59*, 1035–1043. [CrossRef] [PubMed]
11. Jin, X.B.; Robert Jeremiah, R.J.; Su, T.L.; Bai, Y.T.; Kong, J.L. The New Trend of State Estimation: From Model-Driven to Hybrid-Driven Methods. *Sensors* **2021**, *21*, 2085. [CrossRef] [PubMed]
12. Zhu, J.; Cherubini, A.; Dune, C.; Navarro-Alarcon, D.; Alambeigi, F.; Berenson, D.; Ficuciello, F.; Harada, K.; Kober, J.; LI, X.; et al. Challenges and Outlook in Robotic Manipulation of Deformable Objects. *IEEE Robot. Autom. Mag.* **2022**, *29*, 2–12. [CrossRef]
13. Gatouillat, A.; Badr, Y.; Massot, B.; Sejdic, E. Internet of Medical Things: A Review of Recent Contributions Dealing With Cyber-Physical Systems in Medicine. *IEEE Internet Things J.* **2018**, *5*, 3810–3822. [CrossRef]
14. Del Alcazar von Buchwald, R.; Schäfer, A.; Golde, T.; Gaiser, I.; Olszewski, J.D.; Obert, M. Operationstisch mit Lastsensoranordnung. DE 102021107833 A1, 29 September 2022.
15. Puder, A.; Del Alcazar von Buchwald, R. Sicherheitssystem zur Detektion von Fehlern in Medizinischen Tischen (Safety System for Detecting Errors in Medical Tables). German Patent/WIPO Patent DE102022110888A1/WO2023213868A1, 3 May 2022.
16. Puente León, F.; Bauer, S. *Praxis der Digitalen Signalverarbeitung*, 2nd ed.; KIT Scientific Publishing: Karlsruhe, Germany, 2017. [CrossRef]
17. León, F.P.; Jäkel, H. *Signale und Systeme*; De Gruyter Oldenbourg: Berlin, Germany, 2019. [CrossRef]
18. Hawkins, D. *Identification of Outliers*; Springer: Berlin/Heidelberg, Germany, 1980. [CrossRef]
19. Chandola, V.; Banerjee, A.; Kumar, V. Anomaly detection: A survey. *Acm Comput. Surv. (CSUR)* **2009**, *41*, 15. [CrossRef]
20. Awasthi, S.; Travieso-González, C.M.; Sanyal, G.; Kumar Singh, D. (Eds.) *Artificial Intelligence for a Sustainable Industry 4.0*, 1st ed.; Springer eBook Collection, Springer International Publishing and Imprint Springer: Cham, Switzerland, 2021. [CrossRef]
21. Kalman, R.E. A New Approach to Linear Filtering and Prediction Problems. *J. Basic Eng.* **1960**, *82*, 35–45. [CrossRef]
22. Welch, G.; Bishop, G. An Introduction to the Kalman Filter. *Proc. Siggraph Course* **2006**, *8*, 41.
23. Wendel, J. *Integrierte Navigationssysteme: Sensordatenfusion, GPS und Inertiale Navigation*, 2nd ed.; De Gruyter: München, Germany, 2011.
24. van der Merwe, R. Sigma-Point Kalman Filters for Probabilistic Inference in Dynamic State-Space Models. Ph.D. Thesis, OGI School of Science & Engineering at Oregon Health & Science University, Portland, OG, USA, 2004.
25. Labbe, R. *Kalman and Bayesian Filters in Python*; 2020. Github. Available online: https://github.com/rlabbe/Kalman-and-Bayesian-Filters-in-Python (accessed on 1 October 2023).
26. Killourhy, K.S.; Maxion, R.A. Comparing anomaly-detection algorithms for keystroke dynamics. In Proceedings of the 2009 IEEE/IFIP International Conference on Dependable Systems and Networks, Lisbon, Portugal, 29 June–2 July 2009; pp. 125–134. [CrossRef]
27. Dash, C.S.K.; Behera, A.K.; Dehuri, S.; Ghosh, A. An outliers detection and elimination framework in classification task of data mining. *Decis. Anal. J.* **2023**, *6*, 100164. [CrossRef]
28. Liu, F.T.; Ting, K.M.; Zhou, Z.H. Isolation Forest. In Proceedings of the 2008 Eighth IEEE International Conference on Data Mining, Pisa, Italy, 15–19 December 2008; pp. 413–422. [CrossRef]
29. Hornik, K.; Stinchcombe, M.; White, H. Multilayer feedforward networks are universal approximators. *Neural Netw.* **1989**, *2*, 359–366. [CrossRef]
30. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016.
31. Hawkins, S. Outlier Detection Using Replicator Neural Networks. In Proceedings of the Data Warehousing and Knowledge Discovery, Aix-en-Provence, France, 4–6 September 2002.
32. Sakurada, M.; Yairi, T. Anomaly Detection Using Autoencoders with Nonlinear Dimensionality Reduction. In Proceedings of the MLSDA'14, Gold Coast Australia, QLD, Australia, 2 December 2014.
33. Rausch, A.; Sedeh, A.M.; Zhang, M. Autoencoder-Based Semantic Novelty Detection: Towards Dependable AI-Based Systems. *Appl. Sci.* **2021**, *11*, 9881. [CrossRef]
34. Wei, Y.; Jang-Jaccard, J.; Xu, W.; Sabrina, F.; Camtepe, S.; Boulic, M. LSTM-Autoencoder-Based Anomaly Detection for Indoor Air Quality Time-Series Data. *IEEE Sensors J.* **2023**, *23*, 3787–3800. [CrossRef]
35. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]

36. Vennerød, C.B.; Kjærran, A.; Bugge, E.S. Long Short-term Memory RNN. *arXiv* **2021**, arXiv:2105.06756.
37. Chen, T.; Liu, X.; Xia, B.; Wang, W.; Lai, Y. Unsupervised Anomaly Detection of Industrial Robots Using Sliding-Window Convolutional Variational Autoencoder. *IEEE Access* **2020**, *8*, 47072–47081. [CrossRef]
38. Hill, D.J.; Minsker, B.S.; Amir, E. Real-Time Bayesian Anomaly Detection for Environmental Sensor Data. In Proceedings of the Congress-International Association for Hydraulic Research, Venice, Italy, 1–6 July 2007.
39. Hill, D.J.; Minsker, B.S.; Amir, E. Real-time Bayesian anomaly detection in streaming environmental data. *Water Resour. Res.* **2009**, *45*. [CrossRef]
40. Jaber, A.A.; Bicker, R. Industrial Robot Fault Detection Based on Statistical Control Chart. *Am. J. Eng. Appl. Sci.* **2016**, *9*, 251–263. [CrossRef]
41. Sathish, V.; Ramaswamy, S.; Butail, S. Training data selection criteria for detecting failures in industrial robots. *IFAC-PapersOnLine* **2016**, *49*, 385–390. [CrossRef]
42. Cheng, F.; Raghavan, A.; Jung, D.; Sasaki, Y.; Tajika, Y. High-Accuracy Unsupervised Fault Detection of Industrial Robots Using Current Signal Analysis. In Proceedings of the 2019 IEEE International Conference on Prognostics and Health Management (ICPHM), San Francisco, CA, USA, 17–20 June 2019; pp. 1–8. [CrossRef]
43. Vallachira, S.; Orkisz, M.; Norrlöf, M.; Butail, S. Data-Driven Gearbox Failure Detection in Industrial Robots. *IEEE Trans. Ind. Informatics* **2020**, *16*, 193–201. [CrossRef]
44. Weber, M.; Klug, S.; Sax, E. Embedded Hybrid Anomaly Detection for Automotive CAN Communication. In *9th European Congress on Embedded Real Time Software and Systems (ERTS 2018)*; HAL: Toulouse, France, 2018.
45. Weber, M. Untersuchungen zur Anomalieerkennung in Automotive Steuergeräten durch Verteilte Observer mit Fokus auf die Plausibilisierung von Kommunikationssignalen. Ph.D. Thesis, Karlsruhe Institute for Technology, Karlsruhe, Germany, 2019.
46. Grimm, D.; Weber, M.; Sax, E. An Extended Hybrid Anomaly Detection System for Automotive Electronic Control Units Communicating via Ethernet—Efficient and Effective Analysis using a Specification- and Machine Learning-based Approach. In Proceedings of the 4th International Conference on Vehicle Technology and Intelligent Transport Systems. SCITEPRESS—Science and Technology Publications, Funchal, Portugal, 16–18 March 2018; pp. 462–473. [CrossRef]
47. Huang, X.; Zhang, F.; Wang, R.; Lin, X.; Liu, H.; Fan, H. KalmanAE: Deep Embedding Optimized Kalman Filter for Time Series Anomaly Detection. *IEEE Trans. Instrum. Meas.* **2023**, *2*, 1–11. [CrossRef]
48. Sanchez, J.; Corrales, J.A.; Bouzgarrou, B.C.; Mezouar, Y. Robotic manipulation and sensing of deformable objects in domestic and industrial applications: A survey. *Int. J. Robot. Res.* **2018**, *37*, 688–716 [CrossRef]
49. Liu, J.; Wang, Z.; Xu, M. DeepMTT: A deep learning maneuvering target-tracking algorithm based on bidirectional LSTM network. *Inf. Fusion* **2020**, *53*, 289–304. [CrossRef]
50. Kilgore, J.L. Anthropometric variance in humans: Assessing Renaissance concepts in modern applications. *Anthropol. Notebooks* **2012**, *18*, 13–23.
51. Toshniwal, A.; Mahesh, K.; Jayashree, R. Overview of Anomaly Detection techniques in Machine Learning In Proceedings of the 2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Palladam, India, 7–9 October 2020; Volume 4; pp. 808–815.
52. Kiranyaz, S.; Avci, O.; Abdeljaber, O.; Ince, T.; Gabbouj, M.; Inman, D.J. 1D convolutional neural networks and applications: A survey. *Mech. Syst. Signal Process.* **2021**, *151*, 107398. [CrossRef]
53. IEC. *Medical Electrical Equipment: Part 1: General Requirements for Basic Safety and Essential Performance*; ADInstruments Pty Ltd.: Singapore, 2020.
54. Puder, A.; Rumez, M.; Grimm, D.; Sax, E. Generic Patterns for Intrusion Detection Systems in Service-Oriented Automotive and Medical Architectures. *J. Cybersecur. Priv.* **2022**, *2*, 731–749. [CrossRef]
55. Clinical Alarms Task Force. Impact of Clinical Alarms on Patient Safety: A Report From the American College of Clinical Engineering Healthcare Technology Foundation. *J. Clin. Eng.* **2007**, *32*, 22–33.
56. Sendelbach, S.; Funk, M. Alarm Fatigue. *AACN Adv. Crit. Care* **2013**, *24*, 378–386. [CrossRef] [PubMed]
57. Lee, I.; Sokolsky, O.; Chen, S.; Hatcliff, J.; Jee, E.; Kim, B.; King, A.; Mullen-Fortino, M.; Park, S.; Roederer, A.; et al. Challenges and Research Directions in Medical Cyber–Physical Systems. *Proc. IEEE* **2012**, *100*, 75–90. [CrossRef]
58. Cournapeau, D.; Grisel, O.; Varoquaux, G.; Gramfort, A.; Mueller, A. Scikit-Learn: Machine Learning in Python. Available online: https://scikit-learn.org/stable/index.html (accessed on 8 December 2023).
59. TensorFlow. TensorFlow. Available online: https://www.tensorflow.org/ (accessed on 9 November 2023).
60. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In Proceedings of the Advances in Neural Information Processing Systems 32, Vancouver, BC, Canada, 8–14 December 2019; Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R., Eds.; Curran Associates, Inc.: New York, NY, USA, 2019; pp. 8024–8035.