Interaction Analysis in Smart Work Environments through Fuzzy Temporal Logic

zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

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

genehmigte

Dissertation

von

Joris IJsselmuiden

aus Groningen, Niederlande

Tag der mündlichen Prüfung:	17. Juli 2014
Erster Gutachter:	Rainer Stiefelhagen
Zweiter Gutachter:	Jürgen Beyerer

Zusammenfassung

Semantische Beschreibungen von Personengruppen und Interaktionen können mittels logischer Schlussfolgerung aus multimodaler maschineller Wahrnehmung abgeleitet werden. Eine solche automatische Interaktionsanalyse hat viele potentielle Anwendungen in den Bereichen Robotik, intelligente Wohn- und Arbeitsumgebungen, adaptive Benutzerschnittstellen, Videoüberwachung, Entscheidungsunterstützung und Multimediasuche. Eine mögliche Anwendung ist die automatische Protokollierung von Feuerwehrstabsübungen. Mit den heutigen Mitteln ist man nicht in der Lage den Übungsteilnehmern individuelle, aufgabenbezogene Rückmeldung zu bieten. Automatisch generierte Verhaltensprotokolle könnten hier Abhilfe schaffen, da sie für eine semi-automatische Leistungsbewertung der Teilnehmer herangezogen werden können. So können nach der Übung Fragen beantwortet werden wie: "Wer hätte an welcher Gruppenbesprechung teilnehmen sollen?", "Wie lange haben die Teilnehmer gebraucht um einzelnen Aufgaben zu erledigen?", "Welche Flaschenhälse traten auf?", "Welche Ressourcen blieben unbenutzt?", und "Inwiefern wurden Standardarbeitsanweisungen beachtet?" Wenn durch maschinelle Wahrnehmung Identität, Position, Orientierung und Sprachaktivität der Übungsteilnehmer, sowie Information über die Objekte im Raum, bekannt ist, kann das vorgestellte System automatisch die dazu gehörige Gruppenbeschreibungen und -visualisierungen erzeugen; wer macht was mit wem, mit Hilfe von welchen Unterstützungs-Werkzeugen? Ein typisches Beispiel einer solchen Beschreibung ist: "Die Sachgebietsleiter S1 und S2 bearbeiten gemeinsam die Lagekarte."

Um Abhängigkeiten von maschineller Wahrnehmung zu vermeiden, wird sie simuliert mit Hilfe eines speziell dafür entwickelten Softwarewerkzeugs. Eine Feuerwehr-Stabsübung wurde audiovisuell aufgezeichnet und die entsprechenden Personen-Tracks und weitere Attribute der Personen und Objekten im Raum wurden annotiert. Das entwickelte System, das diese Daten als Eingaben benutzt, basiert auf unscharfer metrisch temporaler Logik (FMTL) und Situationsgraphenbäumen (SGTs). Diese Methode zeichnet sich aus durch ihre Ausdrucksfähigkeit und Introspektion. Domänenwissen kann verhältnismäßig leicht formalisiert werden in logischen Formeln und Baumstrukturen, da diese nah an der menschlichen Intuition liegen. Die Methode ermöglicht die Erzeugung der komplexen Modelle, die für diese Fallstudie gebraucht werden. Es wurde ein Satz an neuen FMTL-Regeln und SGTs entwickelt und die Methode wurde erweitert um ein Clustering-Verfahren und ein Parameter-Lernverfahren. Das Clustering vereinfacht die Schlussfolgerung und das Lernverfahren steigert die Leistung des Systems. Die generierten Gruppenbeschreibungen wurden quantitativ gegen eine Grundwahrheit evaluiert, die mittels eines speziell dafür entwickelten Softwarewerkzeugs erzeugt wurde. Auch wurden unter anderem die Laufzeit, auftretende Fehler und das Verhalten bei verrauschten Daten analysiert. Die Ergebnisse sind vielversprechend, aber es wurden auch einige Möglichkeiten für zukünftige Arbeiten definiert.

Acknowledgements

This work is supported by the Fraunhofer-Gesellschaft Internal Programs under Grant 692 026. Thanks to the people that were involved in making this funding possible. I express my gratitude to the students that were involved in this study. Johannes Dornheim for his dedication and valuable contributions. Ioannis Papantonis for his supportive role in the maturation of the developed system. Timo Körner, Daniel Reich, and Sebastian Kaltwang for pleasant cooperation on other topics. I thank Rainer Stiefelhagen for his supervision, insight, fruitful discussions, and friendly attitude. Thanks to David Münch and Ann-Kristin Grosselfinger for their contributions and fertile cooperation. I thank the State Fire Service Institute North Rhine-Westphalia for providing the opportunity to record a staff exercise and for their valuable domain knowledge and feedback. Cheers to Alexander Schick, Florian van de Camp, Michael Voit, Abteilung IAD, and CV:HCI for a great day at the office and for teaching me in the art of German Engineering. Eva Jorritsma, thank you for your great patience and support.

Contents

List of Figures

List of Tables

Notation

These symbols and conventions are mostly applied in Chapter 4.

Symbol	Meaning
$\wedge$	Fuzzy conjunction: semantics explained in Section 2.7.1
$\vee$	Fuzzy disjunction: semantics explained in Section 2.7.1
$\neg$	Fuzzy negation: semantics explained in Section 2.7.1
$\leftarrow$	Fuzzy implication
$\Diamond_{[t_1,t_2]}$	Interval diamond operator: at least once between times t_1 and t_2
$\Box_{[t_1,t_2]}$	Interval box operator: always between times t_1 and t_2
$\Diamond_{dt}$	Relative diamond operator: at time $t = t_0 + dt$
$\Box^{a\%}_{dt_1,dt_2}$	Fractional box operator: at least $a\%$ of time interval $[t_0 + dt_1, t_0 + dt_2]$
!	Cut operator: prevents a rule from querying its remaining conditions
$=..$	Decomposition operator: splits a predicate into a list containing its predicate name and arguments
$V(P)$	Truth value operator: returns the truth value of predicate P

Conventions:

- list variables are boldfaced,
- nested predicate variables are capitalized,
- constant arguments are non-italic to distinguish them from variables allowing one to omit quantifiers (all variables in a logic formula are implicitly universally quantified with scope over the entire formula, for example: $\neg Human(x) \vee Mortal(x)$ stands for: $\forall x(\neg Human(x) \vee Mortal(x))$ which is logically equivalent to: $\forall x(Human(x) \rightarrow Mortal(x))$),
- the situation graph trees (SGTs) in Section 4.2 and the source code snippet in Section 4.1.4 applies an inverted capitalization convention for practical reasons.

Chapter 1

Introduction

Automatic interaction analysis can be achieved through a combination of machine perception and reasoning. This thesis presents a system for logic-based reasoning, applied to interaction analysis in smart work environments. It is based on fuzzy metric temporal logic (FMTL) and situation graph trees (SGTs). The motivation for this work is presented in Section 1.1. Then, the approach is introduced in Section 1.2. The contribution of the thesis is explained in Section 1.3, and Section 1.4 gives the outline of the remaining chapters.

1.1 Motivation

The motivation of this study is to come closer to domain independent reasoning methods for automatic interaction analysis and related problems. Interaction analysis is defined as the generation of semantic descriptions about groups and interactions within them, from multimodal machine perception observing multiple persons and objects. Interaction analysis and the encompassing research field of high-level reasoning still contain many challenges. Although computer vision and other areas of machine perception (e.g. person tracking and human pose estimation) have enjoyed great progress in recent years, corresponding high-level reasoning methods that use such machine perception have not progressed at the same pace. This is in part because machine perception is itself an active and challenging field of research. High-level reasoning often requires input that the state-of-the-art in machine perception cannot provide, or the effort involved in developing such machine perception is too large, causing researchers to focus on other problems. But as machine perception progresses and as the system tasks and perceived world become more complex, e.g. in robotics and smart environments, the need for powerful and flexible reasoning methods that bridge the semantic gap rises.

1.1.1 Suitable Application Domains

High-level reasoning can be used in many application domains. Research in this direction is relevant if computers need to understand the relations between the objects and people in the world in order to describe them or react appropriately. Because the reasoning system presented in this thesis is only connected to underlying machine perception through a generic symbolic interface, any combination of sensors and machine perception components can be used to provide it with input data. The case study that is presented below is based on cameras and microphones, but other types of sensors such as satellite navigation, radar, laser scanners, RFID tags, and pressure sensors can be used just as well. In addition, activities and information in cyberspace can be monitored to obtain further input data.

As the presented reasoning methods can use any combination of machine perception components (human pose estimation, speech recognition, vessel tracking, etc.), the range of possible application domains is wide: multimedia retrieval, robotics, smart homes, ambient assisted living, smart work environments, intelligent user interfaces, smart cities, the internet of things, indoor and outdoor surveillance, air traffic control, decision support for military and civil security, and more. In a complex system of systems, the applied reasoning methods can also facilitate camera control, sensor deployment planning, prediction, information exchange between system components, and top-down knowledge for machine perception components to guide their search and improve their outputs.

The obvious application for video surveillance, reconnaissance, and air traffic control is to guide an operator's focus of attention by emphasizing anomalous situations in a multitude of camera streams or geographic visualizations, reducing cognitive load and increasing detection rates. The presented methods can also be used in sports analysis to automatically classify game situations from tracking data for example. This could allow semantic multimedia retrieval for improved training procedures and for semantically guided media consumption. On a related note, multimedia retrieval for entertainment purposes could benefit from such semantic-gap-bridging technologies as well. Other applications are conceivable in the area of partially automated biological, psychological, and sociological research. Machine perception and subsequent high-level reasoning about the observed behavior of humans and animals could improve the efficiency and reliability of experiments in behavioral science. Such interaction analysis could be used for semi-automatic analysis of group behavior in animals, for cognitive modeling of driving behavior, and for modeling and predicting the dynamics within an urban society.

1.1.2 The Smart Control Room

Although this study focuses on the case study described below, it is part of a larger body of research represented by the Smart Control Room laboratory and related installations at the Fraunhofer IOSB in Karlsruhe, Germany. The Smart Control Room (Figure 1.1) is a research platform for human machine interaction. It uses multiple interaction modalities, most notably (beyond) state-of-the-art computer vision observing the people in the room. Multiple users can interact with multiple large and small screens through person tracking, human pose estimation, gesture recognition, head pose estimation, speech recognition,

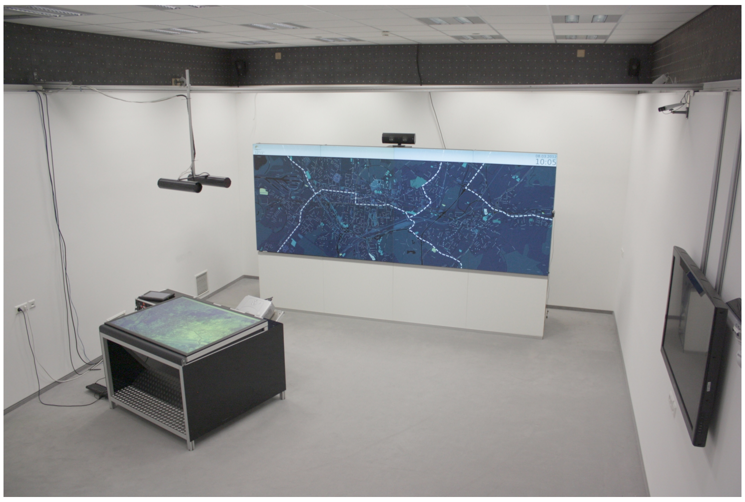


Figure 1.1: The Smart Control Room laboratory at the Fraunhofer IOSB in Karlsruhe, Germany: a research platform for multimodal, multi-user, multi-display interaction through computer vision and related technologies.

and touch control. All interaction modalities and interaction devices are interconnected, allowing seamless and intuitive interaction across multiple screens. For more details about this research platform, see [12, 125].

Using the reasoning methods for interaction analysis presented in this thesis, the Smart Control Room could reason about user behavior on a semantic level, using a world model that contains all available perception modalities. This would allow the automatic generation of reports and visualizations of the situation in the room and enable the room to react more intelligently. For example, based on the observed locations, orientations, speech patterns, and screen contents, two individual workspaces could merge into a collaborative one if the room detects that two users are trying to work together on the same problem. Furthermore, the Smart Control Room could present its users with interactive tools that are appropriate for their current roles and activities. The main demonstrator in the Smart Control Room is a hypothetical crisis response control room of the future, which motivated us to perform the case study described below.

The Smart Control Room laboratory arose from a research project that focused on computer vision for human machine interaction and its application to civil security in the form of smart control rooms. Because of their leading role in German and international crisis response, the fire brigade was chosen as potential end-user group of such a smart control room. However, current fire brigade control rooms are not equiped with the technical means to facilitate this change and their staff is reluctant to change their standard operating procedures to new technology. In other words, these end-users are not ready to adopt the solutions provided by the Smart Control Room laboratory. Novel human machine interaction will likely be adopted by the fire brigade in the near future, but for now, an application is needed that does not influence their current way of working. Based

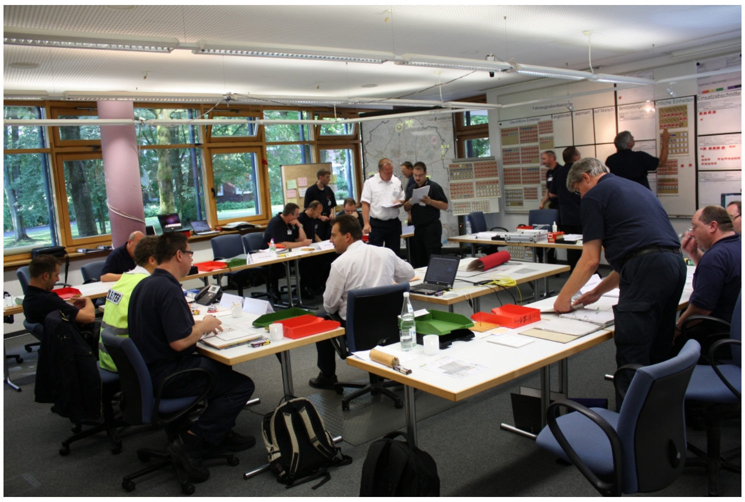


Figure 1.2: Staff exercise at the State Fire Service Institute North Rhine-Westphalia. The presented case study's goal is to model and recognize the interactions between the staff members and the objects in the room. In this image, these are: "conversation" (left/center), "analyzing a document together" (top center), and "editing a display" (top right).

on feedback from potential end-users, the project hence arrived at the case study that is treated in this thesis: automatic report generation for training purposes in crisis response control rooms.

1.1.3 Case Study: Report Generation in Crisis Response Control Rooms

The presented case study is situated at the State Fire Service Institute North Rhine-Westphalia, at one of their staff exercises for crisis response control room operations (Figure 1.2). During such role playing exercises, trainees take on the roles of control room staff while others simulate field units, crisis dynamics, distress calls, and radio communications. We are dealing with control rooms at the highest strategic level, which are only occupied during major catastropic events. Hence, storylines for such exercises are usually concerned with events such as large traffic accidents, widespread fires, or floodings. During the exercise that is the topic of this case study, the storyline was shaped around a collision between a passenger train and a cargo train carrying hazardous material.

In today's staff exercises, individual and task oriented feedback is hard to provide and often neglected. Instructors do not have the tools required to provide such feedback, at least not with enough quality and with manageable effort. Automatically generated behavior reports can improve this situation. They could be used to automatically or semi-automatically assess the performance of the individual participants: How close did they follow standard operating procedures? Who should have been part of which group? How long did it take them to complete specific tasks? Which bottlenecks have occurred? Which available resources were left idle?

When combined with visualizations, audiovisual recordings, and trails of developments in the crisis response software that is being used (field unit status, crisis dynamics, and other context information), such a system could provide a rich information source for feedback and learning, conveniently searchable for specific situations. Furthermore, exercises and subsequent evaluations could become more dynamic, e.g. by evaluating the effect of alternative decisions. Through these support functions, operational and administrative tasks would take less time, leaving more time for training at the strategic level. The presented case study is of scientific and practical interest because of its uniqueness, the large set of persons, objects, and attributes involved, and the described issues with current training solutions.

To enable such a system, the simulated crisis dynamics, field units, etc. need to be modeled, but also the situation within the control room, the latter being the focus of this case study. Hence, the goal is to automatically generate behavior reports during fire brigade control room exercises in order to increase their learning effect. To achieve this, the knowledge base presented in Chapter 4 aims to recognize group behavior during control room operations by modeling and recognizing the different types of person-person interaction and person-object interaction in various group formations. For this, the static and dynamic properties of the persons and objects in the room need to be perceived using machine perception. Second, these properties need to be analyzed by a reasoning engine to generate the behavior reports. If machine perception can perceive the identity, position, orientation, and speech activity of the staff members over time, as well as the state of the objects in the room, the reasoning engine can automatically generate corresponding descriptions and visualizations of group formations and interaction patterns, i.e. of who is doing what with whom, using which support tools. An example of such an automatically generated description is: "First officers S1 and S2 are editing the overview map together." To wrap up this section, Figures 1.3 and 1.4 provide some photos from the exercise that is the focus of the presented case study.

1.1.4 Thesis Goal

Following from this motivation, the goal of the presented study is: development of a reasoning system for interaction analysis and its evaluation in a case study: automatic report generation for training purposes in crisis response control rooms. The developed reasoning system should also be applicable to other application domains.

1.2 Approach

The architecture developed to achieve this goal is visualized in Figure 1.5 and explained below.

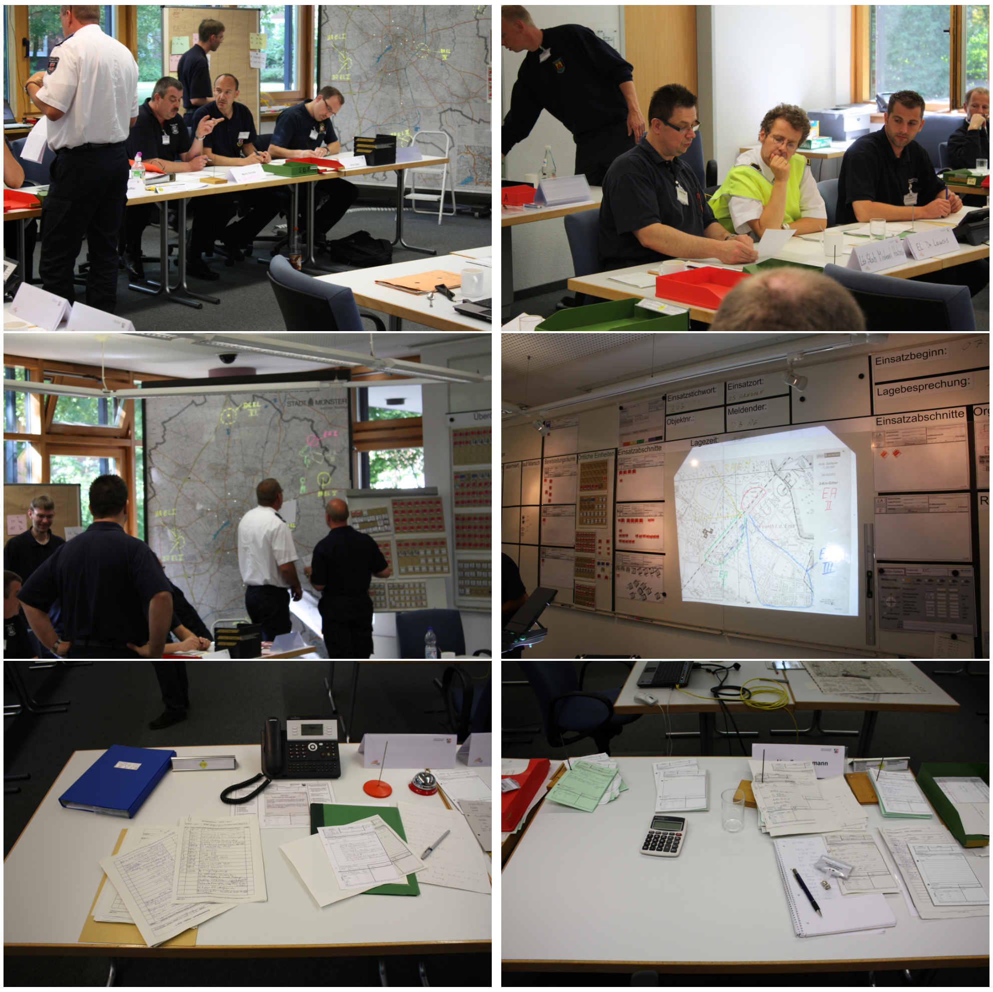


Figure 1.3: Top left: first officer S2 and his assistant engaged in a conversation. Top right: director of operations and his assistant discussing a document. Center left: two staff members occupied with the overview map. Center right: focus area map and unit status table. Bottom left: workspace of director of operation's assistant. Bottom right: another workspace, with the table for strategic planning in the background.

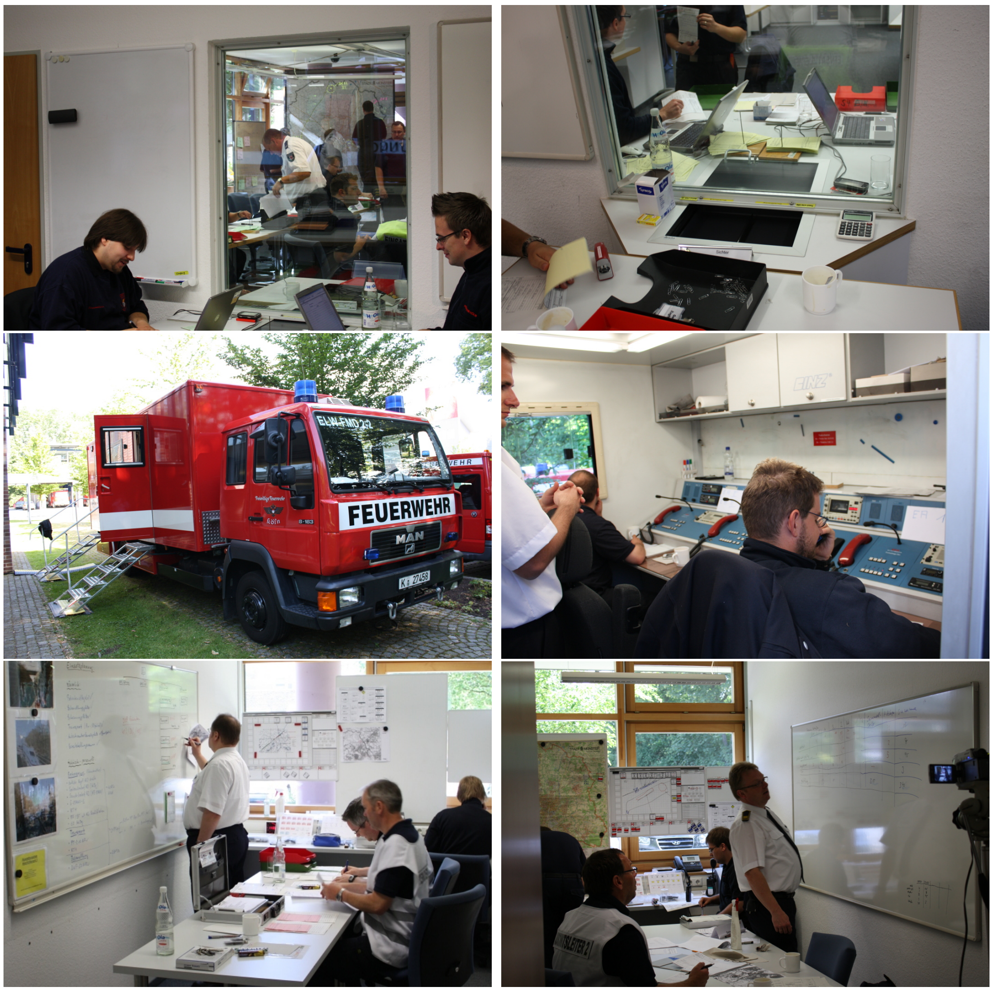
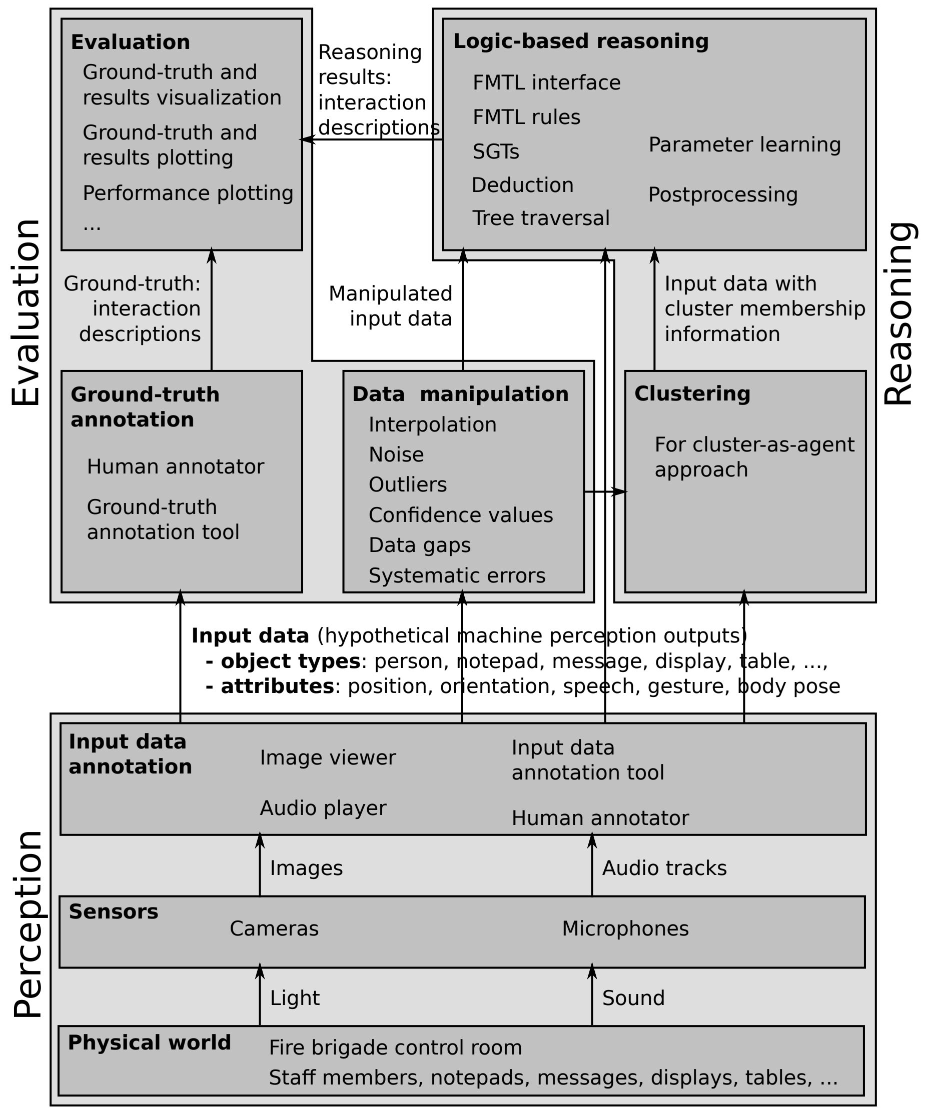


Figure 1.4: Top: both sides of the hatch connecting the control room (visible through the window on the left) to the outside world by paper messages. Center and bottom: simulation of crisis dynamics and field units just outside the control room, in a mobile command center (center, outside and inside) and two separate rooms (bottom), each responsible for an operational area, supposedly at different geographic locations.



Evaluation
Reasoning
Perception
Evaluation
Ground-truth and results visualization
Ground-truth and results plotting
Performance plotting
...
Reasoning results: interaction descriptions
Logic-based reasoning
FMTL interface
FMTL rules
SGTs
Deduction
Tree traversal
Parameter learning
Postprocessing
Ground-truth: interaction descriptions
Manipulated input data
Input data with cluster membership information
Ground-truth annotation
Human annotator
Ground-truth annotation tool
Data manipulation
Interpolation
Noise
Outliers
Confidence values
Data gaps
Systematic errors
Clustering
For cluster-as-agent approach
Input data (hypothetical machine perception outputs)
- object types: person, notepad, message, display, table, ...,
- attributes: position, orientation, speech, gesture, body pose
Input data annotation
Image viewer
Audio player
Input data annotation tool
Human annotator
Images
Audio tracks
Sensors
Cameras
Microphones
Light
Sound
Physical world
Fire brigade control room
Staff members, notepads, messages, displays, tables, ...

Figure 1.5: Schematic of the developed architecture.

1.2.1 Perception

In the future, machine perception should be used to perceive the control room for subsequent reasoning, but the required machine perception is not available. It would require tremendous effort to develop and for some of the required perception modalities it is questionable whether the current state-of-the-art can provide the required quality and accuracy. In order to avoid machine perception dependancy problems and to remain focused on high-level reasoning, this study replaces the required machine perception with annotated input data. Hypothetical machine perception outputs are annotated based on real audiovisual data, using a self-developed data annotation tool. This is a step along the way toward a real-time system containing various automatic machine perception components.

A fire brigade control room and the staff members and objects within it (notepads, messages, displays, tables, etc.) were recorded using five cameras and four microphones. Audio was used as-is while video from all five cameras was sampled at 1*fps* and merged into five-pane images that are convenient for the input data annotation process. The 1*fps* five-pane images and audio tracks are displayed in an image viewer and an audio player. A human annotator analyzes them in order to mimic them in symbolic form using mouse and keyboard and the developed input data annotation tool. This results in so-called hypothetical machine perception outputs, mimicing the machine perception that the reasoning engine requires. This input data consists of symbolic object descriptions: persons, notepads, messages, displays, and a few others. Their attributes describe their position, orientation, speech, gesture, and body pose.

1.2.2 Reasoning

The developed system uses fuzzy metric temporal logic (FMTL) and situation graph trees (SGTs) as the basis for reasoning. The FMTL/SGT approach was chosen, because it provides the expressive power and introspection that is needed for the performed case study. Domain knowledge can be formalized into logic formulas and tree structures relatively easily, because the logic paradigm is intuitive to use and close to human reasoning. The FMTL/SGT approach provides a rich language for reasoning with complex models that are understandable by humans. The annotated input data is read by an interface implemented in FMTL, yielding the atomic facts that form the bottom building blocks (i.e. the terminal leaves) of the reasoning process. The entire network of FMTL rules is grounded in these atomic facts. From a top-down perspective, the SGT traversal algorithm triggers FMTL queries, each starting a logic deduction process. During SGT traversal, reasoning results in the form of situation descriptions are generated. They describe group interactions that occur often during staff exercises. Optimal trapezoid truth function parameters can be learned through maximization of an adapted F-score performance measure. Some of the conducted experiments use a simple form of postprocessing to filter out redundant or otherwise unwanted results.

An adapted clustering algorithm is used in some experiments, as preprocessing before reasoning, to enrich the person descriptions in the annotated input data with cluster membership information. The persons in the room are clustered according to their positions in the room, simplifying the subsequent reasoning process. With the necessary adaptations to the FMTL rules and SGTs, this allows for the so-called cluster-as-agent approach during reasoning. Here, each traversal is performed with a cluster of persons as the center of reasoning (agent) instead of a single person as is the case in the so-called person-as-agent approach. This leads to a more global approach which is more intuitive. Furthermore, it leads to transitive group membership relations, yielding more consistent results for oblong groups and less redundant results in general. Finally, clustering as preprocessing can avoid some redundant traversals and it can replace part of the combinatorial search that is required with the person-as-agent approach, potentially leading to better runtimes.

1.2.3 Evaluation

To evaluate the system, reasoning results need to be compared to annotated ground-truth. A human annotator uses his mouse and keyboard and the ground-truth annotation tool that was developed for this purpose to enrich the annotated input data with situation descriptions that the system should deduce. The main idea is to quantitatively compare the reasoning results to the annotated ground-truth. The system includes tools to visualize them (side by side) and to plot them together on a time axis. The main performance measurement tool plots precision, recall, and F-score values over different truth value thresholds. The evaluation also contains runtime analysis, error analysis, experiments on noisy data, measurements of inter-annotator agreement, measurements of the effect of a new clustering parameter, and measurements of the effect of using parameter learning to optimize FMTL rules. To evaluate the system's robustness, a tool was developed that can add different amounts of noise to any of the annotated object attributes.[1] [2]

1.3 Contribution

In summary, the contribution of this thesis comes from its unique application domain and from the reasoning models that were developed for it, supported by the development of an enabling software toolkit and the conducted evaluation. Fuzzy metric temporal logic (FMTL) and situation graph trees (SGTs) are applied to interaction analysis for the first time. In the past, similar methods have only been applied in traffic and surveillance settings. The contribution of this thesis mainly consists of the development and evaluation

[1]In some cases, the 1*fps* sampling rate of the annotated input data is not sufficient, so the data manipulation tool can be used to interpolate the data with arbitrary rates. But this feature is not used in the evaluation.

[2]Five more manipulations are described in this thesis: adding outliers, confidence values, data gaps, superfluous atomic facts, and systematic errors that are likely to occur when working with real machine perception. These types of imperfections are not used in the evaluation, but they are explained in the thesis and suggestions are made on how they can be handled by the presented system.

of new reasoning models within a novel context rather than newly developed reasoning methods. These new reasoning models recognize groups and interaction patterns rather than movement patterns of one or two vehicles or persons as is the case in previous work. We now discuss the contribution of this thesis in detail.

The case study along which the presented system was developed, provides scientific novelty. It is concerned with the automatic generation of reports during fire brigade control room exercises to increase their learning effect. In previous work, increased situational awareness is achieved by modeling the site of the (simulated) crisis using geographical information systems (GIS) and other software tools, but the situation inside the control room has never been modeled and used. The case study is both unique and challenging, involving many persons and objects and the interaction between them, each with a variety of attributes. *This contribution is found in Section 1.1.3, Chapter 3, and Chapter 4.*

Because of this novel application domain and case study, new reasoning models had to be developed (FMTL rules, SGTs, and supporting components). These are able to recognize multiple group constellations and interaction patterns within them, as opposed to mainly movement patterns of one or two vehicles or persons. Never have such group and interaction models been achieved before. The developed FMTL rules, SGTs, and supporting components implement some powerful fuzzy spatiotemporal concepts that are of scientific and practical interest, applicable to other domains. *This contribution is found in Chapter 4.*

Additionaly, there are two novel components that solved specific problems that have occurred during development. First, the combination of FMTL/SGT reasoning with a customized clustering algorithm as preprocessing (with corresponding changes to FMTL rules and SGTs). Second, learning optimal trapezoid truth function parameters through maximization of an adapted F-score performance measure. *These contributions are found in Section 4.2.*

Several steps toward an integrated development toolkit were completed: new software tools for input data annotation and ground-truth annotation, as well as a new dataset with corresponding ground-truth. The system was evaluated on the presented dataset using precision, recall, and F-score measures. Furthermore, its runtime and the errors that occurred were analyzed. The system's robustness against noise was also evaluated, and the inter-annotator agreement between different ground-truth annotations has been analyzed. Finally, a newly introduced parameter for the clustering algorithm, as well as the application of parameter learning on trapezoid truth functions were evaluated. The toolkit, dataset, and evaluation procedure are of practical and scientific interest because of their unique character, but also because of their applicability to other research problems. *These supporting contributions are found in Chapters 3 and 5.*

1.4 Thesis Outline

That concludes Chapter 1, establishing the motivation, the approach, and the contribution of this thesis. Next, Chapter 2 provides the required background knowledge while discussing related work. Different methods that can be used for interaction analysis are presented as well as how they are applied in literature. Then, the strengths and weaknesses of description-based and statistical approaches are compared, and the FMTL and SGT methods are presented in terms of related work and basic functionality. Chapter 3 explains the case study as well as the dataset that was developed for it: the way they work in crisis response control rooms, how a control room exercise was recorded, and how the data was annotated. The core of the thesis is found in Chapter 4, describing the reasoning models that were developed for the case study, divided into the so-called primary models and secondary models. The reasoning models are evaluated in Chapter 5, first describing the applied evaluation methods and then evaluating the primary and secondary models in a variety of ways. Finally, Chapter 6 provides a conclusion consisting of a summary and a discussion on future work.[3]

[3]This thesis is based on the author's previous work represented by [6, 13]. His publications [3, 5, 7, 8] and supervised student theses [14, 15, 17] are also related, but they are not used in this thesis. Finally, [1, 2, 4, 9, 10, 11, 12, 16, 18, 19] represent the author's work on novel methods for human-machine interaction through computer vision and some other research topics.

Chapter 2

Background and Related Work

First, this chapter introduces the different types of approaches to high-level reasoning for interaction analysis and related problems (Sections 2.1 through 2.4). Then, some relevant studies on ambient intelligence, smart environments, and related application domains are presented (Section 2.5). Before homing in on the methods that are used in this thesis (Section 2.7), a discussion on the strenghts and weaknesses of statistical and description-based approaches is provided (Section 2.6). We start out by summarizing five relevant survey papers. In [20], Aggarwal and Ryoo explain that many computer vision applications in surveillance, patient monitoring, and human-computer interfaces require automated recognition of high-level activities, composed of multiple atomic actions that are performed by a single person. After reviewing recognition methodologies for atomic actions, recognition methodologies for high-level activities are presented and compared. They put special focus on the recognition of human-object interactions and group activities.

Single-layered approaches operate directly on sensor data, whereas hierarchical approaches divide the problem into multiple layers. Single-layered approaches are divided into space-time approaches such as template matching in 3D *xyt*-space and sequential approaches such as hidden Markov models (operating directly on sensor data). They are suitable for the recognition of gestures and actions with sequential characteristics, but they are not suitable for the problem described in this thesis. Hierarchical approaches enable the recognition of high-level activities based on the recognition results of atomic actions. For example, a high-level interaction like "fighting" may be recognized by detecting a sequence of several punching and kicking interactions, which are in turn composed of atomic actions such as "streching hand" and "withdrawing hand". This makes the recognition of high-level activities computationally tractable and understandable by humans, and the detectors for the atomic actions can be reused in the recognition of many different high-level activities. Usually, atomic actions are recognized by the single-layered approaches that are described in [20]. So the major advantage of hierarchical approaches is their ability to recognize high-level activities with more complex structures. They are especially suitable for a semantic-level analysis of interactions between humans and/or objects as well as complex group activities. This is because they can easily incorporate domain knowledge and because they need less training data than single-layered approaches would.[1] Domain knowledge is incorporated by specifying the composition of and the relationships within and between high-level activities. Single-layered approaches are not easily interpretable (they are more black-box-like), preventing a user from incorporating domain knowledge. Furthermore, single-layered approaches would generally require a large amount of training data to recognize high-level activities. For example, single-layered hidden Markov models would need to learn a large number of transition and observation probabilities, since the number of hidden states increases as the activities get more complex.

Lavee et al. [84] describe the problem as the translation of low-level content in video sequences into high-level semantic concepts, with applications such as surveillance, semantic video database indexing, and interactive systems. Turaga et al. [124] highlight applications such as content-based video annotation and retrieval, highlight extraction, and video summarization (in the context of security and surveillance, but also entertainment and personal archiving). They too distinguish between the recognition of atomic actions and high-level activities, the latter being more complex, involving coordinated actions among a small number of humans. Vishwakarma and Agrawal [128] provide another survey along these lines. Finally, Ye et al. [135] illuminate a different research area: situation identification techniques in pervasive computing. They define situation identification as an enabling technology that resolves noisy sensor data and abstracts it into higher-level concepts that are interesting to applications. Concretely, they focus on applications in smart environments and ambient assistant living (e.g. the recognition of activities of daily living). An interesting difference from the surveys discussed above, is that the work that is discussed here, uses many different sensor types beyond video cameras. The books [26, 58, 60] also provide valuable resources on interaction analysis and related topics. As proposed by [20], the available hierarchical approaches and their reasoning methods are divided into statistical, syntactic, description-based, and hybrid approaches below.

2.1 Statistical Approaches

Statistical approaches use probabilistic graphical models such as hidden Markov models or dynamic Bayesian networks to derive situation likelihoods. Berger wrote a standard work on statistical reasoning [32]. In [138], a two-layered hidden Markov model is used to classify group situations in meetings. The first layer uses audio and video features to model person activities and the second layer combines them to group situations. A smart meeting room is demonstrated in [41] that uses an event-driven multilevel dynamic Bayesian network to enable semantic understanding of human behavioral and social signals from sensor data in group interaction scenarios. Similarly, in [57], group situations are detected using dynamic probabilistic networks. Another statistical approach is pre-

[1]Note that the system presented in this thesis does not need training data at all. The use of training data is only shown as an optional extra feature in one of the evaluated models.

sented in [127], recognizing human activities such as cooking, showering and sleeping in a smart home. A different application domain is investigated in [81]: understanding of behavior and workflow in an industrial plant. They use a Bayesian filter supported by hidden Markov models. The user's feedback is incorporated, improving the system through online correction of erroneous classification results. In [119], dynamic Bayesian networks are used for intention recognition during human-machine interaction, allowing intelligent machines and robots to proactively adapt to a human's intentions. The paper discusses an example of a human commanding a mobile robot remotely. In [114], propagation networks are used to recognize the steps involved in a blood glucose monitor procedure from video data of people performing the procedure. Vision-based hand and object tracking and state information measured from the glucose monitor are used to recognize successful and failed execution, indicating which steps are missing in case of failure. This work is motivated by the desire to create assistive technology within a domestic environment. The described task is performed frequently by elderly people who develop late stage diabetes. Finally, Fischer and Beyerer apply dynamic Bayesian networks to two completely different application domains: maritime surveillance [51] and video surveillance of a parking area [50]. SMILE (Structural Modeling, Inference, and Learning Engine) is a C++ implementation for Bayesian networks and related methods, and GeNIe provides a corresponding graphical development environment.[2]

2.2 Syntactic Approaches

Syntactic approaches combine atomic events into complex situations using grammars, mapping spatiotemporal changes in image sequences to events for instance. Syntactic approaches model human activities as a string of symbols, where each symbol corresponds to an atomic action. Human activities are represented as a set of production rules generating a string of atomic actions. They are recognized by a parsing mechanism, similar to the way computer programs and natural languages are parsed. The production rules of (stochastic) context-free grammars naturally lead to a hierarchical representation and recognition of activities.

Gesture recognition and video surveillance applications are demonstrated in [72]. Their system can correctly interpret activities of multiple interacting persons and objects. The lower level detections are performed using probabilistic event detectors and the outputs are used in a stochastic context-free grammar. The corresponding parsing mechanism provides temporal constraints, disambiguation of uncertain low-level detections, and the inclusion of a priori domain knowledge about the structure of temporal events. In [78], activity grammars are learned from video data recorded by a camera that is mounted above the checkout of a convenience store. The system recognizes actions such as "customer removes money from tray" and "employee takes receipt from register", but also specific

[2]http://genie.sis.pitt.edu/

concatenations thereof. A language for human action is proposed in [22], with its own phonemes, morphemes (words), syntax, and machine learning methods. The proposed language is meant to facilitate human-centric interfaces, allowing humans to interact with robots and intelligent machines like they do with other humans. To achieve this, data of human activities from various sensors is analyzed and interpreted. The paper proposes applications in health care, artificial intelligence, and cognitive systems. Finally, [56] presents a new algorithm for plan recognition called ELEXIR (Engine for LEXicalized Intent Recognition) that uses so-called combinatory categorial grammars. It allows the system to postpone commitment to a specific solution, which also reduces runtime.

2.3 Description-Based Approaches

Description-based approaches incorporate domain knowledge in the form of logic formulas and related modeling techniques. This thesis follows a description-based approach. High-level human activities are represented in terms of simpler activities and ultimately in the atomic actions that compose them, describing their temporal, spatial, abstract, and logical relationships. Recognition of the activity is performed by searching for the sub-activities that satisfy the relations specified in its representation. After the development of interval temporal logic by Allen and Ferguson [21] in which a successful axiomatization of time periods was introduced, it became feasible to model situations and actions in terms of temporal logic. The hierarchical logic approach for example [115], combines interval temporal logic with event logic to analyze football games. In [34], the required rules are automatically generated from OWL-DL ontologies. These are combined with a temporal constraint net, consistency checking, and multi-hypothesis tracking. First results are given using examples of airport activities such as aircraft refueling. Limited first order logic is used in [129] to combine several subevents that satisfy predefined temporal and spatial constraints constituting a situation. This system was evaluated on a video dataset displaying a bank attack scenario and train station vandalism. A rule-based recognition system for hierarchically-organized activities is presented in [49]. This system returns only logically consistent scenarios by formulating conflicts as weighted partial MaxSAT clauses. Furthermore, robustness against noise is shown, and the desired level of detail can be adjusted by assigning preferences to clauses of the SAT problem. The system is evaluated in the context of ambient assisted living. Non-monotonic reasoning [23] can provide a means to retract deduced information as new information comes in.

Prolog is the most widespread logic programming language, used in application domains such as natural language processing, theorem proving, expert systems, games, automated answering systems, ontologies, and control systems. It was developed in the 1970s by Colmerauer and Roussel [39] as a declarative progamming language, expressed in terms of relations (facts and rules). Logical deductions are started by running a query over these relations. The Castor logic library [3] is a C++ library that allows the use of the logic programming paradigm from within object oriented programs. Probabilistic logic programs are logic programs in which some of the facts are annotated with probabilities. ProbLog[4] offers a Python implementation for probabilistic logic.

2.4 Hybrid approaches

The statistical and description-based approach can be combined in Markov Logic Networks (MLNs) [45, 107]. Visual event recognition for surveillance is performed in [123], using common sense domain knowledge to overcome noise and missing observations. The knowledge is represented in first-order logic with associated weights expressing confidence. These are used to construct MLNs. They test their system on parking lot surveillance videos containing complex interactions of people and vehicles. Markov Logic Networks are combined with the temporal semantics of event calculus in [116] to recognize the high-level events "meeting", "moving", "fighting", and "leaving an object" in a video surveillance setting. In [77], a video understanding system is developed for scenarios at bus stops, crosswalks, and intersections. Basketball scenes are classified in [92] by tracking players, their hands and feet, and the ball. They formalize the rules of the game as well as physical constraints into spatio-temporal descriptions based on Allen's interval logic (i.e. domain knowledge). Robustness to low-level observation uncertainty is provided by the corresponding MLNs. Moving away from video, satelite navigation data (GPS) is used in [109] to understand human interactions, attempted interactions, and intentions, as opposed to only successful actions of single individuals or statistical properties of groups of people. A GPS-based game of capture the flag is used to evaluate the approach, involving many distinct cooperative and competitive joint activities (such as players capturing enemies). Domain knowledge is formalized from the geometry of the game area, the motion model of the players, and by the rules and dynamics of the game. Again, constraints must not be hard, due to the use of MLNs. By using context information, the system can infer multi-agent activities under substantial amounts of noise or missing data. The reference C++ implementation for MLNs is called Alchemy.[5] ProbCog provides Python and Java implementations for MLNs and Bayesian logic networks.[6] Related literature is provided by [73, 74].

2.5 Smart Environments and Related Application Domains

These hierarchical methods are often applied in the community around ambient intelligence and smart environments, a research area where high-level reasoning and interaction anal-

[3]http://mpprogramming.com/cpp/
[4]http://dtai.cs.kuleuven.be/problog/
[5]http://alchemy.cs.washington.edu/
[6]http://ias.in.tum.de/software/probcog/

ysis are established relatively well. The book [96] provides an overview of this area up to 2010. In particular, we draw inspiration from studies such as [36, 61, 89, 97, 117, 120, 126]. In [130], Waibel and Stiefelhagen were mainly concerned with perceiving humans in smart environments, but the book also contains chapters on the corresponding high-level reasoning problems (p. 121–132, p. 315–340). The state-of-the-art in multimodal fusion for human-computer interaction up to 2010 is described in [121].

In [27], frequent relationships between actions are discovered from a collection of sensor data and the user is able to finetune the system. In [113], a framework is introduced for modeling intelligent environments. It is based on fuzzy transfer learning, allowing the transfer of learned models across different environments. This system predicts temperature development to allow for proactive control. A formal framework based on multi-valued temporal propositional logic is proposed in [87]. Similarly, bigraphs are introduced for the description, design, and analysis of intelligent environments in [68]. Context lattices are introduced in [134], allowing the inclusion of semantic information about the nature and relationships between sensor data and observed activities. The study presented in [105] uses a combination of first-order logic, fuzzy logic, and temporal logic exemplified in a military application.

Markov logic networks are applied to the recognition of activities of daily living in [37], without the use of cameras or wearable sensors. In [82], a training-free method is introduced that generates probabilistic inference systems from causal models for human behavior. A goal-directed human activity computing model that captures the semantic relations between different atomic activities is presented in [132]. From the field of robotics, [131] discusses how manipulation actions are structured in space and time using temporal anchor points where two objects (or hand and object) touch or un-touch each other. This results in a condensed representation by which different manipulations can be recognized and encoded. They also provide examples of robotic manipulation recognition and execution that are based on this representation.

2.5.1 Particular Inspirations for this Thesis

A temporal framework (see Section 2.7.1 – Temporal Modality) is used in [108] to define composite interactions between people in terms of atomic actions. Typical interactions they want to detect are: fighting, greeting, assault, and pursuit. They take advantage from the fact that perception is only concerned with learning and detecting the atomic actions. Complex compositions are provided by the layer on top. They also propose a probabilistic reasoning component that solves for missing and superfluous atomic actions. A classic statistical approach that inspired this thesis is presented in [35], where the perceptual outputs are fed to hidden Markov models after some preprocessing. Here, speech detection, ambient sound detection, tracking, and posture estimation are used to classify social activities in an ambient intelligence setting. In [62], instead of cameras and microphones, they use wearable motion and RFID sensors for recognizing activities of daily living. Nonetheless, they inspired this thesis, since it strives for a general purpose framework, independent of sensor setup and application domain. They use an emerging patterns approach and sliding time windows to classify sequential, interleaved, and concurrent activities.

The work presented in [63] is concerned with storyline extraction from sports videos using and-or graph representations. This approach overcomes some of the limitations of hidden Markov models and dynamic Bayesian networks, because not only the model parameters are learned, but the model structure too. In [133], and-or graphs are used to generate text descriptions for a large dataset containing many different types of videos and images. These last two studies also provide means to generate reports in natural language. In [139], they study semantic information in abstract images created from collections of clip art. It involves a thorough analysis of the semantic importance of visual features, made possible by the use of a large set of abstract data instead of video data. Finally, the author was inspired by applying the methods used in this thesis to video surveillance in [8]. The goal was to recognize situations such as "getting into a car" and "unloading a package from the trunk" in the VIRAT video surveillance dataset. Not only did this show the broad applicability of the applied methods, it also lead to a fruitful cross-fertilization between the two studies. Recent work on the VIRAT dataset is presented in [140].

2.5.2 Literature from Sociology and Crisis Response

In sociology, there is extensive research on group behavior. For example, [80] explains necessary and sufficient conditions for groups as well as the so-called personal space layers. The distance category "personal distance" in which most conversations take place is 45–120cm for western culture. In [75], the related terms "participation structure" and "spatial organization of activity" are introduced. Both [70] and [75] share some insight on scientific practices for interaction analysis. Much like this thesis, they use manual annotation of audiovisual recordings in order to analyze them. But unlike this thesis, they do not use these annotations to automatically deduce semantic situation descriptions, nor do they have the ultimate goal to automate perception.

There are some interesting studies on crisis response management. The Pandora system is presented in [28], a management and training system that integrates computational intelligence with the intelligence of the trainer and the trainees to provide an emotionally engaging augmented/virtual reality training environment for crisis response. In [110], another system for disaster response management and training is presented. The developed software supports the real-time coordination between staff members and organizations. Two related studies [86, 122] aim to improve improvisation skills and collaboration through serious gaming. In these examples, increased situational awareness is achieved by modeling the site of the (simulated) crisis, but the situation inside the control room has never been used. The book [71] provides an overview of the current and future design and ergonomics of control rooms. Finally, [54, 79, 118] provide insight into the current procedures and best practices that are applied in German crisis response and fire brigade management.

2.5.3 Maritime Surveillance

In [40], a system is described for situational awareness against piracy and other maritime threats. Reasoning and related aspects of this project are discussed in [102], and [103] focuses on how the computational overhead can be reduced by applying data structures from the field of moving object databases and methods for behavior estimation using computational movement analysis. The applied methods are description-based. Other studies from the maritime domain, applying statistical methods (probabilistic graphical models) include [44, 51, 83, 85]. The author's own investigations in this direction are represented by [1].

2.6 Discussion

In the second half of the twentieth century, great AI promises were made based on classic, symbolic expert knowledge systems, i.e. the description-based approach. Because these promises were not fulfilled, interest waned and most of the funding stopped. This is sometimes called the AI winter.[7] Later though, AI experienced a revival through machine learning and the statistical approach, with great success. Currently, we are experiencing a resurrection of the description-based approach as well as a tendency toward hybrid systems that combine the best of both worlds. Although off-the-shelf machine learning techniques and the statistical approach as such are very powerful and successful, they do not scale well to knowledge intensive problems that exhibit a certain type of complexity, as further explained below. The shift toward description-based approaches could also have something to do with the fact that our exponentially faster computers are now able to handle their poor worst case time complexity (which is discussed below). When comparing description-based approaches to statistical approaches, it becomes apparent that both have their strenghts and weaknesses. Some problems are better solved with one and some with the other. Furthermore, upcoming hybrid approaches show great promise for a general purpose framework. Because syntactic approaches share many strenghts and weaknesses with description-based approaches, they are not discussed separately here.

2.6.1 Strengths of Description-Based Approaches

The description-based models for representation and reasoning that are used in this thesis are based on formalized domain knowledge rather than learned from training data. Compared to other approaches, domain-knowledge-based representation and reasoning in FMTL and SGTs is intuitive and flexible. The clear boundary between machine perception

[7]The story is actually more complicated, involving many factors besides this one, two periods of reduced interest and funding (1974–1980, 1987–1993), and several smaller episodes. Source: Wikipedia – AI winter

and reasoning makes it easier to improve one without the other. Furthermore, deductions are understandable by humans and completely provable, and existing FMTL rules and SGTs can be adapted to new settings with relatively little effort. The ability for humans to understand the reasoning process is essential to the presented case study. FMTL/SGT expert systems are suitable for knowledge intensive problems with heterogeneous search spaces such as the one presented here. A main advantage of description-based is that modeling is intuitive, because logic can remain relatively close to human language. It is relatively easy and fast to create new models once there is a firm basis to build on. Another main advantage is that in general, logics have greater expressive power than probabilistic graphical models. Aggarwal and Ryoo [20] emphasize that description-based approaches are able to represent and recognize human activities with complex temporal structures, sequential ones as well as concurrent ones. In [31], a case is made for using a description-based approach as opposed to a statistical one for planning and control of robot motion.

2.6.2 Weaknesses of Description-Based Approaches

However, these advantages come at a cost. Model checking in FMTL and many other logics (evaluating whether a given formula holds in a given world model) has a poor worst case time complexity. In practice though, timely and even real-time operation can be achieved in many cases. This is confirmed by Table 5.1, showing the runtimes for the primary experiments in this thesis. The time complexity of model checking depends on the restrictions and extensions that are put on the language and on which algorithm is used to solve the problem. Literature shows that polynomial worst case time complexity can be reached in many cases (as a function of the size of the world model and the length of the formula that has to be evaluated). The review paper [112] describes the worst case and practical time complexities of model checking in three temporal logics with and without certain restrictions and extensions. In [42], the complexity and expressiveness of different forms of logic programming is analyzed: propositional logic programming and Datalog, but also more general logic programming with extensions. Finally, in [76], the complexity of Prolog and quantified Horn clauses in general is analysed. They propose an algorithm that can evaluate a formula in $\mathcal{O}(n^3)$ time. In any case, statistical approaches generally exhibit better runtime behavior than description-based approaches.

Another potential weakness is the lack of the concept "conditional probability". This is a key concept in most statistical approaches, but it is not available in FMTL. Other (probabilistic) logics do have it [52, 88, 98, 106], but in FMTL, one has to make do without. The lack of an operator like $A \rightarrow^{P} B$ (If A then, with probability P, also B) did not lead to any issues during this study, but it might in other cases. Ryoo and Aggarwal [20] state that one of the limitations of description-based approaches is that they are fragile when their input data is noisy or if some of it is missing. Some description-based approaches can handle this problem however [8, 63, 99, 108]. Furthermore, as will be discussed below, if fuzzy logic is deployed correctly, it can handle noise and missing data too.

2.6.3 Strengths of Statistical Approaches

Statistical approaches, as do syntactic ones, provide a probabilistic framework for reliable recognition with noisy and missing inputs. Another strong point of statistical approaches (usually probabilistic graphical models) is that they are well suited for machine learning. It is safe to say that this is one of the greatest success stories in computer science and artificial intelligence. Machine learning opens up pitfalls like overfitting and lack of training data to cover the search space, but above all, it offers great opportunities. Description-based approaches can incorporate machine learning as well, but literature shows that this is less common. Some statistical approaches, like dynamic Bayesian networks for example, can also work well without machine learning. Just like with FMTL/SGTs and other description-based approaches, domain knowledge has to be formalized to obtain model structures and parameters. In general though, this is harder to do in probabilistic graphical models if the domain knowledge that needs to be formalized is complex enough, because they are harder to interpret (more black-box-like). On the other hand, provided that the problem is suitable, domain-knowledge-based dynamic Bayesian networks for example can yield elegant models too [50, 51].

2.6.4 Weaknesses of Statistical Approaches

Statistical approaches generally provide less introspection. Many of them behave like black boxes, providing good solutions given the training data, but without logical explanations for them. Deductions are often not understandable by humans. Furthermore, they usually discover correlation, not causality. Another issue is that instances of off-the-shelf statistical approaches tend to be tweaked to their respective application domains, problem configurations, and training data. Porting them to other settings tends to involve a lot of effort. This issue might be partially solved by adding abstractions and configuration tools that facilitate this porting effort. The methods are based on statistical hypothesis testing. This is great for anomaly detection (rejecting the null hypothesis) and the related problem of classifying statistically different data points. But it fails if the semantics of the data are not solely expressed through statistical difference. This is because data points that lie within a class of statistically similar points cannot be extracted through statistical hypothesis testing.

Aggarwal and Ryoo [20] explain that statistical approaches are especially suitable when recognizing sequential activities. With enough training data, statistical models can reliably recognize activities, even from noisy or partial data. Their major limitation would be the inability to recognize activities with complex temporal structures, such as an activity composed of concurrent actions. For example, HMMs and DBNs have difficulty modeling the relationship of an activity A that occurred during, started with, or finished with an activity B. The edges of HMMs or DBNs specify the sequential order between two nodes, suggesting that they are suitable for modeling sequential relationships, not concurrent relationships. They also state that syntactic approaches have this problem.

2.6.5 Implications

Description-based approaches and statistical approaches both have their strenghts and weaknesses. Some problems are better solved with one and some with the other. When considering planning in robotics for example, low-level motion planning is best solved using a statistical approach and machine learning, whereas high-level action planning is best solved using a description-based approach. In other words, description-based for determining what to do and statistical for determining how to do it. The former problem has a large parameter space, the latter requires a lot of knowlege (semantics other than statistics). The choice also involves a trade-off between runtime and expressive power. Without the expressive power of logic, some problems cannot be solved. If and only if models of a certain complexity are required, logic can be the best choice for developing intelligent systems, taking its potentially poor runtime for granted or improving it through an appropriate adaptation. Finally, hybrid approaches, especially Markov logic networks seem to provide a good balance that combines the strenghts of both approaches. Markov logic networks have more expressive power and they are more intuitive than pure probabilistic graphical models, but they can be faster and they are better suited for machine learning than pure logic.

2.7 Fuzzy Metric Temporal Logic and Situation Graph Trees

The complexity of the presented problem has lead to a hierarchical description-based approach using fuzzy metric temporal logic (FMTL) combined with situation graph trees (SGTs). In this thesis, these methods are applied to interaction analysis in smart work environments for the first time. Nagel [95] provides an overview over the research of his group using the FMTL and SGT methods. They focus their efforts on the understanding of inner-city road traffic scenes. Their cognitive vision system spans the entire spectrum from video data, via conceptual knowledge and temporal development, up to natural language descriptions of the depicted scene. The system mediates between the spatiotemporal geometric descriptions extracted from video and a module that generates natural language text. In [25], an updated design for the cognitive vision system is presented, with detailed accounts of the subsystems for computer vision, knowledge representation, and natural language generation. In [55], the algorithm that extracts vehicle trajectories from monocular video data is presented in detail. These trajectories are imported into a conceptual representation based on FMTL and interpreted as verb phrases describing elementary actions of vehicles. As illustration, here is an example of a natural language output that was generated by their system from a video sequence recorded at an intersection:

> "Vehicle 1 entered the lane. Later vehicle 2 entered the lane. The vehicles formed a pair. Later vehicle 3 entered the lane. In the meantime the vehicles formed a queue. Vehicle 3 was the last vehicle of the queue. Vehicle 1 was the head of the queue."

Nagel et al. also applied these methods to complex scenes recorded at a gas station, yielding natural language outputs such as:

> "Vehicle 1 enters the gas station and stops at the second pump. Later vehicle 1 drives around vehicle 2 and leaves the gas station."

Furthermore, the FMTL and SGT methods were applied in cognitive vision systems for surveillance, deducing semantic descriptions from human movement patterns (among other things). Situations of interest are deduced from trajectories of one or more persons, extracted from videos of pedestrian intersections, parking lots, and public buildings. In [7] for example:

> "Person 1 leaves a bag and walks away (alarm is raised). Over 400 frames later, person 1 returns and picks up the bag (alarm is withdrawn)."

> "Person 1 and person 2 are walking together. At frame 390 they are standing together, at 402 they split up, at 410 person 1 stands together with person 3, and finally at 441, person 1 and person 3 are walking together."

Similarly, the situations "person entering car", "person leaving car", "person loading object", and "person unloading objected" are detected in [8]. In [29], such methods are used to control distributed pan-tilt-zoom cameras, following situations of interest in public buildings. This is used to reduce uncertainties in the observed scene and to maximize the amount of information extracted from it. González et al. [47, 48, 59] have also successfully applied FMTL and SGTs to surveillance applications, generating natural language outputs in English, Spanish, and Catalan, for example:

> "The pedestrian appears by the lower right side. He walks by the lower sideway. He waits to cross. He is waiting with another pedestrian. He crosses by the crosswalk. He walks by the upper sidewalk. He leaves by the upper right side."

> "The person appears by the lower right side. He walks by the cafeteria. He waits at the vending machine. He walks by the cafeteria. He walks among chairs. He sits on a chair. He walks among chairs. He walks by the cafeteria. He leaves by the upper right side."

In [30, 66], similar methods are applied to robot control, although [66] is more of a hybrid approach that incorporates logic reasoning. A rich source of in-depth information on reasoning with FMTL and SGTs is provided by the four dissertations [24, 67, 100, 111].

The situation descriptions that are presented in this thesis differ greatly from the ones listed above and from any of the other related studies described above. In summary, the reasoning process works as follows. Annotated input data are fed into a reasoning engine based on FMTL and SGTs. The reasoning models were implemented using F-Limette[8], an FMTL reasoning engine (similar to Prolog) written in C, and the SGT-Editor[9], a Java application for editing and traversing SGTs. The SGTs constitute the top part of the reasoning process. FMTL rules are called from their nodes to perform the bottom part of the reasoning process. FMTL rules are largely domain independent and typically about spatiotemporal relations, whereas SGTs are more domain specific as they usually constitute abstract relations between the FMTL rules they deploy. Once an FMTL rule base has been established, it stays relatively fixed and it can be used by different SGTs within the same application or even across different application domains. Over the years, the FMTL/SGT methods were incrementally improved while adhering to their validity demonstrated in [7, 24, 25, 55, 59, 111].

2.7.1 Fuzzy Metric Temporal Logic

The FMTL language, introduced by Schäfer and Nagel [95, 111], is a first order logic (or predicate logic) extended with fuzzy evaluation and temporal modality. This section explains both these extensions.

Fuzzy Evaluation

Fuzzy evaluation means that the language uses fuzzy instead of binary truth values. The term fuzzy logic was officialy introduced by Zadeh in [136], but fuzzy logic has been studied since the 1920s by Łukasiewicz, Tarski, and others. It has been successfully applied in control engineering, artificial intelligence, computational linguistics, and other fields. The book [91] provides an extensive overview. A key feature of many intelligent systems is the ability to associate information with truth values between 0.0 and 1.0. But what do these truth values mean? They are often referred to as "degree of belief", "confidence", "probability", or "likelihood", suggesting that the truth values reflect uncertainty, not vagueness, nor both. Degree of belief is the most neutral term, with less semantic bias toward uncertainty than the other three. Berger [32] provides a detailed source on reasoning under uncertainty. Here, we apply the following definition, consistent with the one

[8]http://cogvisys.iaks.uni-karlsruhe.de/Vid-Text/f_limette
[9]http://cogvisys.iaks.uni-karlsruhe.de/Vid-Text/sgt_editor

provided in [24, p. 9]. Having a range of truth values between 0.0 and 1.0 instead of just two (0 and 1) is called fuzziness, which can have at least two different meanings: vagueness and uncertainty.[10]

Vagueness is an inherent property of many (natural) language concepts which has little to do with noise or uncertainty. A statement like "a is close to b" or "a is fast" can be either completely true, completely false, or somewhere in between, depending on the physical distance between a and b and the physical speed of a respectively. Even if perception is flawless, without noise or uncertainty, the truth values of these statements should be able to lie anywhere between 0.0 and 1.0. Uncertainty about information on the other hand, is caused by limited knowledge about the state of the world. In many intelligent systems, it arises through noisy and otherwise imperfect machine perception. In [69, p. 24], a more detailed and subtle analysis of vagueness and uncertainty is given. It states that there are multiple types of each, and that the terms are interrelated in some cases. For many applications, one needs to explicitly handle vagueness, uncertainty, or both. This thesis focuses on handling vagueness, but it also contains some thoughts and experiments on how to handle uncertainty and noise, as well as how to handle vagueness and uncertainty at the same time (below and in Section 3.5). Chapter 4 contains many examples of how the presented system handles vagueness, and Chapter 5 contains the corresponding evaluations, as well as an experiment with noisy data.

Table 2.1 provides the most common semantics for fuzzy conjunction, disjunction, and negation; weak, medium, and strong. Figure 2.1 provides the corresponding graphical representation. F-Limette uses the default semantics (boldfaced in Table 2.1 and Figure 2.1) weak conjunction, strong disjunction, and medium negation. These default semantics are applied throughout Chapters 4 and 5. These particular semantics are applied, because they are the closest to our intuition and because it is common practice when using fuzzy logic reasoning. Fuzzy operator semantics are also discussed in [24, p. 35]. The weak, the medium, as well as the strong semantics fulfill the following logical conditions:

[10]As discussed below, this definition can (and should) be expanded by allowing truth values to express a combination of vagueness and uncertainty, or by having two truth values per atomic fact: one for vagueness and one for uncertainty.

Table 2.1: The most common semantics for fuzzy conjunction, disjunction, and negation; weak, medium, and strong. The semantics that are applied throughout Chapters 4 and 5 are boldfaced. $V(C)$ is the truth value of C.

	Weak	Medium	Strong
$C_1 \wedge C_2$	$\mathbf{min(V(C_1), V(C_2))}$	$V(C_1) \cdot V(C_2)$	$\max(0, V(C_1) + V(C_2) - 1)$
$C_1 \vee C_2$	$\min(1, V(C_1) + V(C_2))$	$(V(C_1) + V(C_2)) - (V(C_1) \cdot V(C_2))$	$\mathbf{max(V(C_1), V(C_2))}$
$\neg C$	$1 - V(C)^2$	$\mathbf{1 - V(C)}$	$1 - \sqrt{V(C)}$

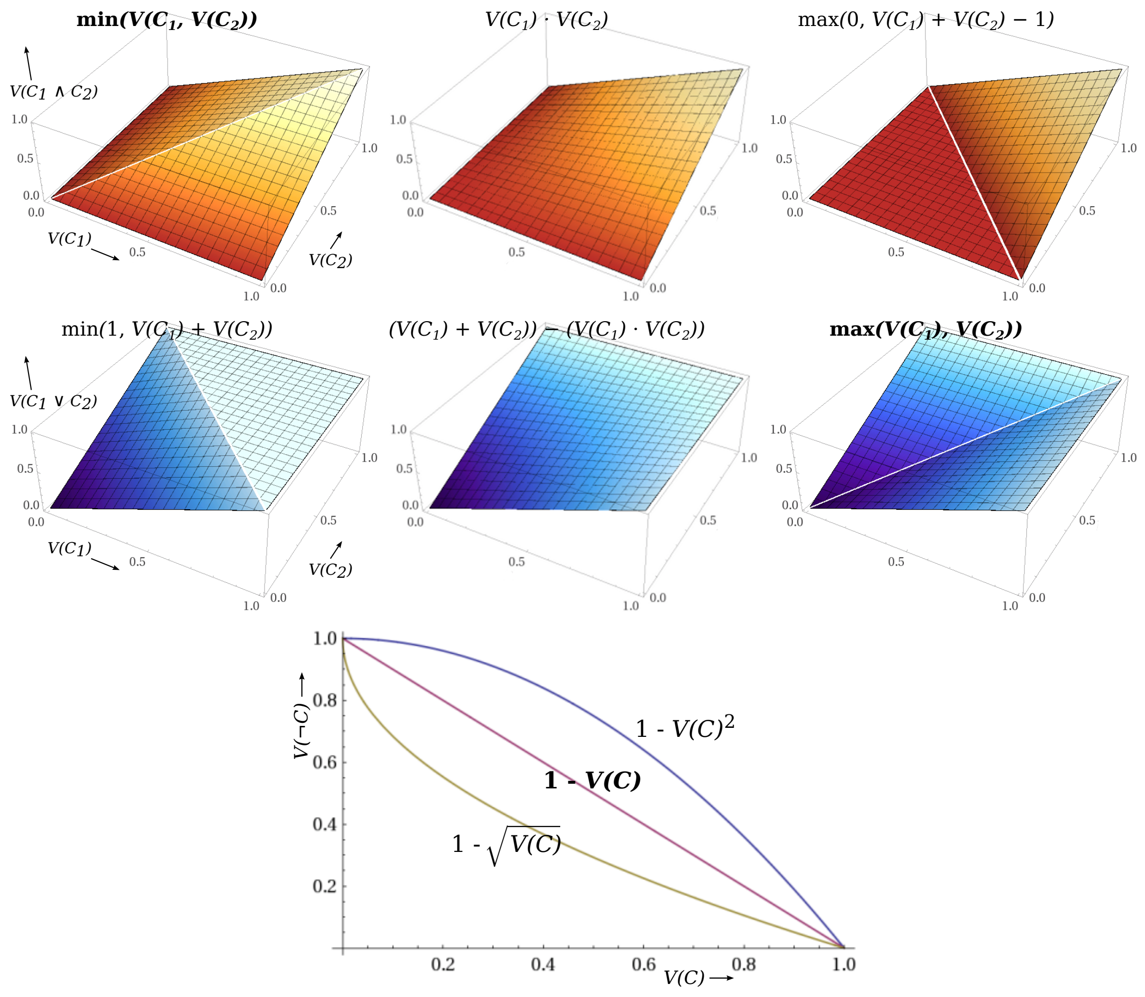


Figure 2.1: A graphical representation of the semantics for fuzzy conjunction, disjunction, and negation; weak, medium, and strong. The semantics that are applied throughout Chapters 4 and 5 are boldfaced. $V(C)$ is the truth value of C.

- conjunction:
  - upper bound (neutrality); $V(C_1) = 1 \quad \Rightarrow \quad V(C_1 \wedge C_2) = V(C_2)$,
  - lower bound; $V(C_1) = 0 \quad \Rightarrow \quad V(C_1 \wedge C_2) = 0$,
  - monotony; $V(C_1) \leq V(C_2) \quad \Rightarrow \quad V(C_1 \wedge C_3) \leq V(C_2 \wedge C_3)$,
  - commutativity; $V(C_1 \wedge C_2) = V(C_2 \wedge C_1)$,
  - associativity; $V((C_1 \wedge C_2) \wedge C_3) = V(C_1 \wedge (C_2 \wedge C_3))$,
- disjunction:
  - upper bound; $V(C_1) = 1 \quad \Rightarrow \quad V(C_1 \vee C_2) = 1$,
  - lower bound (neutrality); $V(C_1) = 0 \quad \Rightarrow \quad V(C_1 \vee C_2) = V(C_2)$,
  - monotony; $V(C_1) \leq V(C_2) \quad \Rightarrow \quad V(C_1 \vee C_3) \leq V(C_2 \vee C_3)$,
  - commutativity; $V(C_1 \vee C_2) = V(C_2 \vee C_1)$,
  - associativity; $V((C_1 \vee C_2) \vee C_3) = V(C_1 \vee (C_2 \vee C_3))$,
- negation:
  - upper bound; $V(C) = 0 \quad \Rightarrow \quad V(\neg C) = 1$,
  - lower bound; $V(C) = 1 \quad \Rightarrow \quad V(\neg C) = 0$,
  - monotony; $V(C_1) < V(C_2) \quad \Rightarrow \quad V(\neg C_1) \geq V(\neg C_2)$.

Temporal Modality

The second extension, temporal modality, enables the modeling of developments along the time axis instead of just at a single point in time. Modern temporal logic was greatly inspired by the work of Prior, founder of tense logic, which is now also known as temporal logic. In his metaphysical work [104], he introduces the temporal modalities past, present, and future as basic ontological categories of fundamental importance for our understanding of time and the world. Pnuelli [101] successfully used temporal logic to reason about the behavioral properties of parallel programs and more generally reactive systems. With interval temporal logic [21], Allen and Ferguson provided a successful axiomatization of time periods that has been applied in many studies since.

The idea of temporal logic is that rule conditions can be grounded in past, current, and future states of the world. Furthermore, FMTL possesses a metric on time, allowing expressions about exact time differences in addition to categorical concepts such as "before" and "after". Temporal modality is essential for modeling the speed of an object for example, and for modeling situations consisting of multiple phases. It can also achieve a smoothing effect against noise, outliers, and brief changes that should be ignored. Chapter 4 uses the following temporal logic operators:

- interval diamond operator $\Diamond_{[t_1,t_2]}$: at least once between times t_1 and t_2
- interval box operator $\Box_{[t_1,t_2]}$: always between times t_1 and t_2,
- relative diamond operator $\Diamond_{dt}$: at time $t = t_0 + dt$, and
- fractional box operator $\Box^{a\%}_{dt_1,dt_2}$: at least $a\%$ of time interval $[t_0 + dt_1, t_0 + dt_2]$.

Horn Fragment and Higher-Order Constructs

Analogous to Prolog, F-Limette reduces FMTL to its Horn fragment FMTHL (Fuzzy Metric Temporal Horn Logic). This means that only formulas of a certain form are allowed,[11] leading to a reduction in time complexity (see Section 2.6.2). Furthermore, F-Limette allows higher-order constructs, leading to an increase in time complexity. In particular, predicates can be nested within other predicates, whereas in standard first order logic, you are only allowed to nest functions in predicates, not other predicates.[12] Examples of predicates nested in other predicates can be found in Formulas 4.25 and 4.45 and in *Filter(**p**, C, **q**)* in Table 4.2. Here, C is instantiated with an entire predicate to be nested. Further examples that indirectly use nested predicates are Formulas 4.33 and 4.42. Here, the predicate to be nested is built from the supplied predicate name and arguments for it from the supplied lists (and constant in the latter case). This behavior is made possible by the predicate *Call(C)*. It calls predicate C and returns its truth value dynamically at runtime. Another powerful example of nested predicates is found in Formula 4.48.

Handling Uncertainty in the Input Data

The presented system contains a tool for input data manipulation (see Section 3.5). This tool was developed because the presented reasoning methods need to work with real machine perception in the future. It can add noise and perform interpolation as well as several other manipulations. One of them is the addition of confidence values (patterns of initial truth values associated with the input data). In [5], the author described on a theoretical level how the applied methods could handle such imperfections, and in particular how initial truth values expressing confidence in the input data might be combined with truth values expressing vagueness. Just like vagueness, confidence in the atomic facts should be reflected in the high-level situation descriptions that are deduced from them. The remainder of this section describes how the presented methods could handle noise, outliers, data gaps, and confidence values. The discussion is an extended version of the one found in [5].

Noise and outliers. The presented approach can inherently handle noise and outliers to a certain degree by applying temporal filtering and fuzzy evaluation. Robustness against noise is evaluated in Section 5.5 in an exemplary fashion. Although the models evaluated here were not optimized for it, the results show that they are robust against small

[11] Horn clauses are disjunctions of literals with at most one positive literal: e.g. $T(a,b) \vee \neg R(a,b) \vee \neg S(a,b)$. They are logically equivalent to the form that is used in Prolog and F-Limette, in this case: $T(a,b) \leftarrow R(a,b) \wedge S(a,b)$. Other forms in Prolog and F-Limette have to be logically equivalent to these.

[12] In first order logic, functions can be nested because they return legal arguments for predicates such as object names and numbers. Examples of functions nested within predicates: *Larger(age(a), age(b))* and *Likes(mother(a), mother(b))*. Predicates return truth values and cannot be nested as such. To distinguish functions from predicates, the former are usually not capitalized. In order to nest predicates within predicates, F-Limette uses a special construct as explained below.

amounts of noise. More modeling effort is required to handle larger amounts of noise (and outliers). Independently of the chosen modeling and reasoning methods, robustness against noise and outliers can be increased by applying outlier detection and smoothing during preprocessing. Furthermore, noise and outliers might be detected by the reasoning process itself, using rules about the data's expected dynamics, potentially even providing machine perception with top-down knowledge about this to improve its outputs or guide sensor and resource deployment.

Data gaps. In logic reasoning, the effects of data gaps can be countered to a certain degree using abduction, where intermediate conditions that can not be deduced are "hallucinated" instead so that reasoning can continue and certain events can be detected despite their missing conditions. Furthermore, interpolation across data gaps can be applied during preprocessing, independently of the chosen modeling and reasoning methods. This effectively turns the data gap problem into a confidence problem, because interpolated data should have appropriate confidence values associated with them that depend on the confidence values in the surrounding data used for interpolation as well as on the temporal distances between new data points and the ones they were calculated from. Confidence values should increase towards an interpolated gap's edge, and large gaps should cause ever lower confidence values as you move to the center. In [8], the author described how to apply abduction as well as interpolation to the FMTL/SGT framework. In [108], a probabilistic reasoning component is proposed that solves for missing and superfluous atomic actions.

Confidence Values. Let us consider an obvious approach to handling confidence values in the input data (from interpolation or other causes), using *DistBetweenCenters(p, q, c)* (Formula 4.11). It calculates the distance between the centers of objects p and q and associates this distance with distance category c. Each input i can have a truth value $P[i]$ between 0.0 an 1.0 that reflects confidence. Let $Position(p, x_p, y_p)$ and $Position(q, x_q, y_q)$ retrieve the center positions on a plane for objects p and q (as listed in Table 4.1), with confidence values $P[Position(p, x_p, y_p)]$ and $P[Position(q, x_q, y_q)]$. The Euclidian distance between the centers of objects p and q are calculated using Formulas 4.16 and 4.21. Confidence values are usually combined through multiplication and d_{pq} depends on p and q, so $P[d_{pq}] = P[p] \cdot P[q]$. The FMTL rule $AssocDistBetweenCenters(d_{pq}, c)$, corresponding to Figure 4.5 (top left), is used to associate distance d_{pq} with distance category c, yielding a vague truth value *V[DistBetweenCenters(p, q, c)] = V[AssocDistBetweenCenters(d_{pq}, c)]* between 0.0 an 1.0, independent of the value of $P[d_{pq}]$. This means that uncertainty and vagueness are represented separately. Optionally, one could use the weak, medium, or strong conjunction semantics from Table 2.7.1 to combine them into a truth value V' reflecting both uncertainty and vagueness: *V'[DistBetweenCenters(p, q, c)] = V[AssocDistBetweenCenters(d_{pq}, c)] $\cdot$ P[d_{pq}].* This is an obvious but naive approach. Multiple theories for combining uncertainty and vagueness need to be evaluated, and the applied heuristics need to be chosen carefully. To find out whether their behavior makes sense, one needs to examine their formal properties in terms of monotonicity, asymptotics, etc. Useful insights in this direction can be found in [43] and in [65] for example. Type-2 fuzzy logic [90, 137] models uncertainty and vagueness as two separate dimensions, adding a third dimension to truth functions such as the ones displayed in Figure 4.5.

2.7.2 Situation Graph Trees

SGTs, introduced by Arens and Nagel [24, 95], are used as an abstraction layer for FMTL. Figure 2.2 shows an abstract SGT example that recognizes moving and stationary groups as well as meetings at tables and interaction with displays. An SGT is a hypergraph that consists of situation graphs, represented by unfilled rounded rectangles. Each situation graph contains one or more situation schemes, represented by filled rectangles. Each situation scheme possesses a name (top segments of filled rectangles in Figure 2.2), one or more conditions (middle segments), and zero or more actions (bottom segments). Typically, there is one SGT traversal per frame per observed person (or vehicle for example). Each traversal normally starts by selecting an agent for that traversal (in the *Root* situation scheme). The selected object becomes the center of the reasoning process during that SGT traversal. During each traversal, all possible paths between the root and the leafs of an SGT are traversed. If a condition along a path cannot be fulfilled, that path is done and the next one is started. The traversal algorithm that is applied in this thesis (adapted from earlier work) is described in [93].

The situation schemes' conditions initiate deductions in FMTL; Prolog-like reasoning processes in F-Limette (FMTL reasoning engine). Each condition returns a truth value between 0.0 and 1.0, depending on the rules that were directly or indirectly evaluated, and ultimately on the atomic facts from the input data. The next condition C_{next} (either within the same situation scheme or in the next, conceptually refined, situation scheme) uses the truth value returned by the previous condition C_{prev} as base truth value. More specifically, weak conjunction semantics (default, see Table 2.7.1) are used as follows: $V'[C_{next}] = min(V[C_{prev}], V[C_{next}])$. During the next step down the tree, the same manipulation is performed, using $V'[C_{next}]$ as $V[C_{prev}]$. This means that the situations deduced along each path of an SGT are always ordered from generic to specific, as their truth values cannot increase along the way. As soon as a condition returns the truth value 0.0, reasoning stops and the next path is started from the *Root* situation scheme. The situation schemes' actions generate the final outputs of the system: situation descriptions, or in embodied settings also actuator commands. The truth value returned by the last condition above each action is also associated with the action, representing vagueness (or uncertainty, or both). Actions below unfulfilled conditions (returning the truth value 0.0) are not executed, and, as we will see in Chapter 5, one can ignore outputs with a truth value

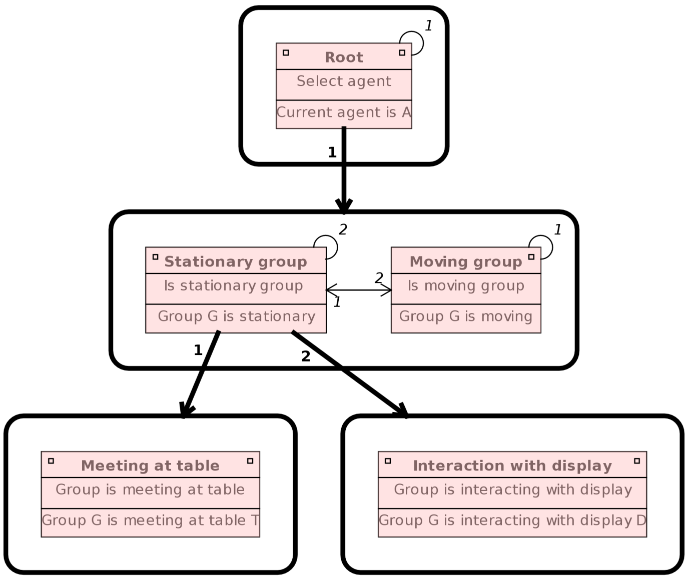


Figure 2.2: An abstract example of a situation graph tree (SGT) that recognizes moving and stationary groups as well as meetings at tables and interaction with displays. During SGT traversal, the conditions (the middle segments of the rectangles) initiate deductions in FMTL (i.e. Prolog-like reasoning processes in F-Limette). The actions (bottom segments) can either be the generation of situation descriptions or actuator commands.

smaller than a certain threshold.

Trying to deduce a more specific situation after a more generic one is called conceptual refinement. This is demonstrated by the situation schemes *Stationary group*, *Meeting at table*, and *Interaction with display* in Figure 2.2. To model temporal dynamics and situations consisting of multiple phases, situation schemes can be connected through temporal edges as demonstrated by the situation schemes *Stationary group* and *Moving group* in Figure 2.2. *Moving group* is only activated if either *Stationary group* or *Moving group* has held (with $V > 0.0$) in the previous frame. The squares in the upper left and upper right corners of situation schemes indicate start situations and end situations in temporal chains, and the circles on their upper right corners represent reflexive temporal edges. The models in Chapter 4 use various temporal FMTL rules, but no temporal SGT edges, with the exception of the SGT displayed in Figures 4.19 through 4.21. Temporal modality at FMTL rule level was introduced in Section 2.7.1.

That concludes Chapter 2 on background and related work. Statistical, syntactic, description-based, and hybrid approaches were introduced, some relevant studies on smart environments and related application domains were discussed, the strenghts and weaknesses of the different types of approaches were compared, and the FMTL/SGT approach was discussed in terms of related work and background.

Chapter 3

Case Study and Dataset

This chapter explains how the dataset for the case study was produced. The dataset and the applied annotation methods form an important part of the presented research and they provide it with extra novelty. The required audiovisual data was gathered at the State Fire Service Institute North Rhine-Westphalia during one of their staff exercises for crisis response control room operations.The annotation methods consist of four parts: audiovisual data recording (Section 3.2), input data annotation (Section 3.3), ground-truth annotation (Section 3.4), and optional input data manipulation (Section 3.5). But we start with a summary of the organisation and workflow of the recorded staff exercise.

3.1 Control Room Organisation and Workflow

The director of operations coordinates the entire affair and has the final responsibility. His first officers are responsible for specific functional areas: unit management (S1), situation assessment (S2), strategy (S3), and supplies (S4). Depending on staff availability, additional functional areas can be added or they can be collapsed together instead. The first officers as well as the director of operations have one or two additional staff members answering to them. Other staff members are concerned with maintaining displays (e.g. maps and unit tables), editing documents, and managing incoming and outgoing messages. Meanwhile, several instructors are offering assistance to the trainees, the director of operations being one of them. Finally, the simulation of the world outside the control room is performed by a combination of instructors and trainees.

The staff members have predefined roles and they follow standard operating procedures as much as possible.[1] The typical workflow consists of periodic cycles of briefings and dynamic control room operations, where briefings contain multiple phases lead by the first officers responsible for the functional areas. During dynamic operations between briefings, the staff members scatter across the room, attending to their displays, documents, and messages. Groups are constantly forming and breaking with lots of discussion going on.

[1]Note that despite standard operating procedures, improvisation is an important skill in crisis management [86].

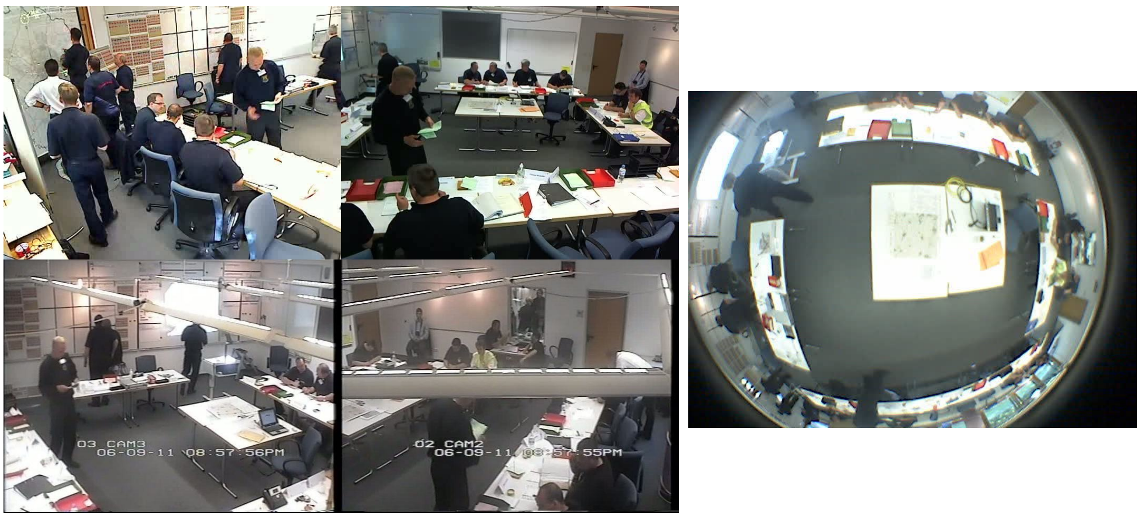


Figure 3.1: Example video data recorded at the fire brigade staff exercise, with views from all five cameras. Such images, sampled from the raw video data at 1*fps*, are used during input data annotation.

3.2 Audiovisual Recording

The six hour long staff exercise was recorded using five cameras and four microphones, providing complete and redundant coverage of the control room. After the recordings, the video footage was sampled at 1*fps*, yielding the five-pane images exemplified by Figure 3.1. These images, along with the recorded audio tracks, are used during input data annotation as described below. An image sampling rate of 1*fps* proved to be sufficient for the current purposes. Human annotators are able to annotate the input data based on 1*fps* images, and the reasoning methods as well as human annotators are able to recognize the targeted situations based on the annotated input data, since the targeted situations do not have fast dynamics.[2]

The first cycle, containing the introductory phase described below, was analyzed thoroughly and two four minute fragments and two ten minute fragments were selected for the annotation process. First, everybody is preparing their workspaces, waiting for the director of operations to introduce the current crisis situation. When he does, everybody stops working and returns to their seats to listen. After the introduction, the director of operations tells his staff to continue their preparations and asks his first officers to join him at the central table for strategic planning. Once this is done, the director of operations addresses the whole room, announcing that everybody must attend to their tasks until the next briefing.

After this initial phase, their behavior becomes highly dynamic. Director of operations, first officers, their subordinates, and supporting staff scatter across the room, attending to their displays, documents, and messages. Groups are constantly forming and breaking,

[2]Computer vision algorithms (especially ones with tracking) could benefit from higher sampling rates.

and there is a lot of discussion going on. In due time, the director of operations calls the next briefing and everybody returns to their seats. After an introduction by the director of operations, each of the first officers stands in front of the appropriate wall display to give a status report on their own functional area. Everybody listens quietly, except for the director of operations who is occasionally asking the presenter questions, sometimes involving one of the other first officers in the discussion. The director of operations concludes the briefing by summarizing the current action plan and everybody gets back to performing dynamic control room operations.

3.3 Input Data Annotation

Because the required machine perception is not yet available, a human annotator has to analyze the audiovisual data in order to produce hypothetical machine perception outputs. This approach avoids machine perception dependancy problems and allows research to focus on high-level reasoning. It is a step along the way toward a real-time system containing various automatic machine perception components. A dedicated input data annotation tool was developed for this purpose (Figure 3.2). The recorded 1*fps* five-pane images exemplified by Figure 3.1 are displayed in an image viewer and the audio tracks are played by an audio player, while a human annotator analyzes them in order to produce corresponding symbolic data using mouse and keyboard. This is done by manipulating the modeled persons and objects in the birdseye view of the input data annotation tool. The reasoning engine uses the result as input data. From the six hours of recorded audiovisual data, two four minute sections and two ten minute sections were selected for annotation. They include data for each phase observed in the control room workflow and the transitions between them: briefings (consisting of multiple phases) as well as dynamic control room operations.[3] The video footage contains all the information needed to annotate the symbolic data, except for the participants' speech activity and some useful auditory context information about the operations in the control room, which are provided by the audio footage.

In the input data annotation tool (Figure 3.2), each person can be moved and rotated, and their body pose, gesture activity, and speech activity can be set. Speech is indicated by a rim around the head and speech-supporting gesticulation by a rim around the right hand. An extended and optionally rotated arm indicates pointing or interaction with displays, notepads, and messages. An extended head indicates looking down and extended legs indicate sitting. Notepads and messages can only be moved around. Of course, functionality for recording, playing back, and navigating through the data is included in the user interface and data files can be saved to be reloaded later. Using this method, between 10 and 20 seconds of input data can be annotated in an hour. This is alright for developing reasoning methods using manageable amounts of data, but a more automatic

[3]Most of the models presented in Chapter 4 recognize group behavior during the dynamic phases.

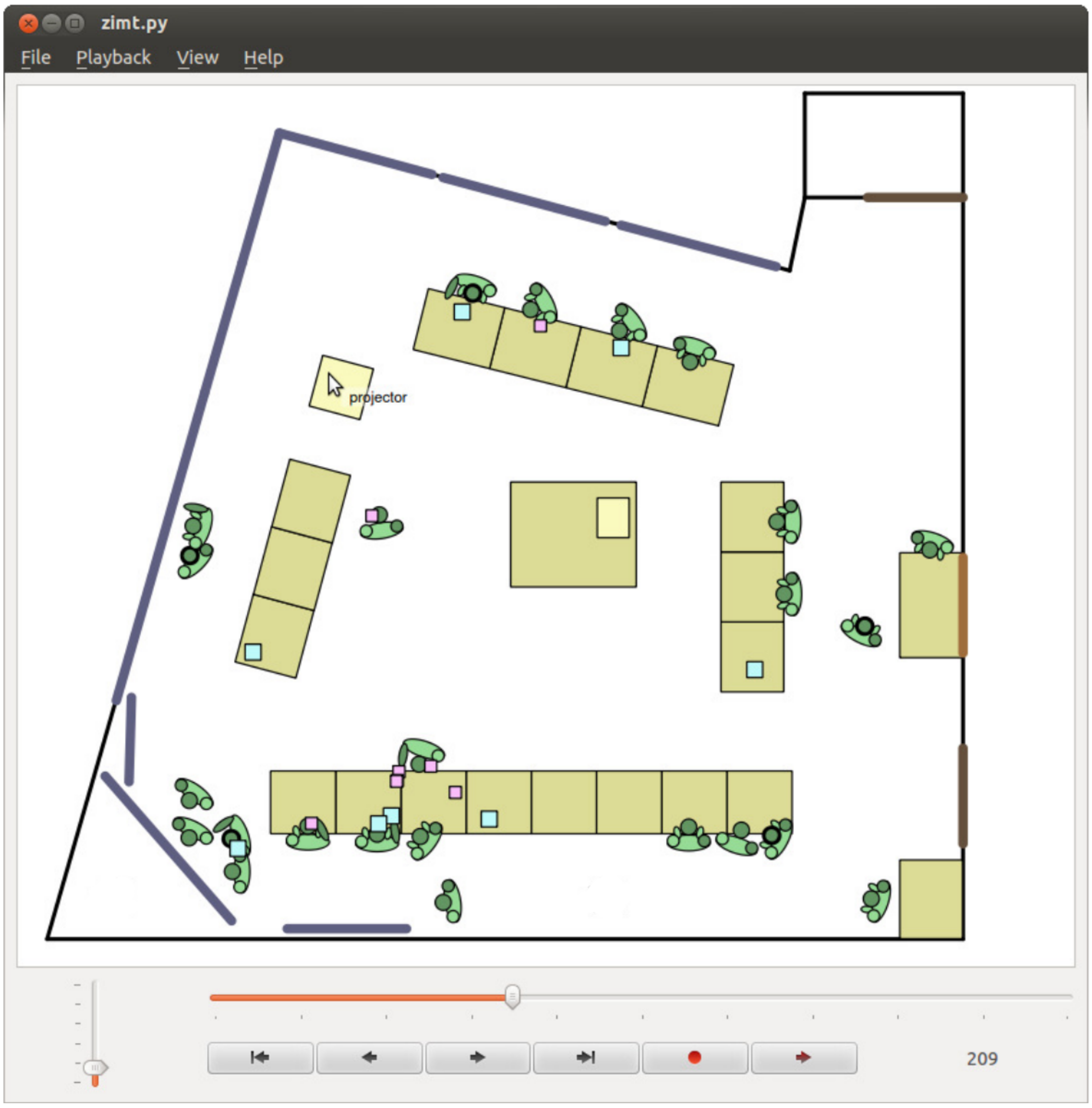


Figure 3.2: Screenshot of the developed input data annotation tool. It is used to annotate hypothetical machine perception outputs for each of the 1*fps* images exemplified by Figure 3.1. This annotated hypothetical machine perception is used as input by the reasoning engine.

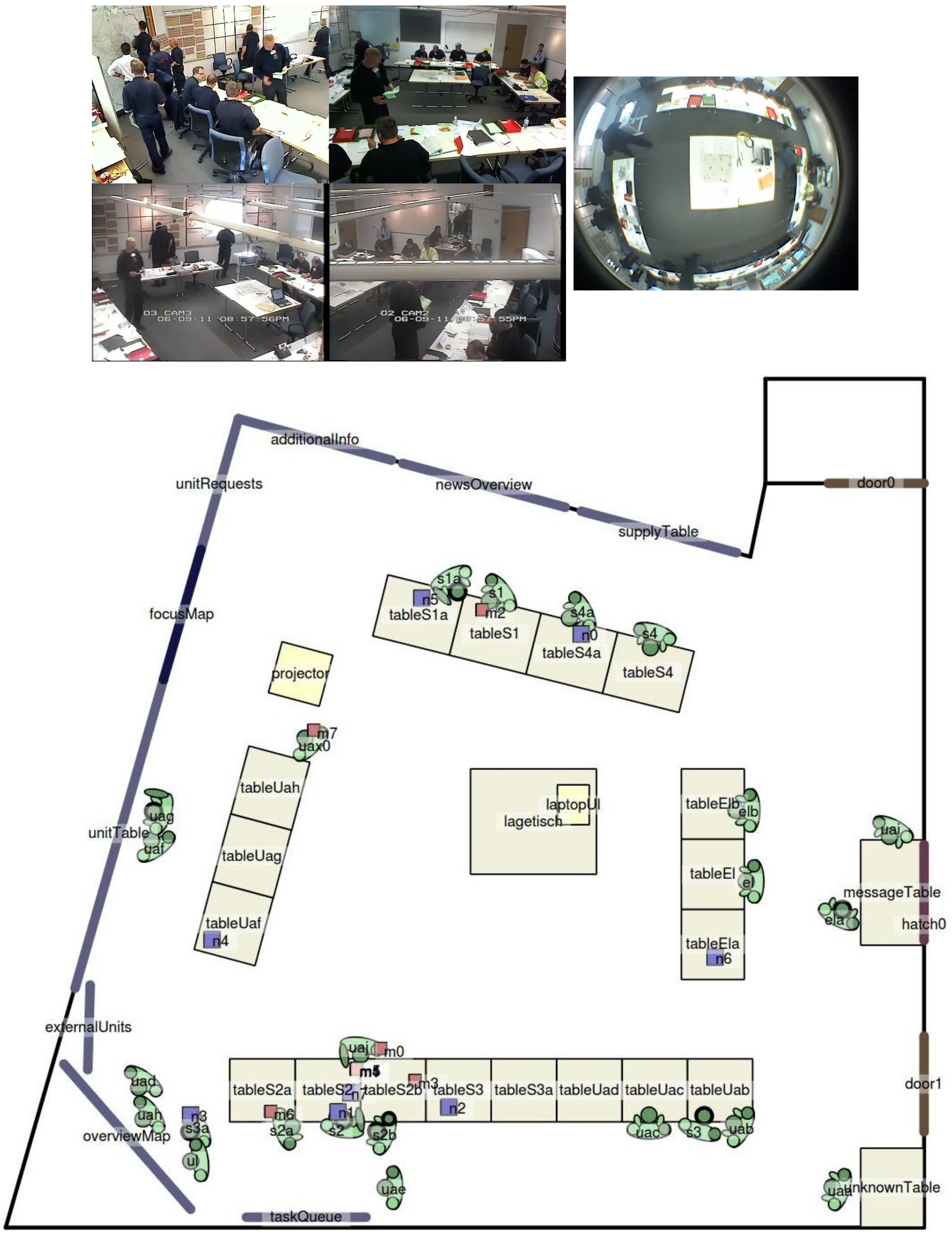


Figure 3.3: An exemplary frame of video data with the corresponding annotated input data.

approach is required in the long run. This can range from e.g. wearable sensors to obtain tracking data automatically while manually annotating other attributes, to computer vision and speech detection, i.e. fully automatic unintrusive perception.

The resulting input data for the reasoning engine consists of symbolic person and object representations over time. The dynamic objects (with attributes that change over time) are notepads and paper messages, and the static objects (with only constant attributes) are: tables, displays, doors, and devices. Their attributes are: name, type, position, horizontal orientation, width, and height (in the horizontal plane), and additionally for persons: gesture activity (two types: one binary for speech supporting gesticulation, one horizontal angle for pointing gestures), speech activity (yes/no), vertical head orientation (up/down), and body pose (sitting/standing). Hence, the data annotations for persons consist of state descriptors of the form *[name:el, type:person, x:774, y:412, w:54, h:20, orientation:144, speech:false, gesture:-18, looking_down:true, sitting:false]*, meaning: person "el" is located at x = 774cm and y = 412cm, has width 54cm and height 20cm, and an orientation of 144°. He is not speaking, pointing 18° left of his orientation, looking down, and not sitting. The persons and objects in Figure 3.2 and similar images throughout the thesis are simply visualizations of such state descriptors.[4]

The described taxonomy is analogous to the one proposed by Fischer and Beyerer [51, p. 44–45]. In 2011, Fraunhofer IOSB organized a study group on term formalization, consisting of the author of this thesis and five other researchers, because there is a lack of concensus in this area. The goal was to arrive at common definitions for some key terms.[5] In this thesis, the following terms are applied:

- A *person* or *object* is a description in the world model of a physical entity in the real world.
- Its *attributes* can be divided into static ones (i.e. properties, a person's name for example) and dynamic ones (i.e. states, an object's position for example). A person or object can also have inferred attributes and inferred relations with other objects, e.g. the degree to which two persons are close to each other.
- A *frame* is the set of all persons and objects and their observed and inferred attributes and relations at a given time.[6]
- A *group* is defined through the models in Chapter 4. They are defined in terms of the proximity, orientations, and interaction patterns of their members (persons and objects).

[4]Annotated input data with higher sampling rates than 1*fps* can be achieved through additional annotation or through interpolation of the existing 1*fps* annotated input data, as described in Section 3.5.

[5]Although the author of this thesis is not one of the authors of [51], he did contribute to it indirectly through the study group.

[6]In [51], the configuration space is the collection of all persons and objects (they call them object representatives) and all possible values of all their attributes. Thus, frames are points in the configuration space and a sequence of frames is a trajectory through the configuration space.

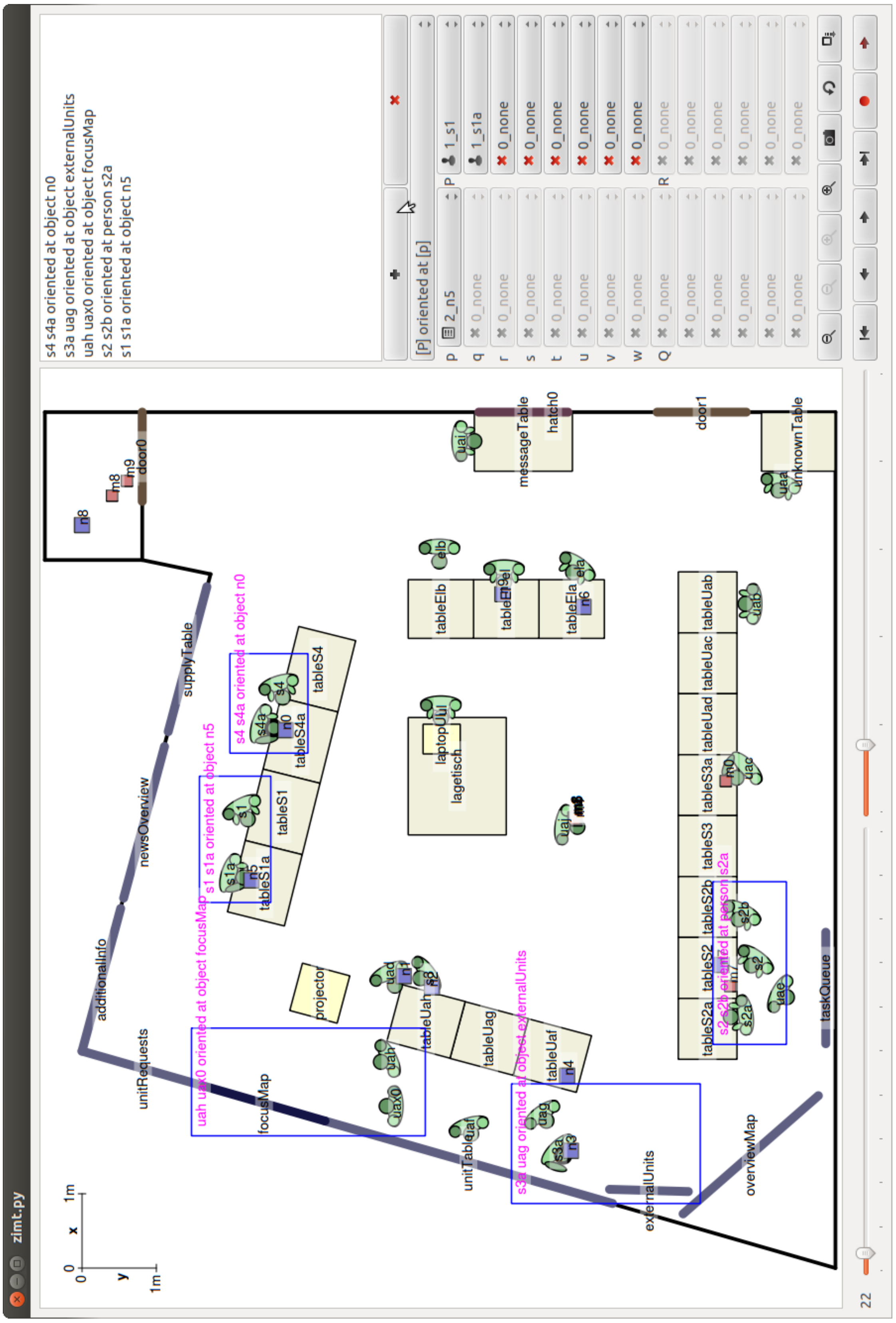
- A *situation* is a semantic statement about the world model that has a truth value between 0.0 and 1.0 associated with it, inferred from the attributes of the objects and persons in the current frame and the surrounding ones. Situations abstract away from the level of detail that is available in the world model. The term *situation description* refers to the reasoning engine's output.

3.4 Ground-Truth Annotation

After the input data has been produced, corresponding ground-truth is required to compare to the reasoning engine's output. These are the situation descriptions that the system should deduce. This is also achieved through manual annotation, in a dedicated software tool developed for this purpose: Figure 3.4. The annotator analyzes the symbolic input data (from Section 3.3) in the birdseye view to determine the correct situation descriptions.

The annotator has to repeatedly select the appropriate situation type as well as the involved persons and objects, using the interaction panel on the right side of Figure 3.4. The resulting situation descriptions (i.e. ground-truth results) are stored in the list in the top-right corner and the corresponding boundingboxes are drawn in the birdseye view. Much like the input data annotation tool, the ground-truth annotation tool offers functionality for recording, playback, navigation, saving, and reloading. With between 50 and 80 seconds of ground-truth per hour, ground-truth annotation is less time-consuming than data annotation (10–20 seconds per hour). Without knowledge of the reasoning process' inner workings, the annotator was instructed to annotate the ground-truth according to his own observations and common sense. Chapter 5 describes how the system is evaluated by comparing the resulting ground-truth to the situation descriptions that are deduced by the reasoning engine.

The following ground-truth situations were annotated, because they represent common group constellations and interaction patterns that occur during dynamic control room operations. Italics signify variables that should be instantiated with a person or object, and bold-italics signify variables that should be instantiated with a list of persons. Between four and six minutes of 1*fps* ground-truth was produced for each of the following situation types. These ground-truth situation types are automatically deduced through logic reasoning by the primary and secondary models presented in Sections 4.1 and 4.2 respectively. Figure 3.5 provides some corresponding visual examples, generated by the developed ground-truth annotation tool.


s4 s4a oriented at object n0
s3a uag oriented at object externalUnits
uah uax0 oriented at object focusMap
s2 s2b oriented at person s2a
s1 s1a oriented at object n5

[P] oriented at [p]

Figure 3.4: Screenshot of the developed ground-truth annotation tool. It is used to annotate ground-truth that can be compared to the reasoning engine's output in order to evaluate its performance.

Primary models:

- ***group*** is listening to *person*,
- conversation between ***persons*** in group ***g***,
- silent group ***g***,

- ***persons*** in group ***g*** are sitting,
- ***persons*** in group ***g*** are standing,
- ***persons*** in group ***g*** are moving,

- ***persons*** are joining ***group***,
- ***persons*** are leaving ***group***,
- constant group ***g***,

- *person* is pointing at *object*,
- *person* is pointing at *object* and ***group*** is looking at it,
- *person* is speaking and pointing at *object*, and
- *person* is speaking and pointing at *object* and ***group*** is looking at it.

Secondary models:

- ***group*** oriented at center,
- ***group*** oriented at object *o*,
- ***group*** oriented at person *p*,

- *person* talking to ***group***,
- *person* talking to ***group*** about *object*,
- ***group*** in dialogue,
- ***group*** in dialogue about *object*,

- *person* carrying *document*,
- *person* moving *document*,
- *person* reading *document*,
- *person* writing *document*,
- *person* picking up *document*, and
- *person* laying down *document*.

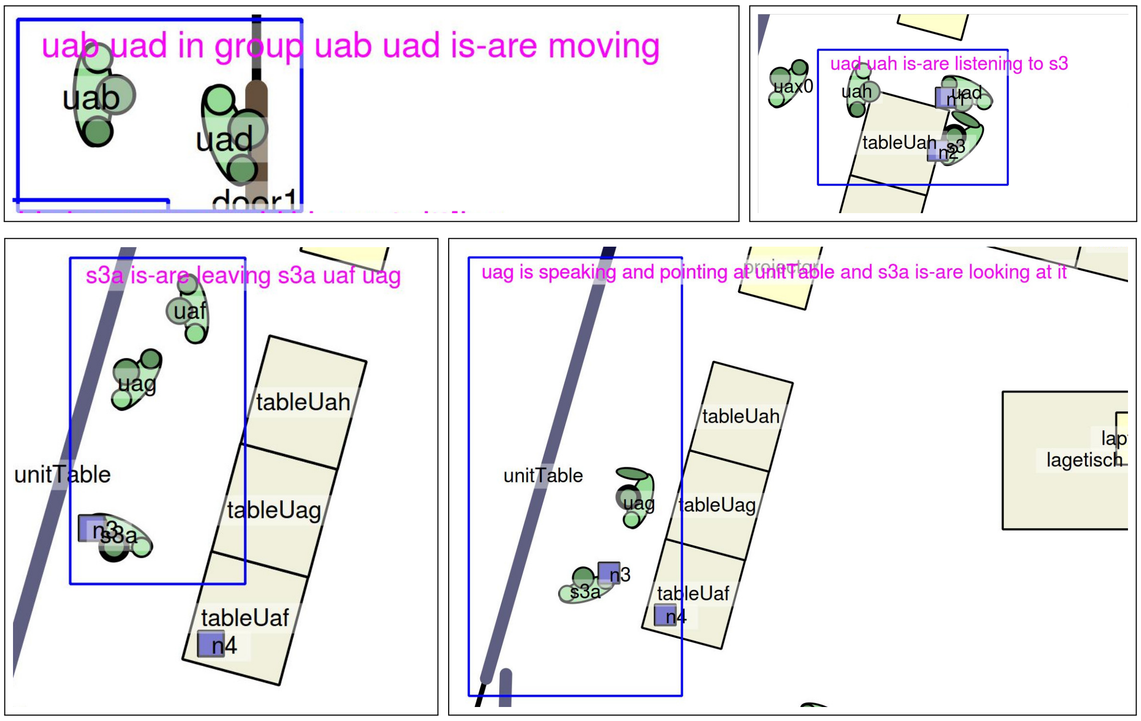

uab uad in group uab uad is-are moving
uab
uad
uad uah is-are listening to s3
uax0
uah
tableUah
s3a is-are leaving s3a uaf uag
uaf
uag
tableUah
unitTable
tableUag
n3
s3a
tableUaf
n4
uag is speaking and pointing at unitTable and s3a is-are looking at it
tableUah
unitTable
uag
tableUag
n3
s3a
tableUaf
n4
lagetisch

Figure 3.5: Some visual ground-truth examples, generated by the developed ground-truth annotation tool.

3.5 Input Data Manipulation

The presented system contains a tool for input data manipulation. It can add noise, perform interpolation, and several other manipulations. This tool was developed because the presented reasoning methods need to work with real machine perception in the future.

Noise. The data manipulation tool can add noise to any of the dynamic person and object attributes: position, orientation, and arm rotation, but also boolean attributes: speaking, sitting, looking down, and gesticulating. This is used to evaluate the system's robustness against noise in Section 5.5. The amount of noise to add to which attribute(s) can be configured, and the resulting noisy data can be visualized in the tools described in Sections 3.3 and 3.4.

Interpolation. In some cases, the $1fps$ sampling rate of the annotated input data is not sufficient. To obtain a more flexible offer of input data, higher sampling rates can be achieved with corresponding annotation effort, or by interpolating the hypothetical machine perception outputs that were already annotated with $1fps$. We chose the latter approach. The data manipulation tool can be used to interpolate the annotated input data with arbitrary rates. Although this feature is not used for the evaluations in Chapter 5, it could allow for the recognition of other situations that have faster dynamics, and for more finegrained temporal modeling. The $1fps$ data can contain large jumps which can cause loss of important data relations, e.g. directional speed. However, standard interpolation methods only work to a certain degree. Changes in direction between frames would still be lost for example. Higher sampling rates can also be required to obtain more suitable data for the subsequent addition of data gaps for example. Adding noise and other features of the data manipulation tool can be used with or without prior interpolation.

Other Manipulations. Five more manipulations are conceivable: outliers, confidence values (patterns of initial truth values associated with the input data), data gaps (missing atomic facts), superfluous atomic facts, and systematic errors that are likely to occur when working with real machine perception. Systematic errors include missing, superfluous, swapped, and drifting person tracks for example, and they can have characteristics of all the other error types described here. When working with real perception, these types of imperfect data can arise from noisy sensors, occlusions in the sensor data, areas without sensor coverage, technical problems with machine perception components, etcetera. Not all of these manipulations were fully implemented in the data manipulation tool and they were not used in the evaluations in Chapter 5. But Section 2.7.1 includes a discussion on how to handle such imperfections in the input data.

That concludes Chapter 3. First, the organisation and workflow within the control room was illustrated. Then, the audiovisual data recording process was explained, followed by the input data annotation methods and the ground-truth annotation methods. Finally, options for the manipulation of input data were discussed.

Chapter 4

Knowledge Base for Interaction Analysis

This chapter forms the core of this thesis, describing the SGTs, FMTL rules, and supporting methods that constitute the developed knowledge base for interaction analysis. The models are divided into the primary models (Section 4.1) and the secondary models (Section 4.2). They use logic reasoning to automatically deduce the ground-truth situation types listed in Section 3.4. Besides its use for the current case study and evaluation, this knowledge base is relevant to the field of high-level reasoning in general, and to many different application domains, such as the ones presented in Section 1.1.1.

After analyzing the data, axioms for group models were formulated on a natural language level and then formalized into SGTs and FMTL rules using common sense knowledge and empirical trials (defined by experts as opposed to learned from training data). Which numeric distances, angles, and speeds correspond to which semantic concepts was also determined through common sense and empirical trials, and by comparing the data to other real-world examples (e.g. from surveillance data and average pedestrian speeds). The primary models, originally published in [6], provide a detailed taxonomic representation of spatiotemporal concepts, interaction patterns, and group constellations that are commonly encountered in control room settings. The secondary models, originally published in [13], are based on the primary models. Their SGTs employ part of the FMTL interface and FMTL rule base from the primary models while other rules were changed or added.

The four primary models use the conventional person-as-agent approach, whereas the first and third secondary models use the cluster-as-agent approach through DBSCAN preprocessing. The difference will be discussed in Section 4.2. The second secondary model uses the person-as-agent approach, employing temporal modality at SGT level instead of the temporal modality at FMTL rule level employed by the other models. Furthermore, the first secondary models was optimized through parameter learning as described in Section 4.2.2. As we will see in Chapter 5, the evaluation of the secondary models is based on the evaluation performed on the primary models, with some interesting extensions: robustness against noisy data and inter-annotator-agreement.

4.1 Primary Models

The four primary models that are described below are used to recognize:

- groups with speaking and listening members (SGT 1),
- groups with sitting, standing, and moving members (SGT 2),
- groups with joining and leaving members (SGT 3), and
- groups oriented at a referred object (SGT 4).

Following from the data analysis and knowledge formalization process, these models represent groups in terms of the proximity of their members and the similarity between their orientations, their orientations toward each other, or orientations toward a mutual object or person. Two of the primary SGTs use this group model as their base, but they have different conceptual refinements. One of them models speech behavior across time within recognized groups and distinguishes between monologues, discussions, and silent groups. The other describes the sitting, standing, and moving members within groups. These refinements are again based on the observed data and on common sense knowledge. The third SGT uses the same group model as its base, but it extends it across time in order to recognize staff members that are joining and leaving groups. Following from the observed data and common sense knowledge, the fourth SGT applies a different strategy. It detects staff members that are referring to objects through pointing gestures and speech. Then, it selects the surrounding staff members that are oriented at the object or person that is being referred to. In Section 4.1.1, the four SGTs for the primary models are presented in detail. All of them use the same FMTL base, presented in Sections 4.1.2 through 4.1.5: interface specification, rule specifications, FMTL formulas, and trapezoid truth functions.

4.1.1 Primary SGTs

Figures 4.1 through 4.4 display the four SGTs for the primary models.[1] *All* FMTL rules that are directly or indirectly used by these four SGTs are listed and explained in Tables 4.1 (interface specification) and 4.2 (rule specifications in alphabetical order). Furthermore, the rules in Table 4.2 that are considered non-trivial and not too verbose are included as formulas in Section 4.1.4.[2] The applied truth value functions are included in Section 4.1.5.[3]

[1]These SGTs were rendered by the SGT-Editor; the same program that is used to edit the SGTs and to run the actual reasoning process.

[2]These formulas are the logic notation equivalent of the actual source code.

[3]These graphs are simply visualizations of further source code.

SGT 1:

- recognizes groups with speaking and listening members,
- is displayed in Figure 4.1, and
- generates outputs of the form:
  - ***group*** is listening to *person*,
  - conversation between ***persons*** in group ***g***, and
  - silent group ***g***.

All four primary SGTs have the same *Root* node, selecting a person as the agent for the current traversal (*SelectPersonAsAgent(p)*, Table 4.1). The first SGT recognizes groups and their speaking and listening members. Its *Group* node selects all persons other than the agent as possible patients (*SelectPersonsAsPatients(p,* ***qq****)*, Table 4.1) and filters them to only the ones that can be considered part of the same group as the agent, using *Filter(****qq****, InSameGroup(p,* elem*),* ***group****)* (Table 4.2 and Formula 4.5).

Then, the node *Silent* checks whether neither the agent nor any of the other persons in the group are speaking during a 5*s* interval, using *Not(InShortInterval(Speaking(p)))*, *Filter(****group****, InShortInterval(Speaking (*elem*)),* ***speakers****)*, and *Empty(****speakers****)* (Tables 4.1 and 4.2, Formula 4.25). If this is the case, *AppendHead(p,* ***group****,* ***group_****)* (Table 4.2) appends agent *p* to ***group***, yielding an output of the form "silent group *p q1 q2 q3*", where *p* is the agent for the current traversal and *q1, q2, q3* are three other persons that fulfill the described conditions. *InShortInterval(C)* uses a 5*s* interval, because this proved to be a suitable time frame for this situation. Similarly, rules can be implemented for *InMediumInterval(C)* and *InLongInterval(C)* for intervals of 7*s* and 9*s* respectively. To make this approach more flexible, one would have to implement a recursive rule called *InInterval(C, i)* where *i* is the length of the interval to consider. Formulas 4.50–4.52 provide another means to make the approach more flexible by applying an arbitrary temporal mask of length 5*s*.

In the other branch, the node *Not_silent* checks to what degree the agent and the other persons in the group are speaking during the same 5*s* interval, using the same FMTL rules. Then, if the agent (*p*) is speaking, the SGT checks whether at least one of the other persons in the group is also speaking: *Filter(****group****, InShortInterval(Speaking(*elem*)),* ***speakers****)* and *Not(Empty(****speakers****))* in node *Conversation.* If so, the SGT appends *p* to ***speakers*** and to ***group*** and outputs something of the form "conversation between *p q1* in group *p q1 q2 q3*". If none of the other persons in the group are speaking (*Empty(****speakers****)* in node *Monologue*), an output of the form "*q1 q2 q3* listening to *p*" is generated.

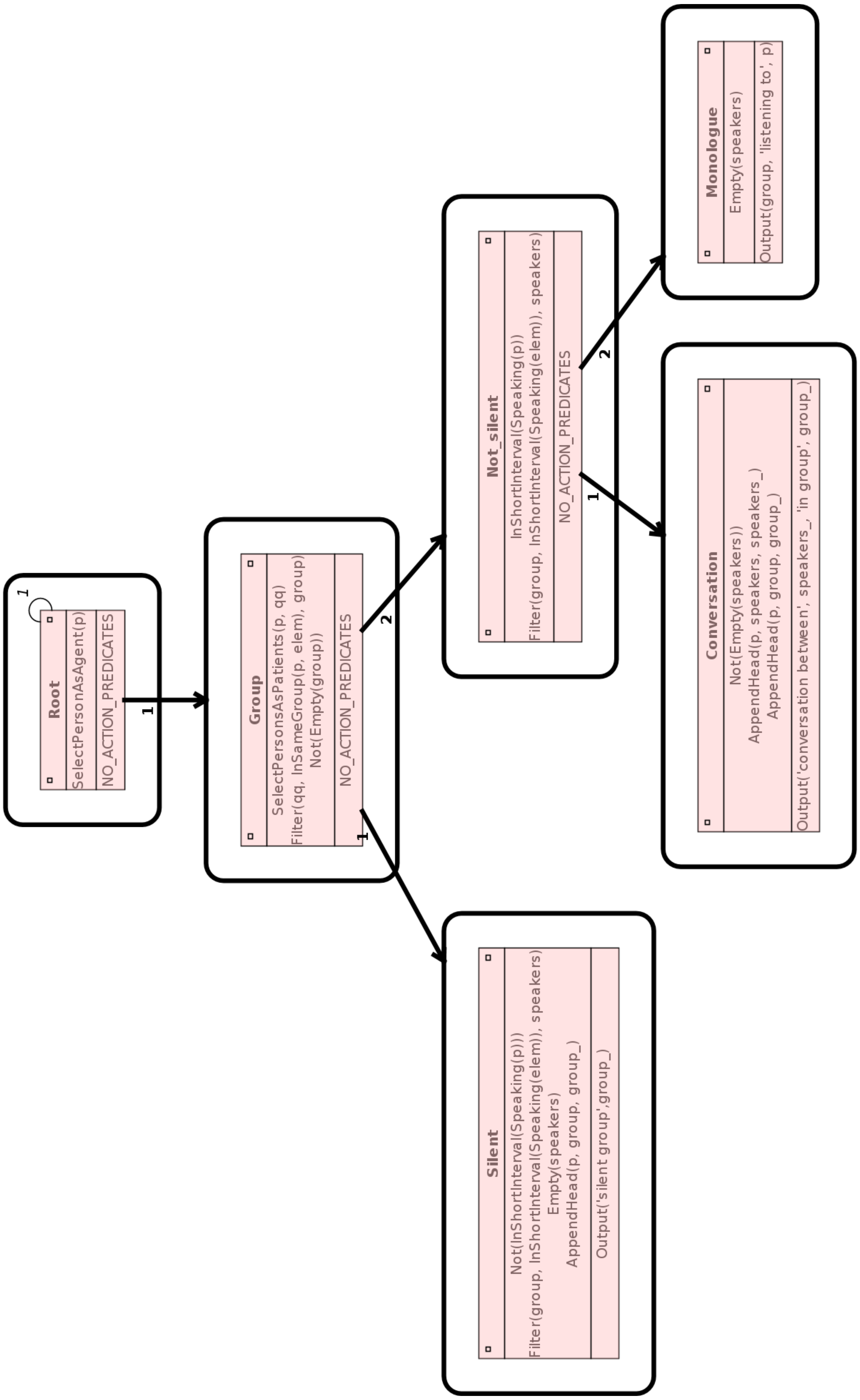

Root
SelectPersonAsAgent(p)
NO_ACTION_PREDICATES
1
1
Group
SelectPersonsAsPatients(p, qq)
Filter(qq, InSameGroup(p, elem), group)
Not(Empty(group))
NO_ACTION_PREDICATES
2
1
Not_silent
InShortInterval(Speaking(p))
Filter(group, InShortInterval(Speaking(elem)), speakers)
NO_ACTION_PREDICATES
2
1
Monologue
Empty(speakers)
Output(group, 'listening to', p)
Conversation
Not(Empty(speakers))
AppendHead(p, speakers, speakers_)
AppendHead(p, group, group_)
Output('conversation between', speakers_, 'in group', group_)
Silent
Not(InShortInterval(Speaking(p)))
Filter(group, InShortInterval(Speaking(elem)), speakers)
Empty(speakers)
AppendHead(p, group, group_)
Output('silent group',group_)

Figure 4.1: SGT 1 recognizes groups and their speaking and listening members.

SGT 2:

- recognizes groups with sitting, standing, and moving members,
- is displayed in Figure 4.2, and
- generates outputs of the form:
  - ***persons*** in group ***g*** are sitting,
  - ***persons*** in group ***g*** are standing, and
  - ***persons*** in group ***g*** are moving.

The second SGT recognizes groups and their sitting, standing, and moving members. Its *Root* and *Group* node are the same as in Figure 4.1, except for the additional condition *AppendHead(p,* ***group_,*** ***group****)* (Table 4.2). Then, the SGT splits into three branches. In the node *Sitting*, the agent's sitting property should hold (Table 4.1) and the persons in the group are filtered to only those that are sitting. The corresponding output is of the form "*p q1* in group *p q1 q2 q3* are sitting" (*p* is the current agent, *q1, q2, q3* are other persons fulfilling the appropriate conditions).

In the node *Standing*, the agent's sitting property should not hold, neither should he be moving (*Not(SpeedInShortInterval(p,* some*))*, Formula 4.15). Using *NegFilter(. . .)* (Table 4.2, the group is then filtered to only the members that are not sitting and not moving, i.e. the ones that are standing. The corresponding output is of the form "*p q1* in group *p q1 q2 q3* are standing". Finally, the node *Moving* uses a similar method to generate output of the form "*p q1* from group *a q1 q2 q3* are moving".

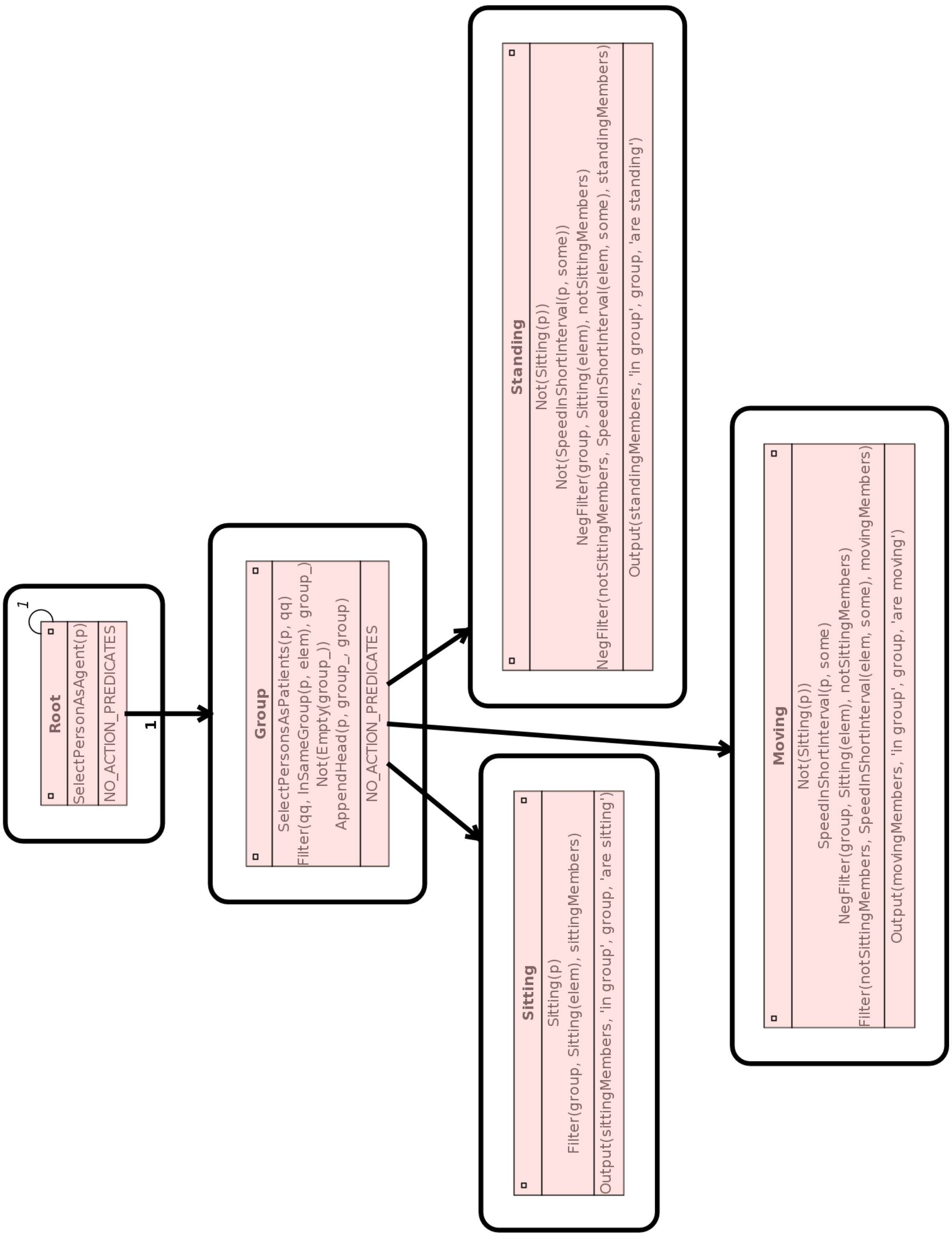
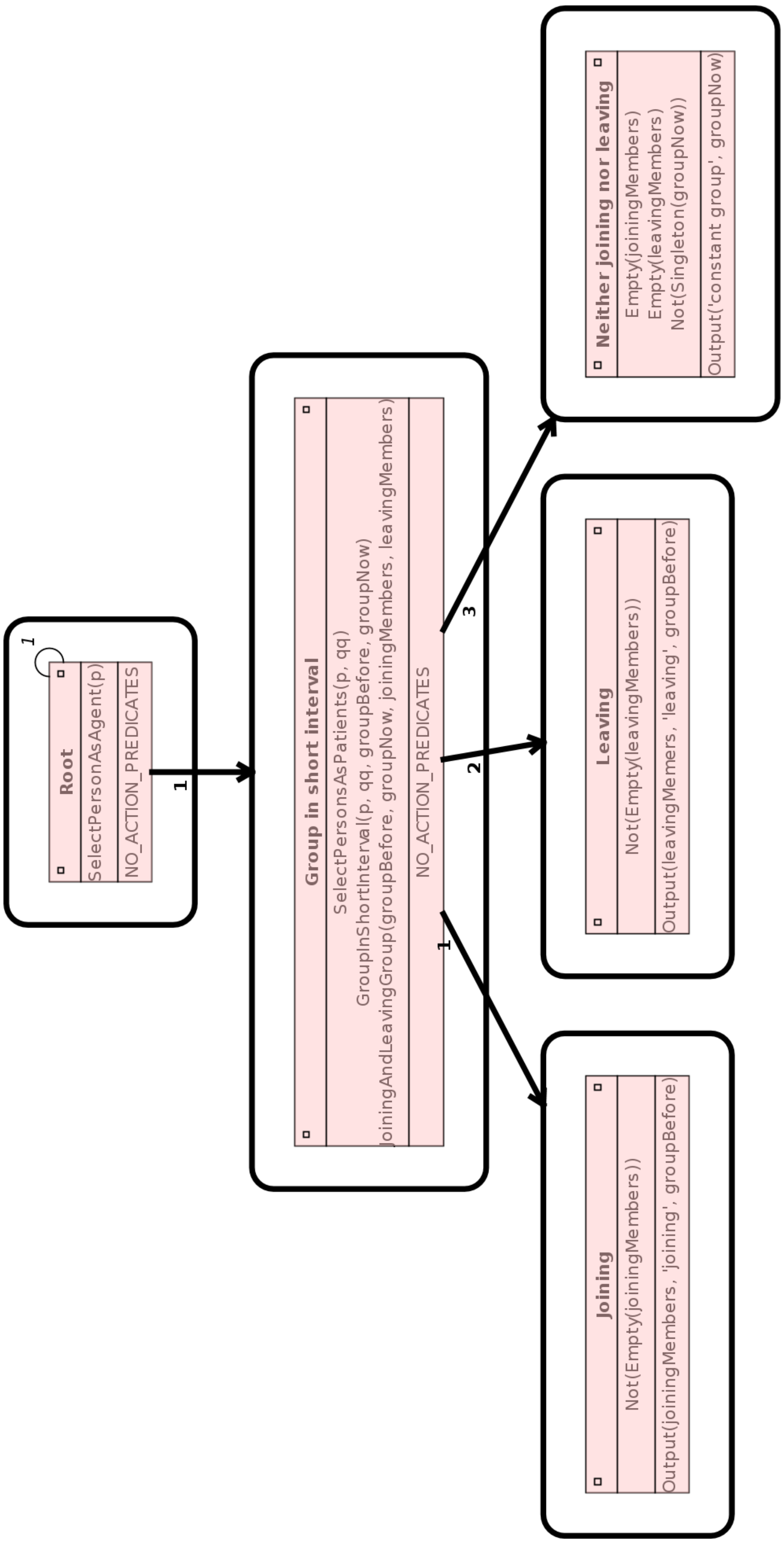

Root
SelectPersonAsAgent(p)
NO_ACTION_PREDICATES
1
1
Group
SelectPersonsAsPatients(p, qq)
Filter(qq, InSameGroup(p, elem), group_)
Not(Empty(group_))
AppendHead(p, group_, group)
NO_ACTION_PREDICATES
Standing
Not(Sitting(p))
Not(SpeedInShortInterval(p, some))
NegFilter(group, Sitting(elem), notSittingMembers)
NegFilter(notSittingMembers, SpeedInShortInterval(elem, some), standingMembers)
Output(standingMembers, 'in group', group, 'are standing')
Sitting
Sitting(p)
Filter(group, Sitting(elem), sittingMembers)
Output(sittingMembers, 'in group', group, 'are sitting')
Moving
Not(Sitting(p))
SpeedInShortInterval(p, some)
NegFilter(group, Sitting(elem), notSittingMembers)
Filter(notSittingMembers, SpeedInShortInterval(elem, some), movingMembers)
Output(movingMembers, 'in group', group, 'are moving')

Figure 4.2: SGT 2 recognizes groups and their sitting, standing, and moving members.

SGT 3:

- recognizes groups with joining and leaving members,
- is displayed in Figure 4.3, and
- generates outputs of the form:
  - ***persons*** are joining ***group***,
  - ***persons*** are leaving ***group***, and
  - constant group ***g***.

The third SGT (Figure 4.3) starts with the usual *Root* node and a node called *Group in short interval.* It uses *GroupInShortInterval(p,* ***qq****,* ***groupBefore****,* ***groupNow****)* (Formula 4.3) to detect the group around agent p during the previous and current frame. Then, it uses *JoiningAndLeavingGroup(****groupBefore****,* ***groupNow****,* ***joiningMembers****,* ***leavingMembers****)* (Formula 4.6) to determine which members join and leave the group during this time. If the list of joining members is not empty, the node *Joining* outputs something like "*q1* joining *p q2 q3*". The node called *Leaving* does the same for leaving members, e.g. "*q1* leaving *p q2 q3*". Finally, in *Neither joining nor leaving*, output of the form "constant group *p q1 q2 q3*" is generated if *Empty(****joiningMembers****)* and *Empty(****leavingMembers****)*.



Root
SelectPersonAsAgent(p)
NO_ACTION_PREDICATES
1
1
Group in short interval
SelectPersonsAsPatients(p, qq)
GroupInShortInterval(p, qq, groupBefore, groupNow)
JoiningAndLeavingGroup(groupBefore, groupNow, joiningMembers, leavingMembers)
NO_ACTION_PREDICATES
1
2
3
Joining
Not(Empty(joiningMembers))
Output(joiningMembers, 'joining', groupBefore)
Leaving
Not(Empty(leavingMembers))
Output(leavingMemers, 'leaving', groupBefore)
Neither joining nor leaving
Empty(joiningMembers)
Empty(leavingMembers)
Not(Singleton(groupNow))
Output('constant group', groupNow)

Figure 4.3: SGT 3 recognizes groups and their joining and leaving members.

SGT 4:

- recognizes groups oriented at a referred object,
- is displayed in Figure 4.4, and
- generates outputs of the form:
  - *person* is pointing at *object*,
  - *person* is pointing at *object* and ***group*** is looking at it,
  - *person* is speaking and pointing at *object*, and
  - *person* is speaking and pointing at *object* and ***group*** is looking at it.

The fourth and last of the primary SGTs recognizes groups that are gathered around an object that is being referred to. After the *Root* node, it uses *PointingAtNearbyObject(p, q, c)* (Formula 4.2) to determine whether the agent is pointing at a nearby object. Then, the nodes *Not speaking* and *Speaking* use *InShortInterval(Speaking(p))* (Formula 4.25) to determine whether the agent is speaking in a $5s$ interval, generating the corresponding output "p pointing at q" or "p pointing at q and speaking". Instead of a $5s$ interval, one could use $7s$ or $9s$, or one could define a more flexible version of *InShortInterval(C)* that takes the desired length of the interval as an argument.

The nodes *Looking 1* and *Looking 2* are identical, except for their outputs. They use *DistBetweenInnerPerimeters(. . .)* (Formula 4.12) to find the persons close to the agent and *LookingAtNearbyObject(. . .)* (Formula 4.1) to filter them to only the ones that are looking at the object that the agent is pointing at. Their outputs are of the form "p pointing at q and *r1 r2 r3* looking at it" and "p pointing at q and speaking and *r1 r2 r3* looking at it" respectively.

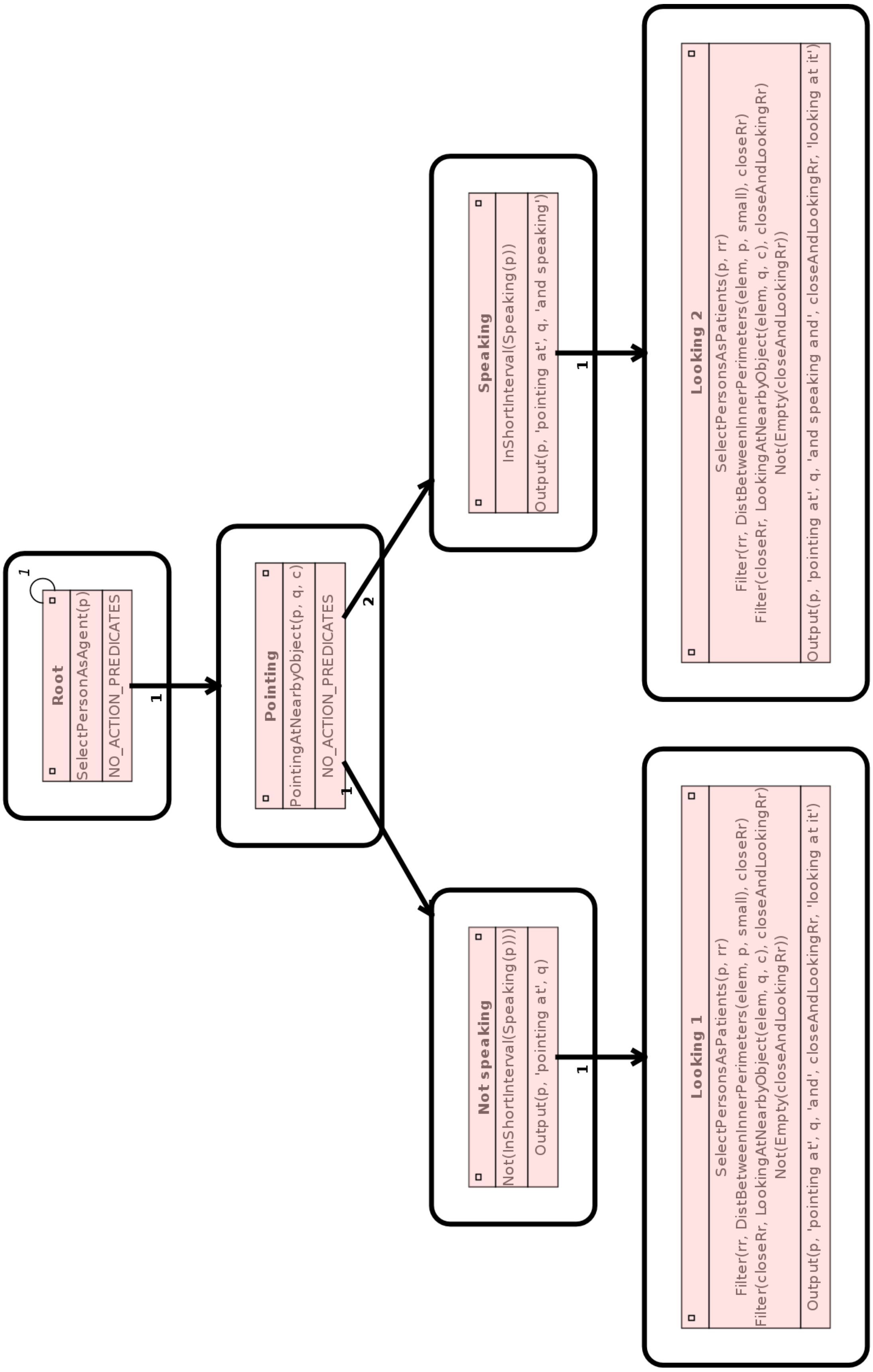

Root
SelectPersonAsAgent(p)
NO_ACTION_PREDICATES
1
1
Pointing
PointingAtNearbyObject(p, q, c)
NO_ACTION_PREDICATES
2
1
Speaking
InShortInterval(Speaking(p))
Output(p, 'pointing at', q, 'and speaking')
1
Looking 2
SelectPersonsAsPatients(p, rr)
Filter(rr, DistBetweenInnerPerimeters(elem, p, small), closeRr)
Filter(closeRr, LookingAtNearbyObject(elem, q, c), closeAndLookingRr)
Not(Empty(closeAndLookingRr))
Output(p, 'pointing at', q, 'and speaking and', closeAndLookingRr, 'looking at it')
Not speaking
Not(InShortInterval(Speaking(p)))
Output(p, 'pointing at', q)
1
Looking 1
SelectPersonsAsPatients(p, rr)
Filter(rr, DistBetweenInnerPerimeters(elem, p, small), closeRr)
Filter(closeRr, LookingAtNearbyObject(elem, q, c), closeAndLookingRr)
Not(Empty(closeAndLookingRr))
Output(p, 'pointing at', q, 'and', closeAndLookingRr, 'looking at it')

Figure 4.4: SGT 4 recognizes groups oriented at an object that is being referred to.

4.1.2 Interface Specification

The interface that connects the primary models to the world outside the reasoning engine is provided in Table 4.1. The interface contains rules for selecting persons and objects as agents and patients during reasoning, rules that read the input data and translate it into the atomic facts that the reasoning process is grounded in, and a rule for generating output, i.e. the situation descriptions that must be compared to the ground-truth. Most of this interface is also used by the secondary models described in Section 4.2.

Table 4.1: The interface specification for the primary models (Section 4.1); rules for selecting persons and objects as agents and patients, rules for obtaining atomic facts from the input data, and a rule for generating output. The secondary models explained in Section 4.2 also use most of this interface.

Rule head	**Explanation**
SelectPersonAsAgent(p)	Select person p as center of reasoning (i.e. agent) for current traversal – *Type(p,*person*)*.
SelectPatient(p, q)	Select person/object q as reasoning subject (i.e. patient) with agent p – $q \neq p$.
SelectPatients(p, $\boldsymbol{q}$*)*	Select list $\boldsymbol{q}$ containing all patients for agent p – $q \in \boldsymbol{q} \Leftrightarrow q \neq p$.
SelectPersonsAsPatients(p, $\boldsymbol{q}$*)*	Select list $\boldsymbol{q}$ with all patients of type person for agent p – $q \in \boldsymbol{q} \Leftrightarrow q \neq p \wedge$ *Type(q,* person*)*.
Type(p, t)	Query type t for person/object p.
Position(p, x, y)	Query position x, y for person/object p.
Size(p, w, h)	Query size w, h (horizontal plane) for person/object p.
Orientation(p, o)	Query horizontal orientation o for person/object p.
Geometry(p, x, y, w, h, o)	Query geometry x, y, w, h, o for person/object p.
Speaking(p)	Query if person p is speaking.
Gesticulating(p)	Query if person p is gesticulating.
ExtendingArm(p, o)	Query hor. orientation o of person p's extended arm.
LookingDown(p)	Query if person p is looking down.
Sitting(p)	Query if person p is sitting.
Output(a, b, ...)	Output string $a + b + \ldots$ (arbitrary nr. of arguments).

4.1.3 Rule Specifications

Table 4.2 provides an alphabetically sorted overview of *all* the FMTL rules that are directly or indirectly used by SGTs 1 through 4 (Figures 4.1 through 4.4). Their use is explained in Section 4.1.1. The rules that are considered non-trivial and not too verbose are included as formulas in Section 4.1.4.

Table 4.2: The rule specifications for the primary models; *all* rules that are directly or indirectly used by SGTs 1 through 4 (Figures 4.1 through 4.4) in alphabetical order. Some of these rules are also used by the secondary models described in Section 4.2.

Rule head	**Explanation**
AbsOrientationOfExtended-Arm(p, o)	Orientation o (in $^\circ$) is the absolute orientation of person p's extended arm (Formula 4.9).
*AppendHead(p, **q**, **r**)*	Append element p to the head of list $\boldsymbol{q}$, yielding list $\boldsymbol{r}$ ($\boldsymbol{r} = \{p\} \cup \boldsymbol{q}$).
AngleBetween-Points(x1, y1, x2, y2, a)	Angle a is the angle of the line between points *(x1, y1)* and *(x2, y2)* (Formula 4.23).
AssocDiffBetweenOriAnd-AngleToCenter(d, c)	Associate difference d (in $^\circ$) with difference category c, where d is the difference between the orientation of a person/object and the angle of the line between its center and another person's/object's center. Category c can either be instantiated (e.g. with "small") which returns an appropriate truth value, or uninstantiated, allowing the rule to return a truth value > 0 for each defined and appropriate category (e.g. more than one of "no, small, medium, large") (Section 4.1.5).
AssocDiffBetween-Oris(d, c)	Associate difference d (in $^\circ$) with difference category c, where d is the difference between the orientations of two persons/objects. Category c can either be instantiated or uninstantiated (Section 4.1.5).
AssocDistBetween-Centers(d, c)	Associate distance d (in cm) with distance category c, where d is the distance between the centers of two persons/objects (Section 4.1.5).
AssocDistBetween-InnerPerimeters(d, c)	Associate distance d (in cm) with distance category c, where d is the distance between the inner perimeters of two persons/objects (Section 4.1.5).
AssocSpeedIn-ShortInterval(v, c)	Associate speed v (in cm/s) with speed category c, where v is the speed of a person/object over $3s$ (Section 4.1.5).
Atanoid(dx, dy, a)	Angle a is the slope for two perpendicular lines dx and dy, like *atan2* (arctangent for signed inputs), but in a different coordinate system.

Continues next page.

Continued from previous page.

BuildGroup(p, $\boldsymbol{q}$, $\boldsymbol{r}$)	List $\boldsymbol{r}$ contains person p and all persons $q \in \boldsymbol{q}$ that fulfill *InSameGroup(p, q)* ($\boldsymbol{r} = \{\text{p}\} \cup \{q \in \boldsymbol{q} : \textit{InSameGroup(p, q)}\}$, Formula 4.4).
CalcDiffBetweenOriAndAngleToCenter(p, q, d)	Calculate difference d (in $^\circ$) between the orientation of person/object p and the angle of the line between p's center and another person/object q's center (Formula 4.19).
CalcDiffBetweenOris(p, q, d)	Calculate difference d (in $^\circ$) between the orientations of two persons/objects p and q (Formula 4.18).
CalcDistBetweenCenters(p, q, d)	Calculate distance d (in cm) between the centers of persons/objects p and q (Formula 4.16).
CalcDistBetweenInnerPerimeters(p, q, d)	Calculate distance d (in cm) between the inner perimeters of persons/objects p and q (Formula 4.17).
CalcSpeedInShortInterval(p, v)	Calculate speed v of person/object p, using a $3s$ interval (Formula 4.20).
Call(C)	Call rule head C and return its truth value. This is used to call rules dynamically at runtime.
DiffBetweenAngles(a, b, d)	Calculate difference d (in $^\circ$) between angles a and b (Formula 4.22).
DiffBetweenAngles(x1, y1, x2, y2, b,d)	Calculate difference d (in $^\circ$) between the angle of the line spanning points *(x1, y1)* and *(x2, y2)*, and angle b (Formula 4.24).
DiffBetweenOriAndAngleToCenter(p, q, c)	Calculate the difference between the orientation of person/object p and the angle of the line between p's center and another person/object q's center. Then associate this difference with difference category c (Formula 4.14).
DiffBetweenOris(p, q, c)	Calculate the difference between the orientations of persons/objects p and q and associate this difference with difference category c (Formula 4.13).
Difference($\boldsymbol{p}$, $\boldsymbol{q}$, $\boldsymbol{r}$)	List $\boldsymbol{r}$ is the set difference of lists $\boldsymbol{p}$ and $\boldsymbol{q}$ ($\boldsymbol{r} = \boldsymbol{p} \setminus \boldsymbol{q}$); $\boldsymbol{r}$ contains all members of $\boldsymbol{p}$ that are not members of $\boldsymbol{q}$.
DistBetweenCenters(p, q, c)	Calculate the distance between the centers of persons/objects p and q and associate this distance with distance category c (Formula 4.11).
DistBetweenInnerPerimeters(p, q, c)	Calculate the distance between the inner perimeters of persons/objects p and q and associate this distance with distance category c (Formula 4.12).
DistBetweenPoints(x1, y1, x2, y2, d)	Calculate distance d (in cm) between points *(x1, y1)* and *(x2, y2)* (Formula 4.21).
Empty($\boldsymbol{p}$)	Determine whether $\boldsymbol{p}$ is an empty list ($\boldsymbol{p} = \emptyset$).
Filter($\boldsymbol{p}$, C, $\boldsymbol{q}$)	Filter list $\boldsymbol{p}$, applying rule head C to its elements. List $\boldsymbol{q}$ contains the elements of $\boldsymbol{p}$ that fulfill rule head C ($\boldsymbol{q} = \{p \in \boldsymbol{p} : p$ fulfills $C\}$). The truth value returned by *Filter($\boldsymbol{p}$, C, $\boldsymbol{q}$)* is the average of the truth values returned by rule head C applied to the elements of $\boldsymbol{p}$. If none fulfill C, $\boldsymbol{q} = \emptyset$, and *Filter($\boldsymbol{p}$, C, $\boldsymbol{q}$)* returns truth value 1.0. Rule head C has the constant "elem" as one of its arguments, which is a placeholder for the elements of $\boldsymbol{p}$.

Continues next page.

Continued from previous page.

*GroupInShortInterval(p, **q**, **r**, **s**)*	List $\boldsymbol{r}$ contains person p and all persons in list $\boldsymbol{q}$ that fulfill *BuildGroup(p, **q**, **r**)* at $t = t_{current} - 1$. List $\boldsymbol{s}$ contains person p and all persons in list $\boldsymbol{q}$ that fulfill *BuildGroup(p, **q**, **s**)* at $t = t_{current}$ (Formula 4.3).
InSameGroup(p, q)	Persons/objects p and q are in the same group if the distance between their centers is small and, either q is oriented at p, or p and q have the same orientation, or they are oriented at the same person/object r (Formula 4.5).
InShortInterval(C)	Call rule head C for every second in interval $[t_{current}-2, t_{current}+2]$ and return a truth value that is proportional to the number of seconds that C is fulfilled (Formula 4.25).
*Intersection(**p**, **q**, **r**)*	List $\boldsymbol{r}$ is the set intersection of lists $\boldsymbol{p}$ and $\boldsymbol{q}$ ($\boldsymbol{r} = \boldsymbol{p} \cap \boldsymbol{q}$); $\boldsymbol{r}$ contains all elements that are in $\boldsymbol{p}$ as well as in $\boldsymbol{q}$.
*JoiningAndLeavingGroup(**p**, **q**, **r**, **s**)*	Given a group at $t = t_1$ ($\boldsymbol{p}$) and at $t = t_2$ ($\boldsymbol{q}$), lists $\boldsymbol{r}$ and $\boldsymbol{s}$ respectively contain the members that joined and left the group between t_1 and t_2 (Formula 4.6).
LookingAt(p, q)	Determine whether person p is looking at person/object q (Formula 4.7).
LookingAtNearbyObject(p, q, c)	Determine whether person p is looking at person/object q and associate the distance between p and q with distance category c (Formula 4.1).
MaxAndMin(a, b, max, min)	Given two numbers a and b, determine their maximum and minimum.
*NegFilter(**p**, C, **q**)*	Filter list $\boldsymbol{p}$, applying rule head C to its elements. List $\boldsymbol{q}$ contains the elements of $\boldsymbol{p}$ that *do not* fulfill rule head C ($\boldsymbol{q} = \{p \in \boldsymbol{p} : \neg(p \text{ fulfills } C)\}$). *NegFilter(**p**, C, **q**)* returns truth value 1.0. Rule head C has the constant "elem" as one of its arguments, which is a placeholder for the elements of $\boldsymbol{p}$.
NormalizeAngle(a, b)	Angle b is the normalized equivalent of angle a.
Not(C)	Invert the truth value returned by rule head C (the F-Limette equivalent of "$\neg$").
OrientedAtSame(p, q, r)	Determine to what degree persons p and q are oriented at the same person/object r, depending on their distances to r as well as their orientations relative to r (Formula 4.10).
PointingAt(p, q)	Determine whether person p is pointing at person/object q (Formula 4.8).
PointingAtNearbyObject(p, q, c)	Determine whether person p is pointing at person/object q and associate the distance between p and q with distance category c (Formula 4.2).
RayHitsObject(p, o, q)	Determine whether the ray emanating from person p's center, and following his orientation o, hits object q.
ReturnTruthValue(v)	When used inside another rule, that rule returns truth value v.
*Singleton(**p**)*	Determine whether $\boldsymbol{p}$ is a list containing only one element.
SpeedInShortInterval(p, c)	Calculate speed of person/object p in a $3s$ interval and associate this speed with speed category c (Formula 4.15).

4.1.4 FMTL Formulas

This section lists the FMTL rules that play a central role during reasoning with the primary models (their use is explained in Section 4.1.1). Many of the conditions in the formulas below are included as formulas themselves, and *all* rule heads and conditions are listed with a brief explanation in Tables 4.1 and 4.2. The trapezoid truth functions that are used by Formulas 4.11 through 4.15 (starting with $Assoc\dots$) are displayed in Figure 4.5. Note that the formulas in this section are the logic notation equivalents of the actual source code. To show the ease of this translation step, the F-Limette source code for Formulas 4.1 through 4.3 is included below the last formula (capitalization is inverted).

$$\begin{aligned}&\mathit{LookingAtNearbyObject}(p,q,c) \leftarrow \\ &\quad \mathit{LookingAt}(p,q) \wedge \mathit{DistBetweenInnerPerimeters}(p,q,c)\end{aligned} \tag{4.1}$$

$$\begin{aligned}&\mathit{PointingAtNearbyObject}(p,q,c) \leftarrow \\ &\quad \mathit{PointingAt}(p,q) \wedge \\ &\quad \mathit{CalcDistBetweenInnerPerimeters}(p,q,d) \wedge \\ &\quad [c = \text{small} \vee (\neg \mathit{LookingDown}(p) \wedge c = \text{notLarge}) \vee \\ &\qquad (\neg \mathit{LookingDown}(p) \wedge \mathit{Type}(q, \text{display}) \wedge c = \text{any})] \wedge \\ &\quad \mathit{AssocDistBetweenInnerPerimeters}(d,c)\end{aligned} \tag{4.2}$$

$$\begin{aligned}&\mathit{GroupInShortInterval}(p,\boldsymbol{q},\boldsymbol{r},\boldsymbol{s}) \leftarrow \\ &\quad \Diamond_{-1} \mathit{BuildGroup}(p,\boldsymbol{q},\boldsymbol{r}) \wedge \mathit{BuildGroup}(p,\boldsymbol{q},\boldsymbol{s})\end{aligned} \tag{4.3}$$

$$\begin{aligned}&\mathit{BuildGroup}(p,\boldsymbol{q},\boldsymbol{r}) \leftarrow \\ &\quad \mathit{Filter}(\boldsymbol{q}, \mathit{InSameGroup}(p,\text{elem}), \boldsymbol{r'}) \wedge \mathit{AppendHead}(p,\boldsymbol{r'},\boldsymbol{r})\end{aligned} \tag{4.4}$$

$$\begin{aligned}&\mathit{InSameGroup}(p,q) \leftarrow \\ &\quad \mathit{DistBetweenCenters}(p,q,\text{small}) \wedge \\ &\quad [\mathit{DiffBetweenOrientationAndAngleToCenter}(q,p,\text{small}) \vee \\ &\qquad \mathit{DiffBetweenOrientations}(q,p,\text{small}) \vee \\ &\qquad \mathit{OrientedAtSame}(p,q,r)]\end{aligned} \tag{4.5}$$

$$\begin{aligned}&\mathit{JoiningAndLeavingGroup}(\boldsymbol{p},\boldsymbol{q},\boldsymbol{r},\boldsymbol{s}) \leftarrow \\ &\quad \mathit{Intersection}(\boldsymbol{p},\boldsymbol{q},\boldsymbol{t}) \wedge \mathit{Difference}(\boldsymbol{q},\boldsymbol{t},\boldsymbol{r}) \wedge \mathit{Difference}(\boldsymbol{p},\boldsymbol{t},\boldsymbol{s})\end{aligned} \tag{4.6}$$

$$\begin{aligned}&\mathit{LookingAt}(p,q) \leftarrow \\ &\quad p \neq q \wedge \mathit{Orientation}(p,o_p) \wedge \mathit{RayHitsObject}(p,o_p,q)\end{aligned} \tag{4.7}$$

$$\begin{aligned}&\mathit{PointingAt}(p,q) \leftarrow \\ &\quad p \neq q \wedge \mathit{AbsOrientationOfExtendedArm}(p,o_{arm_p}) \wedge \\ &\quad \mathit{RayHitsObject}(p,o_{arm_p},q)\end{aligned} \tag{4.8}$$

$$\begin{aligned}&\mathit{AbsOrientationOfExtendedArm}(p,o_{arm_p}) \leftarrow \\ &\quad \mathit{Orientation}(p,o_p) \wedge \mathit{ExtendingArm}(p,o_{arm}) \wedge \\ &\quad o'_{arm_p} = o_p + o_{arm} \wedge \mathit{NormalizeAngle}(o'_{arm_p}, o_{arm_p})\end{aligned} \tag{4.9}$$

$$\begin{aligned}&\mathit{OrientedAtSame}(p,q,r) \leftarrow \\ &\quad \mathit{SelectPatient}(p,r) \wedge \mathit{SelectPatient}(q,r) \wedge \\ &\quad \mathit{DistBetweenCenters}(p,r,\text{notLarge}) \wedge \\ &\quad \mathit{DistBetweenCenters}(q,r,\text{notLarge}) \wedge \\ &\quad \mathit{LookingAt}(p,r) \wedge \mathit{LookingAt}(q,r)\end{aligned} \tag{4.10}$$

$$\begin{aligned}&\mathit{DistBetweenCenters}(p,q,c) \leftarrow \\ &\quad \mathit{CalcDistBetweenCenters}(p,q,d)\wedge \\ &\quad \mathit{AssocDistBetweenCenters}(d,c)\end{aligned} \tag{4.11}$$

$$\begin{aligned}&\mathit{DistBetweenInnerPerimeters}(p,q,c) \leftarrow \\ &\quad \mathit{CalcDistBetweenInnerPerimeters}(p,q,d)\wedge \\ &\quad \mathit{AssocDistBetweenInnerPerimeters(d,\ c)}\end{aligned} \tag{4.12}$$

$$\begin{aligned}&\mathit{DiffBetweenOris}(p,q,c) \leftarrow \\ &\quad \mathit{CalcDiffBetweenOris}(p,q,d)\wedge \\ &\quad \mathit{AssocDiffBetweenOris}(d,c)\end{aligned} \tag{4.13}$$

$$\begin{aligned}&\mathit{DiffBetweenOriAndAngleToCenter}(p,q,c) \leftarrow \\ &\quad \mathit{CalcDiffBetweenOrientationAndAngleToCenter}(p,q,d)\wedge \\ &\quad \mathit{AssocDiffBetweenOrientationAndAngleToCenter}(d,c)\end{aligned} \tag{4.14}$$

$$\begin{aligned}&\mathit{SpeedInShortInterval}(p,c) \leftarrow \\ &\quad \mathit{CalcSpeedInShortInterval}(p,v)\wedge \\ &\quad \mathit{AssocSpeedInShortInterval}(v,c)\end{aligned} \tag{4.15}$$

$$\begin{aligned}&\mathit{CalcDistBetweenCenters}(p,q,d) \leftarrow \\ &\quad \mathit{Position}(p,x_p,y_p) \wedge \mathit{Position}(q,x_q,y_q)\wedge \\ &\quad \mathit{DistBetweenPoints}(x_p,y_p,x_q,y_q,d)\end{aligned} \tag{4.16}$$

$$\begin{aligned}&\mathit{CalcDistBetweenInnerPerimeters}(p,q,d) \leftarrow \\ &\quad \mathit{CalcDistBetweenCenters}(p,q,d_{pq})\wedge \\ &\quad \mathit{Size}(p,w_p,h_p) \wedge \mathit{Size}(q,w_q,h_q)\wedge \\ &\quad \mathit{MaxAndMin}(w_p,h_p,d_{p_{out}},d_{p_{in}})\wedge \\ &\quad \mathit{MaxAndMin}(w_q,h_q,d_{q_{out}},d_{q_{in}})\wedge \\ &\quad d = d_{pq} - 0.5\ d_{p_{in}} - 0.5\ d_{q_{in}}\end{aligned} \tag{4.17}$$

$$\begin{aligned}&\mathit{CalcDiffBetweenOris}(p,q,d) \leftarrow \\ &\quad \mathit{Orientation}(p,o_p) \wedge \mathit{Orientation}(q,o_q)\wedge \\ &\quad \mathit{DiffBetweenAngles}(o_p,o_q,d)\end{aligned} \tag{4.18}$$

$$\begin{aligned}&\mathit{CalcDiffBetweenOriAndAngleToCenter}(p,q,d) \leftarrow \\ &\quad \mathit{Position}(p,x_p,y_p) \wedge \mathit{Position}(q,x_q,y_q)\wedge \\ &\quad \mathit{Orientation}(p,o_p) \wedge \mathit{DiffBetweenAngles}(x_p,y_p,x_q,y_q,o_p,d)\end{aligned} \tag{4.19}$$

$$\begin{aligned}&\mathit{CalcSpeedInShortInterval}(p,v) \leftarrow \\ &\quad \Diamond_{-1}\ \mathit{Position}(p,x_{-1},y_{-1})\wedge \\ &\quad \qquad \mathit{Position}(p,x_0,y_0)\wedge \\ &\quad \Diamond_{1}\ \ \mathit{Position}(p,x_1,y_1)\wedge \\ &\quad \mathit{DistBetweenPoints}(x_{-1},y_{-1},x_0,y_0,d_{-1,0})\wedge \\ &\quad \mathit{DistBetweenPoints}(x_0,y_0,x_1,y_1,d_{0,1})\wedge \\ &\quad v = 0.5\ (d_{-1,0} + d_{0,1})\end{aligned} \tag{4.20}$$

$$\begin{aligned}&\mathit{DistBetweenPoints}(x_1, y_1, x_2, y_2, d) \leftarrow \\ &\quad d_x = x_2 - x_1 \wedge d_y = y_2 - y_1 \wedge d = \sqrt{d_x^2 + d_y^2}\end{aligned} \tag{4.21}$$

$$\begin{aligned}&\mathit{DiffBetweenAngles}(x_1, y_1, x_2, y_2, a_1, d) \leftarrow \\ &\quad \mathit{AngleBetweenPoints}(x_1, y_1, x_2, y_2, a_2) \wedge \\ &\quad \mathit{DiffBetweenAngles}(a_1, a_2, d)\end{aligned} \tag{4.22}$$

$$\begin{aligned}&\mathit{AngleBetweenPoints}(x_1, y_1, x_2, y_2, a) \leftarrow \\ &\quad d_x = x_2 - x_1 \wedge d_y = y_2 - y_1 \wedge \mathit{Atanoid}(d_x, d_y, a)\end{aligned} \tag{4.23}$$

$$\begin{aligned}&\mathit{DiffBetweenAngles}(a_1, a_2, d) \leftarrow \\ &\quad \mathit{MaxAndMin}(a_1, a_2, a_{max}, a_{min}) \wedge d' = a_{max} - a_{min} \wedge \\ &\quad \mathit{NormalizeAngle}(d', d)\end{aligned} \tag{4.24}$$

$$\begin{aligned}&\mathit{InShortInterval}(C) \leftarrow \\ &\quad [\Box_{-2,2}^{100\%} \mathit{Call}(C) \wedge \mathit{ReturnTruthValue}(1.0) \wedge !] \vee \\ &\quad [\Box_{-2,2}^{80\%} \mathit{Call}(C) \wedge \mathit{ReturnTruthValue}(0.8) \wedge !] \vee \\ &\quad [\Box_{-2,2}^{60\%} \mathit{Call}(C) \wedge \mathit{ReturnTruthValue}(0.6) \wedge !] \vee \\ &\quad [\Box_{-2,2}^{40\%} \mathit{Call}(C) \wedge \mathit{ReturnTruthValue}(0.4) \wedge !] \vee \\ &\quad [\Box_{-2,2}^{20\%} \mathit{Call}(C) \wedge \mathit{ReturnTruthValue}(0.2)]\end{aligned} \tag{4.25}$$

```
// Source code for Formula 4.1:
always(lookingAtNearbyObject(P, Q, C) :-
    lookingAt(P, Q) , distBetweenInnerPerimeters(P, Q, C)
).

// Source code for Formula 4.2:
always(pointingAtNearbyObject(P, Q, C) :-
    pointingAt(P, Q) ,
    calcDistBetweenInnerPerimeters(P, Q, D) ,
    (C = small ; (not(lookingDown(P)) , C = notLarge) ;
        (not(lookingDown(P)) , type(Q, display) , C = any)) ,
    assocDistBetweenInnerPerimeters(D, C)
).

// Source code for Formula 4.3:
always(groupInShortInterval(P, Qs, Rs, Ss) :-
    -1 ! buildGroup(P, Qs, Rs) , buildGroup(P, Qs, Ss)
).
```

4.1.5 Trapezoid Truth Functions

The trapezoid truth functions (visualizations of the corresponding source code) that are used by Formulas 4.11 through 4.15 (starting with *Assoc...*) are displayed in Figure 4.5. They associate distances d in cm, angular differences d in $^\circ$, and speeds v in cm/s with appropriate categories. As values on the x-axes increase, the truth values V on the y-axes (for the corresponding categories) increase from 0.0 to 1.0, stay constant at 1.0, and then decrease back to 0.0. The categories that are applied by the primary models have boldfaced labels and colored curves.

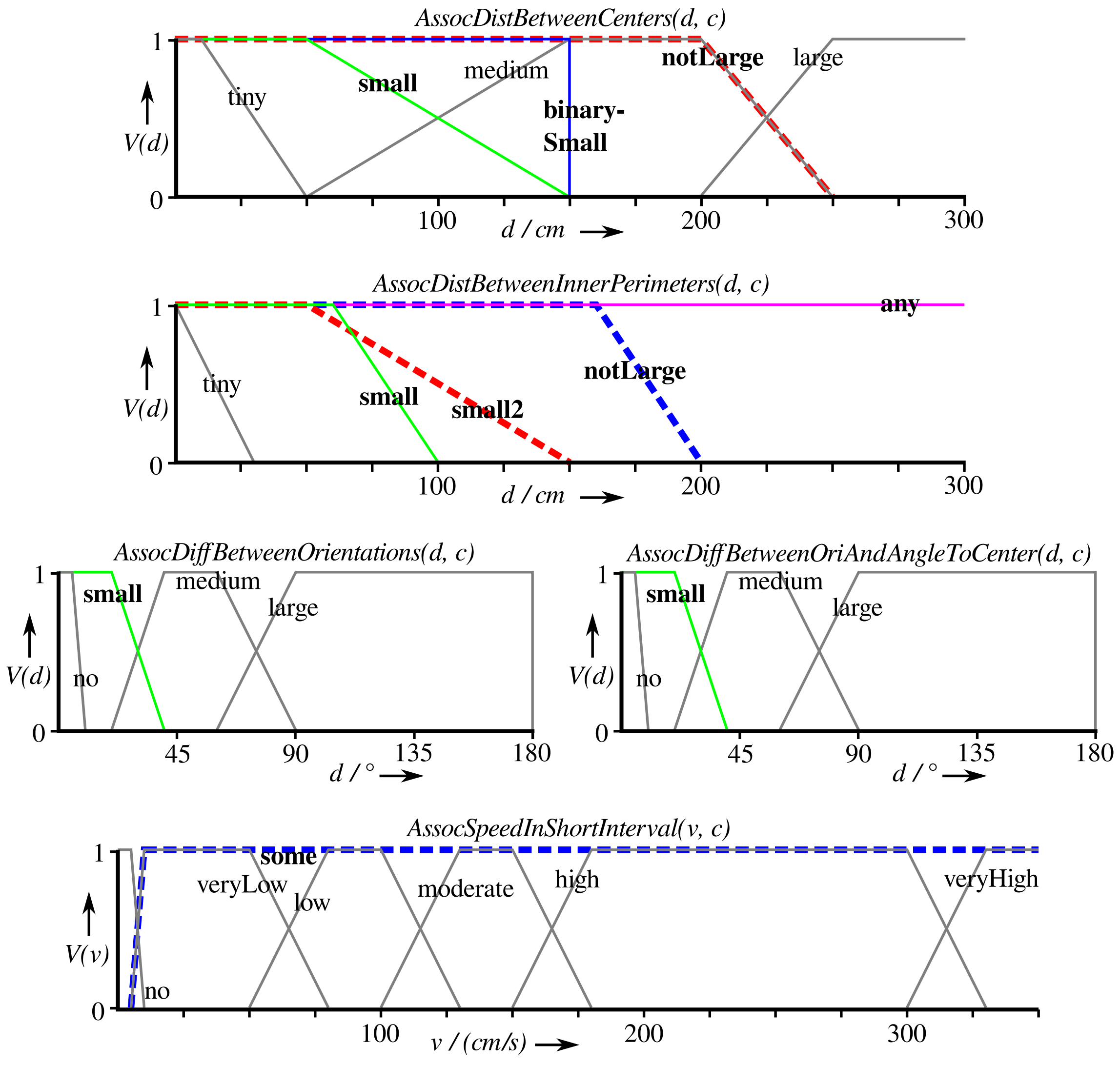


Figure 4.5: The trapezoid truth functions used by Formulas 4.11 through 4.15. They associate distances d in cm, angular differences d in $^\circ$, and speeds v in cm/s with appropriate categories. The categories that are applied by the primary models have boldfaced labels and colored curves.

The formulas for *AssocDistBetweenCenters(d, c)* (top left in Figure 4.5) are included below. These are used in combination with Formulas 4.11 and 4.16. $SP(v)$ in Formula 4.27 is a built-in predicate that makes the rule it is called from return truth value v. The numbers in Formula 4.26 and a, b, c, and d in Formula 4.27 are the values where the trapezoid truth function has its inflections: left slope bottom, left slope top, right slope top, and right slope bottom respectively. In Formula 4.27, the :=-operator assigns a value to v.

$$\begin{aligned}
&AssocDistBetweenCenters(d, c) \leftarrow \\
&\quad [c = \text{tiny} \wedge Trapezoid(d, 0, 0, 10, 50)] \vee \\
&\quad [c = \text{small} \wedge Trapezoid(d, 0, 0, 50, 150)] \vee \\
&\quad [c = \text{binarySmall} \wedge Trapezoid(d, 0, 0, 150, 150)] \vee \\
&\quad [c = \text{medium} \wedge Trapezoid(d, 50, 150, 200, 250)] \vee \\
&\quad [c = \text{large} \wedge Trapezoid(d, 200, 250, 9999, 9999)] \vee \\
&\quad [c = \text{notLarge} \wedge Trapezoid(d, 0, 0, 200, 250)]
\end{aligned} \tag{4.26}$$

$$\begin{aligned}
&Trapezoid(p, a, b, c, d) \leftarrow \\
&\quad [p \geq a \wedge p < b \wedge v := (p - a)/(b - a) \wedge SP(v)] \vee \\
&\quad [p \geq b \wedge p < c \wedge SP(1.0)] \vee \\
&\quad [p \geq c \wedge p < c \wedge v := (d - p)/(d - c) \wedge SP(v)]
\end{aligned} \tag{4.27}$$

4.2 Secondary Models

The three secondary models that are described below are used to recognize:

- center, object, and person oriented groups (SGT 5),
- monologues and dialogues (SGT 6), and
- persons handling documents (SGT 7).

The secondary models were based on the primary models just described, but they introduce various new features. They consist of three SGTs with a corresponding FMTL rule base that uses part of the FMTL interface, rule specifications, formulas, and trapezoid truth functions presented in Sections 4.1.2 through 4.1.5. The secondary SGTs 5 through 7 and some of the additional FMTL rules they deploy will be presented in Section 4.2.3. The model around SGT 5 performs a cluster-based detection of groups that are oriented at the group's center, at an object, or at a person. This model is similar to the model around SGT 1, but it can detect more sophisticated situations. The model around SGT 6 performs cluster-based detection of monologues and dialogues, optionally involving an object. As such, it is the successor of the model around SGT 4, with greater expressive power. Finally, the model around SGT 7 applies the conventional person-as-agent approach to detect individuals handling documents. It is the only SGT in this thesis that uses temporal SGT edges. But before the secondary models are explained in Section 4.2.3, some newly developed concepts will be introduced: a customized clustering algorithm that is used as preprocessing (Section 4.2.1) and a customized algorithm for parameter learning (Section 4.2.2).

4.2.1 Clustering

From the clustering algorithms that are available in literature, DBSCAN was chosen as a preprocessing method for the FMTL/SGT-based reasoning engine. DBSCAN [46]

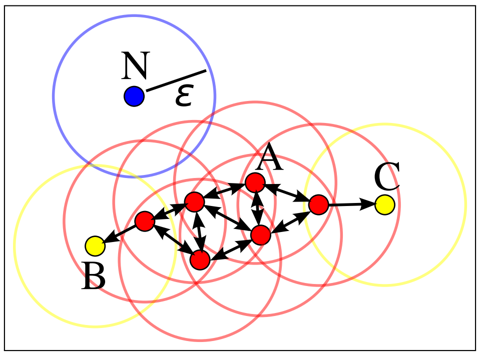


Figure 4.6: Visualization of the DBSCAN algorithm ($minPts = 3$) with six corepoints (A/red), two density-reachable points (B/C/yellow), and one noise point (N/blue). Source: Wikipedia – DBSCAN

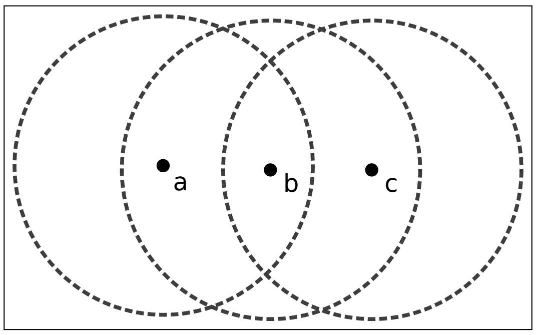


Figure 4.7: Demonstration of the lack of transitivity in the person-as-agent approach. If b is the agent, $\mathit{Group}(a,\ b,\ c)$ holds, but if a or c is the agent, $\mathit{Group}(a,\ b)$ or $\mathit{Group}(b,\ c)$ is deduced instead.

(Density-Based Spatial Clustering of Applications with Noise, Figure 4.6) is a density-based clustering algorithm that uses two parameters: ε and *minPts*. Clusters consist of core points and density-reachable points. Their core points have at least *minPts* neighbors that lie within their ε-neighborhood (A/red in Figure 4.6). Density-reachable points lie within the ε-neighborhood of a core point, but they are not core points themselves (B/C/yellow in Figure 4.6). Noise points are neither core points nor density-reachable points, hence they do not belong to a cluster (N/blue in Figure 4.6).

DBSCAN can be used as preprocessing for the FMTL/SGT reasoning engine, based on the persons' position attributes. This allows for a more global cluster-as-agent approach, as opposed to the more local person-as-agent approach that is usually applied. For each frame of input data, an SGT is traversed once per agent. In the person-as-agent approach, each person is observed as an agent, whereas in the cluster-as-agent approach, each cluster of persons found by the DBSCAN algorithm is observed as an agent. Each cluster of persons is used as a superset within which the FMTL/SGT reasoning engine searches for group situations by considering all possible subsets of the cluster as a potential group.

Advantages of the Cluster-as-Agent Approach and DBSCAN

The (more global) cluster-as-agent approach has several advantages over the person-as-agent approach. First, it is more intuitive, i.e. more consistent with the way humans detect group situations. Second, it ensures that group membership is a transitive relation,[4] consistent with intuition. For the person-as-agent approach, group membership is always reflexive and symmetrical, but not always transitive. Oblong groups are particularly problematic for the person-as-agent approach, as shown in Figure 4.7. The circles represent the group neighborhoods of objects a, b, and c. The following problem can be identified for the person-as-agent approach. If b is selected as the agent, $\mathit{Group}(a,\ b,\ c)$ is deduced, but if a or c is the agent, $\mathit{Group}(a,\ b)$ or $\mathit{Group}(b,\ c)$ is deduced, although the situation is the same. Third, it can significantly improve runtimes by substituting part of the logic's

[4] A relation R is transitive if and only if $R(a, b) \wedge R(b, c) \Rightarrow R(a, c)$.

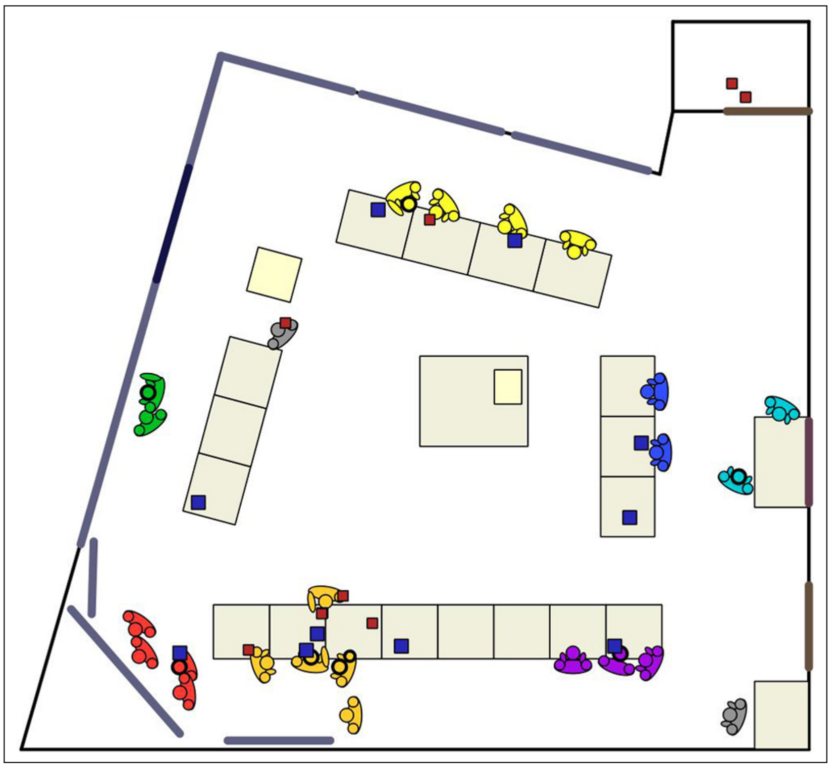


Figure 4.8: An example frame of annotated input data, enriched with outputs from the DBSCAN algorithm ($minPts = 1$, $\varepsilon = 110$cm). Each colour represents a different cluster and the two gray persons are outliers.

combinatorial search and by avoiding redundant traversals. To exploit these advantages, an adapted form of DBSCAN preprocessing was investigated as explained in this section, with corresponding changes to the SGTs and FMTL rules. The resulting models are presented in Section 4.2.3 (around SGTs 5 and 6). They are evaluated in Sections 5.5.1 and 5.5.2.

DBSCAN is well suited for the presented problem, because it uses a dynamic number of clusters, an essential feature in this case. On a related note, one does not have to provide the number of clusters as input, as opposed to k-means and hierarchical clustering. Another important feature of DBSCAN is that it allows individuals to not belong to any cluster (outliers). The two gray individuals in Figure 4.8 show that this is a desired property. Finally, because DBSCAN is a density-based approach, it can detect clusters of varying sizes and shapes (oblong, L-shaped, ring-shaped, etc.). A related but considerably different study using mean shift clustering and FMTL/SGT reasoning is presented in [94].

Visualization and Parameters

The input data annotation tool explained in Section 3.3 can visualize the cluster information generated during DBSCAN preprocessing. An example frame of annotated input data, enriched with cluster information, is provided in Figure 4.8. Each colour represents a different cluster and the two gray persons are outliers. DBSCAN's parameter *minPts* is set

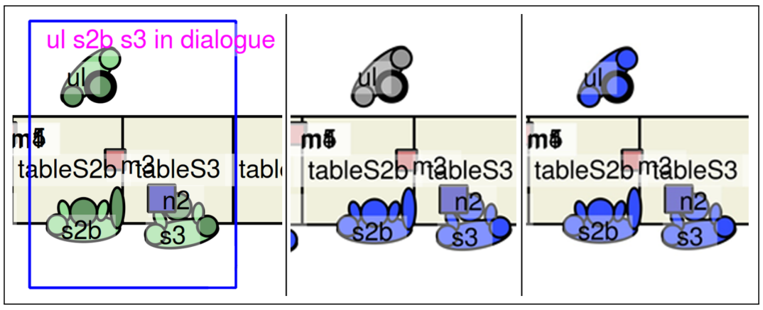


Figure 4.9: The effect of augmenting distances across tables using Equation 4.28; ground-truth (left), clustering without augmentation (center), and clustering with augmentation (right).

to 1, because it should also detect groups consisting of two people, which is not possible for $minPts > 1$. Sensible values for DBSCAN's second parameter ε are 110–120cm, because most group situations in the input data have a density that is smaller than that. Furthermore, the term personal distance, a distance category from anthropology, is 45–120cm for western culture. Most conversations take place within this distance [80].

Modifications to the DBSCAN Algorithm

In order to achieve optimal interaction between DBSCAN and FMTL rules / SGTs, three modifications to the DBSCAN algorithm were designed and implemented: augmentation of distances across tables, cluster tracking, and the use of an additional parameter *maxMembers*. Furthermore, some modifications to the SGTs and FMTL rules were required (discussed further below).

Augmentation of Distances across Tables. Clusters form when the Euclidean distance between two persons is smaller than ε. If there is a table between them, this distance is often larger than ε, although they do belong to the same group. The data shows that the personal distance described above tends to be larger across tables, and that group interactions such as conversations often take place across tables. When using the unmodified Euclidean distance, multiple clusters and outliers are detected instead of the desired single cluster. Since each cluster is used as a superset within which the FMTL/SGT reasoning engine searches for group situations, this leads to errors. To solve this issue, the augmented distance $Dist'(p, q)$ between two persons p and q that is to be compared to ε is defined as:

$$Dist'(p, q) = Dist(p, q) - (1 - \alpha) Dist_T(p, q) \ , \tag{4.28}$$

where $Dist(p, q)$ is the Euclidean distance between p and q, $Dist_T(p, q)$ is the distance between p and q that lies across a table (fraction of $Dist(p, q)$), and α is a weight between 0.0 and 1.0. If $\alpha = 0.0$, the entire distance across the table is ignored, and if $\alpha = 1.0$, the table counts as normal space. The value of α was set to 0.65, because that yields the best results throughout the entire dataset. Figure 4.9 shows the effect of this augmentation. The ground-truth situation on the left can only be detected by the reasoning engine if the augmentation is applied (right). Although *ul* is farther away from *s2b* and *s3* than $\varepsilon = 110cm$, they are allocated to the same cluster as they should be.

Cluster Tracking. During clustering, each cluster is given an ID number. Conventionally, DBSCAN is performed independently of the surrounding frames, and the allocation of cluster IDs is non-deterministic. This lack of coherence in the cluster IDs prevents the FMTL/SGT reasoning from performing temporal reasoning. The addition of cluster tracking to the conventional DBSCAN algorithm makes the cluster IDs depend on the clusters in the previous frames. This is achieved through the so-called Jaccard index which measures the similarity between two clusters. The Jaccard index is defined as the size of the intersection divided by the size of the union of the two clusters:

$$J(a,b) = \frac{\mid a \cap b \mid}{\mid a \cup b \mid} \ . \tag{4.29}$$

Let c be one of the clusters at time t: $c \in clusters_t$. Cluster c gets the same ID as cluster c_{max}, the most similar cluster at $t-1$:

$$c_{max} = \underset{c' \in clusters_{t-1}}{\arg\max} \ J(c,c') \ . \tag{4.30}$$

Figure 4.10 shows three consecutive frames of input data from top to bottom, without cluster tracking on the left and with cluster tracking on the right. The colours that represent the cluster IDs constantly change on the left, whereas on the right they remain constant over time, which is required for subsequent temporal reasoning.[5]

Additional parameter *maxMembers*. The runtime of subsequent reasoning using FMTL rules and SGTs depends on the size of the clusters generated by DBSCAN, because each cluster of persons is used as a superset within which *all* possible subsets are considered as potential groups. For a cluster of size n, there are 2^n possible subsets.[6][7] This exponential relation means that the reasoning's runtime depends heavily on the size of the clusters.[8] If some of the clusters in some of the frames are too large, real-time or timely performance is not possible. Figure 4.11 illustrates the problem. Note that this analysis was performed on a *single* CPU core of an *Intel Core i5-2500K* chip with $3.3GHz$. Runtime can be improved by a factor m where m is the number of cores available for multithreading. For the analysis, 450 frames of input data were used, fed to the model

[5]Because the cluster IDs only depend on the previous frame, this method does not track clusters that briefly disperse or merge.

[6]The set of all possible subsets of a set is called its powerset.

[7]Because the empty set and the singleton sets do not have to be considered as groups, the reasoning engine needs to perform reasoning on $2^n - n - 1$ subsets per cluster.

[8]DBSCAN itself is very fast compared to the subsequent reasoning.

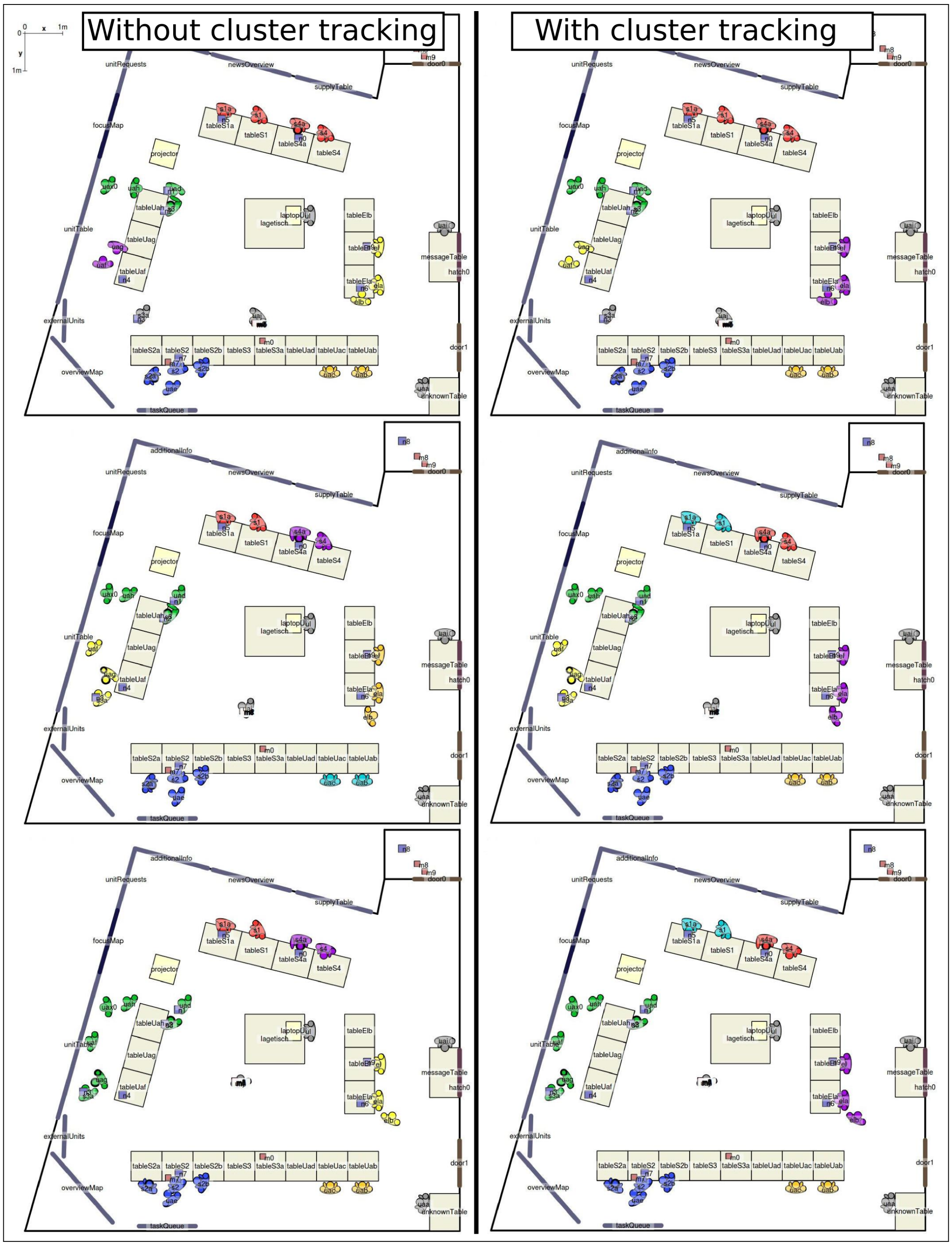


Figure 4.10: Three consecutive frames of input data without cluster tracking (left) and with cluster tracking (right). Without cluster tracking, the colours that represent cluster IDs constantly change.

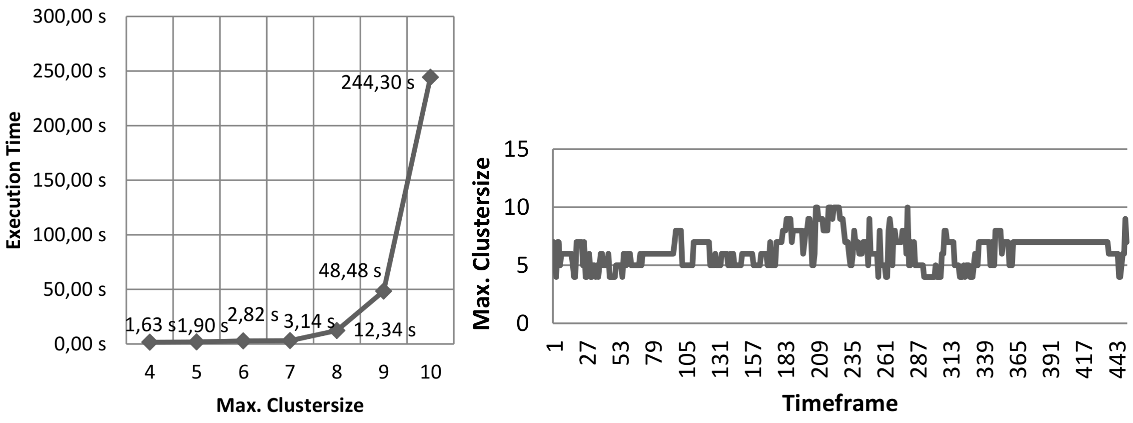


Figure 4.11: Left: the reasoning engine's runtime for selected individual frames. The size of the largest cluster in the frame is displayed along the x-axis. Right: *Max. Clustersize* over 450 frames of input data. Conditions: one CPU core of an *Intel Core i5-2500K* chip with $3.3GHz$, model described in Section 4.2.3 (SGT 5), $\varepsilon = 110cm$.

described in Section 4.2.3 (SGT 5), detecting center, object, and person oriented groups (DBSCAN's $\varepsilon = 110cm$).

Figure 4.11 (left) shows the reasoning engine's runtime for selected frames where the size of the largest cluster in the frame, *Max. Clustersize*, is displayed along the x-axis. Given the fact that only one CPU core was used, and that each frame corresponds to 1 second, real-time performance can be achieved with off-the-shelf hardware for *Max. Clustersize* up to 7 or 8. Then, 9 can still be considered timely performance for some applications. From 10 on, runtimes would be problematic for most applications. Figure 4.11 (right) shows that frames with *Max. Clustersize* > 7 do not occur often (for about 10% of the frames in this graph, and only for one or two clusters per frame). But each time they do, the system slows down considerably. This is why the additional parameter *maxMembers* was introduced. Whenever a cluster size larger than *maxMembers* occurs, DBSCAN is performed on only that cluster again with reduced ε, repeatedly if necessary. Figure 4.12 shows the desired effect. The problematic frames containing large clusters start around $180s$. Without *maxMembers*, the accumulated execution time quickly explodes, as opposed to with *maxMembers* $= 6$, 7, and 8. Using *maxMembers* can lead to reduced reasoning performance, but in Figure 5.19, we see that there is no negative effect on performance in this case.

4.2.2 Parameter Learning

Usually, the parameters of the trapezoid truth functions such as the ones displayed in Figure 4.5 are set manually. Which numeric distances, angles, and speeds correspond to which semantic concepts is determined through common sense and empirical trials, and by comparing the data to other real-world examples (e.g. from surveillance data and average pedestrian speeds). In this section, we discuss how determining these parameters can

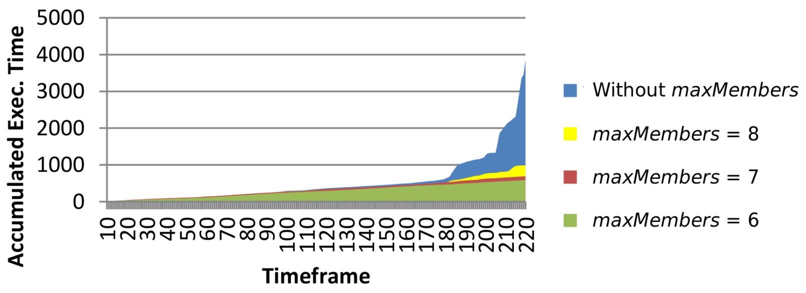


Figure 4.12: Accumulated execution time without *maxMembers* and with *maxMembers* $= 6$, 7, and 8 ($\varepsilon = 110cm$).

be automated (i.e. how they can be learned) using maximization of an adapted F-score performance measure.[9]

Parameter learning was tested in an exemplary fashion on the trapezoid truth functions that are used by the model around SGT 5 below. This should be considered a proof-of-concept only, because the test was performed on a relatively small set of data with the risk of overfitting. The application at hand can further influence what are good parameter values. Hence, parameter learning should also be performed on other data and in other application settings. Furthermore, the F-score measure is based on the subjective view of the ground-truth annotator (as confirmed by Section 5.5.1 – Inter-Annotator Agreement) which can further influence the resulting parameter values. Section 5.5.1 – Effect of Parameter Learning on Performance presents the difference in performance between the model around SGT 5 without parameter learning and with parameter learning, using three different scoring functions.

Learning Strategy. Take the curves labeled "medium" and "small" in the center left graph in Figure 4.5 for example. They are defined by the x-values of their inflection points, medium: $a = 20°$, $b = 45°$, $c = 65°$, $d = 90°$, small: $a = 0°$, $b = 0°$, $c = 20°$, $d = 45°$. Only trapezoids of the latter form ($a = 0$ and $b = 0$) were considered for parameter learning, because trapezoids with $a > 0$ or $b > 0$ do not reflect logical semantics in the tested model. This is often the case when considering distances and angular differences for example. Conveniently, having only trapezoids of this form simplifies the learning strategy, because only the values for c and d have to be learned. Learning consists of two phases. In the first phase, c and d are increased together with a fixed stepsize, from 0 up to the value where a scoring function is maximized, i.e. until increasing c and d further leads to a significant decrease in score. Only small local maxima can occur, so using a threshold on the allowed decrease in score is sufficient. A more sophisticated method like hill climbing was not needed. In the second phase, the optimal block-shaped truth function from the first phase is turned into a trapezoid by decreasing c and increasing d, effectively tilting

[9]Other parameters that could be learned in the context of this thesis are DBSCAN's ε and the weights $\boldsymbol{w}$ that are used by *WeightedInInterval(C, $\boldsymbol{w}$)* (Formulas 4.50–4.52).

the side of the block function while maximizing the scoring function just like in the first phase.

Scoring Function. The applied scoring function (Equations 4.31 and 4.32) is based on Equation 5.4, where α can be set > 1 to emphasize strong precision or < 1 to emphasize strong recall. In Equation 4.31, the value of α depends on the truth value threshold *tvt* (using stepsize 0.1 for example) and a factor x as defined in Equation 4.32. If α does not depend on the truth value threshold, there is no incentive for trapezoid functions that are actually fuzzy (with $d > c$). They will turn out as block functions instead (like "binarySmall" in the top left of Figure 4.5), with $c = d$ at an optimal value given the input data and ground-truth. This is because the ground-truth has binary truth values. There is always a binary reasoning result that has a higher average F_α-score (over all truth value thresholds, Equation 5.4) than the corresponding reasoning result with fuzzy truth values. But following this measure leads to overfitting and the loss of important information.

$$F_{\alpha_x}\textit{-score} \quad = \quad (1+\alpha_x) \cdot \frac{\textit{precision} \cdot \textit{recall}}{\alpha_x \cdot \textit{precision} + \textit{recall}} \tag{4.31}$$

$$\alpha_x = \begin{cases} x \ - \ 2 \cdot \textit{tvt} \cdot (x-1) & \text{if } \ 0.0 < \textit{tvt} \leq 0.5 \\ \frac{1}{x \ - \ 2 \cdot (1-\textit{tvt}) \cdot (x-1)} & \text{if } \ 0.5 < \textit{tvt} \leq 1.0 \end{cases} \tag{4.32}$$

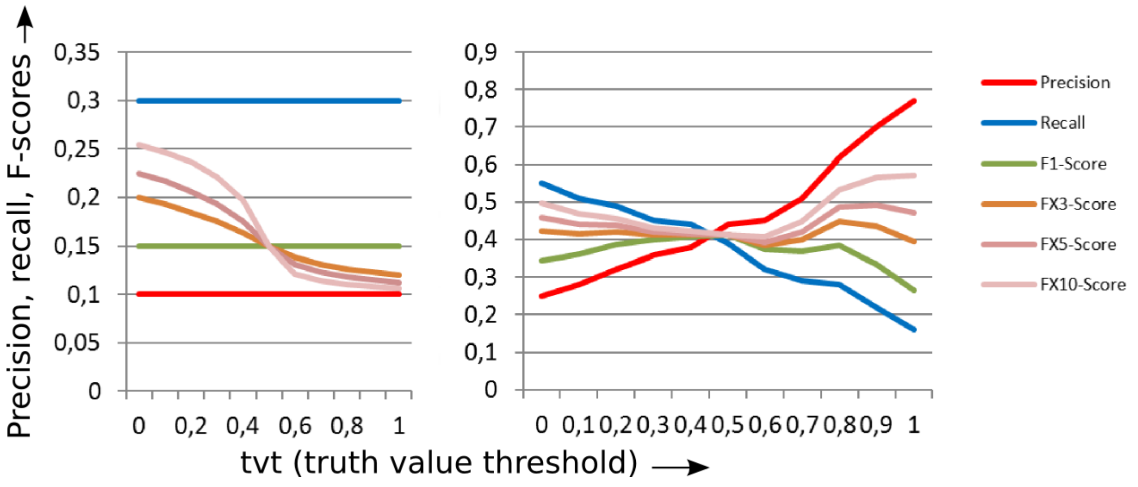


Figure 4.13: Standard F-score, $F_{\alpha_{x=3}}$, $F_{\alpha_{x=5}}$, and $F_{\alpha_{x=10}}$ for given precion and recall values, for results without fuzzy truth values (left) and with fuzzy truth values (right). During parameter learning, the average F_{α_x}-score over all truth value thresholds is maximized.

Figure 4.13 shows the standard F-score (Equation 5.3), $F_{\alpha_{x=3}}$, $F_{\alpha_{x=5}}$, and $F_{\alpha_{x=10}}$ for given precion and recall values.[10] On the left, precision and recall do not vary across truth value thresholds (x-axis, *tvt*) which is the case for results without fuzzy truth values. The measure that is optimized during parameter learning is the average F_{α_x}-score over all truth

[10]F1-Score, FX3-Score, FX5-Score, and FX10-Score in the graph's legend

value thresholds. Fuzzy truth values in the results are desired, because they contain extra information about the degree to which the result holds (because of vagueness and/or uncertainty). This can be used in intelligent systems, by humans in decision support systems, or in the presented case study to filter or sort the deduced situations in the reports that are generated for the staff members. Furthermore, with fuzzy results, one can use a truth value threshold to put more emphasis on precision or recall, without changing anything else. Last but not least, overfitting can be avoided to some extent by using fuzzy truth functions. Fuzzyness is achieved through Equation 4.31 by rewarding high recall values at low truth value thresholds and by rewarding high precision values at high truth value thresholds. The larger the value of x in Equation 4.32, the wider the slopes of the trapezoid truth functions become. This can be used to purposefully influence the amount of fuzzyness in the results.

4.2.3 SGTs and FMTL Rules

This section discusses the secondary SGTs 5 through 7 and some of the additional FMTL rules that they deploy.

SGT 5:

- recognizes center, object, and person oriented groups,
- is displayed in Figure 4.14, and
- generates outputs of the form:
  - ***group*** oriented at center,
  - ***group*** oriented at object o, and
  - ***group*** oriented at person p.

This SGT and the next one use the cluster-as-agent approach as opposed to the person-as-agent approach. DBSCAN preprocessing is used to obtain the required cluster information as described in Section 4.2.1.

Close Groups. During each traversal, the *Root* node in Figure 4.14 selects one of the clusters found by DBSCAN as the current agent. Then, the node *CloseGroup* finds all subclusters within the current agent-cluster. The set of subclusters $\boldsymbol{scs_c}$ contains all subsets of cluster $\boldsymbol{c}$, i.e. $\boldsymbol{scs_c} = \{\boldsymbol{sc} : \boldsymbol{sc} \subseteq \boldsymbol{c}\}$.[11] Next, a distance criterium is applied to each subcluster $\boldsymbol{sc}$ in $\boldsymbol{scs_c}$:[12]

$$\begin{aligned}&\mathit{CloseGroup}(\boldsymbol{sc}, dc) \leftarrow \\ &\quad \mathit{AllTuplesInList}(\boldsymbol{sc}, \boldsymbol{t}) \wedge \\ &\quad \mathit{CalcAvgForTuples}(\boldsymbol{t}, \text{CalcDistBetweenCenters}, d_{avg}) \wedge \\ &\quad \mathit{AssocDistBetweenCenters}(d_{avg}, dc)\end{aligned} \tag{4.33}$$

Here, $\mathit{AllTuplesInList}(\boldsymbol{sc}, \boldsymbol{t})$ generates the list of all possible tuples $\boldsymbol{t}$ within subcluster $\boldsymbol{sc}$. *CalcAvgForTuples(**t**,* CalcDistBetweenCenters*, d_{avg})* applies *CalcDistBetweenCenters(p, q, d)* (Formula 4.16) to all tuples (p, q) in $\boldsymbol{t}$ and stores the average over all returned values d in d_{avg}.[13] Finally, $\mathit{AssocDistBetweenCenters}(d_{avg}, dc)$ (Figure 4.5, top left) associates d_{avg} with distance category dc, determining the truth value V that Formula 4.33 returns. In this case, the SGT in Figure 4.14 has set dc to "binarySmall", so V is 1 if $d_{avg} \leq 150cm$ and 0 otherwise. Truth values < 1 are formed further down this SGT. The subclusters that fulfill Formula 4.33 (the ones that return $V = 1$) are conceptually refined in the three bottom nodes in Figure 4.14 to find center oriented, object oriented, and person oriented groups.

[11] $\boldsymbol{scs_c}$ is the powerset of $\boldsymbol{c}$.

[12] The system is devised in such a way that Formula 4.33 is applied repeatedly until no further instantiations of $\boldsymbol{sc}$ are possible.

[13] Note that *CalcAvgForTuples* can apply other rules than *CalcDistBetweenCenters* as well.

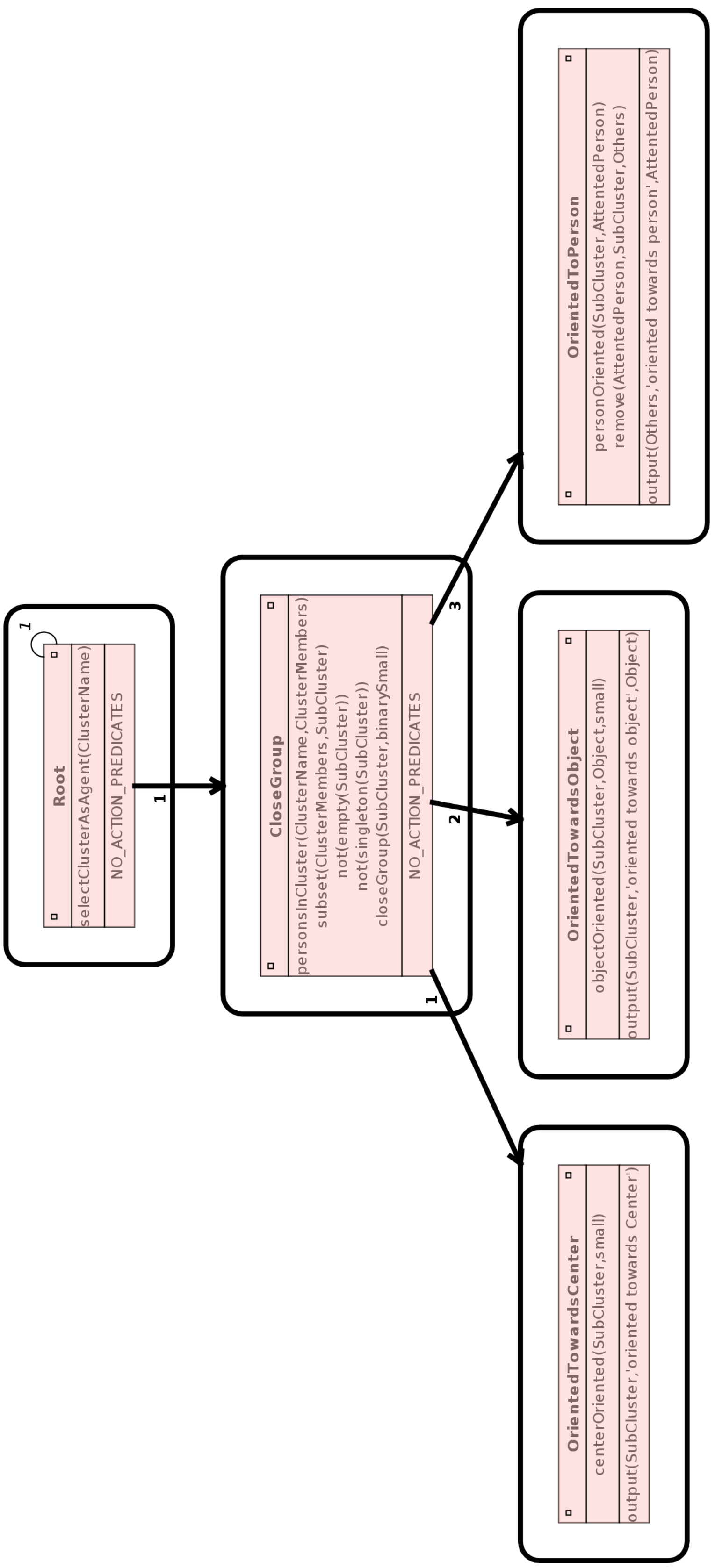

Root
selectClusterAsAgent(ClusterName)
NO_ACTION_PREDICATES
1
1
CloseGroup
personsInCluster(ClusterName,ClusterMembers)
subset(ClusterMembers,SubCluster)
not(empty(SubCluster))
not(singleton(SubCluster))
closeGroup(SubCluster,binarySmall)
NO_ACTION_PREDICATES
3
2
1
OrientedToPerson
personOriented(SubCluster,AttentedPerson)
remove(AttentedPerson,SubCluster,Others)
output(Others,'oriented towards person',AttentedPerson)
OrientedTowardsObject
objectOriented(SubCluster,Object,small)
output(SubCluster,'oriented towards object',Object)
OrientedTowardsCenter
centerOriented(SubCluster,small)
output(SubCluster,'oriented towards Center')

Figure 4.14: SGT 5 recognizes center, object, and person oriented groups using the cluster-as-agent approach.

Groups Oriented Towards Center. The bottom left node in Figure 4.14, *OrientedTowardsCenter*, calculates to what degree groups are oriented towards the center of the group, Formula 4.34, and generates the corresponding output. First, $\mathit{CalcCenter}(\boldsymbol{sc}, x_c, y_c)$ calculates the center (x_c, y_c) of subcluster ***sc*** by averaging the positions of its members. Then, $\mathit{AllTuplesForListAndPoint}(\boldsymbol{sc}, x_c, y_c, \boldsymbol{t})$ generates the list of tuples $\boldsymbol{t} = \{(p, (x_c, y_c)) : p \in \boldsymbol{sc}\}$.

$$\begin{aligned}
&\mathit{CenterOriented}(\boldsymbol{sc}, dc) \leftarrow \\
&\quad \mathit{CalcCenter}(\boldsymbol{sc}, x_c, y_c)\wedge \\
&\quad \mathit{AllTuplesForListAndPoint}(\boldsymbol{sc}, x_c, y_c, \boldsymbol{t})\wedge \\
&\quad \mathit{CalcAvgForTuples}(\boldsymbol{t}, \text{CalcDiffBetweenOriAndAngleToPoint}, d_{avg})\wedge \\
&\quad \mathit{AssocDiffBetweenOriAndAngleToCenter}(d_{avg}, dc)
\end{aligned} \tag{4.34}$$

$\mathit{CalcAvgForTuples}(\boldsymbol{t}, \text{CalcDiffBetweenOriAndAngleToPoint}, d_{avg})$ applies the rule $\mathit{CalcDiffBetweenOriAndAngleToPoint}(p, (x_c, y_c), d)$ to the tuples in $\boldsymbol{t}$ and averages the returned values d (angular differences between persons p and the center of the subcluster (x_c, y_c)), yielding d_{avg}. The value of (x_c, y_c) is then associated with difference category dc, in this case "small" ($\mathit{AssocDiffBetweenOriAndAngleToCenter}(d_{avg}, dc)$, Figure 4.5, center right).

Groups Oriented towards Object. The bottom center node in Figure 4.14 (*OrientedTowardsObject*) uses Formula 4.35 to generate fuzzy outputs of the form "***subcluster*** oriented towards *object*". For this to hold, *object* needs to be a message, notepad, display, laptop, or projector. $\mathit{AssocDistBetweenObjectAndPoint}(o, x_c, y_c, \text{binaryMedium})$ associates the distance between object o and the group's center (x_c, y_c) with distance category "binaryMedium" (trapezoid truth function with inflections at 0, 0, 200, and 200). So if this distance is smaller than $200cm$, *OrientedAt* (Formulas 4.38 through 4.41) can be applied through *ConjunctionForTuples* (Formula 4.36), after *AllTuplesForListAndPoint* supplies all possible tuples between the members of ***sc*** and the object's center (x_o, y_o), to determine to what degree the group's members are oriented at object o.

$$\begin{aligned}
&\mathit{ObjectOriented}(\boldsymbol{sc}, o, dc) \leftarrow \\
&\quad [\mathit{Type}(o, \text{message}) \vee \mathit{Type}(o, \text{notepad}) \vee \mathit{Type}(o, \text{display})\vee \\
&\quad\quad \mathit{Type}(o, \text{laptop}) \vee \mathit{Type}(o, \text{projector})]\wedge \\
&\quad \mathit{CalcCenter}(\boldsymbol{sc}, x_c, y_c)\wedge \\
&\quad \mathit{AssocDistBetweenObjectAndPoint}(o, x_c, y_c, \text{binaryMedium})\wedge \\
&\quad \mathit{Position}(o, x_o, y_o)\wedge \\
&\quad \mathit{AllTuplesForListAndPoint}(\boldsymbol{sc}, x_o, y_o, \boldsymbol{t})\wedge \\
&\quad \mathit{ConjunctionForTuples}(\boldsymbol{t}, \text{OrientedAt}, \text{small})
\end{aligned} \tag{4.35}$$

$$\begin{aligned}
&\mathit{ConjunctionForTuples}([(a, b) \mid \boldsymbol{t}], \mathit{predName}, \mathit{cat}) \leftarrow \\
&\quad C\ =..\ [\mathit{predName}, a, b, \mathit{cat}]\wedge \\
&\quad \mathit{Call}(C)\wedge \\
&\quad \mathit{ConjunctionForTuples}(\boldsymbol{t}, \mathit{predName}, \mathit{cat})
\end{aligned} \tag{4.36}$$

$$\mathit{ConjunctionForTuples}([\,], \mathit{predName}, \mathit{cat}) \tag{4.37}$$

ConjunctionForTuples (Formula 4.36) recursively takes the first tuple (a, b) from list $\boldsymbol{t}$ and combines it into C using the = .. operator, resulting in a predicate with predicate name *predName* and arguments a, b, and *cat*. Calling *Call(C)* and then *ConjunctionForTuples* (recursively) on the rest of the list results in a concatenation of conjunctions with weak conjunction semantics.[14] Formula 4.37 is the recursion's stop condition with truth value 1.0. *OrientedAt(p, q)* (Formulas 4.38 through 4.41 and Formula 4.43 later on) is an improved version of Formula 4.7. Besides the horizontal orientation of p, it also uses p's *lookingDown* attribute and the distance between p and q, in a way that depends on the type of object q. For messages for example, p should be looking down and the distance between p and q should be very small. For displays, p should not be looking down if the distance is medium and he can either look down or not if the distance is small.[15]

$$\begin{aligned}
&\mathit{OrientedAt}(p,q) \leftarrow \\
&\quad [\mathit{Type}(q,\text{notepad}) \vee \mathit{Type}(q,\text{message})] \wedge \\
&\quad \mathit{LookingDown}(p) \wedge \\
&\quad \mathit{DistBetweenInnerPerimeters}(p,q,\text{verySmall}) \wedge \\
&\quad \mathit{DiffBetweenOriAndAngleToCenter}(p,q,\text{small})
\end{aligned} \tag{4.38}$$

$$\begin{aligned}
&\mathit{OrientedAt}(p,q) \leftarrow \\
&\quad \mathit{Type}(q,\text{display}) \wedge \\
&\quad \mathit{Orientation}(p,o_p) \wedge \mathit{RayHitsObject}(p,o_p,q) \wedge \\
&\quad \{[\neg \mathit{LookingDown}(p) \wedge \mathit{DistBetweenInnerPerimeters}(p,q,\text{medium})] \vee \\
&\qquad \mathit{DistBetweenInnerPerimeters}(p,q,\text{small}\}
\end{aligned} \tag{4.39}$$

$$\begin{aligned}
&\mathit{OrientedAt}(p,q) \leftarrow \\
&\quad \mathit{Type}(q,\text{laptop}) \wedge \\
&\quad \mathit{DistBetweenInnerPerimeters}(p,q,\text{small}) \wedge \\
&\quad \mathit{DiffBetweenOriAndAngleToCenter}(p,q,\text{verySmall})
\end{aligned} \tag{4.40}$$

$$\begin{aligned}
&\mathit{OrientedAt}(p,q) \leftarrow \\
&\quad \mathit{Type}(q,\text{projector}) \\
&\quad \mathit{LookingDown}(p) \wedge \\
&\quad \mathit{DistBetweenInnerPerimeters}(p,q,\text{small}) \wedge \\
&\quad \mathit{DiffBetweenOriAndAngleToCenter}(p,q,\text{verySmall})
\end{aligned} \tag{4.41}$$

Groups Oriented towards Person. In the bottom right node of Figure 4.14, outputs of the form "***persons*** oriented towards *person*" are generated. It uses a similar construction as the previous node (Figure 4.14 bottom center), as listed in Formula 4.42. This time, the instantiation of *OrientedAt(p, q)* in Formula 4.43 applies, because q is a person.

[14]weak (default) conjunction semantics, $\min(V(A), V(B))$, see Table 2.7.1

[15]The distance and difference categories used in these formulas are similar to the ones that are used by the primary models.

Table 4.3: Motivation for the conjunct $\neg \mathit{LookingDown}(p) \vee [\mathit{Sitting}(q) \wedge \neg \mathit{Sitting}(p)]$ in Formula 4.43. *S(p)* stands for *Sitting(p)*, *S(q)* for *Sitting(q)*, and *L(p)* for *LookingDown(p)*. *Vertical Allignment?* shows in which cases one would expect vertical alligment from intuition. The underscored $\underline{\mathbf{1}}$ is subject to debate.

S(p)	*S(q)*	*L(p)*	Vertical Allignment?	$\neg L(p)$	$\vee$	$[S(q)$	$\wedge$	$\neg S(p)]$
1	1	1	**0**	0	**0**		0	0
1	1	0	**1**	1	**1**		0	0
1	0	1	**0**	0	**0**		0	0
1	0	0	**1**	1	**1**		0	0
0	1	1	**1**	0	**1**		1	1
0	1	0	$\underline{\mathbf{1}}$	1	**1**		1	1
0	0	1	**0**	0	**0**		0	1
0	0	0	**1**	1	**1**		0	1

$$\begin{aligned}
&\mathit{PersonOriented}(\boldsymbol{sc}, p) \leftarrow \\
&\quad \mathit{Remove}(p, \boldsymbol{sc}, \boldsymbol{sc'}) \wedge \\
&\quad \mathit{AllTuplesForListAndPerson}(\boldsymbol{sc'}, p, \boldsymbol{t}) \wedge \\
&\quad \mathit{ConjunctionForTuples}(\boldsymbol{t}, \text{OrientedAt}, \text{small})
\end{aligned} \tag{4.42}$$

$$\begin{aligned}
&\mathit{OrientedAt}(p, q) \leftarrow \\
&\quad \mathit{Type}(q, \text{person}) \wedge p \neq q \wedge \\
&\quad \{\neg \mathit{LookingDown}(p) \vee [\mathit{Sitting}(q) \wedge \neg \mathit{Sitting}(p)]\} \wedge \\
&\quad \mathit{DiffBetweenOriAndAngleToCenter}(p, q, \text{verySmall}) \wedge \\
&\quad \mathit{DiffBetweenOriAndAngleToCenter}(q, p, \text{smallOrMedium})
\end{aligned} \tag{4.43}$$

The choice for the conjunct $\neg \mathit{LookingDown}(p) \vee [\mathit{Sitting}(q) \wedge \neg \mathit{Sitting}(p)]$ in Formula 4.43 is motivated through Table 6.3. Here, *S(p)* stands for *Sitting(p)*, *S(q)* for *Sitting(q)*, and *L(p)* for *LookingDown(p)*. The column *Vertical Allignment?* shows in which cases one would expect vertical alligment between two persons to hold from intuition. The logical condition in the right column (and in Formula 4.43) corresponds to the natural language expression "For vertical allignment from p to q, p should not be looking down, unless q is sitting and p is not". One could argue that the underscored $\underline{\mathbf{1}}$ in Table 6.3 should be a 0 from intuition, because p would be looking over q's head in this case. Finally, the last two conditions in Formula 4.43 state that p should be clearly oriented horizontallly at q, but q should also be somewhat oriented horizontally at p in order for interaction to take place.

SGT 6:

- recognizes monologues and dialogues,
- is displayed in Figures 4.15 through 4.18, and
- generates outputs of the form:
  - *person* talking to ***group***,
  - *person* talking to ***group*** about *object*,
  - ***group*** in dialogue, and
  - ***group*** in dialogue about *object*.

The SGT overview in Figure 4.15 contains seven nodes. Under the *Root* node, the *SpeakingSituation* node uses temporal criteria to search for speaking persons within each cluster. At the level below that, these are separated into monologues (containing one speaker, "talking to" situations) and dialogues (containing two or more speakers). At the bottom level, these are separated into unspecific monologues and monologues about an object on the one hand, and unspecific dialogues and dialogues about an object on the other hand. The four outputs listed above are generated in the four leaves of this SGT. The nodes from this overview are displayed in detail in Figures 4.16 through 4.18.

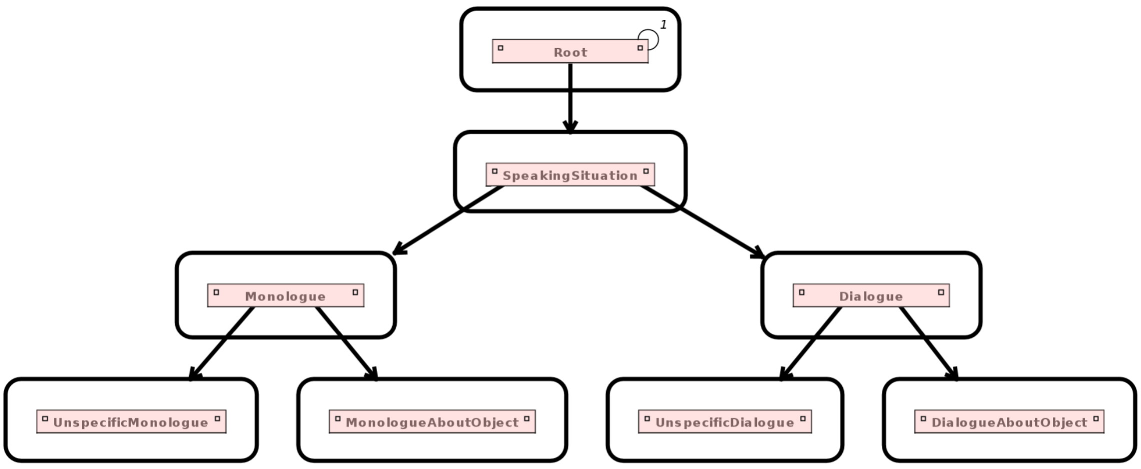


Figure 4.15: SGT 6 recognizes monologues and dialogues using the cluster-as-agent approach – overview, with details in Figures 4.16, 4.17, and 4.18.

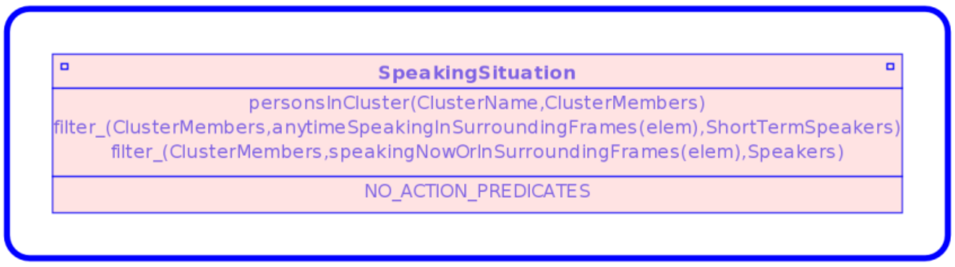

SpeakingSituation
personsInCluster(ClusterName,ClusterMembers)
filter_(ClusterMembers,anytimeSpeakingInSurroundingFrames(elem),ShortTermSpeakers)
filter_(ClusterMembers,speakingNowOrInSurroundingFrames(elem),Speakers)
NO_ACTION_PREDICATES

Figure 4.16: SGT 6: the *SpeakingSituation* node from Figure 4.15.

Speaking Situations. The node that is displayed in Figure 4.16 activates *Filter(**p**, C, **q**)* twice (see Table 4.2) to extract speaking persons from clusters in the current interval. Formulas 4.44 through 4.47 are responsible for this. Formula 4.45 checks whether C holds at least once between $t = -3$ and $t = 3$, yielding *ShortTermSpeakers* in this case. Formula 4.47 checks whether C holds at $t = 0$. If this fails, it goes on to check that C holds at $t = -1$ and at $t = 1$. This is done to filter out speech interrupts that are one frame in length, yielding *Speakers*. Groups around these speaking persons (within the current cluster) are detected further down the SGT.

$$\begin{aligned} &\mathit{AnytimeSpeakingInSurroundingFrames}(p) \leftarrow \\ &\quad \mathit{AnytimeInSurroundingFrames}(\mathit{Speaking}(p)) \end{aligned} \tag{4.44}$$

$$\begin{aligned} &\mathit{AnytimeInSurroundingFrames}(C) \leftarrow \\ &\quad \Diamond_{[-3,3]}\,\mathit{Call}(C) \end{aligned} \tag{4.45}$$

$$\begin{aligned} &\mathit{SpeakingNowOrInSurroundingFrames}(p) \leftarrow \\ &\quad \mathit{NowOrInSurroundingFrames}(\mathit{Speaking}(p)) \end{aligned} \tag{4.46}$$

$$\begin{aligned} &\mathit{NowOrInSurroundingFrames}(C) \leftarrow \\ &\quad [\mathit{Call}(C) \wedge !] \vee \\ &\quad [\Diamond_{-1}\,\mathit{Call}(C) \wedge \Diamond_{1}\,\mathit{Call}(C)] \end{aligned} \tag{4.47}$$

Monologues. On the one hand, the members of the list *Speakers* from the *SpeakingSituation* node are evaluated as monologue-holders (top node in Figure 4.17). Here, *NonSpeakers* consists of the members of *ClusterMembers* that are not in *ShortTermSpeakers* (using *Difference(**p**, **q**, **r**)* from Table 4.2). Each subset of *NonSpeakers* is then investigated as *Listeners* in the two bottom nodes of Figure 4.17. In the bottom left node, *UnspecificMonologue*, Formula 4.48[16] is applied to each possible combination of *Speaker* and *Listeners* within the current cluster, where *Listeners* is a non-empty subset of *NonSpeakers*. *AllTuplesForListAndPerson(**q**, p, **t**)* in Formula 4.48 prepares a list of tupes ***t*** containing pairs of each member of *Listeners* with *Speaker*. Then, *ConjunctionForTuples* (Formula 4.36) is applied to ***t***, predicate *OrientedAt* (Formula 4.43 because we are only dealing with persons), and distance category "small".

[16]The suffix "Smooth" is used for all rules that incorporate the mechanism provided by *WeightedInInterval* (Formula 4.52).

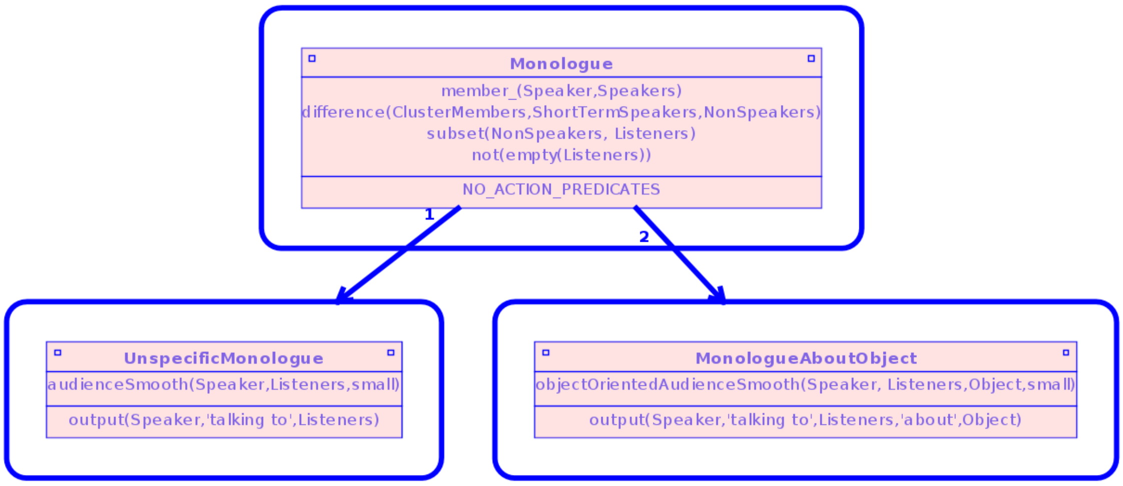

Monologue
member_(Speaker,Speakers)
difference(ClusterMembers,ShortTermSpeakers,NonSpeakers)
subset(NonSpeakers, Listeners)
not(empty(Listeners))
NO_ACTION_PREDICATES
1
2
UnspecificMonologue
audienceSmooth(Speaker,Listeners,small)
output(Speaker,'talking to',Listeners)
MonologueAboutObject
objectOrientedAudienceSmooth(Speaker, Listeners,Object,small)
output(Speaker,'talking to',Listeners,'about',Object)

Figure 4.17: SGT 6: the monologue subtree from Figure 4.15.

$$\begin{aligned}&\mathit{AudienceSmooth}(p,\boldsymbol{q}) \leftarrow \\ &\quad \mathit{AllTuplesForListAndPerson}(\boldsymbol{q},p,\boldsymbol{t})\wedge \\ &\quad \mathit{WeightedInInterval}(\mathit{Conj.ForTuples}(\boldsymbol{t},\mathrm{OrientedAt},\mathrm{small}),\mathrm{shortSymmetric})\end{aligned} \tag{4.48}$$

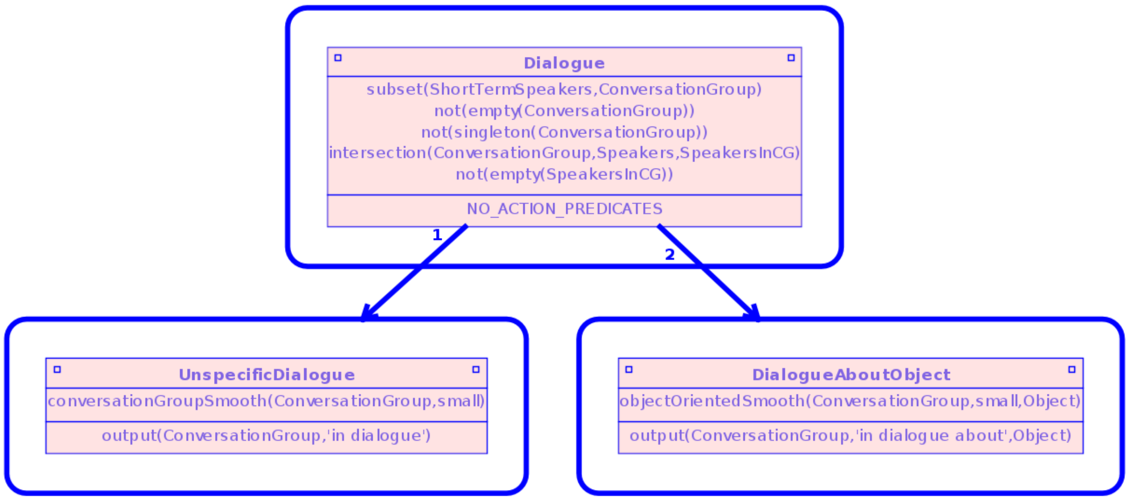
$$\begin{aligned}&\mathit{ObjectOrientedAudienceSmooth}(p,\boldsymbol{q},r) \leftarrow \\ &\quad \mathit{ObjectFilter}(p,\boldsymbol{q},r)\wedge \\ &\quad \mathit{WeightedInInterval}(\mathit{OrientedAt}(p,r,\mathrm{small}),\mathrm{shortSymmetric})\wedge \\ &\quad \mathit{AllTuplesForListAndObject}(\boldsymbol{q},r,\boldsymbol{t})\wedge \\ &\quad \mathit{WeightedInInterval}(\mathit{Conj.ForTuples}(\boldsymbol{t},\mathrm{OrientedAt},\mathrm{small}),\mathrm{shortSymmetric})\end{aligned} \tag{4.49}$$

$$\begin{aligned}&\mathit{WeightedInInterval}(C,\mathrm{shortSymmetric}) \leftarrow \\ &\quad \mathit{WeightedInInterval}(C,[0.0,0.3,0.6,0.3,0.0])\end{aligned} \tag{4.50}$$

$$\begin{aligned}&\mathit{WeightedInInterval}(C,\mathrm{symmetricBoosted}) \leftarrow \\ &\quad \mathit{WeightedInInterval}(C,[0.2,0.3,0.6,0.3,0.2])\end{aligned} \tag{4.51}$$

$$V[\mathit{WeightedInInterval}(C,\boldsymbol{w})] = \sum_{t=-2}^{2} w_t \cdot V[\Diamond_t C] \tag{4.52}$$

This is wrapped in *WeightedInInterval(C, $\boldsymbol{w}$)* (Formula 4.52), a powerful, newly developed fuzzy temporal concept that calculates a weighted average over the truth values of an arbitrary condition C across multiple frames, using arbitrary weights $\boldsymbol{w} = \{w_{t_{-2}}, w_{t_{-1}}, w_{t_0}, w_{t_1}, w_{t_2}\}$ for the interval $t = [-2, 2]$. An intermediate step is required that translates the name of the weights-list into the actual weights: Formula 4.50. In this case, "shortSymmetric" is associated with $\boldsymbol{w} = \{0.0, 0.3, 0.6, 0.3, 0.0\}$. One can define as many of these weights-lists as desired. The weights can add up to one, but they do not have to. If they do not, the result is amplified or dampened. If the resulting truth value is larger than one, it is set to 1.0. In the bottom right node, *MonologueAboutObject*, Formula 4.49 is applied to all combinations of *Speaker*, *Listeners*, and *Object*, following the same principle. *ObjectFilter(p, $\boldsymbol{q}$, r)* makes sure that only the relevant objects r are considered that are in the vicinity of p and $\boldsymbol{q}$. Depending on the type of object r, Formula 4.38, 4.39, 4.40, or 4.41 applies.



Dialogue
subset(ShortTermSpeakers,ConversationGroup)
not(empty(ConversationGroup))
not(singleton(ConversationGroup))
intersection(ConversationGroup,Speakers,SpeakersInCG)
not(empty(SpeakersInCG))
NO_ACTION_PREDICATES
1
2
UnspecificDialogue
conversationGroupSmooth(ConversationGroup,small)
output(ConversationGroup,'in dialogue')
DialogueAboutObject
objectOrientedSmooth(ConversationGroup,small,Object)
output(ConversationGroup,'in dialogue about',Object)

Figure 4.18: SGT 6: the dialogue subtree from Figure 4.15.

Dialogues. Figure 4.18 displays the right branch of Figure 4.15 in detail. Each subset of size > 1 within *ShortTermSpeakers* (from Figure 4.16) is investigated as a dialogue. The application of *intersection(ConversationGroup, Speakers, SpeakersInCG)* followed by *not(empty(SpeakersInCG))* guarantees that at least one of the speakers in the interval $t = [-3, 3]$ is speaking at $t = 0$ or at $t = -1$ *and* $t = 1$, i.e. that part of the dialogue is currently taking place. The bottom left node of Figure 4.18, *UnspecificDialogue*, applies Formula 4.53 to *ConversationGroup*. It determines to what degree two speakers are oriented at each other and uses the weights defined in Formula 4.51. Finally, the bottom right node of Figure 4.18, *DialogueAboutObject*, applies Formula 4.55 to *ConversationGroup*, with *ObjectFilter($\boldsymbol{p}$, q)* making sure that only the relevant objects q in the vicinity of $\boldsymbol{p}$ are considered.

$$\begin{aligned}&\mathit{ConversationGroupSmooth}(\boldsymbol{p}) \leftarrow \\ &\quad \mathit{AllTuplesInList}(\boldsymbol{p}, \boldsymbol{t}) \wedge \\ &\quad \mathit{ConjunctionForTuples}(\boldsymbol{t}, \text{ConversationPartnersSmooth})\end{aligned} \tag{4.53}$$

$$\begin{aligned}&\mathit{ConversationPartnersSmooth}(p, q) \leftarrow \\ &\quad \mathit{WeightedInInterval}(\mathit{OrientedAt}(p, q), \text{symmetricBoosted}) \wedge \\ &\quad \mathit{WeightedInInterval}(\mathit{OrientedAt}(q, p), \text{symmetricBoosted})\end{aligned} \tag{4.54}$$

$$\begin{aligned}&\mathit{ObjectOrientedSmooth}(\boldsymbol{p}, q) \leftarrow \\ &\quad \mathit{ObjectFilter}(\boldsymbol{p}, q) \wedge \\ &\quad \mathit{AllTuplesForListAndObject}(\boldsymbol{p}, q, \boldsymbol{t}) \wedge \\ &\quad \mathit{WeightedInInterval}(\mathit{Conj.ForTuples}(\boldsymbol{t}, \text{OrientedAt}, \text{small}), \text{symmetricBoosted})\end{aligned} \tag{4.55}$$

SGT 7:

- recognizes persons handling documents,
- is displayed in Figures 4.19 through 4.21, and
- generates outputs of the form:
  - *person* carrying *document*,
  - *person* moving *document*,
  - *person* reading *document*,
  - *person* writing *document*,
  - *person* picking up *document*, and
  - *person* laying down *document*.

As opposed to the previous two SGTs, the one in Figure 4.19 uses the person-as-agent approach again. It is the only SGT that makes use of temporal SGT edges. Documents are either messages or notepads, both treated equally by this model. (In the data, ground-truth, and results visualizations throughout this thesis, starting with Figure 3.2, messages are pink or red. Notepads are light or dark blue and slightly larger.) The temporal SGT edges are located in the situation graph containing three situation schemes on the left of Figure 4.19.

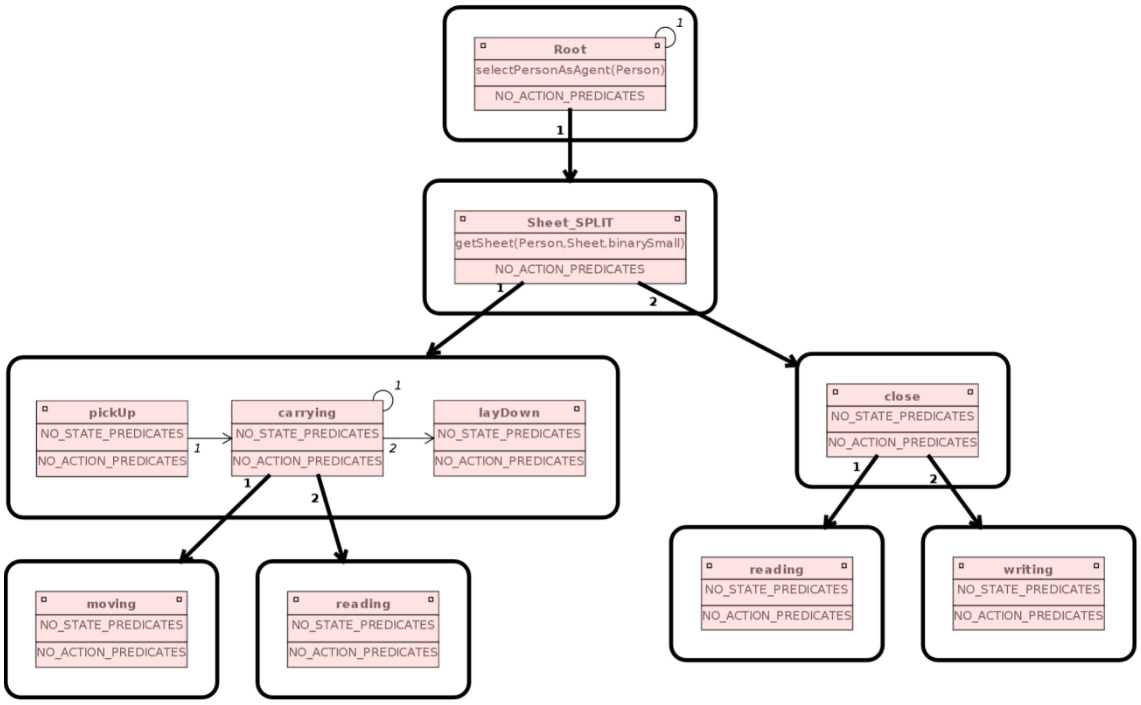


Figure 4.19: SGT 7 recognizes persons that are handling documents, using the person-as-agent approach with temporal SGT edges – overview, with details in Figures 4.20 and 4.21.

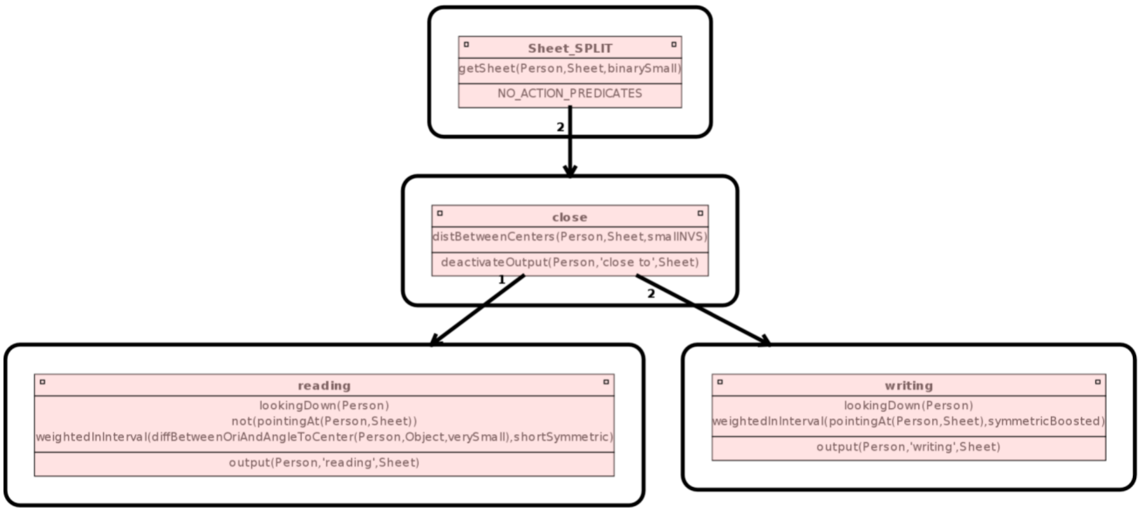

Sheet_SPLIT
getSheet(Person,Sheet,binarySmall)
NO_ACTION_PREDICATES
2
close
distBetweenCenters(Person,Sheet,smallNVS)
deactivateOutput(Person,'close to',Sheet)
1
2
reading
lookingDown(Person)
not(pointingAt(Person,Sheet))
weightedInInterval(diffBetweenOriAndAngleToCenter(Person,Object,verySmall),shortSymmetric)
output(Person,'reading',Sheet)
writing
lookingDown(Person)
weightedInInterval(pointingAt(Person,Sheet),symmetricBoosted)
output(Person,'writing',Sheet)

Figure 4.20: SGT 7: the subtree from Figure 4.19 responsible for documents located on tables.

The right subtree in Figure 4.19 is displayed in detail in Figure 4.20. It detects situations where persons are close to documents on tables. The node *Sheet_SPLIT*, finds all documents that are close enough to the current agent to be considered. Then, the node *close* checks to what degree a document is "close but not very close" to the current agent. The corresponding trapezoid truth function has its inflection points at 32, 32, 37, and 45 so that documents that are very close to the agent ($< 32cm$) are only considered in the left subtree.

This is conceptually refined by the nodes *reading* and *writing* in Figure 4.20. The *reading* node requires looking down and *not* pointing at the document (to distinguish it from writing). Pointing at (i.e. extending arm, Formula 4.8) is the closest thing we have to writing. Finally, reading requires that the agent is sufficiently oriented at the document in the current $3s$ interval. Just like reading, the *writing* node requires looking down. Furthermore, the truth value of writing is determined by how much the agent is pointing at the document in the current $5s$ interval. Again, pointing at means extending one's arm, in this case writing. These two nodes use the same temporal construct as before: *WeightedInInterval*, Formulas 4.50 through 4.52.

The left hand side of Figure 4.19 is displayed in detail in Figure 4.21. It models the handling of documents, from picking up to laying down, with multiple situations in between. The node *pickUp* is the start situation which needs to be detected first. Effectively, the conditions in the *pickUp* node check whether the distance between agent and document is larger than $80cm$ at $t = [-2, -1]$, and that the distance is smaller than $80cm$ at $t = [0, 1, 2]$. Its last condition makes sure that the document is moving between $t = -1$ and $t = 1$ (inflection points 5, 10, 9999, and 9999 cm/s) in order to exclude (common) situations where the agent moves close to a document but does not pick it up.

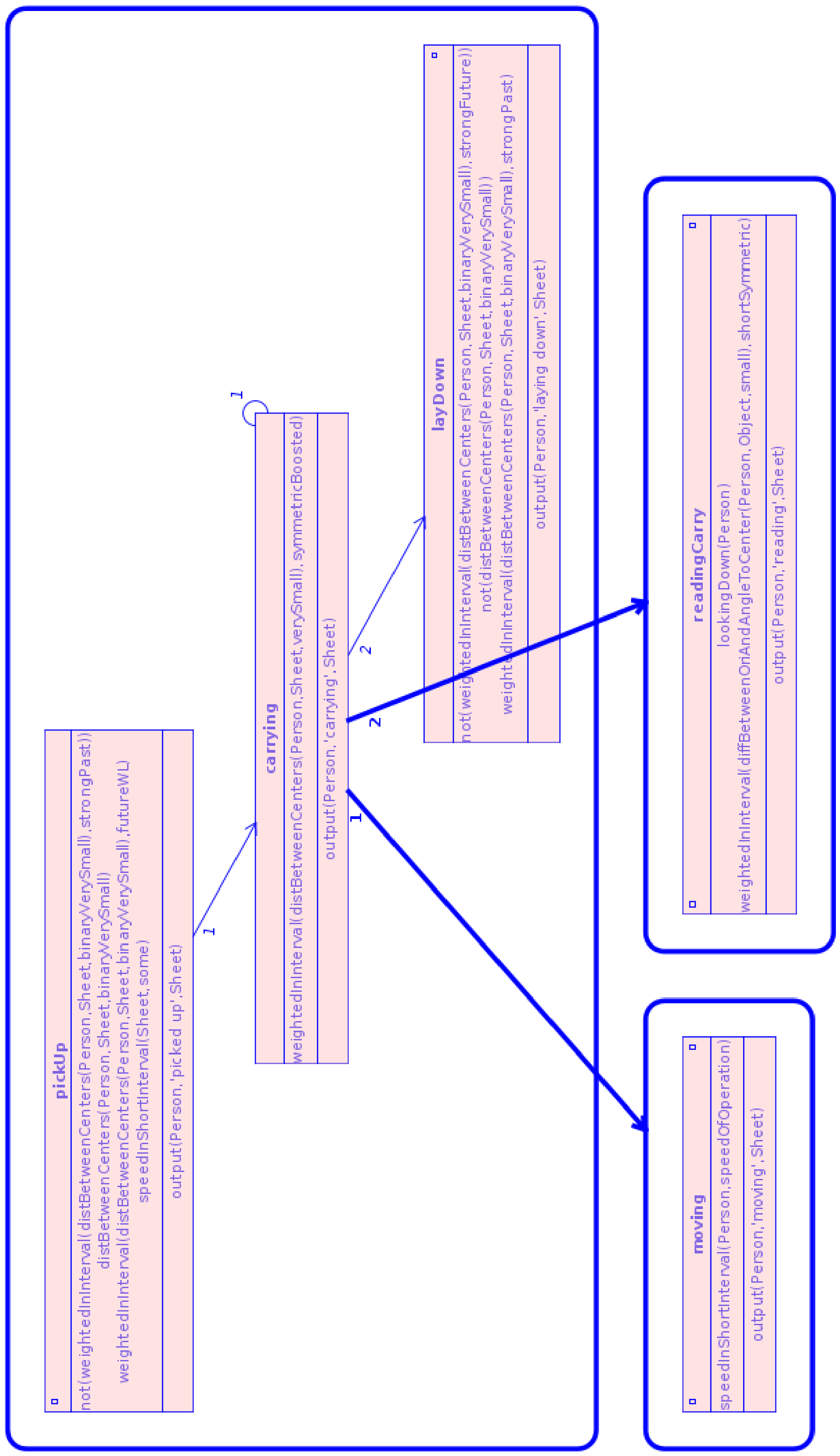

pickUp
not(weightedInInterval(distBetweenCenters(Person,Sheet,binaryVerySmall),strongPast))
distBetweenCenters(Person,Sheet,binaryVerySmall)
weightedInInterval(distBetweenCenters(Person,Sheet,binaryVerySmall),futureWL)
speedInShortInterval(Sheet,some)
output(Person,'picked up',Sheet)
carrying
weightedInInterval(distBetweenCenters(Person,Sheet,verySmall),symmetricBoosted)
output(Person,'carrying',Sheet)
layDown
not(weightedInInterval(distBetweenCenters(Person,Sheet,binaryVerySmall),strongFuture))
not(distBetweenCenters(Person,Sheet,binaryVerySmall))
weightedInInterval(distBetweenCenters(Person,Sheet,binaryVerySmall),strongPast)
output(Person,'laying down',Sheet)
readingCarry
lookingDown(Person)
weightedInInterval(diffBetweenOriAndAngleToCenter(Person,Object,small),shortSymmetric)
output(Person,'reading',Sheet)
moving
speedInShortInterval(Person,speedOfOperation)
output(Person,'moving',Sheet)
1
1
2
2
1

Figure 4.21: SGT 7: the subtree from Figure 4.19 responsible for documents that are being handled.

Only after *pickUp* has been detected, *carrying* can apply for any number of frames, until the end situation *layDown* is detected. Every time *carrying* is detected, the SGT tries to conceptually refine it in the nodes *moving* and *readingCarry*. The former applies if the agent is moving (trapezoid inflection points 20, 50, 9999, and 9999 cm/s) and the latter applies if he is looking down, sufficiently oriented at the document within a $3s$ interval. Finally, the end situation *layDown* is detected if the distance between agent and document is smaller than $80cm$ at $t = [-2, -1]$ and larger than $80cm$ at $t = [0, 2]$.

That concludes Chapter 4, the core of this thesis. It first explained the primary models and their SGTs and FMTL rules. Then, clustering and parameter learning were introduced and the SGTs and FMTL rules of the secondary models were discussed.

Chapter 5

Evaluation

This chapter presents the experiments that were performed to evaluate the presented system. First, the applied evaluation methods are introduced in Section 5.1. Then, the primary models described in Section 4.1 are evaluated in Section 5.2. The corresponding error analysis and runtime are presented in Sections 5.3 and 5.4. Finally, the secondary models described in Section 4.2 are evaluated in Section 5.5, introducing some additional evaluation methods.

5.1 Evaluation Methods

All the models described in Chapter 4 were evaluated in the experiments described in this chapter. The goal was to have the situation graph trees (SGTs) produce the same situation descriptions as a human annotator. Input data for these experiments was annotated using the method described in Section 3.3. These input data are fed into the SGTs presented in Chapter 4 and the results are compared to appropriate ground-truth that was annotated using the method described in Section 3.4. From the 28 minutes of available input data, four minutes were selected for ground-truth annotation, containing group constellations and interaction patterns that occur frequently during dynamic control room operations. Each model evaluation uses more than 1,000 ground-truth results. On average, there are six ground-truth results per second for each model.

The situation descriptions generated by the system were compared to the ones produced by a human annotator in a performance evaluation counting false negatives (*fn*), false positives (*fp*), and true positives (*tp*), in order to calculate precision, recall, and F-score (harmonic mean of precision and recall): Equations 5.1 through 5.3. This was repeated for truth value thresholds between 0.1 and 1.0 (with step size 0.1). Reasoning results with truth values smaller than the applied truth value threshold are rejected, so increasing the truth value threshold leads to higher precision but lower recall. The F-score uses the harmonic mean (as opposed to the arithmetic mean) of precision and recall in order to punish unbalanced ratios between precision and recall. We want them both to have decent values. The F_α-score (Equation 5.4) can be used to put more emphasis on either precision ($\alpha > 1$) or recall ($\alpha < 1$). The experiments in this chapter use the neutral F-score in Equation 5.3. But Equation 5.4 forms the basis for the scoring function that is used for parameter learning: Equations 4.31 and 4.32. The motivation behind this scoring function is explained in detail in Section 4.2.2 – Scoring Function.

$$\begin{aligned}
\textit{precision} &= \frac{tp}{tp + fp} && (5.1)\\
\textit{recall} &= \frac{tp}{tp + fn} && (5.2)\\
\textit{F-score} &= 2 \cdot \frac{\textit{precision} \cdot \textit{recall}}{\textit{precision} + \textit{recall}} && (5.3)\\
F_\alpha\textit{-score} &= (1+\alpha) \cdot \frac{\textit{precision} \cdot \textit{recall}}{(\alpha \cdot \textit{precision}) + \textit{recall}} && (5.4)
\end{aligned}$$

Note that the presented problem is not a classification problem between a few classes. Instead, the situation template (e.g. "conversation between ***persons*** in group ***g***") must be classified correctly, and all involved persons and objects (e.g. instantiating the list variables ***persons*** and ***g***) must be consistent with the annotated ground-truth. In Section 5.5.1, some additonal evaluation methods are introduced and performed. They measure inter-annotator agreement, robustness against noise, and the effects of DBSCAN parameter *maxMembers* and parameter learning (for trapezoid truth functions) on performance.

To introduce the evaluation procedure, Figure 5.1 shows a complete example frame with ground-truth annotations (top) and corresponding reasoning results with their truth values (bottom), rendered by the ground-truth annotation tool described in Section 3.4 and a similar tool that can visualize reasoning results. The displayed frame is from the first evaluation described below. It contains the situation descriptions "***group*** is listening to *person*", "conversation between ***persons*** in group ***g***", and "silent group ***g***".

5.2 Primary Models – Experiments

This section provides the evaluation results for the four primary models presented in Section 4.1: "groups with speaking and listening members", "groups with sitting, standing, and moving members", "groups with joining and leaving members", and "groups oriented at referred object". All four primary models were evaluated on 240 frames of annotated input data (four minutes worth).

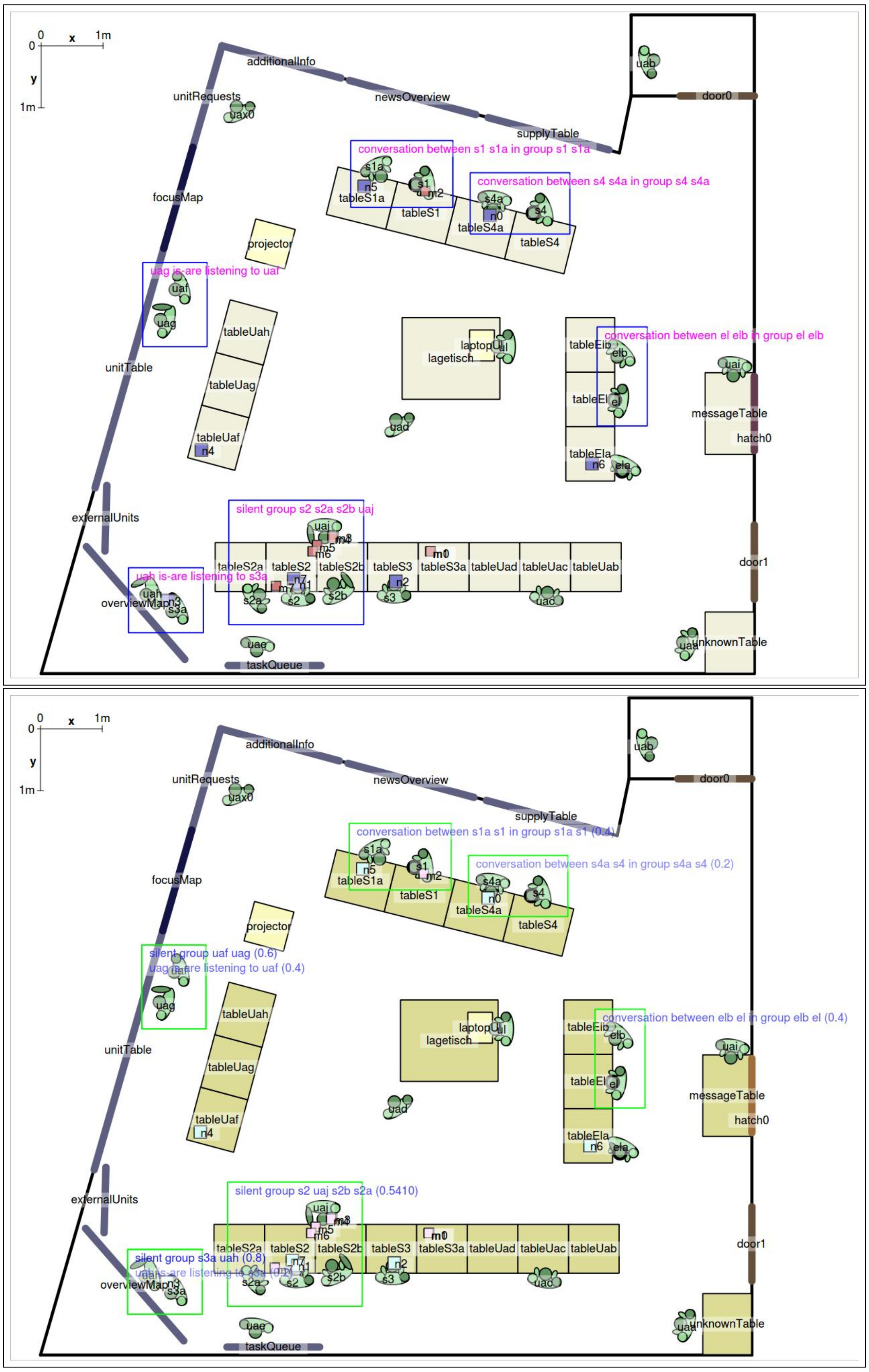

0
x
1m
y
1m
additionalInfo
unitRequests
uax0
newsOverview
supplyTable
uab
door0
conversation between s1 s1a in group s1 s1a
s1a
s1
n5
m2
tableS1a
tableS1
conversation between s4 s4a in group s4 s4a
s4a
s4
n0
tableS4a
tableS4
focusMap
projector
uag is-are listening to uaf
uaf
uag
tableUah
tableUag
tableUaf
n4
unitTable
laptopUil
lagetisch
tableElb
tableEl
tableEla
conversation between el elb in group el elb
elb
el
n6
ela
uai
messageTable
hatch0
uad
externalUnits
silent group s2 s2a s2b uaj
uaj
m8
m5
m6
m0
tableS2a
tableS2
tableS2b
tableS3
tableS3a
tableUad
tableUac
tableUab
uah is-are listening to s3a
n7
n1
n2
uah
overviewMap
n3
s3a
s2a
s2
s2b
s3
uac
door1
uae
taskQueue
uaa
unknownTable
conversation between s1a s1 in group s1a s1 (0.4)
conversation between s4a s4 in group s4a s4 (0.2)
silent group uaf uag (0.6)
uag is-are listening to uaf (0.4)
conversation between elb el in group elb el (0.4)
silent group s2 uaj s2b s2a (0.5410)
silent group s3a uah (0.8)
uah is-are listening to s3a (0.2)

Figure 5.1: Complete example frame with ground-truth annotations (top) and corresponding reasoning results and truth values (bottom), from the experiment in Section 5.2.1.

5.2.1 Groups with Speaking and Listening Members (SGT 1)

Note that all experiments in this chapter contain images like the ones in this section. The explanation below applies to all of them. In the first experiment, groups are classified with respect to which members are speaking, distinguishing between groups listening to a single speaker, conversations, and silent groups (SGT 1 in Figure 4.1). This SGT generates situation descriptions of the form "***group*** listening to *person*", "conversation between ***persons*** in group ***g***", and"silent group ***g***". Example ground-truth (left) and the corresponding reasoning result (right) are visualized in Figure 5.2. In this example, three persons are close together with two of them oriented at the third one (s3), satisfying the conditions for *InSameGroup(p, q)* (Formula 4.5). Because only s3 is speaking between t = *-2* and t = *2* (*InShortInterval(Speaking(p))*, Formula 4.25), the output "uad uah is-are listening to s3" is generated. Further examples from this experiment are provided in Figure 5.1.

Each of the three situation descriptions generated by this SGT is further exemplified by the graphs in Figure 5.3, where the truth values of specific situations are plotted over time. As the positions, orientations, and amounts of speech of the involved persons vary, the truth values of the recognized situations in Figure 5.3 vary with them. The goal is to maximize the correspondence between the bold line (annotated ground-truth) and the thin line (reasoning results). The system performance for this experiment is displayed in Figure 5.4. Precision, recall, and F-score are plotted as a function of the applied truth value threshold (below which results are rejected, higher truth value thresholds leading to higher precision but lower recall). The bottom graphs show the separated performance for each type of situation description generated by the SGT, in this case: "***group*** is listening to *person*", "conversation between ***persons*** in group ***g***", and "silent group ***g***" (from left to right). The most important facts about Figure 5.4 are:

- best overall performance (F-score > 0.6) for truth value thresholds 0.2, 0.3, and 0.4,
- overall number of results (gray area in top graph) in good proportion to number of ground-truth situations (dashed horizontal line),
- "***group*** is listening to *person*" (bottom left): number of results (gray area) small compared to number of ground-truths (dashed horizontal line), causing low recall,
- "conversation between ***persons*** in group ***g***" (bottom center): only low truth value results (proportional to amount of speech in group in $3s$ interval, causing a preference for low truth value thresholds), too little reasoning results compared to ground-truth (causing low recall), and
- precision decreases when truth value threshold approaches 1.0, because only few results with $V \geq$ *truth value thershold* remain (not enough for statistical significance) and *tp* decreases faster than *fp* (this also applies to other performance graphs below).

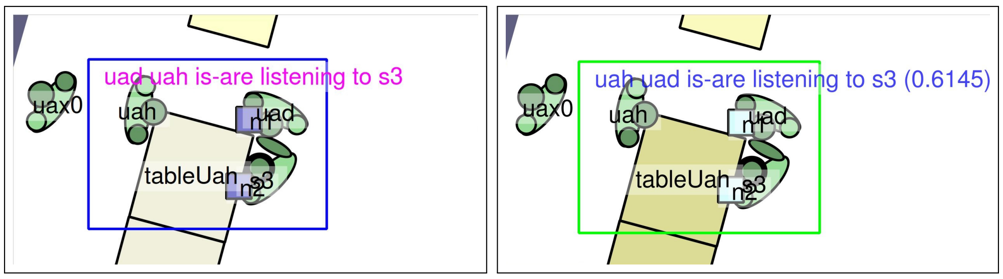

uad uah is-are listening to s3
uax0
uah
uad
n1
tableUah
s3
n2
uah uad is-are listening to s3 (0.6145)
uax0
uah
uad
n1
tableUah
s3
n2

Figure 5.2: Example ground-truth (left) and corresponding reasoning results with their truth values (right) from the experiment “groups with speaking and listening members”.

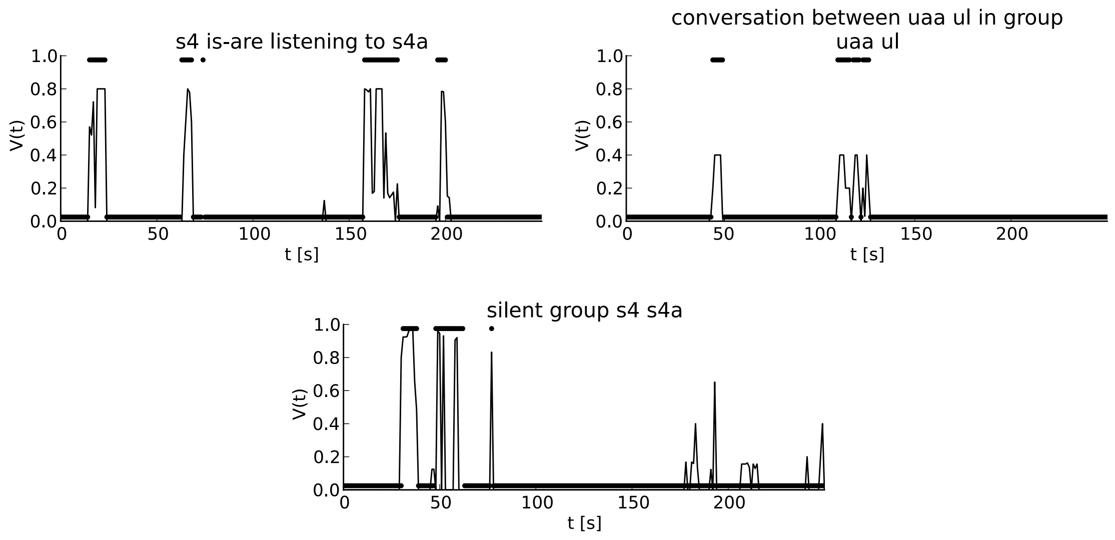
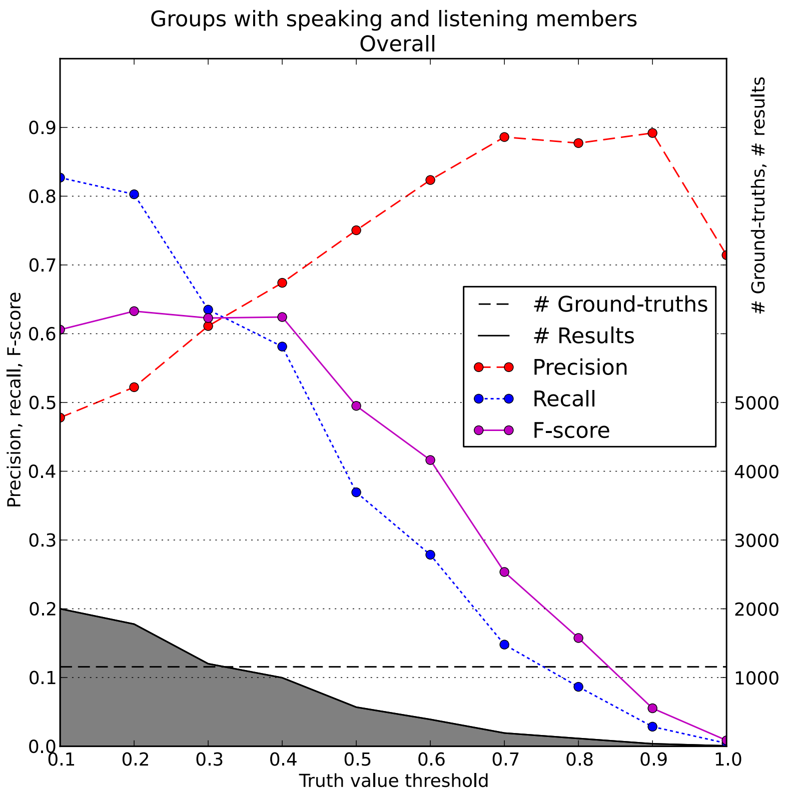


Figure 5.3: Truth values V over time for some example ground-truth and corresponding reasoning results, from the experiment “groups with speaking and listening members”. The bold line shows the annotated ground-truth and the thin line shows the reasoning results.



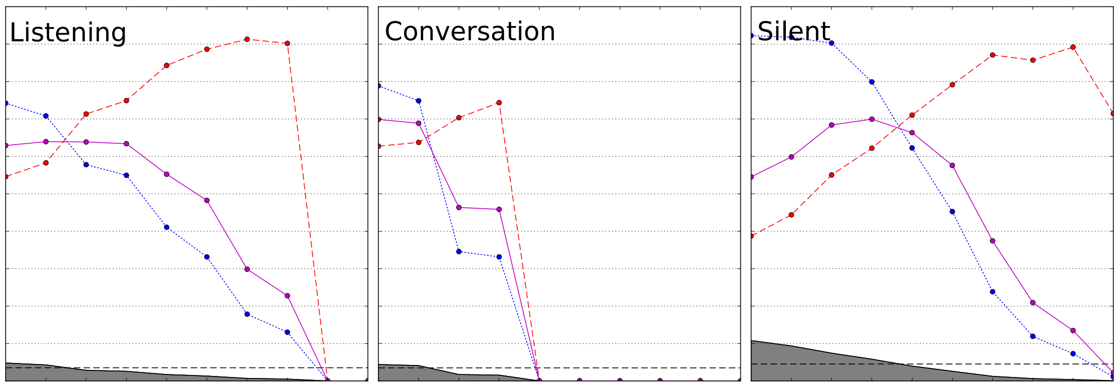


Figure 5.4: System performance for the experiment "groups with speaking and listening members": precision, recall, and F-score over varying truth value thresholds, and the number of ground-truth results and reasoning results involved (the latter depending on the truth value threshold). The bottom graphs, from left to right, show the separated performance for each type of situation description generated by the SGT: "***group*** is listening to *person*", "conversation between ***persons*** in group ***g***", and "silent group ***g***".

5.2.2 Groups with Sitting, Standing, and Moving Members (SGT 2)

Here, groups are assorted with respect to which members are sitting, standing, and moving (SGT 2 in Figure 4.2). This SGT generates situation descriptions of the form "***persons*** in group ***g*** are sitting", "***persons*** in group ***g*** are standing", and "***persons*** in group ***g*** are moving". Some examples are visualized in Figure 5.5 and plotted over time in Figure 5.6. In Figure 5.5, uab and uad belong to the same group, because they are close together and they have appropriate orientations; *InSameGroup(p, q)*, (Formula 4.5). Their positions across three frames change enough for the system to output "uab uad in group uab uad are moving"; *SpeedInShortInterval(p, c)* (Formula 4.15). The system performance for this experiment is displayed in Figure 5.7. The most important facts about Figure 5.7 are:

- best overall performance (F-score ≈ 0.6) for truth value thresholds 0.4 and 0.5,
- overall number of results in good proportion to number of ground-truths, and
- "***persons*** in group ***g*** are moving" (bottom right): relatively poor performance, small number of situations present.

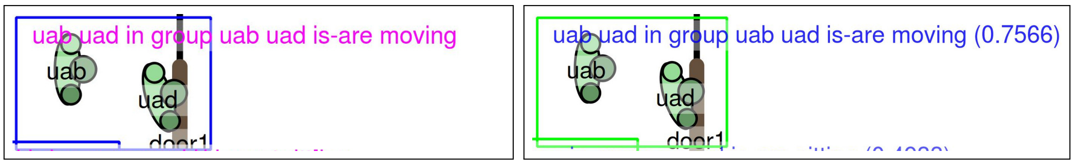

uab uad in group uab uad is-are moving
uab
uad
door1
uab uad in group uab uad is-are moving (0.7566)
uab
uad
door1

Figure 5.5: Example ground-truth (left) and corresponding reasoning results with their truth values (right) from the experiment "groups with sitting, standing, and moving members".

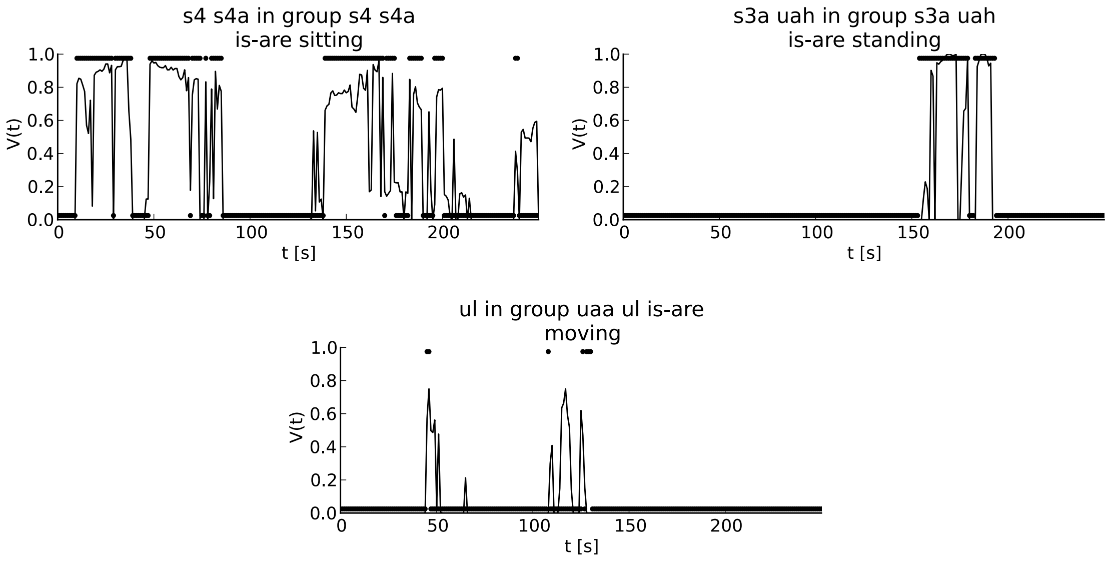


Figure 5.6: Truth values V over time for some example ground-truth and corresponding reasoning results from the experiment "groups with sitting, standing, and moving members". The bold line shows the annotated ground-truth and the thin line shows the reasoning results.

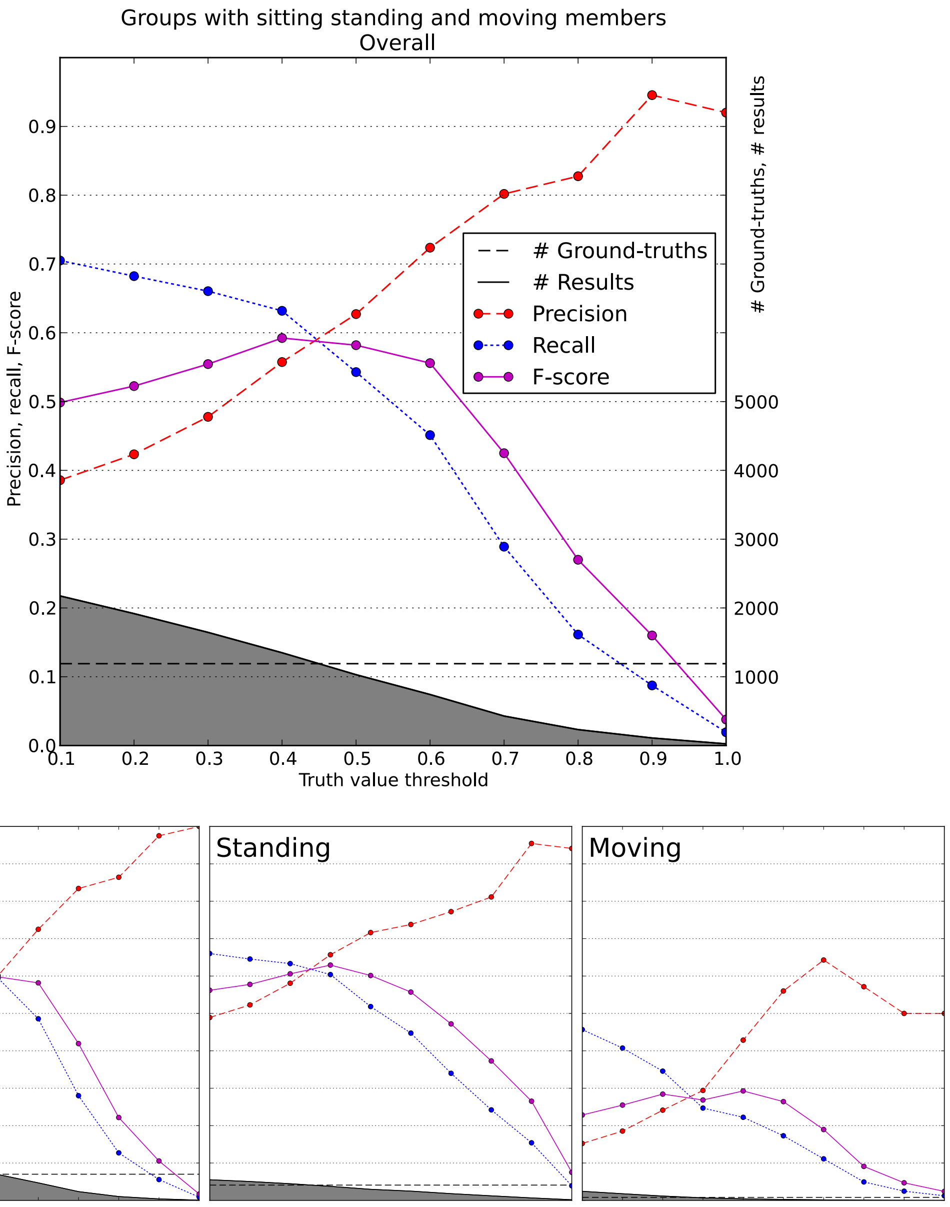


Figure 5.7: System performance for the experiment "groups with sitting, standing, and moving members": precision, recall, and F-score over varying truth value thresholds, and the number of ground-truth results and reasoning results involved (the latter depending on the truth value threshold). The bottom graphs, from left to right, show the separated performance for each type of situation description generated by the SGT: "***persons*** in group ***g*** are sitting", "***persons*** in group ***g*** are standing", and "***persons*** in group ***g*** are moving".

5.2.3 Groups with Joining and Leaving Members (SGT 3)

This experiment deals with groups and persons that are joining and leaving them (SGT 3 in Figure 4.3). The situation descriptions are: "***persons*** are joining ***group***", "***persons*** are leaving ***group***", and "constant group ***g***". Example screenshots and truth value over time plots are provided in Figures 5.8 and 5.9. The left column in Figure 5.8 shows a group of three persons. In the center column, the next frame, s3a is not part of the group anymore, due to his changed position and orientation. This is achieved using *GroupInShortInterval(p, **q**, **r**, **s**)* and *JoiningAndLeavingGroup(**p**, **q**, **r**, **s**)*, (Formulas 4.3 and 4.6), yielding the output "s3a is leaving s3a uaf uag". In the next frame (right column), the system outputs "constant group uaf uag", because s3a has left. System performance is displayed in Figure 5.10. The most important facts about Figure 5.10 are:

- best overall performance (F-score > 0.6) for truth value thresholds 0.3 and 0.4,
- overall number of results in good proportion to number of ground-truths, and
- "constant group ***group***" (bottom right): relatively good performance and large number of situations present.

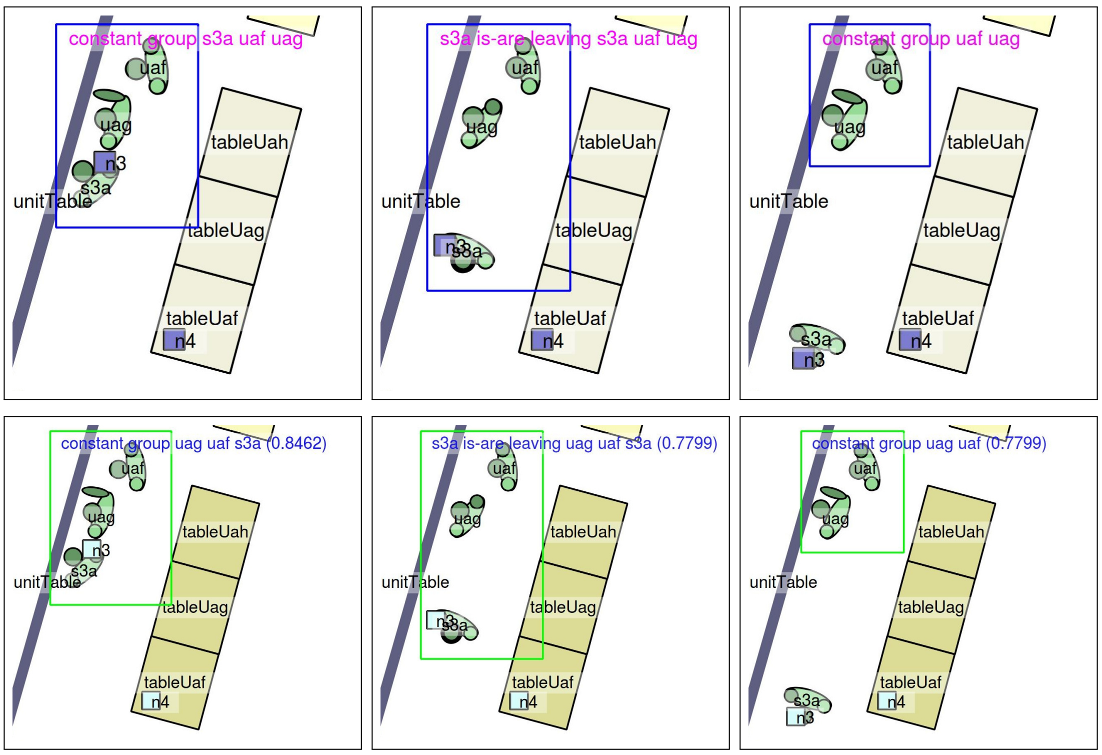

constant group s3a uaf uag
s3a is-are leaving s3a uaf uag
constant group uaf uag
constant group uag uaf s3a (0.8462)
s3a is-are leaving uag uaf s3a (0.7799)
constant group uag uaf (0.7799)
uaf
uag
n3
s3a
n4
unitTable
tableUah
tableUag
tableUaf

Figure 5.8: Example ground-truth (top) and corresponding reasoning results with their truth values (bottom) from the experiment “groups with joining and leaving members”, with three consecutive frames from left to right.

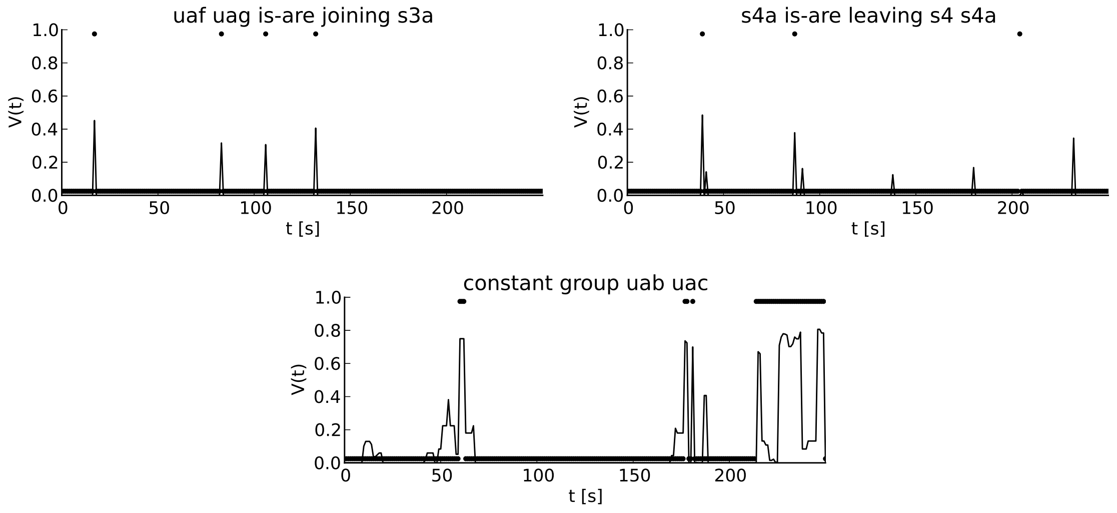


Figure 5.9: Truth values V over time for some example ground-truth and corresponding reasoning results from the experiment “groups with joining and leaving members”. The bold line shows the annotated ground-truth and the thin line shows the reasoning results.

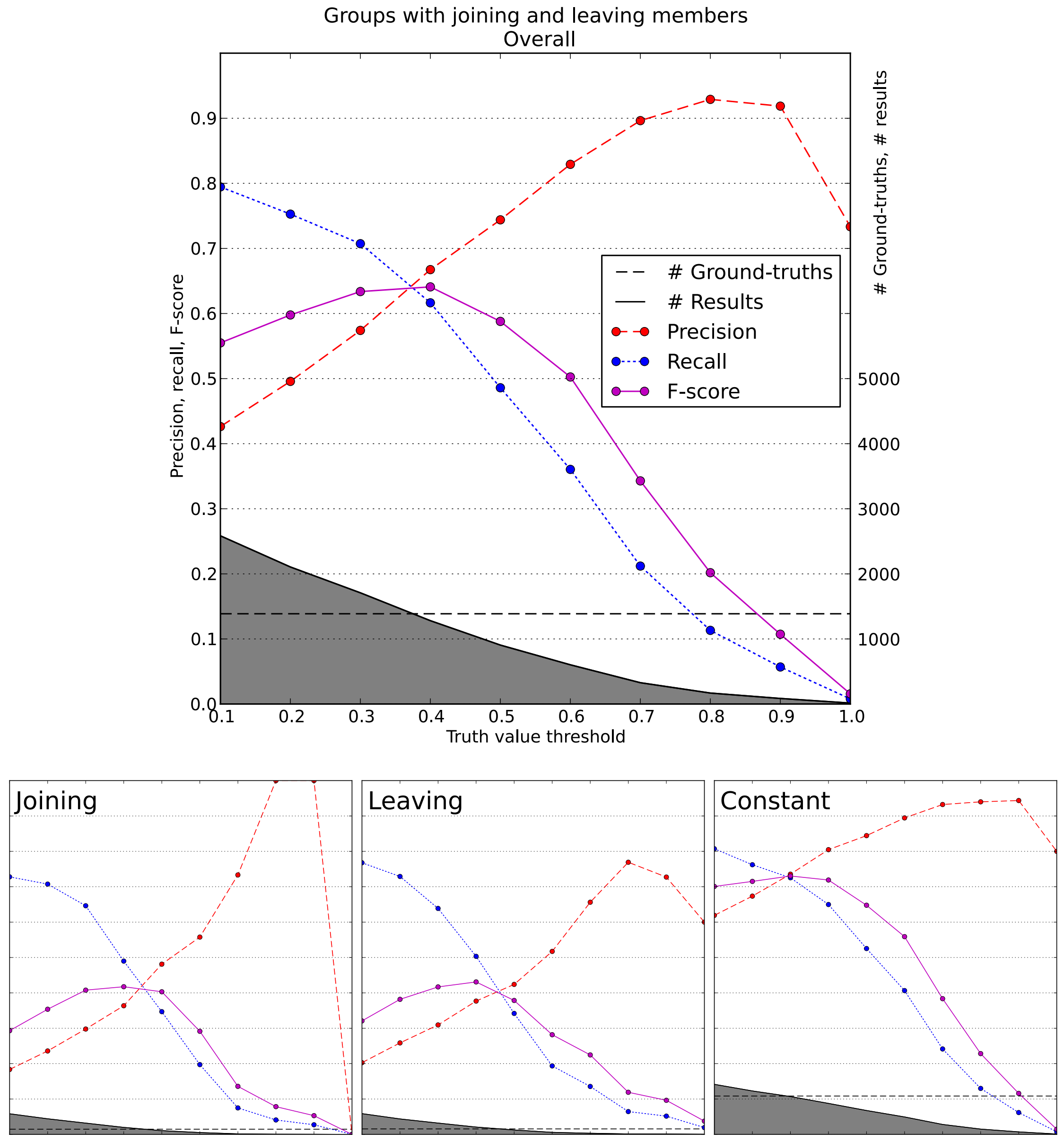


Figure 5.10: System performance for the experiment "groups with joining and leaving members": precision, recall, and F-score over varying truth value thresholds, and the number of ground-truth results and reasoning results involved (the latter depending on the truth value threshold). The bottom graphs, from left to right, show the separated performance for each type of situation description generated by the SGT: "***persons*** are joining ***group***", "***persons*** are leaving ***group***", and "constant group ***g***".

5.2.4 Groups Oriented at Referred Object (SGT 4)

The fourth and last primary model detects groups in which one of the members is referring to an object, and others are oriented at it (SGT 4 in Figure 4.4). The situation descriptions generated by this SGT are: "*person* is pointing at *object*", "*person* is pointing at *object* and ***group*** is looking at it", "*person* is speaking and pointing at *object*", and "*person* is speaking and pointing at *object* and ***group*** is looking at it". The corresponding example screenshots, truth value over time plots, and system performance are provided in Figures 5.11, 5.12, and 5.13. For Figure 5.11, the reasoning engine checks whether the current agent (uag) is pointing at an object (unitTable); *PointingAtNearbyObject(p, q, c)*, Formula 4.2. Because s3a is in the same group as uag (*InSameGroup(p, q)*, Formula 4.5), the output "uag is speaking and pointing at unitTable and s3a is looking at it" is generated. The most important facts about Figure 5.13 are:

- best overall performance (F-score ≈ 0.6) for truth value thresholds 0.7 and 0.8,
- overall number of results large compared to number of ground-truths, causing low precision, and
- best performance and most situations present for "*person* is pointing at *object*".

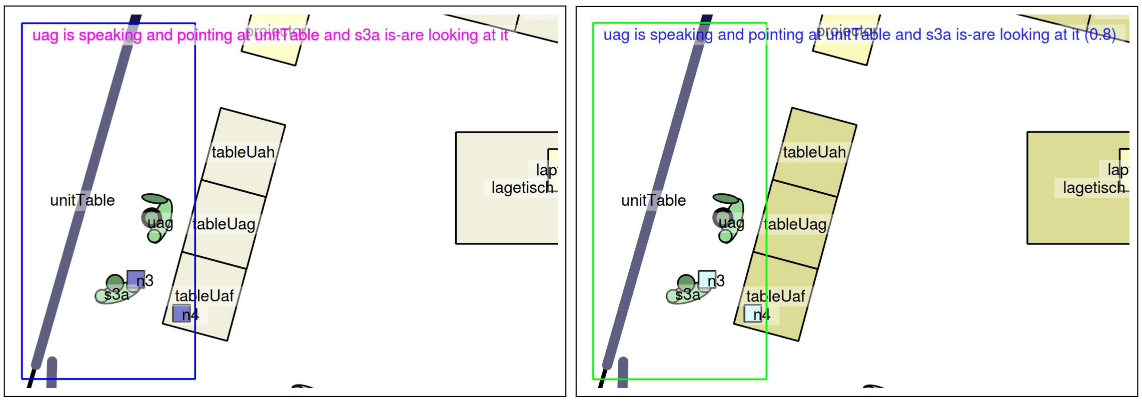

uag is speaking and pointing at unitTable and s3a is-are looking at it
projector
tableUah
tableUag
tableUaf
unitTable
uag
s3a
n3
n4
lap
lagetisch
uag is speaking and pointing at unitTable and s3a is-are looking at it (0.8)

Figure 5.11: Example ground-truth (left) and corresponding reasoning results with their truth values (right) from the experiment “groups oriented at referred object”.

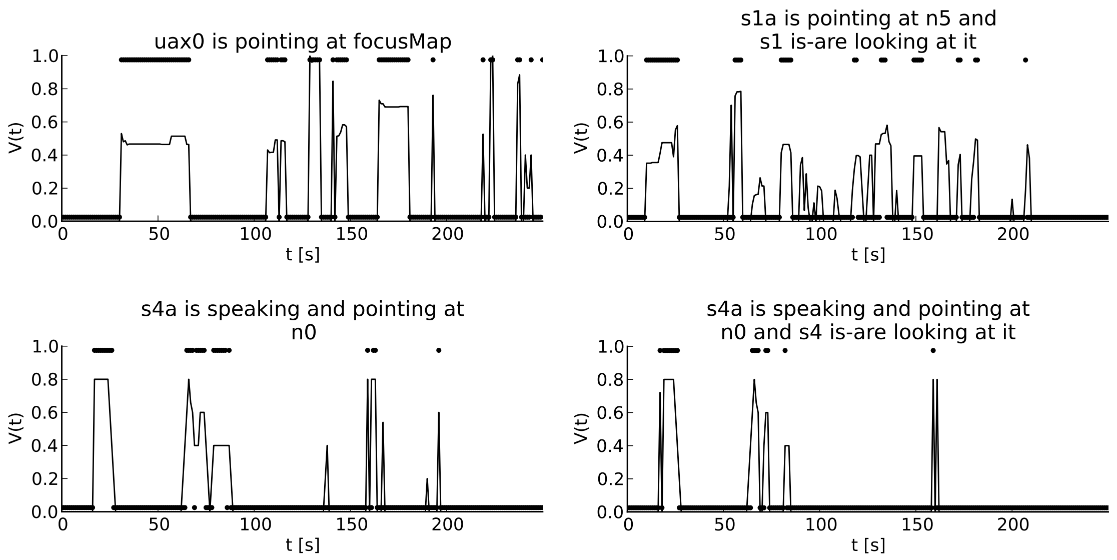
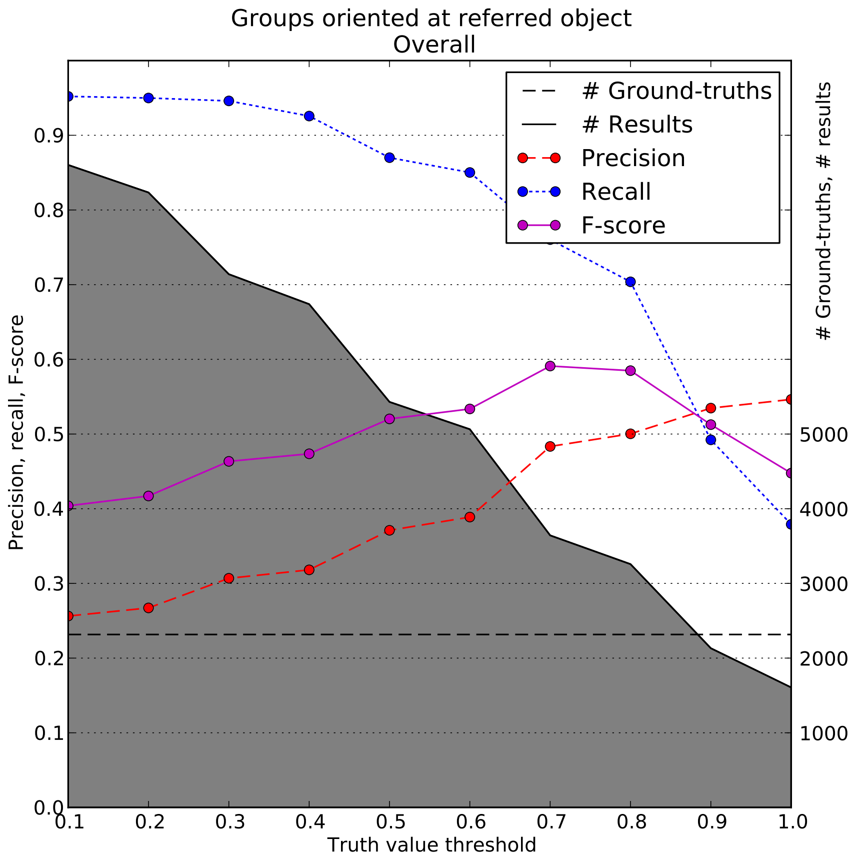


Figure 5.12: Truth values V over time for some example ground-truth and corresponding reasoning results from the experiment “groups oriented at referred object”. The bold line shows the annotated ground-truth and the thin line shows the reasoning results.



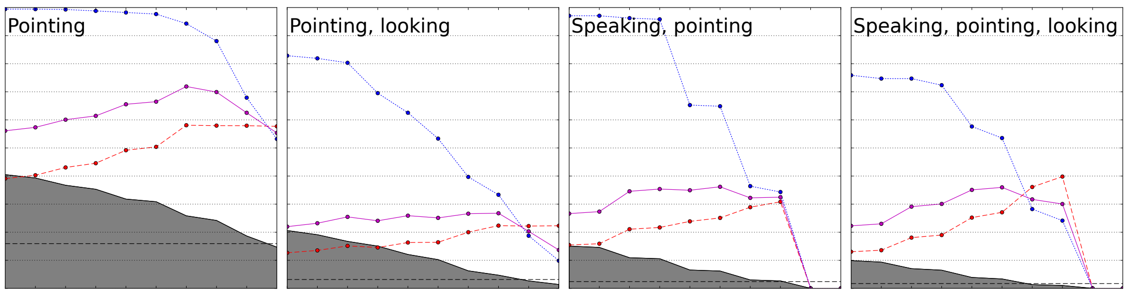


Figure 5.13: System performance for the experiment "groups oriented at referred object": precision, recall, and F-score over varying truth value thresholds, and the number of ground-truth results and reasoning results involved (the latter depending on the truth value threshold). The bottom graphs, from left to right, show the separated performance for each type of situation description generated by the SGT: "*person* is pointing at *object*", "*person* is pointing at *object* and ***group*** is looking at it", "*person* is speaking and pointing at *object*", and "*person* is speaking and pointing at *object* and ***group*** is looking at it".

5.3 Primary Models – Error Analysis

Section 5.3.1 suggests ways to improve performance and Section 5.3.2 puts the applied evaluation criteria into perspective.

5.3.1 Possible Optimizations

Some false negatives are caused by correct results with low truth values. In such cases, correct results are ignored because of the applied truth value threshold. An example thereof with a truth value of 0.2 is visible in the top-right of the images in Figure 5.1. Detecting this result using a truth value threshold ≤ 0.2 is not feasible, because it would lead to many false positives elsewhere in the data. The opposite effect also occurs: wrong results with truth values higher than the applied truth value threshold. Wrong results with low truth values on the other hand, can be filtered out using an appropriate truth value threshold. The models need to be improved to solve these cases.

One of the conditions for the recognition of a group is the proximity of its members. False negatives occur if this condition is not met. Sometimes, one of the group members is standing at a distance from the rest of the group. Human annotators can usually deduce from the context that he is still part of the group, whereas our current knowledge base applies the proximity between group members as a necessary condition. Optimizing or learning the parameters (see Section 4.2.2) of the trapezoid truth functions in Figure 4.5 could solve some of these cases, effectively increasing the system's separating power by changing the proximity conditions. Other cases cannot be solved like this. They would require more complex models instead.

Further errors are caused by either too many or too few group members. More specifically, annotators tend to select the largest possible groups, whereas the reasoning engine sometimes prefers a subgroup thereof, due to failing conditions for some of the persons involved. Such errors might be solved by applying more subtle (fuzzy and temporal) conditions, or by optimizing the trapezoid truth functions in Figure 4.5.

In other cases, two groups are interpreted as one because of their close proximity and harmonic orientations. This could again be partially solved by optimizing the trapezoids, and by modeling more subtle fuzzy temporal conditions, paying more attention to the members' attributes in neighboring frames. Furthermore, single groups are sometimes interpreted as two groups when the members are not close enough together. The applied proximity condition could be combined with fuzzy temporal interaction conditions to cover such cases. There are also cases where persons are recognized as members of two groups simultaneously. Annotators seem to apply a constraint against this that our knowledge base does not yet have. Finally, a staff member passing through or passing by a group is interpreted as joining, belonging to, and leaving the group, leading to further errors. Additional conditions should check how long the staff member stays and whether he is interacting with the group.

To achieve a more powerful knowledge base with more fine-grained control, more FMTL rules should be wrapped in rules like *InShortInterval(C)* (Formula 4.25), combining the power of temporal modality and fuzzy evaluation. Alternatives to *InShortInterval(C)* are provided by the secondary models in Formulas 4.45, 4.47, and 4.50–4.52. Especially these last ones could bring a considerable increase in expressive power and performance. Furthermore, the slopes of the trapezoid truth functions could be widened to facilitate fuzziness (the advantage of which is explained in Section 4.2.2 – Scoring Function), and the rule *RayHitsObject(p, o, q)* in particular (Table 4.2) could benefit from more fuzzy conditions. For some temporal modeling (e.g. of speed), a higher temporal resolution than 1*fps* would be beneficial, which can be approximated by interpolating the current data annotations.[1]

5.3.2 Performance Criteria

One could reconsider the applied performance criteria. For some applications, a hard truth value threshold would not be necessary. One could report and visualize all results along with their truth values, and have a human operator decide which results are the most interesting. Under such criteria (truth value threshold ≤ 0.01) one would require high recall with reasonable precision. Figures 5.4, 5.7, 5.10, and 5.13 show that this can be achieved. One can also perform the presented evaluations while allowing for small temporal offsets to show that some correct situations are detected too early or too late, but within a few seconds from the annotated ground-truth. For some applications, these results would still be valuable. Similarly, the evaluations can be performed while allowing for partial group member match. This means that the list variables in the results and ground-truth only have to match by a fraction between 0.0 and 1.0. Results with partial group member match would still be valuable for some applications. This idea is similar to the performance metric "word error rate" from speech recognition. Performance increases if such concessions are made.

5.4 Primary Models – Runtime

For many applications, real-time performance is essential. The described system provides its results frame-by-frame, but whether this happens in real-time, depends on the situation at hand. In this case study, the system achieves near-real-time performance and actual real-time is within reach. Table 7.1 lists the runtimes for the four experiments explained above, i.e. how long it took to process the four minutes of 1*fps* input data that were used for each experiment. Runtime depends on many factors that are determined by the application domain at hand. In the current case study, the frame rate is only 1*fps*, leading to faster runtimes. However, as many as 25 persons and 35 objects are involved, which

[1]Standard interpolation methods only work to a certain degree. Changes in direction between frames are still lost for example.

Table 5.1: Runtime for the primary models on a desktop computer with a six core Intel Xeon W3690 CPU: total runtime for each experiment (with $240s$ of $1fps$ input data) and average runtime per frame of input data.

	Runtime [$\boldsymbol{s}$]	Runtime [$\boldsymbol{s}/\mathit{frame}$]
Groups with speaking and listening members	1131	4.7
Groups w. sitting, standing, and moving members	959	4.0
Groups with joining and leaving members	2602	10.8
Groups oriented at referred object	563	2.3

leads to slower runtimes. The applied hardware and parallelization strategy also have a profound effect on runtime, in this case a desktop computer with a six core Intel Xeon W3690 using CPU multithreading. If two or more SGTs have to be evaluated in parallel, they share computing resources, effectively multiplying these runtimes by the number of SGTs that is running, unless they are running on separate machines. Finally, runtimes can be improved by optimizing the SGT traversal algorithm; by avoiding redundant evaluations and by preserving reasoning results that can be used again in other parts of the traversal or in different traversals across time for example.

5.5 Secondary Models

Below, the performance of the three models presented in Section 4.2 is evaluated.

5.5.1 Center, Object, and Person Oriented Groups (SGT 5)

The model around SGT 5 (Figure 4.14) was evaluated on 220 frames of annotated input data (3:40 min.). DBSCAN preprocessing was performed with $\varepsilon = 120cm$ and *maxMembers* = 7.[2] Figures 5.14 through 5.16 provide example results for the three situations that this model detects: ***group*** oriented at center, ***group*** oriented at *object*, and ***group*** oriented at *person*. In each of these figures, the result visualization on the left corresponds to the results and ground-truth plotted on the right. Besides the correct result[3] "s2b s3 oriented at person s4", Figure 5.16 shows some false positives, because the conditions for a center oriented group and a person oriented group sometimes hold simultaneously in constellations like this.

Performance. Figure 5.17 (top) presents the overall performance of this model. Figure 5.17 (bottom left) shows relatively low performance for center oriented groups. The gray area and dotted line at the bottom of this graph, representing number of results (depending on the truth value threshold) and number of ground-truth situations, suggest that this is partly due to the fact that too little ground-truth situations of this type were annotated. The instructions to the ground-truth annotator might have been too strict. Or it is due to the fact that too many results were detected, or, most likely, a combination of both. This is confirmed by low precision scores (red curve). The performance for object oriented groups (bottom center) is relatively high and the proportion of the number of results and ground-truth situations is good. Finally, for person oriented groups (bottom right), performance is low and there are too many ground-truth situations compared to the amount of results, reflected by low recall scores.

[2]DBSCAN's parameters ε and *maxMembers* are explained in Section 4.2.1, the latter towards the end of that section.

[3]i.e. the result that corresponds to the ground-truth annotations. Given the surrounding frames and *uac*'s interaction with message *m0*, the annotator did not classify him as part of this group.

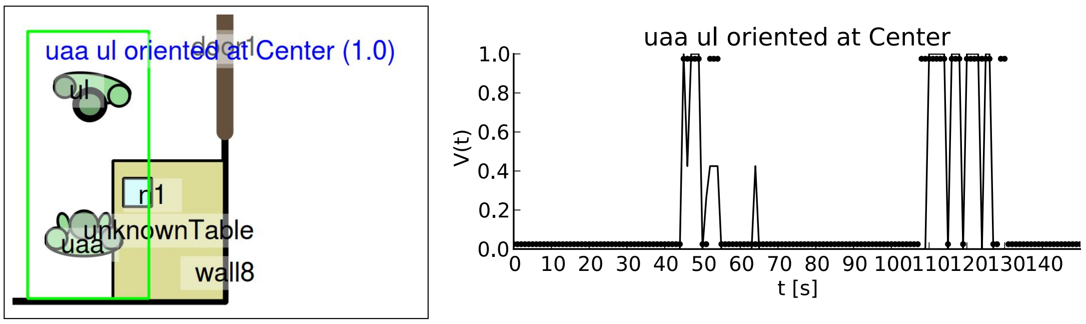


Figure 5.14: Visualized result (left) and corresponding results and ground-truth over time (right) for the situation ***group*** oriented at center.

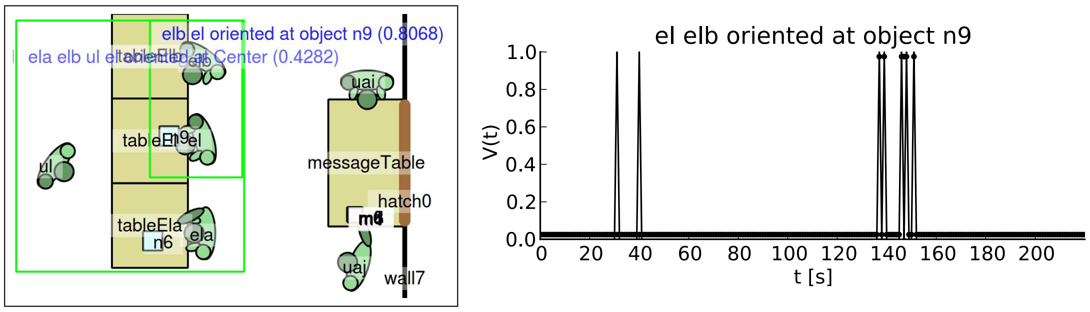


Figure 5.15: Visualized result (left) and corresponding results and ground-truth over time (right) for the situation ***group*** oriented at *object*.

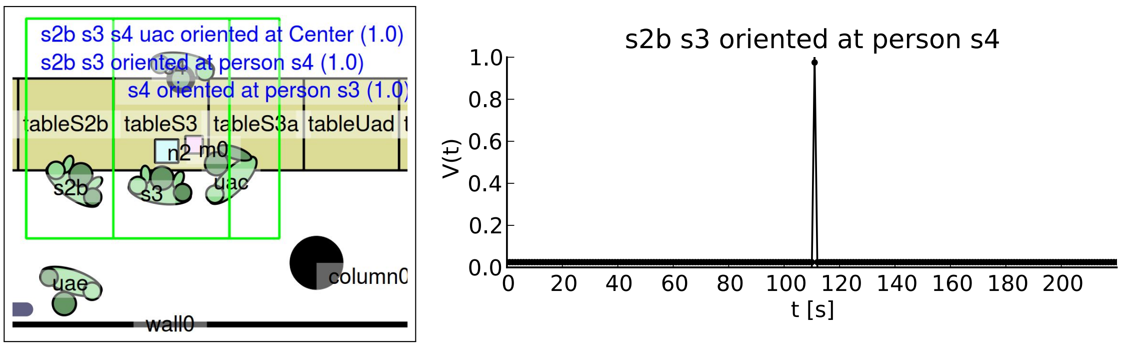


Figure 5.16: Visualized result (left) and corresponding results and ground-truth over time (right) for the situation ***group*** oriented at *person*.

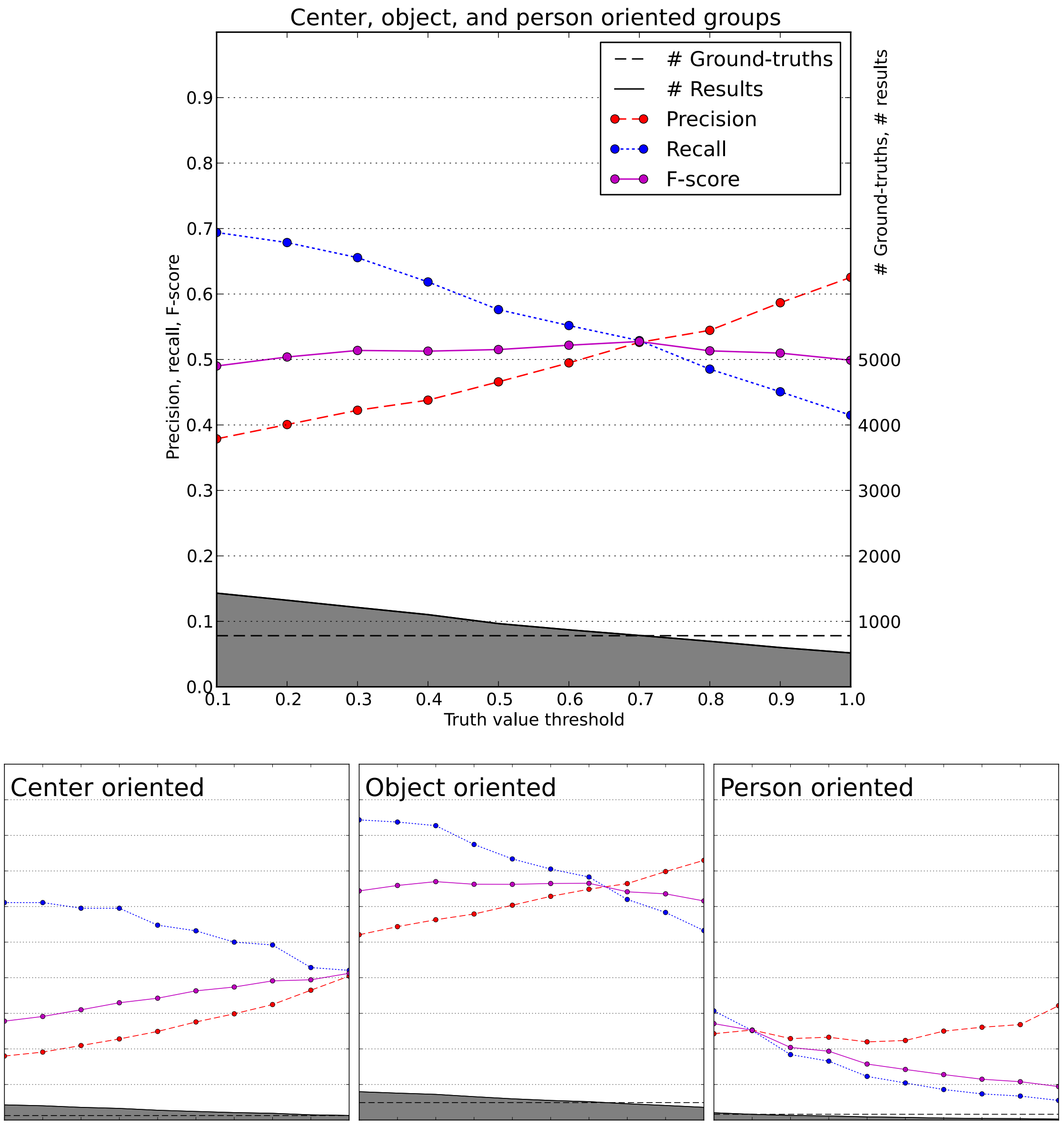


Figure 5.17: Overall performance for the model described in Section 4.2.3 (top) and performance for the three separate situations that it detects: ***group*** oriented at center, ***group*** oriented at *object*, and ***group*** oriented at *person*.

Inter-Annotator Agreement. Two out of three situations detected by this models show relatively poor performance, but is this entirely due to the models themselves? Besides the considerations under Section 5.3.2, one should keep in mind that the ground-truth annotations themselves are subjective interpretations of the input data. Low performance can be caused by a failing model or by the fact that a situation is inherently hard to classify by humans. In the latter case, agreement between reasoning results and ground-truth cannot be achieved, even with a good model.

To determine how much performance loss comes from a failing model and how much comes from a situation that is inherently hard to classify by humans, ground-truth from multiple annotators needs to be compared quantitatively. The most common way of doing this is using Cohen's kappa [38] for two annotators and Fleiss' kappa [53] (multi-annotator generalization of Scott's pi) for more than two annotators. Recent adaptations of these measures include [33, 64]. However, such measures are designed for "normal" classification problems where each instance belongs to exactly one class. Our problem space is different, because the system needs to generate complex descriptions instead of merely classifications, making these measures unusable. The F-score itself can be used instead, on two sets of ground-truth as opposed to on one set of reasoning results and one set of ground-truth. Note that Cohen's kappa compensates for the amount of agreement that occurs by chance and the F-score measure does not. In our problem space though, there are very many possible situations and only a few that are actually annotated, so this chance agreement is negligible.

Inter-annotator agreement was anlyzed for 220 frames with the model "center, object, and person oriented groups" (SGT 5) and ground-truth from three annotators. Tables 7.2 through 7.4 show the results of this analysis for the three different situations detected by this model: center oriented group, object oriented group, and person oriented group. Each cell reports the F-score for the comparison of two different ground-truth sets. The bottom row shows the F-scores for the three ground-truth sets evaluated against the reasoning results.[4] Finally, the two numbers in the right column report the average F-scores reflecting all comparisons between annotators and all comparisons between annotator and reasoning respectively.[5] When looking at the bottom rows of these tables, we see that reasoning performance fluctuates across different annotators, reflecting the ground-truth's subjectivity. This is also reflected by the comparisons between the different annotators.

For center oriented groups (Table 5.2) the F-score that reflects average inter-annotator agreement is 0.513, significantly higher than 0.353, the average maximum F-score that reflects reasoning performance. This means that there is room for improvement in detecting this situation, but that only a fraction of the way between 0.353 and 1.0 can be solved

[4]Here, the *maximum* F-scores over all truth value thresholds is used, because the reasoning results have fuzzy truth values. This is not needed when comparing ground-truth to ground-truth, because there are no fuzzy truth values then.

[5]Not to be confused with the average F-score over all truth value thresholds that is used elsewhere in this thesis.

through additonal modeling effort, because this situation is inherently hard to detect objectively. For object oriented groups (Table 5.3) the F-scores for inter-annotator agreement and reasoning performance are both 0.657. The fact that inter-annotator agreement equals reasoning performance suggests that the model can recognize situations of this types about as well as humans can. Additionaly, the relatively high value of these F-scores means that this situation type is easier to recognize than the other two (for reasoning and annotators alike). Finally, person oriented groups (Table 5.4) are hard to detect objectively for both annotators and reasoning, reflected by the relatively low average F-scores for both. However, the difference between the two average F-scores is relatively small, suggesting that reasoning performs almost as well as humans do.

Table 5.2: Inter-annotator agreement for center oriented groups. Each cell reports the F-score for comparison between two annotators. The bottom row reports the maximum F-score (over all truth value thresholds) from comparison between annotators and reasoning results. The right column reports the average F-scores between all annotators and between annotators and reasoning.

	Annotator 1	Annotator 2	Annotator 3	Average
Annotator 1	-	0.44	0.61	↘
Annotator 2	0.44	-	0.49	↘
Annotator 3	0.61	0.49	-	0.513
Reasoning	0.40	0.31	0.35	0.353

Table 5.3: Inter-annotator agreement for object oriented groups (see caption of Table 7.2).

	Annotator 1	Annotator 2	Annotator 3	Average
Annotator 1	-	0.67	0.66	↘
Annotator 2	0.67	-	0.64	↘
Annotator 3	0.66	0.64	-	0.657
Reasoning	0.67	0.62	0.68	0.657

Table 5.4: Inter-annotator agreement for person oriented groups (see caption of Table 7.2).

	Annotator 1	Annotator 2	Annotator 3	Average
Annotator 1	-	0.41	0.40	↘
Annotator 2	0.41	-	0.46	↘
Annotator 3	0.40	0.46	-	0.423
Reasoning	0.39	0.33	0.39	0.370

Robustness against Noise. In order to evaluate SGT 5's robustness against noise, the tool described in Section 3.5 was used to add increasing amounts of noise to the *position*, *orientation*, *lookingDown*, and *sitting* attributes of the persons in the annotated input data. The result over 150 frames is visualized in the four graphs in Figure 5.18. Each of these four graphs plots the average F-score[6] over the amount of noise on one of the four attributes mentioned above. They contain curves for the overall average F-score (turquois) as well as the separate average F-scores for each of the three situation types the model detects (purple, yellow, and red). Each colored curve is the average over four separate runs, visualized by the gray curves underneath. Four runs were performed for each graph, because the tool described in Section 3.5 adds noise in a non-determinisitic fashion, i.e. these are repeated random experiments. At 0 noise (left-most x-value) the average F-score is the same in all four graphs in Figure 5.18, because this corresponds to a normal experiment without noise.[7] The horizontal lines in the bottom two graphs indicate that the attributes *lookingDown* and *sitting* are not used in the conditions of all three situations that are recognized by SGT 5.

Effect of DBSCAN Parameter *maxMembers* on Performance. Towards the end of Section 4.2.1, the additional parameter for DBSCAN *maxMembers* was introduced to facilitate real-time/timely performance. Figure 5.19 shows that, while having a profound positive effect on runtime, the use of *maxMembers* does not have a negative effect on system performance in this particular case. For *maxMembers* $= 6$ (right), precision, recall, and F-score are almost identical to the graph on the left, without *maxMembers*.[8]

[6]Average F-score: $F_{avg} = \frac{\sum_{tvt=0.1}^{1.0} F_{tvt}}{9}$, where *tvt* stands for truth value threshold, with stepsize 0.1, and F_{tvt} stands for the F-score at truth value threshold *tvt*, as visualized in Figure 5.17 for example.

[7]Note that the average F-scores at 0 noise in Figure 5.18 do not correspond to Figure 5.17, because different input data were used. Furthermore, the average F-scores at 0 noise in the top right graph do not correspond to the other three graphs, because it was generated using a slightly different set of data and ground-truth than the other three.

[8]Note that this comparison was performed using trapezoid truth functions with learned parameters (see Section 4.2.2). Hence, the left graph in Figure 5.19 corresponds to the center right graph in Figure 5.20 rather than the top graph in Figure 5.17.

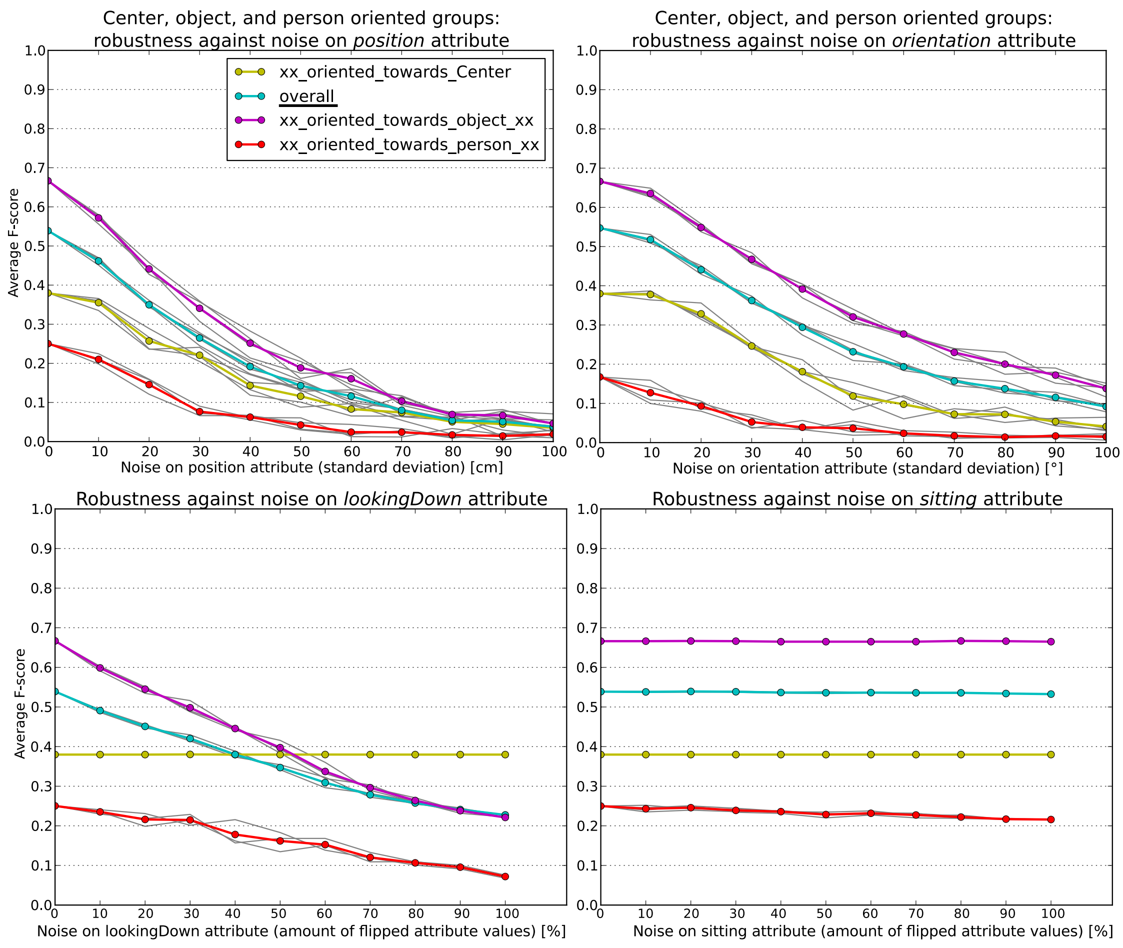


Figure 5.18: Center, object, and person oriented groups, robustness against noise.

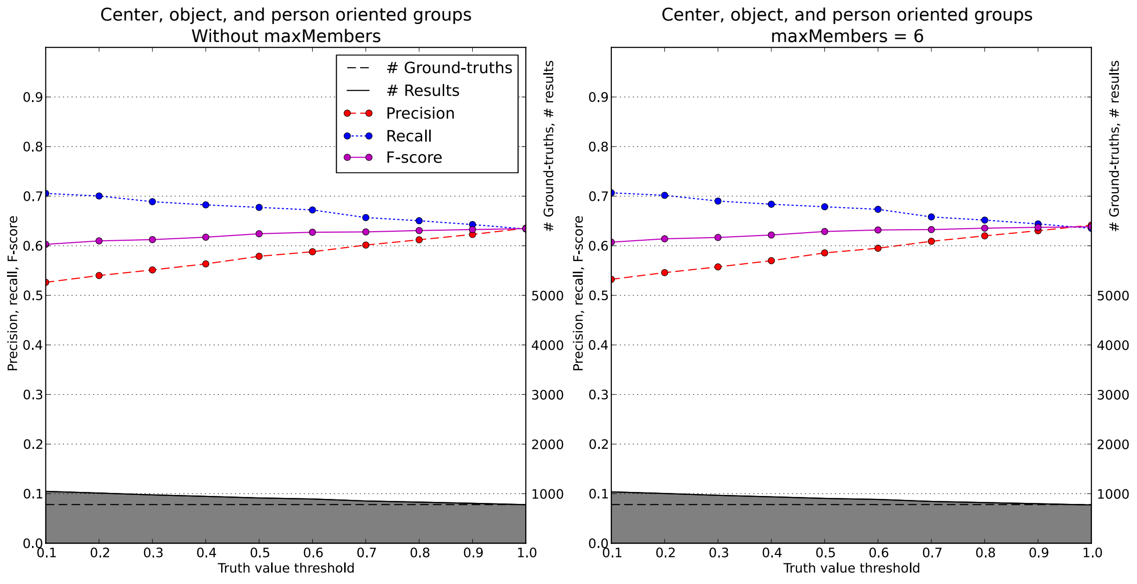


Figure 5.19: Center, object, and person oriented groups, effect of *maxMembers*.

Effect of Parameter Learning on Performance. Section 4.2.2 discusses a method for learning the parameters of trapezoid truth functions using maximization of the adapted F-score performance measure F_{α_x}-score (Equations 4.31 and 4.32). The evaluation of this method for "center, object, and person oriented groups" (SGT 5) is visualized in Figure 5.20. These graphs have the same axes and legend as Figure 5.17 (top). Note that the left graph corresponds to Figure 5.17 (top) because this experiment was performed under the same conditions as the standard performance evaluation of this model.

Between the left (parameters of trapezoid truth functions manually determined) and center left graph (parameter learning using standard F-score) we see a significant performance increase. Then, in the center right (parameter learning using $\mathrm{F}_{\alpha_{x=3}}$-score) and right graph (parameter learning using $\mathrm{F}_{\alpha_{x=10}}$-score), maximum performance stays the same while the fuzzyness of the results increases. The advantage of this increase in fuzzyness is explained in detail in Section 4.2.2 – Scoring Function. Increased fuzzyness is shown by the steeper slopes in opposite directions of the red and blue curves (precision and recall) as well as the increased angle of the gray area below. For the sake of completeness, the dark purple curves in the center right and right graph show the $\mathrm{F}_{\alpha_{x=3}}$-score and $\mathrm{F}_{\alpha_{x=10}}$-score respectively. The center right graph provides a good balance between high average performance over all truth value thresholds and fuzzyness of the results (the advantage of which is explained in Section 4.2.2 – Scoring Function).

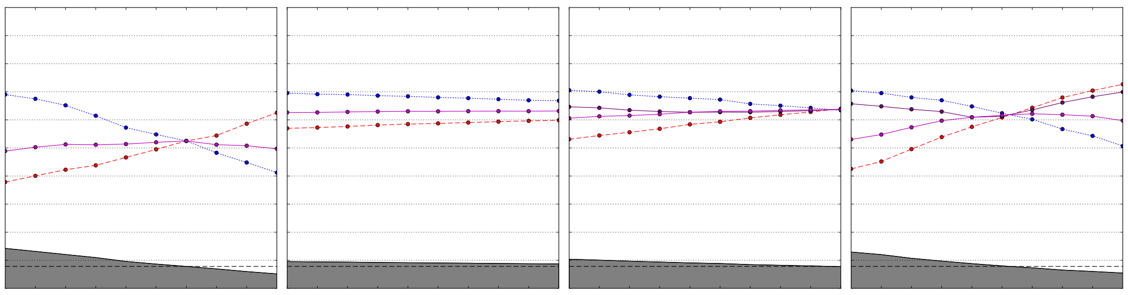


Figure 5.20: Evaluation of parameter learning for "center, object, and person oriented groups" (same axes and legend as Figure 5.17, top). Left: parameters of trapezoid truth functions manually determined, center left: parameter learning using standard F-score, center right: parameter learning using $\mathrm{F}_{\alpha_{x=3}}$-score, and right: parameter learning using $\mathrm{F}_{\alpha_{x=10}}$-score. The dark purple curves in the center and right graph show the $\mathrm{F}_{\alpha_{x=3}}$-score and $\mathrm{F}_{\alpha_{x=10}}$-score respectively.

5.5.2 Monologues and Dialogues (SGT 6)

The model from Section 4.2.3 was evaluated on 250 frames with DBSCAN's $\varepsilon = 120cm$ and $maxMembers = 7$. Figures 5.21 and 5.22 provide example result visualizations with corresponding ground-truth and result plots for two of the situations that this model detects: *person* talking to ***group*** about *object*, and ***group*** in dialogue about *object*. In the two consecutive frames visualized in Figure 5.21, and in the frames around it, *s4* and *s4a* speak alternatingly while being close to each other and oriented at each other, so a dialogue is detected. Furthermore, they are close to and oriented at *n0* during these frames, so the output “s4a s4 in dialgoue about n0” is generated. In Figure 5.22, similar conditions hold around the display *unitReqeusts* while *uax0* is speaking, in the visualized frame and the frames around it, so the output “uax0 talking to uab about unitRequests” is generated. Figure 5.23 (top) shows the overall performance for this model. In the bottom graphs, we see that the model performs relatively well for three out of four situations. “Unspecific monologue” (right center) shows low performance. The proportion of number of results and number of ground-truth situations and the low recall curve tell us that the reasoning engine finds too little situations of this type. This could be due to the fact that the conditions of the other three situations apply more strongly.

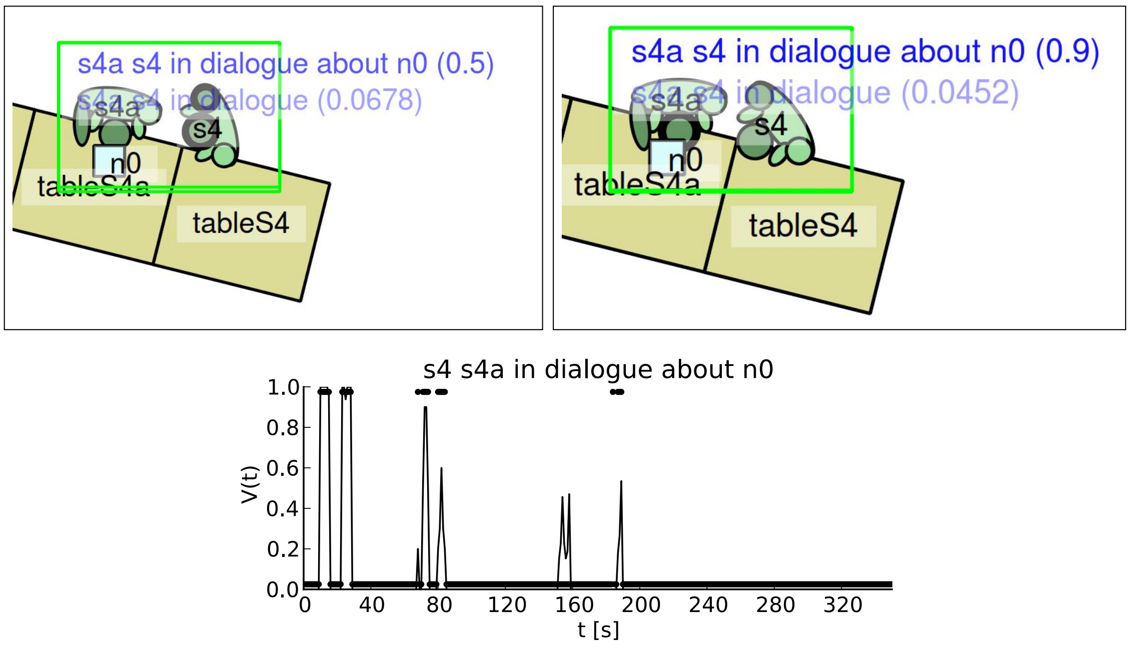


Figure 5.21: Two consecutive frames of a visualized result (top) and corresponding results and ground-truth over time (bottom) for the situation ***group*** in dialogue about *object*.

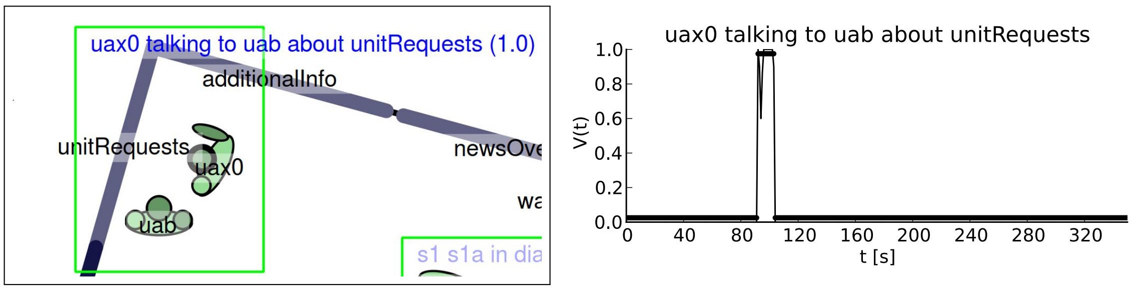


Figure 5.22: Visualized result (left) and corresponding results and ground-truth over time (right) for the situation *person* talking to ***group*** about *object*.

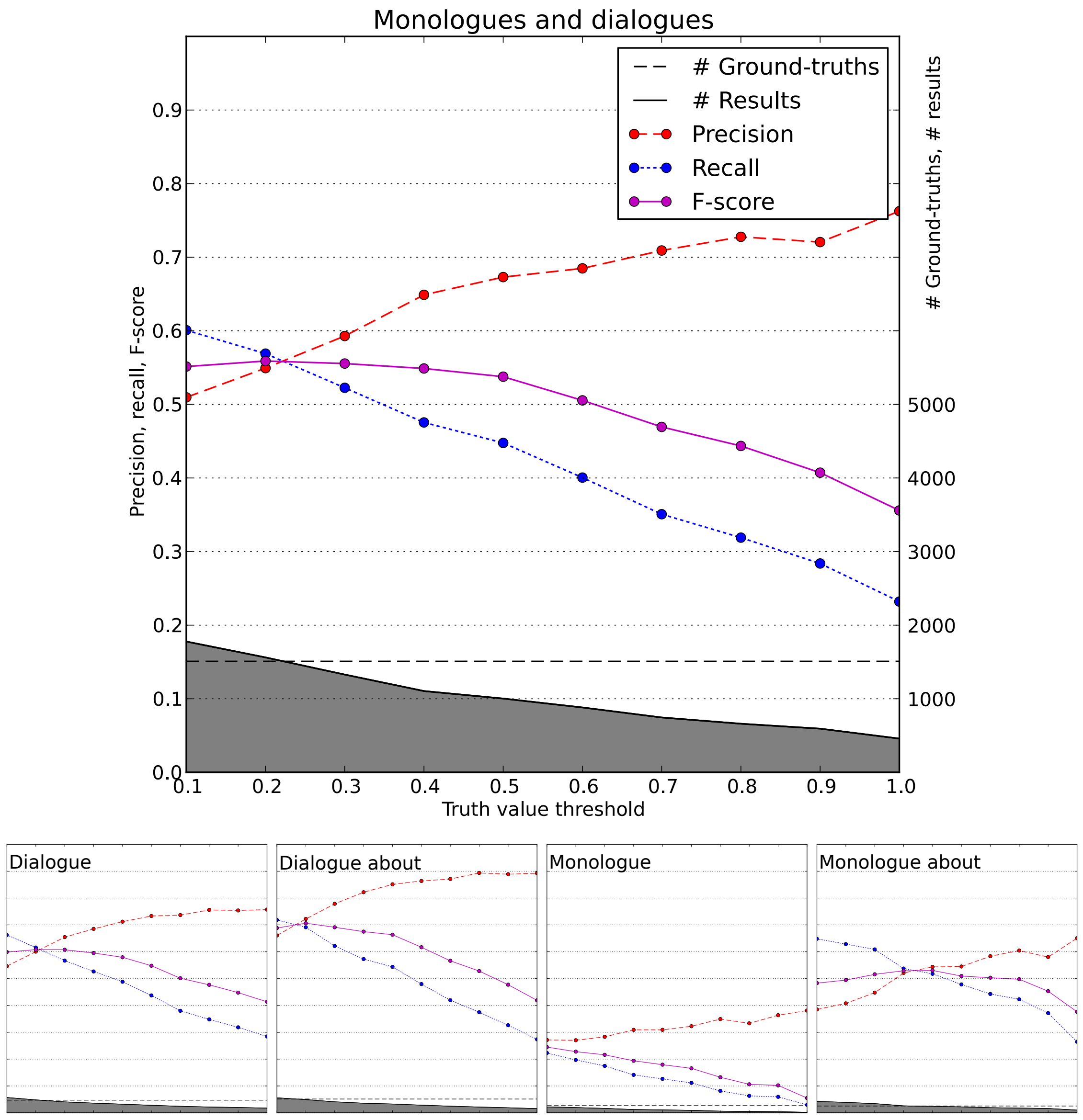


Figure 5.23: Overall performance for the model described in Section 4.2.3 (top) and performance for the four separate situations that it detects: *person* talking to ***group***, *person* talking to ***group*** about *object*, ***group*** in dialogue, and ***group*** in dialogue about *object*.

5.5.3 Handling Documents (SGT 7)

The very last was performed on 200 frames, without the use of DBSCAN, because this model detects situations involving individuals and single documents as opposed to groups. Figure 5.24 displays four consecutive frames; an example result of the temporal part of the SGT from Figure 4.21. After "ul picked up n1" is detected, the system goes on to find "ul carrying n1". The situation "ul moving n1" is not yet detected, because *ul*'s speed is minimal. In Figure 5.25, we find another set of three consecutive frames, showing the second half of the temporal chain. The situation is correctly described as "ul moving n1" followed by the end state "ul laying down n1". Finally, examples of correctly detected writing and reading situations are displayed in Figure 5.26 (not consecutive frames).

The overall performance for this model and the performance for the six detected situations is provided in Figure 5.27. Again, performance varies across the situation types that are offered by the model. Just like in Section 5.5.1, this model's robustness against noise was evaluated by analyzing the average F-score (over all truth value thresholds) for different amounts of noise on the object attributes involved. The result is shown in Figure 5.28. Each colored curve is the average over multiple runs (gray curves); two for the *position* attribute, three for *orientation*, and two for *lookingDown*. The overall performance is visualized by the red curve. Note that this model is sensitive to noise on the *position* attribute, compared to the top left graph in Figure 5.18, because the trapezoid truth functions that are used to associate distances between persons and documents with distance categories use low values to avoid false positive detections. The bottom graphs in Figure 5.28 show that noise on the attributes *orientation* and *lookingDown* have little influence.

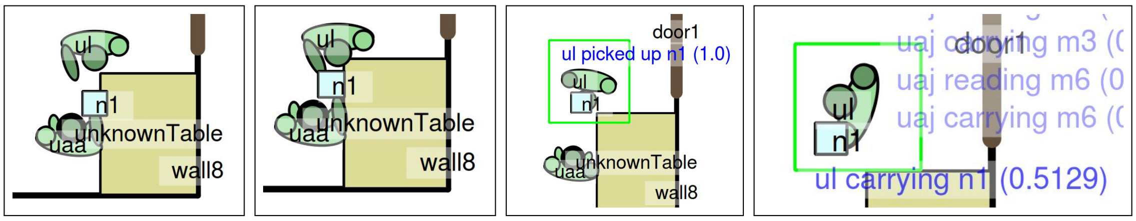


Figure 5.24: Four consecutive frames of visualized results for the situations *person* picked up *document* and *person* carrying *document*.

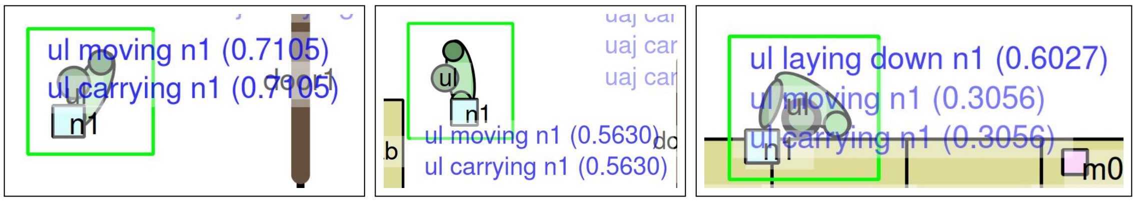


Figure 5.25: Three consecutive frames of visualized results for the situations *person* moving *document*, and *person* laying down *document*.

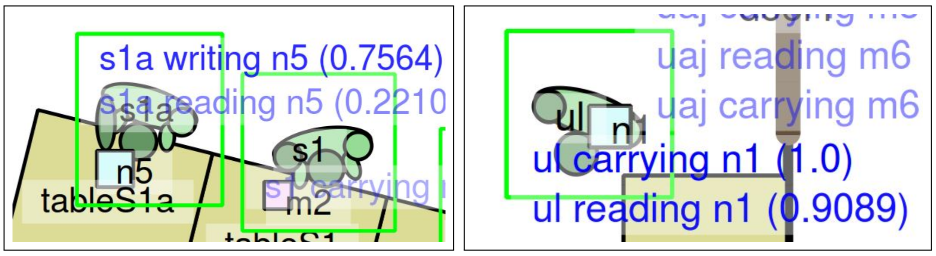


Figure 5.26: Two non-consecutive frames of visualized results for the situations *person* writing *document* and *person* reading *document*.

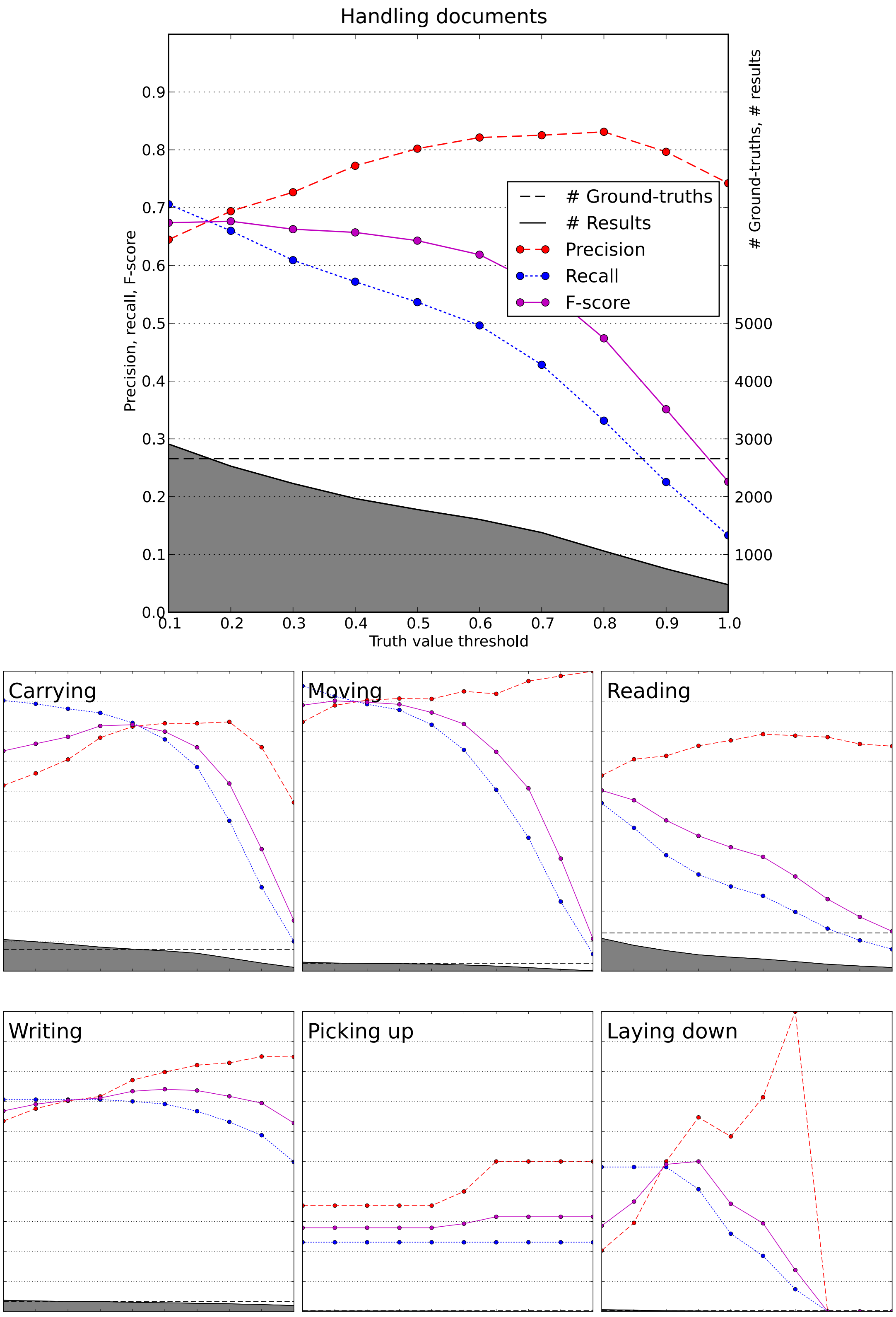


Figure 5.27: Overall performance for the model described in Section 4.2.3 (top) and performance for the six separate situations that it detects: *person* carrying *document*, *person* moving *document*, *person* reading *document*, *person* writing *document*, *person* picking up *document*, and *person* laying down *document*.

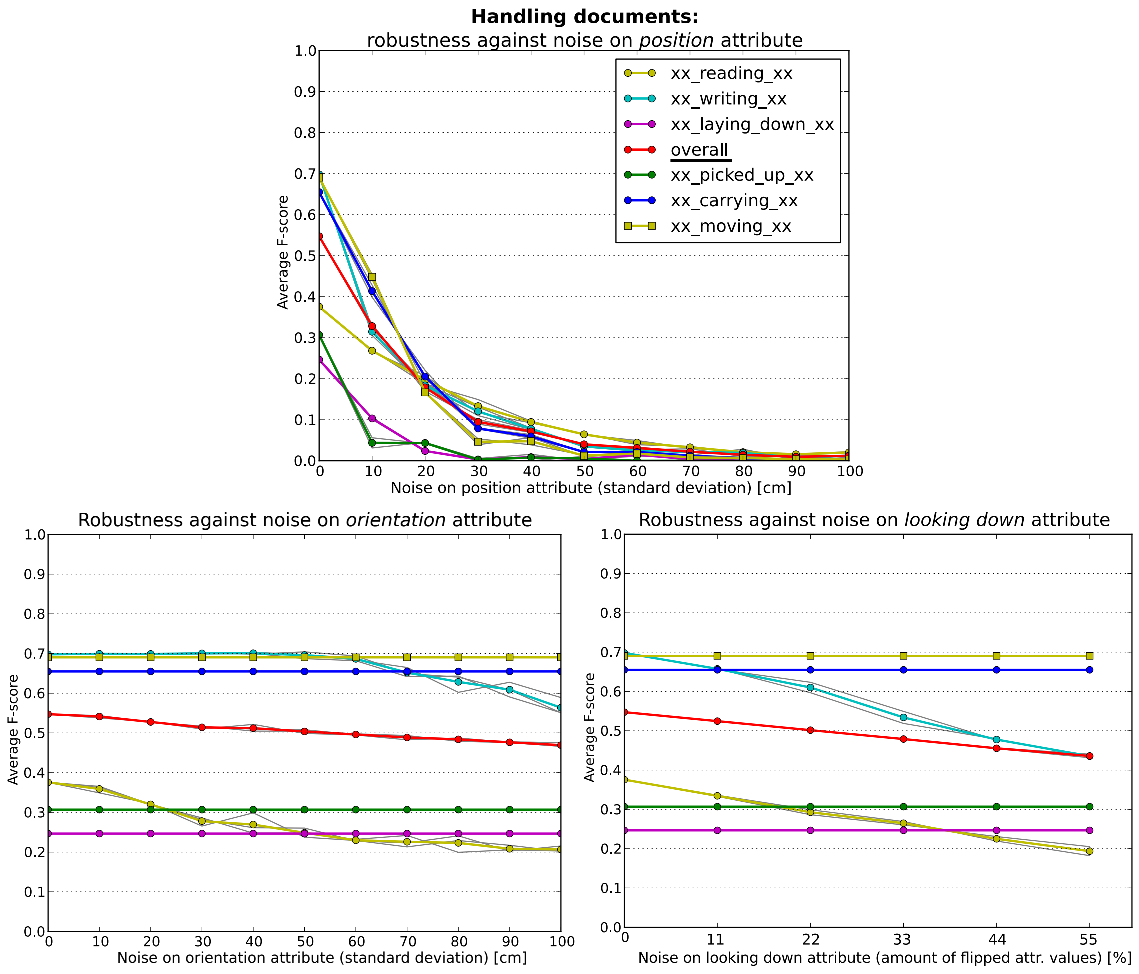


Figure 5.28: Persons handling documents, robustness against noise.

That concludes Chapter 5. The applied evaluation methods were introduced in Section 5.1. Then, the primary models described in Section 4.1 were evaluated in Section 5.2. The corresponding error analysis and runtime were presented in Sections 5.3 and 5.4. Finally, the secondary models described in Section 4.2 were evaluated in Section 5.5, introducing some additional evaluation methods.

Chapter 6

Conclusion

The thesis is concluded with a summary and a discussion on future work.

6.1 Summary

The goal of this study was to develop a reasoning system for interaction analysis and its evaluation in a case study: automatic report generation for training purposes in crisis response control rooms. The developed reasoning system should also be applicable to other application domains. Situation descriptions about groups and interactions are generated from annotated hypothetical machine perception, using fuzzy metric temporal logic (FMTL) and situation graph trees (SGTs). The FMTL/SGT approach was chosen, because it provides the expressive power and introspection that is needed for the performed case study. Domain knowledge can be formalized into logic formulas and tree structures relatively easily, because the logic paradigm is intuitive to use and close to human reasoning. The FMTL/SGT approach provides a rich language for reasoning with complex models that are understandable by humans. The annotated input data consists of person tracks, object information, and information about gestures, body pose, and speech activity, used as input by the FMTL/SGT reasoning engine to deduce situation descriptions: various group constellations and interaction patterns that can be used for automatically generated behavior reports. The described problem is not a classification problem between a few classes. Situations must be classified correctly and all involved persons and objects must be consistent with ground-truth.

The contributions of this thesis are as follows. The case study along which the presented system was developed, is both unique and challenging. In previous work, increased situational awareness is achieved by modeling the site of the (simulated) crisis using geographical information systems and other software tools, but the situation inside the control room has never been modeled and used. This contribution is found in *Section 1.1.3, Chapter 3, and Chapter 4*. Further scientific novelty comes from the reasoning models that were developed. The FMTL/SGT methods were applied to interaction analysis for the first time. In the past, similar methods have only been applied in traffic and surveillance settings. The developed models can recognize multiple group constellations and interaction patterns within them, as opposed to mainly movement patterns of one or two vehicles or persons. The developed reasoning models implement some powerful and widely applicable fuzzy spatiotemporal concepts. This contribution is found in *Chapter 4.* Additionaly, FMTL/SGT reasoning was combined with a customized clustering algorithm, and the use of parameter learning in the FMTL/SGT context was investigated. These contributions are found in *Section 4.2.* Supporting contributions come from the enabling software toolkit that was developed and the evaluation that was conducted. These supporting contributions are found in *Chapters 3 and 5.* Overall, the study showed promising results, but also room for improvement. Ideas for future work that were gathered during the study are summarized below.

6.2 Future Work

There are plenty of opportunities to build upon the work presented in this thesis. One can create new models and perform additional evaluations. The performed case study can be extended upon and the presented research can be applied to other application domains. Finally, improvements and alternatives to the applied reasoning methods should be investigated.

6.2.1 Additional Models and Evaluations

The FMTL rules, SGTs, and supporting components (such as parameter learning) can be improved. Some ideas for this have been presented in Section 5.3. Furthermore, more sophisticated experiments can be performed using the presented dataset, with some new, more powerful models and new ground-truth. For example, one could model briefings (a recurring situation in control room operations) and their distinct phases, as well as persons underway between specific locations, and the delivery and processing of messages (extending the model from Section 4.2.3). The current models use the proximity between members as a necessary condition for groups. Future models should also be able to recognize groups in which members are far apart, based on their orientations, interaction patterns, and context information. The models in Chapter 4 can provide the basis for more sophisticated models.

Chapter 5 provides a wide range of evaluation types that can be performed in the future: overall and situation specific performance, runtime analysis, error analysis, inter-annotator agreement, effect of parameter learning on performance, and robustness against noise (plus other types of imperfections, see Section 2.7.1 – Handling Uncertainty in the Input Data). Another option is the use of n-valued or fuzzy ground-truth instead of binary ground-truth. Furthermore, the alternative performance criteria under Section 5.3.2 should be investigated further (e.g. using a performance metric that is similar to "word error rate"). Last but not least, it would be interesting to compare the behavior and performance of different models, e.g. models with and without clustering as preprocessing,[1] or between different reasoning methods entirely (see Section 6.2.3 below).

6.2.2 Extended Case Studies and Other Application Domains

End-users, human science experts, and developers of crisis response software should be involved in further development. The current case study is focused on the physical attributes of the people and objects in the room, but the system can be improved by taking into account more domain specific attributes i.e. context information: field unit status, crisis dynamics, staff roles, and more sophisticated object information. These could largely be obtained by monitoring developments in the crisis response software that is used in many control rooms. Speech recognition (e.g. keyword analysis through close-talking microphones) would be another valuable source of context information. Such context information would allow one to model more sophisticated domain knowledge in FMTL and SGTs to deduce a richer set of situation descriptions that is of greater use to potential end-users. Finally, one needs to consider the ethical implications and liability issues following from such technologies.

The presented system can be applied to other application domains such as the ones described in Section 1.1.1. A first step could be to deduce a generic approach and best practices from the presented case study. These could then be applied to other application domains, preferably with fully automatic machine perception instead of annotations. Whether the machine perception that is required for the presented case study and similar problems can be obtained through cameras and microphones alone is questionable, at least for the near future. Additional sensors such as RFID tags could help to provide the data quality that the models need. In parallel, their use in other application domains should guarantee that the methods and models are not tailormade for a specific application, but broadly applicable with minor reconfiguration.

6.2.3 Improved and Alternative Methods

The applied methods can be improved through better system development processes. The reasoning models were implemented using F-Limette, an FMTL reasoning engine (similar to Prolog) written in C, and the SGT-Editor, a Java application for editing and traversing SGTs. These programs are outdated, better developer support tools are required, and the re-use of existing code should be facilitated. The features and usability of the developed data annotation, ground-truth annotation, and evaluation tools could also be improved in the future. Additional features that could be added to the FMTL/SGT framework include asserting new knowledge at runtime (already possible to some extent), checking the consistency of reasoning results, and adding conditional SGT edges that could enforce mutual exclusivity between branches for example (by deducing a situation and setting

[1]Although the SGTs without clustering in Figures 4.1 and 4.4 are similar to the SGTs with clustering in Figures 4.15 and 4.14 respectively, their behavior and performance could not be compared directly.

it as a negative constraint for another situation). Finally, runtimes can be improved by optimizing the SGT traversal algorithm; by avoiding redundant evaluations and by preserving reasoning results that can be used again in other parts of the traversal or in different traversals across time for example.

For sure, statistical, syntactic, hybrid, and other description-based approaches need to be investigated as well (see Chapter 2) with and without the use of parameter learning and structure learning. Promising candidates include Markov logic networks, Bayesian logic networks, and-or graphs, and dynamic Bayesian networks. Clustering algorithms could also play a role here. The clustering strategy can be extended across multiple frames, a fuzzy version of DBSCAN can be developed relatively easily, and additional attributes can be incorporated, increasing expressive power and potentially replacing (part of) the logic reasoning on top.

Bibliography

Own Publications

[1] Y. Fischer, J. IJsselmuiden (2011) *A Bayesian network approach to maritime situation assessment.* Maritime Anomaly Detection Workshop (MAD).

[2] J. IJsselmuiden, R. Stiefelhagen, F. van de Camp, A. Schick, M. Voit (2009) *Towards a smart control room for crisis response using visual perception of users.* International Conference on Information Systems for Crisis Response and Management (ISCRAM), poster only.

[3] J. IJsselmuiden, R. Stiefelhagen (2010) *Towards high-level human activity recognition through computer vision and temporal logic.* German Conference on Artificial Intelligence (KI), 426–435.

[4] J. IJsselmuiden, T. Körner, A. Schick, R. Stiefelhagen (2010) *Interaktionstechniken für große Darstellungsflächen.* Fachausschussitzung Anthropotechnik, Deutsche Gesellschaft für Luft- und Raumfahrt.

[5] J. IJsselmuiden, A.K. Grosselfinger, D. Münch, M. Arens, R. Stiefelhagen (2012) *Automatic behavior understanding in crisis response control rooms.* Conference on Ambient Intelligence (AmI), 97–112.

[6] J. IJsselmuiden, D. Münch, A.K. Grosselfinger, M. Arens, R. Stiefelhagen (2014) *Automatic understanding of group behavior using fuzzy temporal logic.* Journal of Ambient Intelligence and Smart Environments (JAISE), to appear.

[7] D. Münch, J. IJsselmuiden, M. Arens, R. Stiefelhagen (2011) *High-level situation recognition using fuzzy metric temporal logic, case studies in surveillance and smart environments.* Workshop on Analysis and Retrieval of Tracked Events and Motion in Imagery Streams (ARTEMIS) @ ICCV.

[8] D. Münch, J. IJsselmuiden, A.K. Grosselfinger, M. Arens, R. Stiefelhagen (2012) *Rule-based high-level situation recognition from incomplete tracking data.* Symposium on Rules, Research Based and Industry Focused (RuleML).

[9] D. Reich, F. Putze, D. Heger, J. IJsselmuiden, R. Stiefelhagen, T. Schultz (2011) *A real-time speech command detector for a smart control room.* Conference of the International Speech Communication Association (INTERSPEECH), 2641–2644.

[10] A. Schick, F. van de Camp, J. IJsselmuiden, R. Stiefelhagen (2009) *Extending touch: towards interaction with large-scale surfaces.* ACM Interactive Tabletops and Surfaces Conference (ITS), 117–124.

[11] R. Stiefelhagen, F. van de Camp, J. IJsselmuiden, M. Voit, A. Schick (2012) *Videobasierte Wahrnehmung des Menschen und innovative Mensch-Maschine Interaktion – Anwendungen für Kontrollräume und Leitstellen.* Karlsruher Leittechnisches Kolloquium, 214–222.

[12] M. Voit, F. van de Camp, J. IJsselmuiden, A. Schick, R. Stiefelhagen (2013) *Visuelle Perzeption für die multimodale Mensch-Maschine-Interaktion in und mit aufmerksamen Räumen.* at – Automatisierungstechnik, 784–791.

Supervised Student Theses

[13] J. Dornheim, J. IJsselmuiden (advisor) (2014) *Methodische Erweiterung einer Prozesskette zur Interaktionsanalyse mittels unscharfer temporaler Logik.* Master's thesis, Karlsruhe University of Applied Sciences, Faculty of Computer Science and Business Information Systems.

[14] I. Grigoriev, J. IJsselmuiden (advisor) (2013) *Fuzzy temporal modeling of group constellations and interaction patterns in smart environments.* Bachelor's thesis, Karlsruhe Institute of Technology, Department of Informatics.

[15] S. Kaltwang, J. IJsselmuiden (advisor) (2011) *High-level activity recognition through interaction modeling and graph analysis.* Diploma thesis, Karlsruhe Institute of Technology, Department of Informatics.

[16] T. Körner, J. IJsselmuiden (advisor) (2010) *Benutzeroberflächen für die Zeigegesten-Interaktion mit einer Videowand.* Diploma thesis, Karlsruhe Institute of Technology, Department of Informatics.

[17] I. Papantonis, J. IJsselmuiden (advisor) (2013) *Automatic behavior understanding in smart work environments using fuzzy metric temporal logic and situation graph trees.* Bachelor's thesis, Karlsruhe Institute of Technology, Department of Informatics.

[18] D. Reich, F. Putze, D. Heger, J. IJsselmuiden (advisors) (2009) *Integration of automatic speech recognition into a smart control room.* Study thesis, Karlsruhe Institute of Technology, Department of Informatics.

[19] D. Reich, F. Putze, D. Heger, J. IJsselmuiden (advisors) (2011) *A real-time speech command detector for a smart control room.* Diploma thesis, Karlsruhe Institute of Technology, Department of Informatics.

Related Work

[20] J.K. Aggarwal, M.S. Ryoo (2011) *Human activity analysis: A review.* ACM Computing Surveys 43(3), 16:1–16:43.

[21] J.F. Allen, G. Ferguson (1994) *Actions and events in interval temporal logic.* Logic and Computation 4, 531–579.

[22] Y. Aloimonos, G. Guerra-Filho, A. Ogale (2009) *The language of action: A new tool for human-centric interfaces.* Human Centric Interfaces for Ambient Intell., 95–131.

[23] G.A. Antonelli (2012) *Non-monotonic logic.* In: E.N. Zalta (ed.), The Stanford Encyclopedia of Philosophy,
http://plato.stanford.edu/archives/win2012/entries/logic-nonmonotonic/.

[24] M. Arens (2005) *Repräsentation und Nutzung von Verhaltenswissen in der Bildfolgenauswertung.* PhD thesis, University of Karlsruhe, Department of Informatics.

[25] M. Arens, R. Gerber, H.H. Nagel (2008) *Conceptual representations between video signals and natural language descriptions.* Image and Vision Comp. 26(1), 53–66.

[26] J.C. Augusto, C.D. Nugent (editors) (2006) *Designing smart homes, The role of artificial intelligence.*

[27] A. Aztiria, J.C. Augusto, R. Basagoiti, A. Izaguirre, D.J. Cook (2012) *Discovering frequent user-environment interactions in intelligent environments.* Personal and Ubiquitous Computing 16, 91–103.

[28] L. Bacon, L. MacKinnon, A. Cesta, G. Cortellessa (2013) *Developing a smart environment for crisis management training.* Journal of Ambient Intelligence and Humanized Computing 4, 581–590.

[29] N. Bellotto, B. Benfold, H. Harland, H.H. Nagel, N. Pirlo, I. Reid, E. Sommerlade, C. Zhao (2012) *Cognitive visual tracking and camera control.* Computer Vision and Image Understanding 116(3), 457–471.

[30] N. Bellotto (2012) *Robot control based on qualitative representation of human trajectories.* AAAI Symposium on Designing Intelligent Robots: Reintegrating AI.

[31] C. Belta, A. Bicchi, M. Egerstedt, E. Frazzoli, E. Klavins, G.J. Pappas (2007) *Symbolic planning and control of robot motion, Grand challenges of robotics.* Robotics and Automation Magazine 14(1), 61–70

[32] J.O. Berger (1985) *Statistical decision theory and Bayesian analysis.*

[33] P.K. Bhowmick, P. Mitra, A. Basu (2008) *An agreement measure for determining inter-annotator reliability of human judgements on affective text.* Workshop on Human Judgements on Affective Text, 58–65.

[34] W. Bohlken, B. Neumann (2009) *Generation of rules from ontologies for high-level scene interpretation.* Symp. on Rule Interchange and Applic. (RuleML), 93–107.

[35] O. Brdiczka, M. Langet, J. Maisonnasse, J.L. Crowley (2009) *Detecting human behavior models from multimodal observation in a smart home.* Automation Science and Engineering 6(4), 588–597.

[36] O. Brdiczka (2010) *Integral framework for acquiring and evolving situations in smart environments.* Journal of Ambient Intelligence and Smart Environments (JAISE) 2(2), 91–108.

[37] P. Chahuara, A. Fleury , F. Portet, M. Vacher (2012) *Using markov logic network for on-line activity recognition from non-visual home automation sensors.* Conference on Ambient Intelligence (AmI), 177–192.

[38] J. Cohen (1960) *A coefficient of agreement for nominal scales.* Educational and Psychological Measurement 20 (1), 37–46.

[39] A. Colmerauer, P. Roussel (1993) *The birth of Prolog.* ACM SIGPLAN Notices 28(3), 37–52.

[40] B. Culik, T. Lehmann, C. Zebermann (2012) *PITAS: Pirate and terrorist aversion system.* Future Security, 337–346.

[41] P. Dai, H. Di, L. Dong, L. Tao, G. Xu (2008) *Group interaction analysis in dynamic context.* IEEE Transactions on Systems, Man, and Cybernetics 38(1), 275–282.

[42] E. Dantsin, T. Eiter, G. Gottlob, A. Voronkov (2001) *Complexity and expressive power of logic programming.* ACM Computing Surveys 33(3), 374–425.

[43] L. Demey, B. Kooi, J. Sack (2013) *Logic and probability.* In: E.N. Zalta (ed.), The Stanford Encyclopedia of Philosophy,
http://plato.stanford.edu/archives/spr2013/entries/logic-probability/.

[44] G. de Vries, W.R. van Hage, M. van Someren (2010) *Comparing vessel trajectories using geographical domain knowledge and alignments.* Conference on Data Mining Workshops (ICDMW), 209–216.

[45] P. Domingos, S. Kok, H. Poon, M. Richardson, P. Singla (2006) *Unifying logical and statistical AI.* National Conference on Artificial intelligence, 2–7.

[46] M. Ester, H.P. Kriegel, J. Sander, X. Xu (1996) *A density-based algorithm for discovering clusters in large spatial databases with noise.* Conference on Knowledge Discovery and Data Mining (KDD), 226–231.

[47] C. Fernández, P. Baiget, F.X. Roca, J. González (2011) *Determining the best suited semantic events for cognitive surveillance.* Expert Systems with Applications 38, 4068–4079.

[48] C. Fernández, P. Baiget, F.X. Roca, J. González (2011) *Augmenting video surveillance footage with virtual agents for incremental event evaluation.* Pattern Recognition Letters 32(6), 878–889.

[49] C. Filippaki, G. Antoniou, I. Tsamardinos (2011) *Using constraint optimization for conflict resolution and detail control in activity recognition.* Conference on Ambient Intelligence (AmI), 51–60.

[50] Y. Fischer, J. Beyerer (2012) *Defining dynamic bayesian networks for probabilistic situation assessment.* Conference on Information Fusion (FUSION), 888–895.

[51] Y. Fischer, J. Beyerer (2012) *A top-down-view on intelligent surveillance systems.* Conference on Systems (ICONS), 43–48.

[52] T. Flaminio, F. Montagna (2005) *A logical and algebraic treatment of conditional probability.* Archive for Mathematical Logic 44(2), 245–262.

[53] J.L. Fleiss (1971) *Measuring nominal scale agreement among many raters.* Psychological Bulletin 76(5), 378–382.

[54] *Feuerwehr Dienstvorschrift 100* (1999) Beschlossene Fassung des AFKzV.

[55] R. Gerber, H.H. Nagel (2008) *Representation of occurrences for road vehicle traffic.* Artificial Intelligence 172(4-5), 351-391.

[56] C. Geib (2009) *Delaying commitment in plan recognition using combinatory categorial grammars.* Int. Joint Conference on Artificial Intelligence (IJCAI), 1702–1707.

[57] S. Gong, T. Xiang (2003) *Recognition of group activities using dynamic probabilistic networks.* International Conference on Computer Vision (ICCV), 742–749.

[58] S. Gong, T. Xiang (2011) *Visual analysis of behaviour, From pixels to semantics.*

[59] J. González, D. Rowe, J. Varona, F.X. Roca (2009) *Understanding dynamic scenes based on human sequence evaluation.* Image and Vision Comp. 27(10), 1433–1444.

[60] B. Gottfried, H. Aghajan (editors) (2009) *Behaviour monitoring and interpretation, Smart environments.*

[61] C. Gouin-Vallerand, B. Abdulrazak, S. Giroux, A.K. Dey (2013) *A context-aware service provision system for smart environments based on the user interaction modalities.* Journal of Ambient Intelligence and Smart Environments (JAISE) 5(1), 47–64.

[62] T. Gu, Z. Wu, X. Tao, H.K. Pung, J. Lu (2009) *EpSICAR: An emerging patterns based approach to sequential, interleaved and concurrent activity recognition.* Conference on Pervasive Computing and Communications (PerCom), 1–9.

[63] A. Gupta, P. Srinivasan, J. Shi, L. Davis (2009) *Understanding videos, constructing plots, Learning a visually grounded storyline model from annotated videos.* Conference on Computer Vision and Pattern Recognition (CVPR), 2012–2019.

[64] K. Gwet (2008) *Computing inter-rater reliability and its variance in the presence of high agreement.* British Journal of Mathematical and Statistical Psychology 61-1, 29–48.

[65] T. Hailperin (1984) *Probability logic.* Notre Dame Journal of Formal Logic 25(3), 198–212.

[66] M. Hanheide, A. Peters, N. Bellotto (2012) *Analysis of human-robot spatial behaviour applying a qualitative trajectory calculus.* Symposium on Robot and Human Interactive Communication (RO-MAN), 689–694.

[67] H. Harland (2011) *Nutzung logikbasierter Verhaltensrepräsentationen zur natürlichsprachlichen Beschreibung von Videos.* PhD thesis, Karlsruhe Institute of Technology, Department of Informatics.

[68] M. Henson, J. Dooley, A. Al Malaise Al Ghamdi, L. Whittington (2012) *Towards simple and effective formal methods for intelligent environments.* Conference on Intelligent Environments (IE), 251–258.

[69] C. Herzog (2002) *Das Methodenpaket IeMAX mit dem Fuzzy Simulationsmodell FLUCS – Entwicklung und Anwendung eines Entscheidungsunterstützungssystems für die integrative Raumplanung.* PhD thesis, Christian Albrechts University Kiel.

[70] E. Hornecker (2005) *Videobasierte Interaktionsanalyse – der Blick durch die (Zeit)Lupe auf das Interaktionsgeschehen kooperativer Arbeit.* In: Informationsarbeit neu verstehen, Methoden zur Erfassung informatisierter Arbeit.

[71] T. Ivergard, B. Hunt (2008) *Handbook of control room design and ergonomics: A perspective for the future, second edition.*

[72] Y. Ivanov, A. Bobick (2000) *Recognition of visual activities and interactions by stochastic parsing.* Pattern Analysis and Machine Intelligence 22(8), 852–872.

[73] D. Jain, S. Waldherr, M. Beetz (2009) *Bayesian logic networks, extended version.* Technical Report IAS-2009-03, Intelligent Autonomous Systems Group, Technical University Munich.

[74] D. Jain (2011) *Knowledge engineering with Markov logic networks: A review.* Workshop on Dynamics of Knowledge and Belief (DKB).

[75] B. Jordan, A. Henderson (1995) *Interaction analysis: foundations and practice.* Journal of the Learning Sciences, 4(1), 39–103.

[76] M. Karpinski, H. Kleine Büning, P.H. Schmitt (1988) *On the computational complexity of quantified Horn clauses.* Lecture Notes in Computer Science 329 (Workshop on Computer Science Logic Karlsruhe, FRG), 129–137.

[77] A. Kembhavi, T. Yeh, L. Davis (2010) *Why did the person cross the road (there)? Scene understanding using probabilistic logic models and common sense reasoning.* European Conference on Computer Vision (ECCV), 693–706.

[78] K.M. Kitani, Y. Sato, A. Sugimoto (2008) *Recovering the basic structure of human activities from noisy video-based symbol strings.* Pattern Recognition and Artificial Intelligence 22, 1621–1646.

[79] K. Klösters, F. Sölken (2005) *Führen in Großschadenlagen.*

[80] H. Korte, B. Schäfers (2010) *Einführung in Hauptbegriffe der Soziologie, 8. Auflage.*

[81] D.I. Kosmopoulos, N.D. Doulamis, A.S. Voulodimos (2012) *Bayesian filter based behavior recognition in workflows allowing for user feedback.* Computer Vision and Image Understanding 116(3), 422–434.

[82] F. Krüger, K. Yordanova, C. Burghardt, T. Kirste (2012) *Towards creating assistive software by employing human behavior models.* Journal of Ambient Intelligence and Smart Environments (JAISE) 4(3), 209–226.

[83] R.O. Lane, D.A. Nevell, S.D. Hayward, T.W. Beaney (2010) *Maritime anomaly detection and threat assessment.* Conference on Information Fusion (FUSION), 1-8.

[84] G. Lavee, E. Rivlin, M. Rudzsky (2009) *Understanding video events: A survey of methods for automatic interpretation of semantic occurrences in video.* IEEE Transactions on Systems, Man, and Cybernetics 39(5), 489–504.

[85] R. Laxhammar, G. Falkman, E. Sviestins (2009) *Anomaly detection in sea traffic, A comparison of the Gaussian mixture model and kernel density estimators.* Conference on Information Fusion (FUSION), 756-763.

[86] B. Ley, V. Pipek, C. Reuter, T. Wiedenhoefer (2012) *Supporting improvisation work in inter-organizational crisis management.* Conference on Human Factors in Computing Systems (CHI), 1529–1538.

[87] Z. Lu, J.C. Augusto, J. Liu, H. Wang, A. Aztiria (2012) *A system to reason about uncertain and dynamic environments.* Artificial Intelligence Tools 21(5).

[88] E. Marchioni, L. Godo (2004) *A logic for reasoning about coherent conditional probability: A modal fuzzy logic approach.* In: Logics in Artificial Intelligence, 213–225.

[89] S. McKeever, J. Ye, L. Coyle, C. Bleakley, S. Dobson (2010) *Activity recognition using temporal evidence theory.* Journal of Ambient Intelligence and Smart Environments (JAISE) 2(3), 253–269.

[90] J.M. Mendel (2001) *Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions.*

[91] K. Michels, F. Klawonn, R. Kruse, A. Nürnberger (2013) *Fuzzy Regelung – Grundlagen, Entwurf, Analyse.*

[92] V. Morariu, L. Davis (2011) *Multi-agent event recognition in structured scenarios.* Conference on Computer Vision and Pattern Recognition (CVPR), 3289–3296.

[93] D. Münch, K. Jüngling, M. Arens (2011) *Towards a multi-purpose monocular vision-based high-level situation awareness system.* Workshop on Behaviour Analysis and Video Understanding @ ICVS.

[94] D. Münch, E. Michaelsen, M. Arens (2012) *Supporting fuzzy metric temporal logic based situation recognition by mean shift clustering.* German Conference on Artificial Intelligence (KI), 233–236.

[95] H.H. Nagel (2004) *Steps toward a cognitive vision system.* AI Magazine 25(2), 31–50.

[96] H. Nakashima, H. Aghajan, J.C. Augusto (editors) (2010) *Handbook of ambient intelligence and smart environments.*

[97] F. Pecora, M. Cirillo, F. Dell'Osa, J. Ullberg, A. Saffiotti (2012) *A constraint-based approach for proactive, context-aware human support.* Journal of Ambient Intelligence and Smart Environments (JAISE) 4(5), 347–367.

[98] N. Pfeifer, G.D. Kleiter (2006) *Inference in conditional probability logic.* Kybernetica 42(4), 391–404.

[99] C.S. Pinhanez, A.F. Bobick (1998) *Human action detection using PNF propagation of temporal constraints.* Conference on Computer Vision and Pattern Recognition (CVPR), 898–904.

[100] N. Pirlo (2011) *Zur Robustheit eines modellgestützten Verfolgungsansatzes in Videos von Straßenverkehrszenen.* PhD thesis, Karlsruhe Institute of Technology, Department of Informatics.

[101] A. Pnueli (1981) *The temporal semantics of concurrent programs.* Theoretical Computer Science 13(1), 45–60.

[102] T. Polomski, H.J. Klein (2013) *How to improve situational awareness using piracy attack patterns.* Future Security, 97–106.

[103] T. Polomski, H.J. Klein (2013) *TrIMPI: A data structure for efficient pattern matching on moving objects.* GI-Workshop on Foundations of Databases.

[104] A. Prior (1957) *Time and Modality.*

[105] P. Rangel, J.G. Carvalho Junior, M.R. Ramirez, J.M. de Souza (2010) *Context reasoning through a multiple logic framework.* Conference on Intelligent Environments (IE), 116–121.

[106] M. Raskovic, Z. Ognjanovic, Z. Markovic (2004) *A logic with conditional probabilities.* In: Logics in Artificial Intelligence, 226–238.

[107] M. Richardson, P. Domingos (2006) *Markov logic networks.* Machine Learning 62(1–2), 107–136.

[108] M. Ryoo, J. Aggarwal (2009) *Semantic representation and recognition of continued and recursive human activities.* Computer Vision 82, 1–24.

[109] A. Sadilek, H. Kautz (2012) *Location-based reasoning about complex multi-agent behavior.* Artificial Intelligence Research 43, 87–133.

[110] M.R. Saeed (2012) *A system for disaster response process management.* Master's thesis, Royal Institute of Technology Sweden.

[111] K.H. Schäfer (1996) *Unscharfe zeitlogische Modellierung von Situationen und Handlungen in Bildfolgenauswertung und Robotik.* PhD thesis, University of Karlsruhe, Department of Informatics.

[112] Ph. Schnoebelen (2002) *The complexity of temporal logic model checking.* Advances in Modal Logic 4, 1–44.

[113] J. Shell, S. Coupland (2012) *Towards fuzzy transfer learning for intelligent environments.* Conference on Ambient Intelligence (AmI), 145–160.

[114] Y. Shi, Y. Huang, D. Minnen, A. Bobick, I. Essa (2004) *Propagation networks for recognition of partially ordered sequential action.* Conference on Computer Vision and Pattern Recognition (CVPR), 862–869.

[115] J.M. Siskind (2001) *Grounding the lexical semantics of verbs in visual perception using force dynamics and event logic.* Artificial Intelligence Research 15(1), 31–90.

[116] A. Skarlatidis, G. Paliouras, G.A. Vouros, A. Artikis. (2011) *Probabilistic event calculus based on markov logic networks.* Symposium on Rules, Research Based and Industry Focused (RuleML).

[117] T. Springer, A.Y. Turhan (2009) *Employing description logics in Ambient Intelligence for modeling and reasoning about complex situations.* Journal of Ambient Intelligence and Smart Environments (JAISE) 1(3), 235–259.

[118] C. Spielvogel, R. Reissig-Hochweller, K. Trautmann, P. Kappes, T. Brunner (2013) *Taschenbuch Stabsarbeit.*

[119] K. Tahboub (2006) *Intelligent human-machine interaction based on dynamic Bayesian networks probabilistic intention recognition.* Intelligent and Robotic Systems 45, 31–52.

[120] H.J. ter Horst, A. Sinitsyn (2012) *Structuring reasoning for interpretation of sensor data in home-based health and well-being monitoring applications.* Journal of Ambient Intelligence and Smart Environments (JAISE) 4(5), 461–476.

[121] J.P. Thiran, F. Marques, H. Bourlard (editors) (2010) *Multimodal signal processing, Theory and applications for human-computer interaction.*

[122] Z.O. Toups, A. Kerne, W.A. Hamilton (2011) *The team coordination game: Zero-fidelity simulation abstracted from fire emergency response practice.* ACM Transactions on Computer-Human Interaction 18(4), 23:1–23:37.

[123] S. Tran, L. Davis (2008) *Event modeling and recognition using markov logic networks.* European Conference on Computer Vision (ECCV), 610–623.

[124] P. Turaga, R. Chellappa, V. Subrahmanian, O. Udrea (2008) *Machine recognition of human activities: A survey.* Circuits and Systems for Video Technology 18(11), 1473–1488.

[125] F. van de Camp, R. Stiefelhagen (2013) *GlueTK: A framework for multi-modal, multi-display human-machine-interaction.* Conference on Intelligent User Interfaces (IUI), 329–338.

[126] T.L.M. van Kasteren, G. Englebienne, B.J.A. Kröse (2010) *Activity recognition using semi-markov models on real world smart home datasets.* Journal of Ambient Intelligence and Smart Environments (JAISE) 2(3), 311–325.

[127] T.L.M. van Kasteren, G. Englebienne, B.J.A. Kröse (2011) *Hierarchical activity recognition using automatically clustered actions.* Conference on Ambient Intelligence (AmI), 82–91.

[128] S. Vishwakarma, A. Agrawal (2012) *A survey on activity recognition and behavior understanding in video surveillance.* The Visual Computer, 1–27.

[129] V.T. Vu, F. Bremond, M. Thonnat (2003) *Automatic video interpretation: a novel algorithm for temporal scenario recognition.* International Joint Conference on Artificial Intelligence (IJCAI), 1295-1300.

[130] A. Waibel, R. Stiefelhagen (editors) (2010) *Computers in the human interaction loop.*

[131] F. Wörgötter, E.E. Aksoy, N. Krüger, J. Piater, A. Ude, M. Tamosiunaite (2013) *A simple ontology of manipulation actions based on hand-object relations.* Autonomous Mental Development 5(2), 117–134.

[132] J. Xiang, J. Tian, A. Mori (2011) *Goal-directed human activity computing.* Journal of Ambient Intelligence and Smart Environments (JAISE) 3(2), 127–145.

[133] B.Z. Yao, X. Yang, L. Lin, M.W. Lee, S.C. Zhu (2010) *I2T: Image parsing to text description.* Proceedings of the IEEE 98(8), 1485–1508.

[134] J. Ye, S. Dobson (2010) *Exploring semantics in activity recognition using context lattices.* Journal of Ambient Intelligence and Smart Environments (JAISE) 2(4), 389–407.

[135] J. Ye, S. Dobson, S. McKeever (2011) *Situation identification techniques in pervasive computing: A review.* Pervasive and Mobile Computing 8(1), 36–66.

[136] L.A. Zadeh (1965) *Fuzzy Sets.* Information and Control 8, 338–353

[137] L.A. Zadeh (1975) *The concept of a linguistic variable and its application to approximate reasoning – I.* Information Sciences 8 199–249.

[138] D. Zhang, D. Gatica-Perez, S. Bengio, I. McCowan (2006) *Modeling individual and group actions in meetings with layered HMMs.* Transactions on Multimedia 8(3), 509–520.

[139] C.L. Zitnick, D. Parikh (2013) *Bringing semantics into focus using visual abstraction.* Conference on Computer Vision and Pattern Recognition (CVPR), 3009–3016.

[140] Y. Zhu, N.M. Nayak, A.K. Roy-Chowdhury (2013) *Context-aware modeling and recognition of activities in video.* Conference on Computer Vision and Pattern Recognition (CVPR), 2491–2498.

Curriculum Vitae

	Year	Activity	Institution
Education	2008–2013	PhD in Computer Science	Karlsruhe Inst. of Technology
	2004–2006	Master in Artificial Intelligence	University of Groningen
	2001–2003	Bachelor in Artificial Intelligence	University of Groningen
	1994–2000	Pre-University Sec. Education	Werkman College Groningen
Work	$\geq$ 2014	Postdoc	Wageningen University and R.C.
	2008–2013	Scientific employee	Fraunhofer IOSB Karlsruhe
	2006	Teacher Advanced Logic	University of Groningen
	2006	Teacher Introductory Logic	University of Groningen
	2005	Teacher Neural Networks	University of Groningen
	2005	Assistant teacher Adv. Logic	University of Groningen
	2004	Assistant teacher Adv. Logic	University of Groningen
	2004	Dev./author for web-platform	University of Groningen
	2003	Assistant teacher Intr. Logic	University of Groningen