

UNIVERSITY OF NICE - SOPHIA ANTIPOLIS - FRANCE  
DOCTORAL SCHOOL STIC  
SCIENCES ET TECHNOLOGIES DE L'INFORMATION ET DE LA COMMUNICATION

# T H E S I S

to obtain the title of

**PhD of Science**

Specialty : COMPUTER SCIENCE

by

Guido PUSIOL

## Discovery of human activities in video

Advisor: Francois BREMOND

Affiliation: INRIA, PULSAR Team

May 2012

*Reviewers:*

Dr. Ram      NEVATIA      Director, Institute for Robotics and Intelligence Systems  
University of Southern California, USA

Dr. Xavier      ROCA      Director, Computer Science Department  
Universitat Autònoma de Barcelona, Spain

*Advisor:*

Dr. Francois      BREMOND      Director, STARS research team  
INRIA - Sophia Antipolis, France



To my beloved parents, Nora and Daniel...



## Acknowledgments



# Contents

|          |  |           |
|----------|--|-----------|
| <b>1</b> | <b>Introduction</b>  | <b>1</b>  |
| 1.1      | Introduction . . . . .   | 1         |
| 1.1.1    | The <i>Semantic Gap</i> problem . . . . .                            | 3         |
| 1.1.2    | Bridging the <i>Semantic Gap</i> . . . . .                           | 5         |
| 1.2      | Contributions . . . . .  | 8         |
| 1.3      | Outline . . . . .  | 9         |
| <b>2</b> | <b>State of the Art</b>  | <b>11</b> |
| 2.1      | State of the Art . . . . .   | 11        |
| 2.2      | Single-Layer Approaches . . . . .                                    | 15        |
| 2.2.1    | Space-Time Approaches . . . . .                                      | 16        |
| 2.2.2    | Sequential Approaches . . . . .                                      | 26        |
| 2.3      | Hierarchical <i>Approaches</i> . . . . .                             | 34        |
| 2.3.1    | Statistical Approaches . . . . .                                     | 35        |
| 2.3.2    | Syntactic Approaches . . . . .                                       | 37        |
| 2.3.3    | Description-Based Approaches . . . . .                               | 39        |
| <b>3</b> | <b>Perceptual Information</b>  | <b>43</b> |
| 3.1      | Global Tracks . . . . .  | 44        |
| 3.1.1    | General tracker . . . . .  | 45        |
| 3.1.2    | Depth tracker . . . . .  | 46        |
| 3.1.3    | Simple tracker, <i>our ad-hoc single person tracker</i> . . . . .    | 47        |
| 3.2      | Braking a video into chunks . . . . .                                | 50        |
| 3.3      | Pixel Tracklets (PT) . . . . .                                       | 51        |
| 3.4      | Perceptual Feature Chunks . . . . .                                  | 51        |
| 3.5      | Summary . . . . .  | 52        |
| <b>4</b> | <b>Contextual Information</b>  | <b>53</b> |
| 4.1      | The importance of Contextual Information for activity understanding  | 53        |
| 4.2      | Topology: A single-resolution spatio-temporal contextual model . . . | 55        |
| 4.2.1    | Topology representation . . . . .                                    | 56        |
| 4.2.2    | Topology training input . . . . .                                    | 57        |
| 4.2.3    | Topology Learning, computing the scene regions . . . . .             | 57        |
| 4.2.4    | Computing the scene region parameters . . . . .                      | 59        |
| 4.2.5    | Clustering with K-Means . . . . .                                    | 59        |
| 4.2.6    | Distance issues . . . . .  | 60        |
| 4.3      | Topology examples . . . . .  | 61        |
| 4.4      | Topology Updating . . . . .  | 62        |
| 4.4.1    | The alignment of topologies . . . . .                                | 62        |
| 4.4.2    | Updating algorithm . . . . .   | 70        |

|          |  |           |
|----------|--|-----------|
| 4.4.3    | Statistical Analysis . . . . .                                       | 71        |
| 4.5      | The Scene Model, a Multi-Resolution representation . . . . .         | 72        |
| 4.5.1    | Selection of the most relevant abstraction level . . . . .           | 73        |
| <b>5</b> | <b>Primitive Events</b>  | <b>75</b> |
| 5.1      | Problem Definition . . . . .   | 75        |
| 5.2      | Primitive Events . . . . .   | 76        |
| 5.3      | Building a Primitive Event . . . . .                                 | 76        |
| 5.4      | The Duration and $ID_{Chunk}$ attributes . . . . .                   | 77        |
| 5.5      | The Type attribute . . . . .   | 77        |
| 5.5.1    | Type representation . . . . .  | 78        |
| 5.5.2    | Type computation . . . . .   | 78        |
| 5.5.3    | Type Example . . . . .   | 78        |
| 5.5.4    | Type Ambiguity . . . . .   | 79        |
| 5.5.5    | Type Quantity . . . . .  | 79        |
| 5.5.6    | Discussion . . . . .   | 79        |
| 5.6      | The Local Dynamics Attribute . . . . .                               | 80        |
| 5.6.1    | Local Dynamics representation . . . . .                              | 81        |
| 5.6.2    | Local Dynamics Abstraction . . . . .                                 | 81        |
| 5.7      | The additional Logic attribute . . . . .                             | 87        |
| 5.8      | Primitive events sequence . . . . .                                  | 87        |
| <b>6</b> | <b>Activities: Discovery, Modeling and Recognition</b>               | <b>91</b> |
| 6.1      | Input: Primitive Event sequences . . . . .                           | 92        |
| 6.2      | Approach: Activity Discovery . . . . .                               | 92        |
| 6.2.1    | Seeking for activity patterns . . . . .                              | 93        |
| 6.2.2    | Behind the <i>Stay</i> and <i>Change</i> activity patterns . . . . . | 95        |
| 6.2.3    | Two main achievements . . . . .                                      | 98        |
| 6.3      | Activity discovery examples and HCI . . . . .                        | 99        |
| 6.3.1    | Homecare Gerhome . . . . .   | 99        |
| 6.3.2    | Hospital . . . . .   | 101       |
| 6.3.3    | Sleeping Monitor . . . . .   | 101       |
| 6.3.4    | The Office . . . . .   | 104       |
| 6.4      | Activity Models . . . . .  | 105       |
| 6.4.1    | Activity neighborhood . . . . .                                      | 106       |
| 6.4.2    | Human Interaction . . . . .  | 107       |
| 6.4.3    | The Modeling Procedures . . . . .                                    | 108       |
| 6.4.4    | Hierarchical Activity Models (HAM) . . . . .                         | 112       |
| 6.4.5    | Multiresolution-Histograms models (MH) . . . . .                     | 119       |
| 6.5      | Activity Similarity . . . . .  | 121       |
| 6.5.1    | The similarity of HAM - Hierarchical Activity Models . . . . .       | 121       |
| 6.5.2    | The similarity of MH - Multiresolution Histograms model . . . . .    | 122       |
| 6.6      | Activity Recognition . . . . .                                       | 123       |



|          |   |            |
|----------|---|------------|
| <b>7</b> | <b>Evaluation</b>   | <b>125</b> |
| 7.1      | Evaluation Scenes . . . . .   | 126        |
| 7.1.1    | Monocular Video Camera Scenes - <i>MCS</i> - . . . . .  | 126        |
| 7.1.2    | Depth Camera Scene - <i>DCS</i> - . . . . .   | 128        |
| 7.1.3    | The - <i>DCS</i> - datasets . . . . .   | 129        |
| 7.1.4    | The dataset summary and highlights . . . . .  | 130        |
| 7.2      | The target activities . . . . .   | 131        |
| 7.3      | Recognition . . . . .   | 138        |
| 7.3.1    | Learning activity models, evaluation procedure . . . . .  | 139        |
| 7.3.2    | Performance measurements: . . . . .   | 140        |
| 7.3.3    | Configuration . . . . .   | 141        |
| 7.3.4    | Recognition results, monocular camera dataset . . . . .   | 142        |
| 7.3.5    | Recognition results, depth camera datasets . . . . .  | 147        |
| 7.3.6    | Comparison: Multi-Resolution Histograms vs. Hierarchical<br>Activity Models . . . . .   | 148        |
| 7.3.7    | Comparison: Multi-Resolution vs. Single-resolution approach<br><i>HM-basic</i> . . . . .  | 148        |
| 7.3.8    | Comparison: Our proposed - <i>HAM</i> and <i>HM</i> - vs. state of<br>the art - <i>SA1</i> - [Zouba 2010], in HOME-CARE . . . . . | 150        |
| 7.3.9    | Ranking . . . . .   | 152        |
| <b>8</b> | <b>Conclusions</b>  | <b>159</b> |
| 8.1      | Conclusions . . . . .   | 159        |
| 8.1.1    | Closing the loop . . . . .  | 160        |
| 8.1.2    | Summary . . . . .   | 160        |
| 8.1.3    | Take it home message . . . . .  | 161        |
| 8.2      | Future Work . . . . .   | 162        |
| 8.2.1    | In the short term . . . . .   | 162        |
| 8.2.2    | In the long term . . . . .  | 163        |
| 8.3      | A final thought . . . . .   | 163        |
| <b>A</b> | <b>Appendix</b>   | <b>165</b> |
| A.1      | K-Means Clustering, Distances . . . . .   | 165        |
| A.1.1    | Euclidean distance . . . . .  | 165        |
| A.1.2    | City-block distance . . . . .   | 165        |
| A.1.3    | The Pearson correlation coefficient . . . . .   | 166        |
| A.1.4    | Uncentered correlation (cosine of the angle) . . . . .  | 167        |
| A.1.5    | Spearman rank correlation . . . . .   | 168        |
| A.1.6    | Kendall's $\tau$ . . . . .  | 168        |
| A.2      | Natural Language Processing for Activity Modeling . . . . .   | 170        |
| A.2.1    | A discovered activity and the underlying relationship with lan-<br>guage models . . . . .   | 171        |
| A.2.2    | Frequency models . . . . .  | 172        |
| A.2.3    | n-gram Models . . . . .   | 175        |

|                     |                          |            |
|---------------------|--------------------------|------------|
| A.2.4               | uni-gram model . . . . . | 175        |
| A.2.5               | bi-gram model . . . . .  | 175        |
| <b>Bibliography</b> |                          | <b>177</b> |

# Introduction

---

## Contents

---

|  |          |
|--|----------|
| <b>1.1 Introduction</b> . . . . .                | <b>1</b> |
| 1.1.1 The <i>Semantic Gap</i> problem . . . . .  | 3        |
| 1.1.2 Bridging the <i>Semantic Gap</i> . . . . . | 5        |
| <b>1.2 Contributions</b> . . . . .               | <b>8</b> |
| <b>1.3 Outline</b> . . . . .                     | <b>9</b> |

---

## 1.1 Introduction

From the beginning, human beings have been trying to understand their own nature. To achieve such a goal, humanity has a long history of use in psychology and philosophy as holders of the fundamental concepts for human psyche. From a scientific point of view, the study of human beings occurs partially via their behavior. Understanding individuals allows to establish general principles which ends up by their benefit, and consequently the society. In most of the cases, the principles are deduced from the analysis of empirical and measurable evidences (i.e. psychoanalysis). Nevertheless, the gathering and analysis of information is a non fully objective manual task.

Nowadays, technical and scientific advances make computers to evolve rapidly, becoming affordable and powerful. There are gadgets everywhere, collecting, transmitting, and warehousing human generated content (or signals) which is inefficiently handled by other human operators. Humans have a limited capacity to understand the underlying relationships in large amounts of data. The need of automatic systems capable of analyzing big amounts of data is imperative. Consequently, cognitive systems (*CS*) can be put in a central position to collaborate with other domains to understand and simplify qualitatively perceptual information.

In the last decades, an interdisciplinary domain has raised within *CS* which aims at solving the problems mentioned before. The domain is composed of techniques such as Data Mining, Image Processing and Analysis, Graph Theory, Pattern Recognition, Artificial Intelligence, Computer Graphics and Machine Learning. It is from this interdisciplinary domain, where we are contributing in this work to the

understanding of human activities.

This work presents a complete framework for video analysis. The framework is capable of automatically discover, model and recognize activities in videos. We address the full chain of required stages, from low-level vision to semantical activity interpretation. Scientifically, this manuscript proposes new solutions for cognitive systems open challenges which are discussed later. From a social point of view, we contribute by addressing a real world problem which is the care of elder people in their own homes. This way, we aim at closing the loop of research applied to real social problems.

### *A social issue*

More than 2 billion people will turn over 65 years old by the year 2050. It is of crucial importance for the research community to help aging adults to living independently as long as possible. Nursing homes and hospitals are expensive, especially due to the staff they require, and could quickly become saturated by those people with chronic diseases. Moreover, the loss of independence that follows when people must leave their own homes for nursing homes is a major factor of stress and participates to the deterioration of patients' health. Keeping the elders in their homes with the help of well-placed technology benefits on both fronts [Ross 2004]. Monitoring elderly people's activities in their homes through video surveillance gives good indicators of their health. The understanding of daily activities is key and is a topic that remains open. In the literature the computational approaches assume usually prior knowledge of the activities and the environment [Brdiczka 2009]. This knowledge is used explicitly to model the activities in a supervised manner. Given the wide range of human activities, for a system to be usable in real-life conditions, it needs to be generic and to require minimal supervision.

### *Scientific challenges*

Reaching the high level of semantic interpretation of a scene from the low level of image processing means jumping over a semantic gap. **Bridging the semantic gap remains an open challenge.** The semantic gap problem is detailed in the next section 1.1.1. Finding a solution to this gap is one of our main goals. Other challenges are related to typical vision problems (e.g. illumination), data abstraction and reduction (e.g. clustering), data modeling (e.g. parametrization), human computer interface (i.e. display), etc. Nevertheless, we consider each of these last challenges local to particular tasks in this work and are discussed in their corresponding chapter.

### 1.1.1 The *Semantic Gap* problem

The semantic gap is the lack of correlation between the semantic categories that a user requires and the perceptual features that the systems can offer. The problem can be found in most signal processing domains (i.e. audio, video, image, radio frequency). In any domain, it can be seen as the difference between what we can measure (*quantitative*) from a signal and what the signal actually means (*qualitative*).

In the visual domain the problem is discussed by Smeulders et al. [Smeulders 2000], where it is defined as "The lack of coincidence between the information that one can extract from the visual data and the interpretation that the same data have for a user in a given situation...".

The idea of a semantic gap is not new. The following are historical references to the problem.

- "*Un bon croquis vaut mieux qu'un long discours*" - A good sketch is better than a long speech.

*Napoleon Bonaparte (1769 - 1861).*

- "*A picture shows me at a glance what it takes dozens of pages of a book to expound*".

*Ivan Turgenev (1862).*

- "*A picture is worth 100 words*".

*Fred R. Barnard (1981).*

More recently, an all embracing statement of the gap is provided by *Richard Hamming (1925 - 1998)*: "*The purpose of computing is insight, not numbers*".

For images, the gap problem can be illustrated as we show in the figure 1.1, where the semantical interpretation of the displayed images is a challenging task. An immediate observation would be that the provided data is ambiguous and insufficient to perform a semantical interpretation, and that additional information would be required. In many cases, the additional information is the context of the scene. For example, in Fig. 1.1 (d) observing that the painting roller is yellow is a hint that the person is painting the light rays and not the night darkness.

For video, the gap problem inherits the complexity of image processing and adds *time* as a new data dimension<sup>1</sup>. One of the advantages of including temporal information is the enabling of the interpretation of movement. Such a thing, is hard to do with a single image. For example, in Fig. 1.1 (e) it is difficult to infer that

---

<sup>1</sup>A video is a sequence of images (or frames) with an order relationship defined by the acquisition in time.

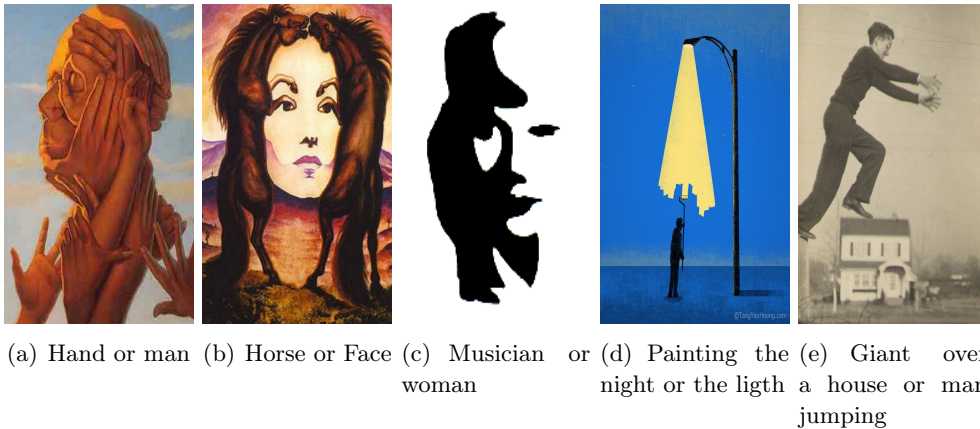
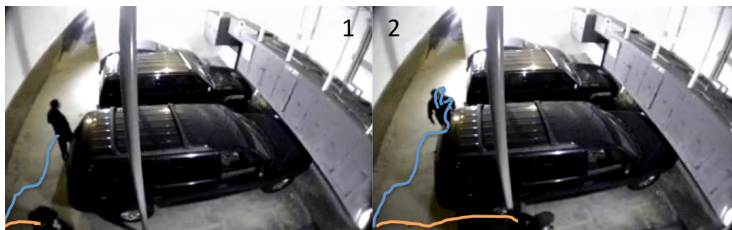


Figure 1.1: Examples of different semantic interpretation of the same visual data.



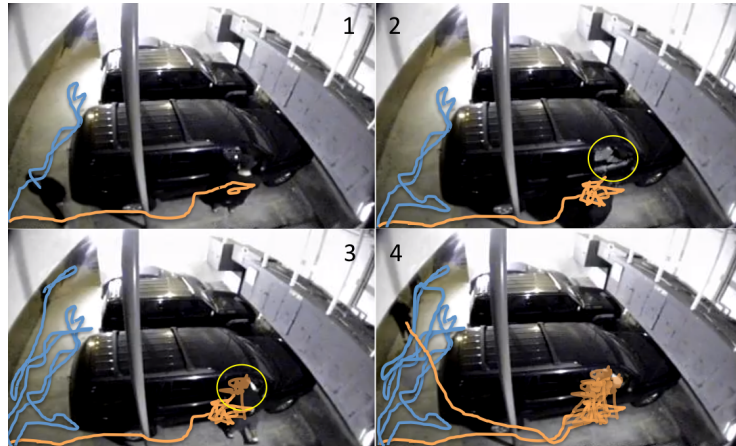
(a) Example of ambiguous behavior in a parking lot. The trajectories are not enough to describe semantically the intentions of the people (i.e. steal a car)

Figure 1.2: Ambiguous people behavior in a parking lot.

the person jumps without first understanding the person motion and his relative position to the house.

The gap problem for video can also be shown using conceptual video images. For example, the figure 1.2 show snapshots of trajectories described by two people in a parking lot. The possible trajectory interpretations are two activities: 1) Normal: "The people are searching for their car". 2) Unfrequent: "The persons aim at stealing a car". Nevertheless, further information is required to make a decision between the two possibilities.

Using more (quantity) and better (less noise) information it is possible to understand semantically the scenario. For example, the continuation of the parking lot scenario is illustrated in the Figure 1.3. In (1 to 4), the analysis of the trajectories can describe **unfrequent paths**; detecting unfrequent events such as in (2) where the **car window is broken**; and, contextual knowledge such as in (3), where its **not normal to get in your own car through the window**. All of the previous hints helps to bridge the gap leading to deterministic semantic interpretations.



(a) Are the persons searching for their car, or searching for one to steal.

Figure 1.3: Person stealing a car in a parking lot.

### 1.1.2 Bridging the *Semantic Gap*

In video processing, the gap is located between the low-level visual features and high-level semantic concepts (labels) understandable by the user.

The low-level features are quantitative (numbers) information obtained from vision-based techniques, capable of collecting but not interpreting the numerical information. The interpretation of the low-level features is qualitative information. It is in the qualitative information space where the user can find semantical interpretation of the low-level data. The figure 1.4 illustrates the problem of bridging the semantic gap in the activity recognition domain.

The research community is very aware about the existence of the semantical gap. In most of the cases, the literature approaches address the gap issue by **avoiding it**. This is, the methods jump over the gap doing a specific transformation of quantitative to qualitative information (and the inverse) in a single step.

In terms of taxonomy of the approaches, they can be classified by the strategies of information processing and knowledge ordering. Two main classes are well defined: *Bottom-Up* and *Top-Down*. In any case, the gap bridge aims at linking low-level information and data models of the interesting occurrences (meaningful abstractions). The interface between the final user and the data is performed through the activity models as it is illustrated in the figure 1.5. The two types of approach are complementary paradigms and can be confronted:

1. *Top-Down* (also known as step-wise design): These approaches start by tools for the creation and management of ontologies, and they link low level features to built ontologies. As a result of the efforts in this area there is a large set

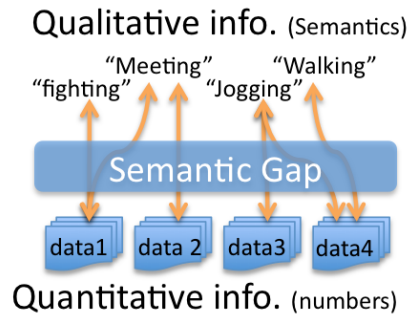


Figure 1.4: The semantical gap problem in activity recognition.

of custom based ontologies, defining important concepts that can guide our research in video analysis. The ontologies are representing what experts expect to extract from the video content. The top-down approaches emphasize planning and a complete understanding of the system sub-parts. This implies that they require a big amount of prior contextual knowledge. Contextual information can be directly injected to the system. Generally, they have a compositional structure of different abstraction layers. At each layer small sub-systems are being solved, contributing to a general solution. A top-down approach is generally assisted by the usage of "black boxes", which makes easier to manipulate the system sub-parts. In most of the cases these approaches are fully **supervised** which can be considered as a drawback in complex real environments. For long term activities, it is hard to know and describe all possible variations of an activity performed by an individual. Further, the manual modeling takes time and it is hard to describe activities of low semantical abstractions. More over, the perceptual low-level information usually arrives with noise which is hard to understand and model manually.

2. *Bottom-Up*: These approaches are usually coming from the Computer Vision, Signal Processing and Audio analysis communities. This is where the research community tends to focus in very specific applications and challenges (e.g. surveillance, traffic, crowd analysis). These approaches generally extract and handle particular types of information that are relevant to a specific content. The applications have a well defined goal and a good understanding of the context of the extracted data allowing to build ad-hoc approaches for a particular domain. Bottom-up approaches are characterized by the lack of human intervention in the data processing. This type of procedure is known as **unsupervised**. Mainly these approaches use prior knowledge of the type of data to be processed. The unsupervised approaches are always linked to a data driven learning procedure. The procedure will produce models of interesting video events and those models can be later found in new low level sequences. The drawback of fully unsupervised bottom-up approaches is that



they can only establish a similarity between a learnt model and low level data, due to the lack of a proper semantical layer. Also, they cannot retrieve semantical interpretation of the low-level features. Nevertheless, bottom-up approaches can have different degrees of non supervision which are related to the complexity of the prior knowledge provided by humans and to the understanding of the data to be processed.

The figure 1.5 shows an illustration of bridging the gap using the methods described above.

Summarizing, *Bottom-Up* approaches bridge the semantic gap by learning the activity models which can be directly retrieved to the users. Nevertheless semantical interpretation is missing. In the other hand, *Top-Down* approaches can interpret semantically the video, and this way bridging the gap. But manually described activity models are usually a weak characterization of an activity. We propose to use an intermediate solution, a third type of approaches:

3. Our approach *is hybrid*: We propose to take the best of both types of approach described above. We process automatically low-level information up to a point where it can provide semantically explainable descriptions of the human motion (e.g. person moves from A to B). Such a thing is achieved by combining data driven learning procedures to understand small video chunks and learnt contextual information (interesting scene regions and objects). The combination is a *Bottom-up* technique which at certain point can be seen as a traditional *Top-Down* approach where the manual annotations are replaced by learning techniques. We combine the automatically learnt information to build an intermediate layer of semantical information characterized as primitive events. The intermediate layer has the property of being located just in the middle of the semantical gap. We use pattern matching techniques to discover and model automatically activities. The interesting models are presented to the user which may add semantical labels (as in *Top-Down*). Finally, the video interpretation is achieved by building new models automatically and by comparing them with labeled ones.

In other words, our method does not avoid the semantical gap, but it proposes an intermediate layer of primitive events. This layer can be understood by humans and labeled semantically, relaxing significantly the amount of supervision without losing the benefits of learning. When high-level semantics are required for a final application, this approach can be classified as semi or weakly supervised.

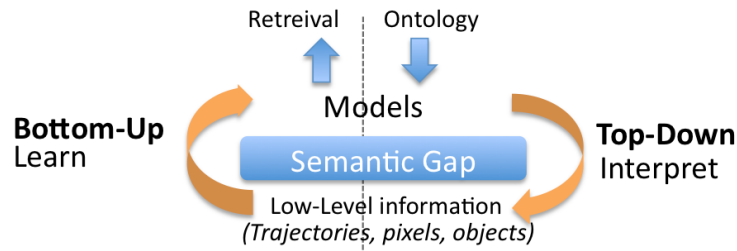


Figure 1.5: Bottom-up and Top-down, the main techniques to bridge the semantical gap.

## 1.2 Contributions

The contributions of this work can be summarized by the following items:

- **Dynamic length video chunks:** On our understanding it is the first approach which proposed braking down a long video into sizable chunks of video representing only meaningful information on demand. This reduces the storage space without losing activity information. Moreover this dynamic decomposition is done in real time.
- **Combination of local and global motion:** Most of the literature approaches detect abstract high-level activities using global object trajectories or short-term actions using pixel based trajectories. We propose to combine both types of trajectory to understand high-level activities reinforced with finer descriptors of local motion.
- **Learning scene models:** We propose an algorithm to automatically discover and characterize the scene regions of interest, where an agent interact with objects (a Topology). The algorithm builds multi-resolution topologies of the scene which corresponds to the model of the scene. Also, we adapt from the graph theory domain an alignment algorithm used to update and compare scene models of different agents.
- **Primitive Events:** This is probably the most important contribution of this work. We build an intermediate layer of primitive events between low-level information and high-level semantics. The layer is automatically learnt. It has the property of being easily understand by humans. Consequently, allowing modifications and providing explanations of the perceptual low-level information (e.g. tracking errors are easily characterized).
- **Activity discovery:** We propose to extract particular event patterns to discover activities at multiple semantical resolutions. Therefore, the activities and sub-activities are automatically discovered marking the start and end of

an activity in a single computation. This method overcomes sliding window approaches which requires trying several sizes to find the right temporal length.

- **Activity modeling:** We propose two hierarchical methods to model an activity. The models are learnt with the particularity of using activity descriptors (building blocks) of different resolutions. Also, the models characterize the relationships among the multiple resolutions. The multiple resolutions represent the activity sub-activities. Most of the literature works learning multi-resolution models build their different abstraction layers from data incoming from a single resolution (i.e. re-clustering the elements of a cluster).

### 1.3 Outline

This thesis goes from the low-level to the high-level video interpretation corresponding to the order of the chapters. A brief description of the chapter organization is itemized as follows.

- *Chapter 2*  
**Related Work:** We explore the literature approaches which address the problem of understanding short, mid and long-term activities. We follow a taxonomy to categorize the approaches in single-layer (mostly unsupervised) and hierarchical (supervised) approaches. Also we describe similar attempts to ours explaining the critical points where they succeed and fail for discovering long-term activities.
- *Chapter 3*  
**Perceptual Information:** This chapter explains how we propose to extract from a video the building blocks (trajectories) for the rest of this work. First, we describe 3 methods to detect and track the global position of an object. Second, we explain how to use the global position to brake a long video into meaningful clips. Third, we explain how we reinforce the motion description by computing sparse pixel-based tracklets. Finally, a compact representation is built to reduce the amount of data necessary to characterize a small amount of activity (i.e. an action).
- *Chapter 4*  
**Contextual information:** We explain how we learn a single resolution contextual model called the scene Topology. We proceed by extending the contextual information to multiple resolutions, building a scene model which is characterized by several topologies. We propose a compact representation of the scene model as well as updating procedures. Finally, we explain the type of analytical information that can statistically be inferred from a learnt scene model to understand basic general scene properties (i.e. the kitchen is the most used space in a house).

- *Chapter 5*

**Primitive Events:** We fuse the learnt information described in the chapters 4 and 5, to build a basic atomic entity named primitive event. The primitive events are abstractions of low-level information capable of describing semantics without ever being labeled by users. For example, typical primitive events are the transitions of an object between the different regions of the scene, or an object remaining static for a long period of time. Also, we attach to the primitive events processed pixel-based motions which can describe the main local dynamics of an object (i.e. the waving gesture of a person). Finally, we explain how a sequence of primitive events is built from an incoming video.

- *Chapter 6*

**Activities: Discovery, Modeling and Recognition:** In this chapter we explain how to extract activities from sequences of primitive events by pattern matching. The activities can be extracted from different resolution levels. The extraction is equivalent to the discovery of the start and end marks of an activity. The activities are automatically clustered by coherent properties and displayed to the user. The user can add semantics which leads to a new activity model. We propose 2 new multi-resolution activity modeling techniques. Finally, a modeled activity is recognized in a new video computing its similarity with new activity models.

- *Chapter 7*

**Evaluation:** In this chapter we evaluate the proposed approach in 3 different scenes. The tests take place by varying several possible configurations of the whole system to demonstrate its flexibility (i.e. the stages of the framework can be replaced). The evaluation scenes are mostly related to home-care applications. We compare our method to other state of the art methods. Also, we compare multi-resolution and single-resolution modeling techniques. Finally, the recognition procedure is evaluated using an activity model learnt with a single training prototype. This last evaluation demonstrates the accuracy of the system using the minimal requirements.

- *Chapter 8*

**Conclusions:** In this chapter we summarize the strengths of this work and the points which still require improvement. We propose other possible applications and extensions. Finally, we discuss the future work that can build over the proposed approach.

# State of the Art

---

## Contents

|            |                                |           |
|------------|--------------------------------|-----------|
| <b>2.1</b> | <b>State of the Art</b>        | <b>11</b> |
| <b>2.2</b> | <b>Single-Layer Approaches</b> | <b>15</b> |
| 2.2.1      | Space-Time Approaches          | 16        |
| 2.2.2      | Sequential Approaches          | 26        |
| <b>2.3</b> | <b>Hierarchical Approaches</b> | <b>34</b> |
| 2.3.1      | Statistical Approaches         | 35        |
| 2.3.2      | Syntactic Approaches           | 37        |
| 2.3.3      | Description-Based Approaches   | 39        |

---

## 2.1 State of the Art

The goal of activity recognition is to automatically understand ongoing activities from an unknown video (i.e. a sequence of image frames). In a simple case where a video is broken into clips to contain only one instance of an activity, the objective of the system is to correctly classify the clips into its activity category (a discriminative approach). In our case, it is more complex, the continuous recognition of human activities must be performed by detecting the starting and ending times of all activities from an input video. Furthermore, the total number of possible activities in unknown and several activities occur in the same time.

A majority of work in activity recognition focuses on relatively short activities (or actions) that have a duration of seconds, rather than minutes, hours or even days. The recognition of small time scale activity is a field that is still poorly understood, **long-term** and **high-level** activities is a field even less explored, making it challenging. For example, there is not yet a formal definition of high-level activities. In larger time scales, the properties of the activities are different than in smaller scales. For example, they present larger variances in the execution than shorter activities. Such a situation makes hard the usage of existing recognition algorithms. The length of the activities carry set of technical problems which makes hard the evaluation of ad-hoc methods. the long term videos is require of annotation techniques that could be carried by humans and still maintain ground truth detail for experimentation. Furthermore, these activities cannot be developed

in the laboratory (as with short actions), they require to be performed in real scenes and the movements need to be as natural as possible, requiring of logistics, proper environments, robust hardware. Finally, the recognition algorithms need to consider the cost of processing and configuration as one of the variables of handling efficiently large amounts of data.

The activity understanding methods are, in most of the cases, designed to work for particular scenarios. In the rest of this work the words *scene* and *scenario* are used in reference of the environment where an activity takes place and the activities occurring in the environment. The scenes can be describing *outdoor* and *indoor* environments. The outdoor scenes generally represent unconstrained areas where agents can move freely interacting with objects. The indoor scenes contain a richer set of objects that the agents interact with. Conceptually, the scene objects play an important roll in activity understanding. In most of the cases a scene can be classified as a *structured* or a *unstructured* scene. We consider that a scene is *structured* when the objects of the scene define a set of possible scenarios, and this scenarios usually follow a constrained spatio-temporal order of events. For example, the frequent activities described by the car paths in a traffic junction is structured and defined by the allowed road way and traffic light changes. In the other hand the activities described by a person in an apartment are not defined entirely by the objects in the scene. For example, several activities can occur at the kitchen, and when an activity occurs ("preparing meal") there is no specific ordering of events (the stove is not going to be necessarily used). This difference of *structured* and *unstructured* is important due to the fact that most of the literature approaches are designed to work for a specific type of scene, being the second category of activities the most challenging to recognize.

The type of activities that can be understood have wide variations. In many cases the variations depend on the agents (e.g. "humans", "cars"). Humans are agents capable of producing complex unstructured motions to perform activities, most of the methods addressing the problem of human activity understanding at home can be ported to other structured domains (i.e. traffic surveillance). There are various types of human activities. We categorize them depending on their complexity and length. The complexity of an activity is defined by the variations of the characteristic events defining an activity and the amount of events that can be perceived. For example, a person "preparing meal" can be perceived by the interactions defined by the person and the kitchen equipment which are more complex to describe and recognize than the ones of a person "sitting".

We define 4 categories of human activities: gestures, actions, interactions, and behaviors. Gestures are movements of a person body part, and can be seen as atomic components describing meaningful motion. For example, "raising a leg" is a gesture. Actions are single-person motion that may be composed of multiple gestures organized temporally, such as "waving," and "punching." Interactions are

human activities that involve two or more persons and/or objects. For example, "two persons fighting" is an interaction between two humans and "a person is preparing meal" is a human-object interaction involving a human and objects. In our case we are interested in long term activities which are in most of the cases human-object interactions of long duration. In this work we refer to these long term interactions as "activities". Finally, a behavior is an high level description of the person state such as "angry", "active". In most of the cases, the behavior is understood over a set of interactive activities of long duration (e.g. days, weeks). Other activity categories could include the understanding of groups of people, nevertheless this subject is beyond the scope of this work.

We are interested in different type of long term-activities, but mainly we make focus in a particular class. The activities can be find in healthcare and assisted living applications. The activities are the called Activities of Daily Living (ADLs). A set of ADLs are originally proposed by Katz et al. [Katz 1963], and have evolved over time into a set of activities used by physicians to measure the well-being of elderly patients. The ADLs contains activities such as: bathing, dressing, transferring, continence and feeding. The ADLs are extended to a set of Instrumental Activities of Daily Living (IADLs) proposed by Lawton and Brody [MP 1969], which is composed of activities such as: using phone, preparing meal, housekeeping, doing laundry, transportation, taking medicaments, etc. Finally, ADLs and IADLs is a challenging set of activities that we aim at recognize automatically.

Subsets of ADLs and IADLs have been recognized wearable sensors [Stikic 2008, Zouba 2009]. The drawback is that for some people the wearable sensors can be invasive, also in many cases the monitored users forget to put on the sensors (e.g. wrist) making the data acquisition impossible. Other approaches overpass the previous limitations using non-wearable sensors such as cameras. The usage of general vision sensors carry a large set of challenges which we aim at address with the work proposed in this work.

The goal of this chapter is to provide an overview of the state-of-the-art of activity recognition methodologies strongly oriented to human agents. We aim at explaining the benefits and drawbacks of the proposed literature techniques, also the reasons and tendencies adopted by recent work.

Activity understanding in videos is an active field which changes fast. There are different works that summarize and describe the contributions of the last years. For human activity understanding, recent work include the surveys of [Turaga 2008] and [Aggarwal 2011]. The last one introduces an approach-based taxonomy structured as a tree to categorize the literature approaches. In this chapter we adopt such a taxonomy to extend the surveys to general activity recognition methods and more up to date techniques.

We discuss the different types of approach designed for activity understanding. The approaches not only can be categorized but also inter- and intra- class comparisons can be made. The comparisons aim at showing the difference of our proposed approach and including non measurable conceptual differences such as:

- The method can recognize activity in structured and/or in unstructured scenes.
- The method works or not for activities with loosely time constraints.
- The method are designed to handle long duration activities (hours), or are oriented to recognize short actions.
- The approach is supervised, unsupervised or semi-supervised.
- About the capabilities of the methods to provide semantical descriptions of the activities.
- About the type of applications where the methods are tested.
- About the difficulty of training and updating, automatically or manually the activity models. When its supervised, the difficulty of describing the models by hand. When its unsupervised, the amount of training data required and the time needed to train a model.
- About the activity models, the different generative and discriminative approaches.

A first classification of the literature approaches consists in differentiating **Single-layered** and **Hierarchical** approaches. Single-layered approaches model and recognize activities directly from the video. Usually, single-layered approaches can recognize short actions or activities that contain events ordered sequentially. In most of the cases, single-layered approaches use a dense set of descriptors to fill complex models. In the other hand, hierarchical methods are similar to the "divide and conquer" algorithm, where an activity is recognized based in the recognition first of simpler sub-activities. A particularity of hierarchical approaches is that the composition of sub-activities allows the semantical description of complex long-term activities.

**Single-layered** approaches are decomposed into two sub-classes depending on the activity modeling structures. The sub-classes are: space-time and sequential approaches. The space-time approaches view an input video volume where they compute descriptive features (e.g. trajectories). In the other hand the sequential approaches characterize a video as a sequence of perceptive events.

**Space-time** approaches are classified in three groups depending on the features they use for modeling an activity. The classes are: video volume (i.e. a sequence of



images -pixel level-), object trajectories, and local features (e.g. pixel-based points of interest). The sequential approaches are classified in two sub-classes depending on the use of exemplar-based or model-based recognition techniques.

**Hierarchical** approaches are classified in three categories depending on the recognition procedures: statistical, syntactic, and description-based approaches.

For statistical approaches, the activity models are state-based and concatenated hierarchically (e.g., layered HMMs). The Syntactic approaches use a grammar syntax such as a stochastic context-free grammar (SCFG) to model activities. Finally, Description-based approaches model human activities by describing sub-events and also their temporal, spatial, and logical attributes.

## 2.2 Single-Layer Approaches

These approaches can recognize human activities directly from video data, where an activity is a type of image sequences (or chunks). A video chunk is classified into a set of predefined classes, where each class represents an activity. The classification is performed by matching algorithms and the process of classification is what is called the activity recognition procedure.

For real-time recognition on the fly, most of the approaches adopt a sliding window technique to build a template video chunk. In most of the cases, these approaches are more effective when the activity models are learned (unsupervised). Due to the sequential nature of these approaches, they can recognize simple and short human actions (e.g. "walking", "jumping", "waving") and activities and long constrained object activities (i.e. "car turn left"). In unstructured scenes, the sequential events of a long activity trend to have variations hard to characterize by these methods. Nevertheless, for long constrained activities, these methods can model the sequence of events.

The single-layered approaches are grouped into two classes: **space-time** and **sequential** approaches.

Space-Time approaches model a human activity as a particular video chunk (3D volume) in a spatio-temporal dimension or as a set of features extracted from the chunk. The chunks are built by concatenating image frames, and the chunks are compared to recognize activity. The group of sequential approaches treat an activity as a sequence of perceptive events. In most of the cases, an activity is represented as a sequence of feature vectors, where the feature vectors are computed from the images composing the chunks. The activity recognition is achieved seeking event sub-sequences matching with a modeled feature vector.

### 2.2.1 Space-Time Approaches

The video piece of an activity instance can be represented as a 3-D XYT spatio-temporal chunk. Space-time approaches can recognize activities by analyzing the space-time video chunks.

Typically the activity recognition in space-time approaches is as follows. First, based on training video chunks, the system learns the activity models. Second, for a new unseen video, the system breaks the video into chunks sequentially. Each new chunk builds a test activity model which is compared with the learnt model (template) to measure the similarity. There are several variations of the volumetric space-time representation. The abstraction of the image chunks into more meaningful features (e.g. trajectories) allows the understanding of more complex activities and the filtering of low-level vision problems (illumination changes). Among the features, many systems represent an activity as a set of object-based trajectories if it is capable of tracking a target object (e.g. humans, cars). Other features may include feature points such as pixel-based motion tracklets, human joint positions, interesting points, visual words, etc. The whole idea of video abstraction into other features is to build compact and discriminative rich descriptors of the raw video information.

The recognition procedure is similar for most of the space-time approaches, where the recognition is achieved by the template-matching of activity models learnt from any type of low-level features. The template-matching algorithms usually correspond to generative approaches, where not the whole set of possible activities is known. Other type recognition approaches are neighbor-based approaches. These approaches usually are followed by discriminative methods, where the whole possible set of activities is modeled, allowing the inter-classes similarity estimations. For neighbor-based approaches the system maintains a set of sample features (e.g. trajectories), and the recognition is achieved by matching an unseen input with all or part of the features.

#### 2.2.1.1 *Space-Time Volumes (STVs)*

These approaches assume to have two volumes of video (learn and test chunks). The main challenge is to define methods to measure the similarity between these volumes. The approaches are mostly applied to human action recognition. They need to measure the similarity between the movements described in the two video volumes. The similarity is measured depending on the type of representation used to describe the video volumes.

A first type of representation is called global, where the region of interest (ROI) of an object is considered as a whole. The ROI is generally obtained by background subtraction and tracking. For human actions, the segmented person's silhouette can be used to represent video. When the silhouette is obtained, there are many ways to

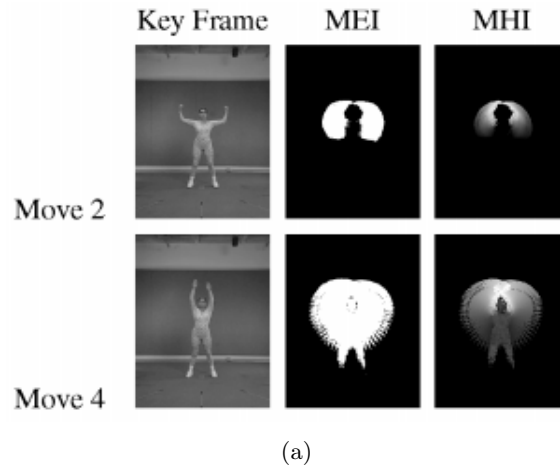


Figure 2.1: Example of the Bobick and Davis [Bobick 2001] 2D motion images

encode the motion information. One of the earliest approaches is proposed by Bobick and Davis [Bobick 2001]. They extract human silhouettes from 2-D images and accumulate the differences between frames of an action sequence. The result is a 2-dimensional binary motion energy image (MEI) and a scalar-valued motion-history image (MHI), where the pixel intensities are a function of the silhouette motion—see figure 2.1(a). The system works in real-time and is a template-based recognition approach that uses  $Hu$  moments to measure the similarity between templates.

In general, methods using a global representation are sensitive to noise, partial occlusions and variations in the viewpoints. Also, the computation of silhouettes is not an easy task requiring controlled environments to achieve accurate segmentations. To partly overcome these issues, grid-based and optical flow-based approaches divide spatially the observation into cells, each of which encodes part of the observation locally. The correlations are usually found locally for each grid patch and propagated to a global volumetric score.

The approach proposed by Shechtman and Irani [Shechtman 2005] estimates optical flow in a 3-D STV. They compute a 3-D STV-template as similarity measurement between two volumes. The similarity measurement is defined hierarchically, where at every 3D location  $(x, y, t)$  of the volume, they compute a small 3D patch around it. Each patch characterizes the flow of the local motion. The local and global similarities between two volumes are obtained by accumulating the local correlation of the patches of both volumes at the same location. For new videos, the system seeks for all possible 3-D volume segments that matches the template (i.e. sliding windows). Their system was able to recognize various types of human actions, including ballet movements, pool dives, and waving. The work of Ke et al. [Ke 2007] uses over-segmented STVs to model activities. Their system applies a hierarchical mean-shift clustering technique to group

similarly colored voxels, and obtains several segmented volumes. The idea is to find only the agent volume and to measure their similarity to the model. Recognition is done by searching for a subset of over-segmented STVs that are similar to the shape contained in the modeled template. The actions are modeled deterministically using Support vector machines (SVM). The system is able to recognize actions from the KTH database [Schuldt 2004] and tennis plays from videos with complex backgrounds.

Also, Rodriguez et al. [Rodriguez 2012] uses synthesizing filters to analyze 3D volumes. They extend the maximum average correlation height (MACH) originally developed for 2D images, now for 3D volumes. Each action is learnt using a synthesized filter fitting the training volume. The recognition is achieved in a new video applying the learnt MATCH. The authors apply the method to movie videos recognizing actions such as hitting and kissing.

In general, STVs are computed as a set of dense descriptors to train the activity models, which is good for the recognition of short actions, but difficult to apply for long term activities. Other difficulties of applying these models to long term activities can be summarized as follows.

- The use of dense descriptors can describe short actions with high detail. Nevertheless, the models require big amount of training data and are hard to update and modify.
- Noisy data is not handled. The background is usually modeled as a part of an action.
- It is hard to recognize actions when multiple-persons are present and when the actions cannot be spatially segmented.

The last point discussed above is addressed by the computation of trajectories which are build as temporal links of interesting pixels or objects. These types of feature are semantically rich and we discuss some approaches which uses them in the next section.

### 2.2.1.2 *Space-Time Trajectories*

In trajectory-based approaches, an object's trajectory is generally represented as a sequence of 2D or 3D points over time.

Human actions can be estimated using the trajectories defined by the body parts (e.g. head, hands, etc.). The body parts estimation is a complex low-level task used to extract the joint positions of a person at each frame.

Using the trajectories described by the joints different approaches are presented. For example, Sheikh et al. [Sheikh 2005] uses an affine projection to obtain a set of normalized trajectories of 13 joints while a person performs an action. Also,

Yilmaz and Shah [Yilma 2005] compare actions of different videos using a set of joint trajectories in 4D (x, y, z, t). The approach of Campbell and Bobick [Campbell 1995] proposes the representation of joint trajectories as curves in low-dimensional phase spaces. Each curve is modeled as a cubic polynomial form and the k most meaningful curves are used as an action model. The system is capable of recognizing simple ballet movements, and the joints are tracked using markers attached to the agents. Other types of features are described by the work of Rao and Shah's [Rao 2001] which track the position of hands in 2D images using a skin detector. The method models actions characterizing the peaks of the hand trajectories and the intervals of time between them. The recognition is a template-based matching procedure capable of detecting actions such as "opening a cabinet" in an office.

The advantage of the usage of joints is the capability of understanding and modeling finer human activities with compact representations. In general, the usage of object-based trajectories is suitable for long term activity understanding. Nevertheless, the approaches described above only focus on short actions. We describe the above methods due to their particular connection with our work. In our work, we use joint trajectories to recognize complex composed activities, which compose the activities described by both hands. The problem of the methods described above, is the difficulty to detect and track the joints in monocular cameras. We overpass such a limitation using of low-cost RGB-D cameras and using a recent approach [Shotton 2011] to track the human joints accurately.

### 2.2.1.3 *Space-Time* Local descriptors

Local representations characterize the observations as a collection of local descriptors or patches. Using local representations the object localization and background subtraction are not required. Usually the use of local descriptors allows representations to be invariant to changes of appearance, occlusions, rotations and scale. Local descriptors describe small windows (2D) in an image or cuboids (3D) in a video volume.

Patches or local descriptors are sampled either densely or at space-time interest points. Local descriptors summarize an image or video patch in a representation that is ideally invariant to background clutter, appearance and occlusions, and possibly to rotation and scale. The spatial and temporal size of a patch is usually determined by the scale of the interest point. Figure 2.2 shows cuboids at detected interest points.

In this section we use equally the terms local features, local descriptors, and interest points. In most of the cases, the systems follow a set of stages:

1. Extract specific local descriptors designed to capture the local motion from a

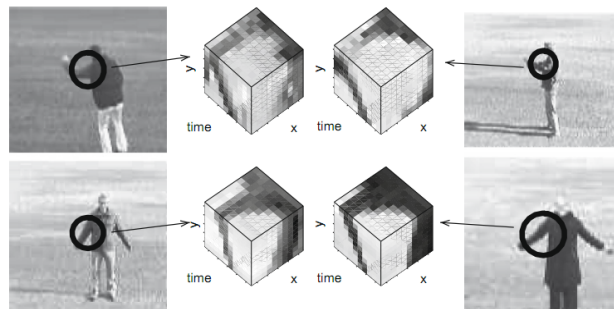


Figure 2.2: 3D cuboids reprinted from [Laptev 2003, Laptev 2005]

### 3-D STV.

2. Combine the descriptors to characterize activities while considering their spatio-temporal correlations or ignoring them.
3. Recognition algorithms are applied to classify the activities.

Some approaches extract local features at each frame, and concatenate them in time to characterize the activity in a STV. For example, the early work of Chomat and Crowley [Chomat 1999] proposed the idea of using local appearance descriptors to characterize an action. They combine motion energy fields with Gabor filters to capture the motion in a STV. They build spatio-temporal appearance descriptors of the motion orientation. The actions are modeled as a set of histograms over the features and the recognition is achieved applying bayesian rule over a set of histograms of a new clip. They can detect simple short human actions such as: "go left", "come". The importance of the work is the introduction of appearance descriptors applied to activity recognition. Similar to Chomat et al., Zelnik-Manor and Irani [Zelnik-Manor 2006] describe an STV as a bag-of-words of different temporal scales. Each word is the intensity gradient orientation of an image local patch. To represent only the moving areas the patches with low temporal variance are ignored. This restricts the approach to detection against non-moving background. They present an inter-histogram distance which ignores the original 3D position of the extracted features.

Other approaches extract sparse space-time local interest points from STV. The space-time interest points describe the locations in a volume where sudden changes of movement occur. Laptev and Lindeberg [Laptev 2003, Laptev 2005] extended the Harris corner detector used for object detection to 3D. They detect scale-invariant interesting points (STIP) searching spatio-temporal corners for non-constant motion patterns. STIP are those points where the local neighborhood has a significant variation in both the spatial and the temporal domain. The scale of the neighborhood is automatically selected for space and time individually. It is possible to detect the changes of direction, splitting and merging of image

structures, they use the descriptors to segment a person from the background. Schuldt et al. [Schuldt 2004] uses the STIP points and classify multiple actions by applying SVMs. This last work introduces as a consequence the KTH actions database widely used for action recognition benchmarking.

The recognition of actions using sparse local interest points from a STV is adopted by several researchers. As its suggested by Laptev and Lindenberg, the usage of sparse points to characterize motion is sufficient to represent short actions. Nevertheless, one drawback of the above methods is the relatively small number of stable interest points. The issue is addressed by Dollar et al. [Dollar 2005], who apply Gabor filtering on the spatial and temporal dimensions individually. The number of interest points is adjusted by changing the spatial and temporal size of the neighborhood in which local minima are selected. A small volume called cuboid is associated to each interest point (see fig. 2.3 (b)). Each cuboid captures pixel appearance (brightness) values of neighborhood interesting points. They use k-means to cluster the training cuboids of an action, building a bag of words. They model an action as a histogram of cuboid word (cluster type). They are capable to classify facial expressions and some human actions of the KTH dataset.

Similarly, Blank et al. [Blank 2005] compute local descriptors at each frame. They first stack silhouettes over a given sequence to form a STV -see fig. 2.3 (a)-. Then the solution of the Poisson equations used to derive local space-time saliency and orientation features. Global features for a given temporal range are obtained by calculating weighted moments over these local features. The authors have applied a simple nearest neighbor classification with an Euclidean distance to recognize actions. Simple actions such as walking, jumping, and bending in the Weizmann dataset, also simple ballet movements are successfully recognized.

Niebles et al. [Niebles 2008] uses the same approximation as [Schuldt 2004] but first smooths the data to reduce its dimensionality by Principal Component Analysis (PCA). They present an unsupervised method for human actions learning and classification over the features extracted by [Dollar 2005]. Their activity "recognition" (discovery) method is generative and models an action as a bag of spatio-temporal descriptors of the object appearance. The approach borrows from the "Natural Language" domain, the method: probabilistic Latent Semantic Analysis (pLSA). Using pLSA they recognize actions in a probabilistic fashion. As a result, they are able to recognize actions from KTH dataset and some skating actions.

Other spatio-temporal extractors have been proposed. For example, Yilmaz and Shah have proposed an action recognition approach to extract sparse features called action sketches. The sketches are built using differential geometric properties on a STV surface (contour) such as maxima and minima in the space-time domain. The method is sensitive to noise on the surface. Scovanner et al. [Scovanner 2007] proposes the 3-D version of the SIFT descriptor [Lowe 1999] constructing a word

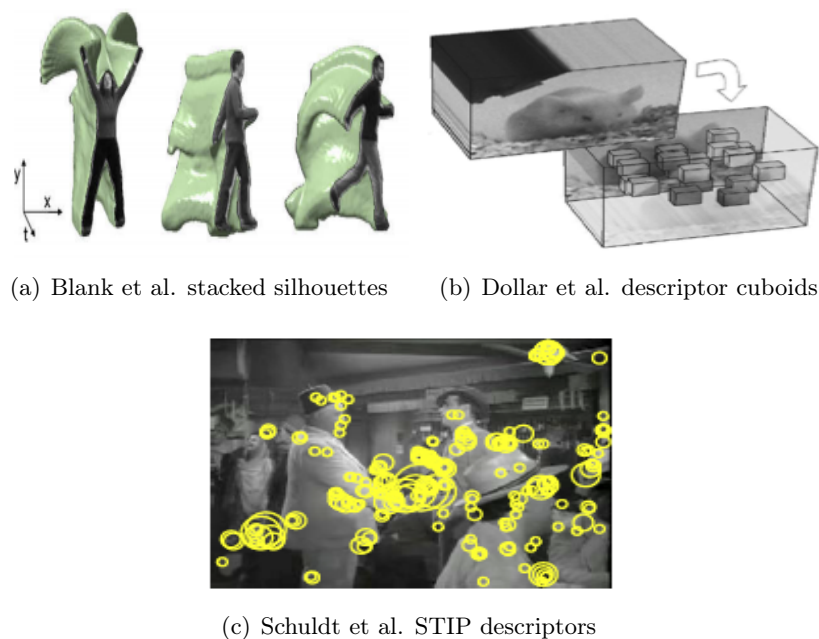


Figure 2.3: Examples of different Space-Time descriptors

co-occurrence matrix, and iteratively merge words with similar co-occurrences until the difference between all pairs of words is above a specified threshold. This leads to a reduced codebook size and similar actions are likely to generate more similar distributions of codewords. Also, Liu et al. [Liu 2008a] present a methodology to prune cuboid features to select the meaningful ones. They use a combination of the space-time descriptors and spin images, which globally describe an STV. A co-occurrence matrix of the features and the action videos is constructed. The matrix is decomposed into eigenvectors and subsequently projected onto a lower-dimensional space. Bregonzio et al. [Bregonzio 2009] propose a detector of cuboid descriptors where the features are selected in a similar fashion as [Liu 2008a]. The method calculates the difference between subsequent frames to estimate the focus of attention and Gabor filtering is used to detect salient points within these particular regions. Rapantzikos et al. [Rapantzikos 2009] improves the cuboid descriptors by their extension to characterize color and motion, improving the previous approaches.

Most of the approaches described above do not consider the spatial nor the temporal relationships of the local descriptors for modeling an activity. These types of method correspond to the family of bag-of-words (BoW). The term BoW in computer vision is borrowed from natural language processing (NLP) and it means that to represent a text, the ordering of appearance of words is ignored. For example, "a good book" and "book good a" are the same under these models.

The BoW approaches described above, show that basic short actions can



be recognized accurately when the temporal and spatial information is ignored. The bag-of-words approaches are particularly successful for basic periodic actions of controlled environments (e.g. KTH dataset). Nevertheless, they fall short when they intend to understand mid-term and long-term activities from the real world. Recently, the interest has turned into characterizing spatial and temporal relationships between the local descriptors.

One approach which characterizes the spatial-temporal distributions of the local features is Wong et al. [Wong 2007]. The approach extends pLSA by including the location of a centroid of correlated motion (e.g. the global movement of an arm), and building an implicit shape model (pLSA-ISM). The authors achieve good results recognizing actions of the KTH dataset. The main difference of pLSA-ISM method and [Niebles 2008] is the description of the relative position of the local features and the center of global motion. The combination of shape and local descriptors can be seen as the combination of global and local representations in space. Also, Savarese et al. [Savarese 2008] proposed a technique to capture spatio-temporal co-occurrences of codewords in local neighborhoods. The problem is the selection of the size of the codebook, where big size codebook introduces noise and small codebooks is not sufficient for discrimination. The codebook size problem is addressed by Liu and Shah [Liu 2008b] who use maximization of mutual information. Basically, the technique merges two codewords if they have similar distributions. Also, they use pyramidal spatio-temporal matching to handle temporal information. Laptev et al. [Laptev 2008] build spatio-temporal histograms by dividing an entire volume into several grids. The method calculates how space-time points are distributed in a grid volume, and measures which descriptors fall in the grid cells.

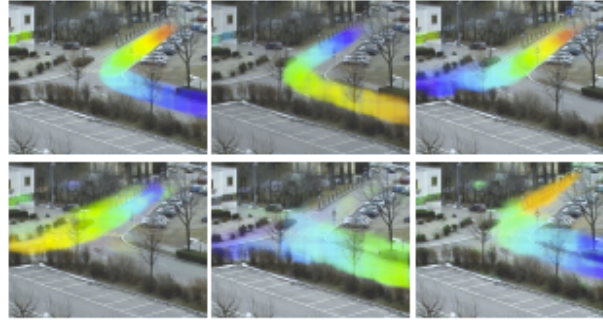
Correlations between descriptors can also be obtained by tracking features. Sun et al. [Sun 2009] calculate SIFT descriptors around interest points in each frame and use Markov chains to determine tracks of these features. Similar work by Messing et al. [Messing 2009] extracts trajectories using the KLT tracker. In both cases, tracks are summarized in a log-polar histogram of track velocities. Also, Oikonomopoulos et al. [Oikonomopoulos 2009] fit B-splines to the STV boundary that is formed by a coherent region of saliency responses. Song et al. [Yao 2009] track points between frames. They fit a triangulated graph to these points to detect and recognize human actions. These methods assume static background and motion due to objects in the background generates feature tracks that do not belong to the person.

*The trend of using topic models:*

Due to the complexity of the relationships between local descriptors, the research community is using probabilistic modeling techniques to characterize activities. *Complex real world data require complex models.* There is a trend on the usage of topic models introducing hidden variables that correspond to action categories.

In most of the cases, the topic model based methods first break the videos into chunks of a few frames or seconds. Documents are created from these clips by quantizing pixel motion at different locations in the images. Wang et al. [Wang 2009b] take a semi-supervised approach and use a semi-latent Dirichlet allocation (S-LDA); the drawback is the manual pre-defined configuration required. Kuetter et al. [Kuetter 2010] represent activities using static co-occurrences of words losing the temporal relationships between words. Also, following the exchangeability assumption and representing each action as a bag-of-words [Wang 2009a] results in losing the temporal dependencies among the words. In structured or semi-structured scenes the long-term or mid-term activities in a video appear temporary constrained. Hospedales et al. [Hospedales 2009] address the temporal constraints by modeling the sequence of activities as a Markov models. The drawback is that the method requires to configure manually a fixed set of topics and that the method is not fully generative. The main challenge in topic-models based activity modeling is the selection of an appropriate model, that is, the configuration of the number of topics needs to be automatic. The estimation of the number of topics can be seen as an analogous problem to set the number of clusters for a clustering technique. Addressing the problem of setting the number of topics, recently, Emonet et al. [Emonet 2011] uses generative Non-parametric Bayesian methods such as Hierarchical Dirichlet Process (HDP) to code spatio-temporal words. The words characterize temporal window which encode meaningful pixel-based motion tracklets. Using HDP they can handle automatically an infinite numbers of topics in theory, while in practice only a finite set of topics becomes important (most populated). The approach is capable of successfully finding interesting motifs (temporal window) automatically from traffic scenes -see fig. 2.4 (a)-.

The topic-model based approaches have the advantage of building models of complex mid-term activities directly from low-level descriptors directly. The recognition of activities is notably accurate in structured scenes and they are capable of detecting interesting unfrequent activities. Nevertheless, in most of the cases they use Gibbs sampling method to learn the hidden model variables. Therefore a first problem is the time required to learn the activity models. A second drawback is that due to the complexity of the learnt models, it is hard to modify manually or explain the reasons of success or failure of the system. A third drawback of the above methods is the lack of global object descriptors making impossible the individual recognition of activities occurring at the same time but in different locations,



(a)

Figure 2.4: Reprint from [Emonet 2011]. Top 6 recovered motifs, explaining more than 95% of the data. Time is represented using a gradient of color from violet to red. Displayed motifs are all composed of 11 seconds

this is, the scene motion is modeled as a whole. Fourth, they still require the configuration of a sliding windows to compute codewords, which makes the recognition process computationally expensive. Finally, the methods are not evaluated for indoor unstructured scenes we are in the process of comparing our approach with these topic-model based methods.

#### 2.2.1.4 *Space-Time Comparison*

Using space-time volumes often has difficulties in handling speed and motion variations. Recognition approaches using space-time joint trajectories can analyze human motion in detail and are view-invariant in most cases. The problem is the detection and tracking of body parts. The spatio-temporal local descriptor-based methods are the most recent. They are robust under vision problems such as: illumination changes and noise. In general they only require low-level vision tasks avoiding difficult foreground subtraction or object classification. Also, it is proved that understanding the relationships of low-level local features it is possible to extract automatically interesting mid-term activities (HDP). The major limitation of the space-time feature-based approaches is that they are not suitable for modeling more complex activities.

Summarizing, the space-time approaches **are suitable for the recognition of temporarily short actions**, gestures and movements. They have been tested on public action datasets (e.g. KTH, Weizmann). Most of the approaches characterize local motions (e.g. "arm movement") but they lack of global description of the scenario and its objects, which makes hard the recognition of long-term activities (e.g. "Person sitting in a chair").

## 2.2.2 Sequential Approaches

Sequential approaches can understand activities by analyzing sequences of features (i.e., feature vectors). They recognize activities in a new video when a learnt sequence is observed. These approaches first extract a vector of features (e.g. a long object-based trajectory) describing the status of an object in the scene (e.g. position) at each time instant. The sets of vector features are analyzed to understand activities by applying similarity measures. The similarity measures compare the new sequences with model sequences representing activities. The models are defined manually or using automatic techniques such as clustering.

We classify the sequential approaches into two categories by using a methodology-based taxonomy: exemplar-based recognition approaches and state model-based recognition approaches.

Exemplar-based sequential approaches describe classes of human actions by using directly the training samples. They maintain a characteristic sequence per activity or a set of training sequences per activity (cluster), and match them (similarity) with a new sequence to recognize its activity.

State-based sequential approaches represent a human activity by constructing a model that is trained to generate sequences of feature vectors corresponding to the test activity. The likelihood (or posterior probability) that a given sequence is generated by each activity model is calculated to recognize the activities.

Sequential approaches are applied to short, mid-term and long-term activity recognition. In particular long trajectories in conjunction with external knowledge of the scene regions of interest is applied to understand complex activities. We divide the state and exemplar-based approaches into two sub categories *short actions* and *long activities* to clarify the final application of the described methods.

### 2.2.2.1 Exemplar-Based Approaches.

#### *long activities*

For long term activities, the used sequences are reduced to long object-based trajectories. Recent work aims at understanding the scene jointly with the trajectories occurring in it.

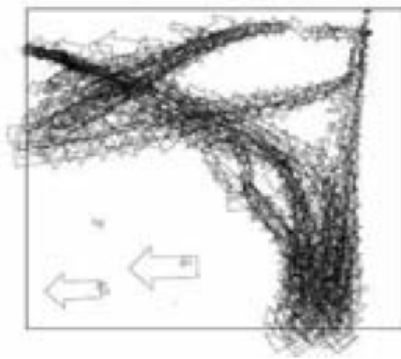
Conceptual data can be learnt by analyzing the object trajectories. The analysis is performed in two steps: 1) patterns on specific regions of the scenario by using clustering techniques; 2) some regions of the scenario can be semantically described (e.g. road, sidewalk) depending on its perceptual properties.



(a) Fernyhough [Jonathan H Fernyhough 1996]



(b) Makris [Makris 2005a]



(c) Johnson and Hogg [Johnson 1996]



(d) Hu et al. [Hu 2006]

Figure 2.5: Scene models of different approaches

Early attempts are done by Johnson and Hogg [Johnson 1996] who propose a method to learn the probability density functions of trajectories. The object movement is described as a sequence of flow vectors. Each vector element represents the position and velocity of the object in the plane. The patterns of trajectories are formed with two competitive learning networks which are connected by neurons. Both neural networks are learnt using vector quantization.

Fernyhough et al. [Jonathan H Fernyhough 1996] recognizes frequent trajectory paths in outdoor scenes by extending trajectories with the spatial extent occupied by the size of the blobs of tracked objects. The method is easy to configure requiring no prior information. One drawback is the lack of discrimination between two objects in the same path but moving in different directions. Also, the method cannot be extended to other domains and requires perfect full trajectories to learn the models. An example of the learnt paths is illustrated in 2.5 (a).

The lack of semantical scenario descriptions is addressed by Makris and Ellis

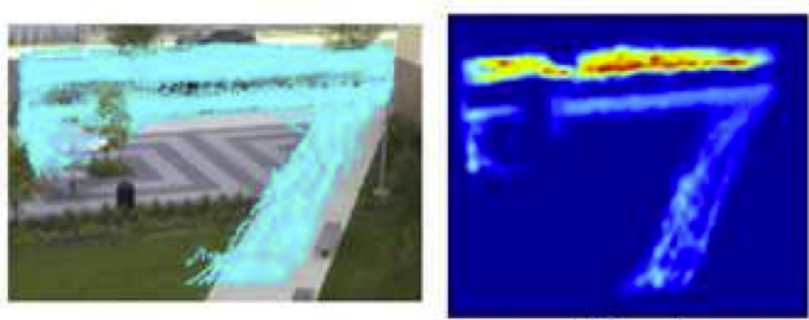
[Makris 2005a]. They learn the entry and exit zones and routes from a set of input trajectories. The set of start/end trajectory points is used to learn entry/exit regions applying Expectation-Maximization (EM). Also, they learn paths by comparing a new trajectory with all exemplar paths previously learnt by a simple similarity measure. The likelihood is used to update an existing path or to initialize a new one. An example of the learnt paths is illustrated in 2.5 (b).

Hu et al. [Hu 2006] drastically improve the learning process by utilizing a complete trajectory as the learning input. They cluster trajectories hierarchically over spatial and temporal features. The resulting motion patterns are represented with a chain of Gaussian distributions. The learnt sequential patterns are used to detect unfrequent activities. The work is a complete framework that proposes clusters and tracks foreground pixels using an adaptive background-foreground segmentation. The tracking procedure also attaches to each object position other appearance features (e.g. color, size, etc.). The motion patterns are learnt using fuzzy K-means clustering algorithm in an hierarchical fashion. The patterns are used to build chains of Gaussian distributions characterizing statistical examples of the activities (paths). Unfrequent activities are seek by the deviations of new motion and the learnt examples. An example of the learnt paths is illustrated in 2.5 (c).

An approach to on-line learning trajectory clusters is presented by Piciarelli et al. [Piciarelli 2006a]. Clusters are represented in a tree like structure, where each edge corresponds to a shared part of long trajectories. The idea is to find the shared spatio-temporal points of trajectories and to build connected clusters in a tree like structure where each cluster represents a bi-directional path. Based on the Euclidean distance they define a new distance measure to decide whether to merge or split the tree paths. The method can detect unfrequent activities but it cannot provide semantical descriptions of the learnt paths. Other similar hierarchical clustering techniques [Antonini 2004, Li 2006a, Patino 2008] allow multi-resolution activity modeling by changing the number of clusters, but the clustering quality depends on the way to decide when a cluster should be merged or not.

Recent contributions extend the set of features used to learn scenario models. Basharat et al. [Basharat 2008] builds a probabilistic map of the scenario (see Fig. 2.6) which combines the trajectory positions and the object size. This way, unfrequent object sizes can be detected. At each frame a vector is created  $(t, x, y, w, h)$ , where  $t$  is the time,  $(x, y)$  is the position and  $(w, h)$  is the 2D size of a bounding box. They compute the relative transformation of the observations in a temporal window and build a transition vector. The transition vectors are later modeled using Gaussian Mixture Models. The method can detect unfrequent activities but is sensitive to the training dataset and to the parametrization.

Adaptive Methods [Piciarelli 2006b, Makris 2005b], where the number of clusters adapts over time, makes possible on-line modeling without the constraint



(a)

Figure 2.6: Scene model learnt by Basharat et al. [Basharat 2008].

of maintaining a training dataset. In these methods it is difficult to initialize a new cluster preventing outlier inclusion. Other methods [Calderara 2007, Bobick 1997] use dynamic programming based approaches (DTW) to classify activities. These methods are quite effective when time ordering constraints hold.

A survey of trajectory-based activity analysis for visual surveillance is presented by [Morris 2008a]. They describe techniques that use trajectory data to define a general set of activities that are applicable to a wide range of scenes and environment. Events of interest are detected by building a generic topographical scene description from underlying motion structure as observed over time. The scene topology is automatically learned and is distinguished by points of interest and motion characterized by activity paths.

### *short actions*

Efros et al. [Efros 2003] present a method for recognizing actions at a distance. Each agent is perceived as a small, 30 pixel, blob-size. They use optical flow descriptors to compute the space-time volume of each tracked person. The motion descriptors are blurry motion channels, and they convert optical flow into a spatio-temporal motion descriptor for each frame. They interpret a video as a sequence of concatenated motion descriptors obtained from the optical flow. The nearest neighbor classification algorithm is applied to the sequence of descriptors for the action recognition. The similarities are calculated frame-to-frame between two sequences and detecting diagonal patterns in the frame-to-frame similarity matrix they achieve the classification of unseen actions. Their system is able to classify ballet movements, tennis plays, and soccer plays, even from moving cameras.

Lubliner et al. [Lubliner 2006] present a methodology that recognizes human activities by modeling them as linear-time-invariant (LTI) systems. They transform a sequence of images into a sequence of silhouettes. From the silhouettes

they extract two types of representations: the width and Fourier descriptors. An activity is characterized as a LTI system. With LTI they can capture the changes in silhouette features. The activities are modeled and classified deterministically using SVMs. Four types of simple actions, slow walk, fast walk, walk sidewise and walk with a ball are correctly recognized.

Veeraraghavan et al. [Veeraraghavan 2006a] describe an activity as a function of time that describes parameter changes. The main contribution of the approach is the explicit modeling of inter- and intra-personal speed variations of activity executions. They learn nonlinear descriptions of an activity in terms of the speed variations. The main idea is to handle partial changes of the execution of an activity. They model an action execution with two functions: 1) a function that describes the changes of the features over time; 2) a function of possible time warping. They extend the classical DTW algorithm to include the function (2) when it attempts to compute the similarity of two sequences. Different actions such as "picking up", "waving", "pushing" are recognized with the approach.

### 2.2.2.2 State Model-Based Approaches

State model-based (or graphical model) approaches are the sequential approaches that represent a human activity as a model composed of a set of states and the transitions among them. The models are trained statistically for each type of activity or manually designed supposing that certain events need to happen. The models are designed to generate a sequence of events (transitions between states) with a certain probability. Usually, each activity has its own corresponding model. For each model, the probability of the model generating an observed sequence of feature vectors is calculated to measure the likelihood between the action model and the input image sequence. In most of the cases the recognition is achieved using either the maximum likelihood estimation (MLE) or the maximum a posteriori probability (MAP) classifiers.

In most of the cases the approaches describing generative models are based on Hidden Markov Models (HMMs) and Dynamic Bayesian networks (DBNs). An activity is represented as a set of hidden states. An agent belongs to one state at each time frame, and each state generates an observation. For the next frame, the system considers the probability of a state transition to evaluate the new state of the agent. When the state transition and observation probabilities are learnt, the unseen activities are recognized by calculating the probability of a given sequence generated by an activity model.

The HMMs need to make two assumptions to keep tractable the modeling of the joint distribution over representation and labels. First, state transitions are conditioned only on the previous state, not on the state history ("the Markov



assumption"). Second, observations are conditioned only on the current state, so subsequent observations are independent.

An early Markov Model approach is proposed by Galata et al. [Galata 2001] presenting a Variable-Length Markov Models (VLMM). The VLMMs unlike HMMs can capture behavioral dependencies that may have a variable or long time scale. In that work, the learning is done assuming an invariant background, where landmarks never change position. The method is not tested under a noisy dataset, where unobserved behaviors may appear.

HMMs have been used in a large number of approaches. The work of Yamato et al. [Yamato 1992] adopted HMMs for the first time. HMMs were at the time, widely used for speech recognition. At each frame, their system converts a binary foreground image into an array of meshes. They cluster grid-based silhouette mesh features to form a compact codebook of observations. An activity is learnt by an HMM that is probabilistically describing a particular sequence of silhouette meshes. The training of an HMM is done efficiently using a labeled dataset and the standard Baum-Welch algorithm. They build one HMM per action. The action recognition is performed finding the HMM that can generate a new observed sequences with the highest probability. They use the Viterbi algorithm to perform the recognition. The Viterbi algorithm [Viterbi 1967] is a dynamic programming algorithm for finding the most likely sequence of hidden states -called the Viterbi path-. They are capable of finding basic tennis plays such as "backhand stroke", but most important, they show that HMMs are capable of producing reliable activity models.

Starner and Pentland [Starner 1995] use standard HMMs to recognize gestures. In particular they aim at recognizing the American Sign Language (ASL). Human hands are tracked and sequences of features extracted from the position and shape of the hands are used to train HMMs. For each gesture a single HMM is trained with position and shape descriptors. Finally they use the Viterbi algorithm to recognize the correspondent ASL word through a sequence of observations.

Jiang et al. [Jiang 2009] consider trajectories as dynamic processes. They use Dynamic Hierarchical Clustering (DHC) to build clusters representing frequent paths. Each cluster is modeled with a HMM. Therefore a set of hierarchical HMMs are learned corresponding to each cluster but no hierarchical relationships are described. The method proposes a scenario updating method, where for each new cluster or updated (trajectory merging) a new HMM is trained and all the trajectories in the database are re-classified. The method uses several iterative algorithms such as the expectation-maximization algorithm to perform the scenario updating. The updating procedure allows error corrections producing more reliable activity models (HMMs).

In addition, approaches using variants of HMMs have also been developed for

human activity recognition (e.g. [Oliver 2000, Natarajan 2007, Park 2004]). Similar to previous frameworks, they build HMMs-based single-layered model (HMM) for each activity they wanted to recognize, and used visual features from the scene as observations directly generated by the model. The methods which extend HMMs are designed to handle more complex activities including relationships between multiple actions and including temporal observations such as the duration of the actions.

Brand and Kettner [Brand 2000] learn activities using HMMs. The training is achieved by first minimizing the entropy of joint distributions. Also, Oliver et al. [Oliver 2000] propose a variant of HMM to model human-human interactions. They name it coupled HMM (CHMM). Basic HMMs are limited to model the activity of a single agent. The incapacity of HMMs to understand interactions of multiple-agents is addressed with CHMMs. They can model complex human interactions coupling multiple HMMs, each one representing an agent. They couple the hidden states of two HMMs by specifying their dependencies. Their system is able to recognize activities such as "two person approaching, meeting and continuing together".

Natarajan and Nevatia [Natarajan 2007] develop a recognition algorithm using coupled hidden semi-Markov models (CHSMMs), which extend previous CHMMs [Brand 2000] by modeling the duration of an activity staying in a state. In the case of basic HMMs and CHMMs, the probability of a person staying in an identical state decays exponentially as time increases. In contrast, each state in a CHSMM has its own duration that best models the activity that the CHSMM represents. As a result, they were able to construct a statistical model that captures the characteristics of activities which improves the HMMs and CHMMs. Similar to Oliver et al. [Oliver 2000], they test their system for the recognition of human-human interactions. Due to the CHSMMs' ability to model the duration of the activity, the recognition accuracy using CHSMMs is better than other simpler statistical models. Also, Lv and Nevatia [Lv 2007] have designed a CHMM-like (graph model) structure called the Action Net to construct a view-invariant action recognition system. The system compares synthetic 3-D human poses and input frames to match action sequences, which are tracked using the Viterbi algorithm.

Park and Aggarwal [Park 2004] use a Dynamic Bayesian Networks (DBN) to recognize gestures of two interacting persons. They recognize gestures such as "stretching an arm" and "turning the head". The method builds a tree-structured DBN to capture the dependent nature among the motions of the body parts. A DBN is an extension of an HMM which is composed of multiple conditionally independent hidden nodes. The nodes generate observations at each time tick (frame) directly or indirectly. In their work, a gesture is modeled as state transitions of hidden nodes (i.e., poses) in time. Each pose generates a set of features corresponding to the body part. The features include: locations of skin regions, curvature points, the ratio and the orientation of each body-part.

Comparisons between the state and exemplar based approaches can be summarized:

- In general, the exemplar-based approaches provide more flexibility for the recognition system, in the sense that multiple sample sequences (which may be different) can be maintained. In addition, exemplar-based approaches are able to cope with **less training** data than the state model-based approaches.
- State-based approaches are able to make a probabilistic analysis of the unseen activities. A state-based approach can estimate a posterior probability of an activity, enabling the easy composition of high level activities. The principal limitations of these approaches is the amount of training data required, which increases while the activity we aim at recognizing becomes more complex.

Sequential approaches consider sequential relationships among features, thereby claiming the detection of complex actions and interactions (i.e. non periodic activities such as sign languages). The recognition of interactions of two persons are reported in [Oliver 2000, Natarajan 2007] where sequential structures are really important.

Another way of modeling activity is adopting a discriminative model. The Conditional Random Fields (CRF) are suited for this purpose. Sminchisescu et al. [Sminchisescu 2005] use a chain of CRFs and they compare CRFs, HMMs and Maximum Entropy Markov Models (MEMMs). They show that CRFs outperform both MEMMs and HMMs when using larger windows, to consider a longer observation history. Also, different variants of CRFs are proposed. Wang and Suter [Wang 2007] defines a Factorial CRF (FCRF), which is a generalization of the classical CRF, where the structure and parameters are repeated over a sequence of state vectors, which can be regarded as a distributed state representation allowing the modeling of complex interactions between labels.

For mid-term and long-term activities, the state-of-the-art approaches are mainly analyzing object trajectories. They are in most of the cases, exemplar based approaches focusing on exploring different clusters over big amount of data. The technique usually explore different techniques to reduce the feature dimensionality without losing description. These types of method are widely used in the traffic domain, where the scene is well structured. Nevertheless, they do not interpret the relationship among the trajectories due to the fact that they usually are complete (i.e. a car in a road crossing the scene). The lack of understanding of the interactions among long trajectories disables the possibility of understanding complex activities with several agents.

To interpret semantical complex mid-term activities, contextual information is introduced to the methods to aid the activity interpretation. For example,

[Porikli 2004] captures spatio-temporal relations in trajectory paths, allowing high-level analysis of an activity, which is very suitable for detecting abnormalities. These methods require prior domain knowledge which is set manually (i.e. scene regions) so the adaptability in time is poor.

Recently, the attention is turned into combine contextual information and object-based trajectories over the scene to provide semantical interpretations. R. Hamid et al. [Ham 2009a] merge the scene topology and sensorial information, modeling sequences of event (n-grams) to discover and classify activities. The method requires manual specification of the scene.

To avoid the manual description of the scene, the approach of Morris and Trivedi [Morris 2008b] learn scene points of interest (POI) and model the transitions between POIs with HMMs encoding trajectory points. The points are similar to Makris et al. [Makris 2005a] semantical regions. The regions are based on the entry and exit complete trajectory points. This approach is suitable to detect abnormal activities and performs well when used in structured scenes (i.e., the usual trajectory paths are well defined, such as in a highway). The method is hard to apply off the shelf to long-term activities which are loosely constraint due to: 1) it requires complete tracks, and in long-term activities the possibility of a tracker error is high; 2) it can not analyze the activity inside a trajectory, when a person is tracked during one hour only the start and end position are considered; 3) the method lacks of description of local motions which is important to have finer descriptions of an activity. Nevertheless, the method follows our proposed work idea of learning the contextual information to provide semantical interpretation.

Naturally, most of the long term activities are composed of sub-activities. The previously described approaches lack of descriptions for composing sub-activities which in many cases implies to understand the characteristics of the modeled activities (i.e. how many times a person uses the fork while eating). In the next section we explore methods which are built hierarchically and are capable of characterizing sub-activities.

## 2.3 Hierarchical Approaches

In general Hierarchical approaches enable the recognition of high-level activities based on first the recognition of simpler activities. The motivation is to follow the "divide and conquer" algorithm, where simpler activities (easier) are recognized and used for recognizing complex activities of higher level of semantics. For example, the recognition of the activity "preparing meal" may be recognized as a sequence of "entering to the kitchen", "using the utensils", "using the stove" interactions. In general the simpler activities are called sub-activities or sub-events.

Hierarchical representations have a set of benefits over the single-layered representations:

- The recognition process is computationally tractable.
- The activity models are conceptually understandable, since most of the model parts are understandable and describing a whole.
- The blocks used to describe an activity are not redundant. A block is usually describing a sub-activity which can be used several times in a model.
- They are capable of incorporating human knowledge (i.e., prior knowledge of the activity) into an activity representation much easier to process.
- Hierarchical approaches are especially suitable for providing a semantic analysis of interactions of humans and/or objects. When modeling high-level activities, non-hierarchical approaches tend to have complex structures and observation features which are not easily interpretable, preventing a user from guiding the recognition with prior knowledge. Hierarchical methods model activity as an organization of semantically interpretable sub-activities, allowing the easy incorporation of prior knowledge.
- They can recognize high-level activities with more complex structures.
- Hierarchical approaches model the activities with a lesser amount of training data and can recognize them more efficiently in new datasets.

Usually activity patterns describing coherent motion are described as a primitive-level (or atomic-level) of actions or sub-activities, the high-level activities are characterized by representing the relationships between the primitive-level sub-activities in an hierarchical fashion. In most of the cases the primitive-level sub-activities are recognized using a single-layered recognition approach described in the previous section.

We brake the hierarchical-based approaches in 3 groups corresponding to the techniques used at the core of the representations: statistical approaches (*multi-resolution states*), syntactic approaches (*grammars*), and description-based approaches (*manual descriptions*).

### 2.3.1 Statistical Approaches

Statistical approaches use graph-based models to recognize activities. In the case of hierarchical statistical approaches, multiple layers of graph-based models (usually two levels of HMMs and DBNs) are used to characterize sequential structures of the activities. The first level, aims at recognizing primitive-activity directly from

the video (like single-layered approaches). The second level interpret the sequence of primitive-activities as observations generated by the higher level activity models.

Multi-resolution HMMs are proposed by Oliver et al. [Oliver 2002]. They introduce the Layered Hidden Markov models (LHMMs). It is a two layered HMM-based approach. The bottom layer is composed of HMMs which recognizes actions by matching the models with the sequence of video feature vectors. The top layer of the HMMs are recognizing high-level sequential activities of the sequences of actions (observations). The system treats the recognized actions as observations generated for the upper layer HMMs. By its nature, all sub-events of an activity are required to be strictly sequential in each LHMM. The human activity in a room: "giving a presentation" is recognized based on the recognition of actions. Nevertheless, the LHMMs are trained separately and with fully labeled data. But getting labeled training data is time consuming in case of the application to real scenarios.

Nguyen et al. [Nguyen 2005] proposes the Abstract Hidden Markov Memory Mode-based approach for recognizing high-level human activities, which is a 2-layers HMM to recognize sequential activities similar to [Oliver 2002]. Zhang et al. [Zhang 2006] also proposes a two-layers HMM approach to recognize group activities in a room. The system recognizes primitive-level actions (i.e. speaking) using the lower-layer HMMs. The group activities (i.e. monologue, presentation) are recognized with the upper HMM layer. Also, Yu and Aggarwal [Yu 2006] propose a block-based 2-layers HMM which is applied to the recognition outdoor activity such as "a person climbing a fence".

Hierarchical HMMs are also applied to the recognition of daily activities. In such a context, Duong et al. [Duong 2005] address the problem of recognizing activities of daily living (ADL). They introduce the Switching Hidden Semi-Markov Model (S-HSMM), a two-layered extension of the hidden semi-Markov model (HSMM) for the modeling task. Activities are modeled with the S-HSMM in two layers: the bottom layer are atomic activities and their duration is obtained using HSMMs; the top layer represents a sequence of interactions between the atomic activities trying to describe high-level activities. They show that the proposed S-HSMM performs better than the HSMMs and the hierarchical hidden Markov model in the recognition of frequent and unfrequent activities.

Other types of hierarchical approaches include the use of DBNs as the core technique. Gong and Xiang [Gong 2003] extends the HMMs to build dynamic probabilistic networks (DPNs). The method is used to detect mostly group activities of the loading and unloading of cargo in trucks. Recently, Dai et al. [Dai 2008] uses DBNs to recognize group activities in a conference room similar to Zhang et al. [Zhang 2006]. Damen and Hogg [Damen 2009] builds Bayesian Networks (BNs) using a Markov Chain Monte Carlo (MCMC) for hierarchical analysis of bicycle-related activities. The BNs are used to represent the relationships between

primitive-level actions and the networks are updated with the MCMC searching for structures that can explain the ongoing action sequences.

Shi et al. [Shi 2004] propose propagation network (P-net) disposed hierarchically. A P-net is defined in a similar way as a HMM. In a P-net an activity is a set of state nodes, the state transitions and observations are probabilities. A P-net allows the activation of multiple state nodes simultaneously, which extends the single-node activation of an HMM. Therefore, a P-net allows the recognition of concurrent sub-activities (e.g. the activities A occurs when B and C), as well as sequential activities. They successfully recognize the activity of a person performing an experiment in a laboratory.

The statistical approaches can recognize constrained activities even with noisy input data. The drawback of these approaches is the inability to recognize activities with complex temporal structures (concurrent activities). For example, an activity "A" starts when the activities "B" and "C" ends, occurs or starts. Such a situation is in general a limitation of the HMM-based and DBN-based approaches, where the edges of two nodes describe a sequential relationship of two sub-activities but not the concurrence between them.

### 2.3.2 Syntactic Approaches

Syntactic approaches describe an activity as a string. The string is composed by symbols. The symbols represent primitive-level actions or sub-activities. Similar to previous approaches the sub-activities need first to be recognized to compose strings.

In most of the cases, an activity is described as a set of production rules. The rules are able to generate sequences of primitive actions which describe a high-level activity. The activity recognition is achieved using parsing techniques over generated strings. In general the parsing techniques are borrowed from the domain of programming languages.

In most of the cases the approaches use as a core technology Context-Free Grammars (CFGs) and Stochastic Context-Free Grammars (SCFGs). The rules of CFGs usually have a hierarchical representation or can be easily structured semantically, similar is the case for the recognition procedures.

The early work of Brand et al. [Brand 1996] uses simple and non-probabilistic grammars to recognize sequences of discrete behaviors. They use foreground segmented blobs as input information; they use also a causal model to disambiguate and parse the input a coherent of action. The causal constraints are drawn from studies of infant perceptual development; as with infants, they precede and may possibly even bootstrap the ability to reliably segment still objects. Their output produces a script of the causal evolution of the scene-output that supports

higher-level reasoning.

Ivanov and Bobick [Ivanov 2000] propose a hierarchical 2-layer system based on SCFGs to recognize high-level activity. The first layer uses HMMs for the recognition of primitive-level actions. The top layer uses stochastic parsing techniques for the recognition of activities based on the recognized actions. They encode a set of stochastic production rules capable to describe all possible activities within the environment. Also, Moore and Essa [Darnell Moore 2002] uses SCFGs for the recognition of activities. Their approach is intended to recognize multitask activities extending [Ivanov 2000]. They are able to recognize activities such as "dealer added card to house" occurring in a game of blackjack.

Minnen et al. [Minnen 2003] adopted SCFGs on the segmentation problem of multiple objects. They show that the semantic processing of activities using CFGs may help the segmentation of objects. They recognize the activity of a person working on the Tower of Hanoi problem without any information on the object appearance.

Joo and Chellappa [Joo 2006] extend SCFGs with an "attribute grammar". Their grammar adds semantic tags and conditions to the production rules. Their grammar can describe feature constraints as well as the temporal constraints of primitive-level actions. They are able to recognize activities in a parking lot by tracking cars and humans. The activities (e.g. parking, picking up, and walk though) are recognized with the position of cars and humans, they are also able to discover infrequent activities.

The syntactic approaches can provide semantical interpretation of video events in real time, and the activity models are easy to modify and update. Nevertheless, the approaches have drawbacks when applied to real world scenarios:

- They are inherently limited for the recognition of high-level activities composed of concurrent sub-activities.
- For the recognition of an activity, the temporal ordering of the sub-activities appearing in a string needs to be strictly sequential. Such a condition restricts the detection of loosely constrained activities.
- They are strictly supervised approaches. The approaches assume that all the events are parsed with the production rules. This is that the user needs to provide a set of production rules to handle all possible events (model the perceptive universe manually). Therefore, the user needs to know all possible unknown observation variations.

To overcome the strictly manual definitions of the production rules, Kitani et al. [Kitani 2007] proposes an approach that aims at learning grammar rules from observations, in an unsupervised manner. Nevertheless, the automatization of this



procedure brakes the main advantage of retrieving manually defined semantics online, and reassembles more to a clustering technique.

### 2.3.3 Description-Based Approaches

A description-based approach explicitly characterizes the spatio-temporal structures of modeled activities. High-level activity is characterized by sub-activities in terms of their temporal, spatial and logical relationships. The recognition is achieved by detecting sub-activities satisfying a defined activity model relationship. In most of the cases, the description-based approaches can handle activities build of concurrent sub-activities.

These approaches usually associate a temporal interval to the occurrence of sub-activities. The temporal interval is used to specify the type of relationships existing between ongoing events. The work of Allen et al. [ALLEN 1983, ALLEN 1994] defines a set of temporal predicates: before, meets, during, overlaps, starts, finishes, equal. The predicates define a set of sequential and concurrent relationships between events which are widely used by literature approaches to characterize high-level activities.

Pinhanez and Bobick [Pinhanez 1998] adopted the Allen's Interval Algebra network (IA-network) to describe the temporal structure of activities. In an IA-network, the sub-activities are nodes and the temporal relationships are described with typed edges. They define a procedure to transform an Allen's IA-network into a *Past, Now, Future* network (PNF-network). A PNF-network can describe the same temporal information as in a IA-network, and making it computationally tractable. The primitive-level actions are manually labeled from the video; the system is able to recognize activities in a kitchen environment even when one of the sub-activities is not provided. The system has two main drawbacks: a sub-activity network has to be specified redundantly if it is used multiple times; all sub-activity relations need to be described in a network form.

Nevatia et al. [Nevatia 2003] design VERL, which is a language to describe human activities. They define 3 levels of activity and organize them hierarchically: primitive events, single-thread composite events, and multithread composite events. Allen's temporal predicates, spatial predicates, and logical predicates are used to represent human activities by specifying their necessary conditions. BNs are used for primitive-activity (event) recognition; HMMs are used for the recognition of single-thread composite events which are sequential. A heuristic algorithm is designed for the recognition of the interactions between multiple persons. The system is probabilistic. Nevertheless the system is not capable to handle the low-level perceptive errors, propagating the low-level errors to the high-level activity recognition. Also, The Vu et al. [Vu 2003] defines a language which is similar to [Nevatia 2003] but extending the representation to characterize infinite

hierarchical layers. Another language is defined by Hakeem et al. [Hakeem 2004] named CASEE, which can represent activity as a conjunction of casual temporal relationships.

Another type of approach widely used are based on Petri Nets (PNets) [Zaidi 1999, Nam 1999, Ghanem 2004]. Using PNets it is possible to define the temporal ordering of the sub-activities composing an activity with a graph, which eases the visualization and control by humans. Each graph node usually represents a temporal state of the completion of a sub-activity. The recognition is achieved by sequentially matching tokens in the graph. [Zaidi 1999] shows that Allen's temporal predicates can be represented using PNets. Nam et al. [Nam 1999] uses the Petri nets for the recognition of hand gestures from videos. Ghanem et al. [Ghanem 2004] uses PNets to recognize interactions between humans and vehicles in a parking lot. The main limitation of PNets is that the high-level activities need to be described in terms of sequential sub-activities, which is not possible for complex scenes and activities.

Recently, Borzin et al. [Borzin 2007] extend PNets to include stochastic timed transitions to the PNets formalisms. The delay between the state transitions is learnt from training examples, using a negative exponential probability function. The resulting formalism is named Generalized Stochastic Petri Net (GSPN), representing the coexistence of immediate transitions and stochastic timed transitions. To tackle with the uncertainty of the detection and tracking processes (if any), Albanese et al. [Albanese 2008] proposes to attach a probability score to the PNet tokens for each object.

In general, due to its deterministic top-down modeling nature the main limitation of description-based approaches is the sensitiveness to the noise or errors that can be introduced by low-level perceptual components. This means that the methods do not learn the errors and therefore they cannot handle them leading to a discrepancy between the manually described model and the real perceptual information acquired.

Addressing the low-level noise, Ryoo and Aggarwal [Ryoo 2008] propose a probabilistic framework that uses a logistic regression to model the probability distribution of an activity, and uses it to detect the activity when some of its sub-activities are misclassified. They compensate the possible failure of the atomic-level components (i.e., no primitive-level action detected) borrowing the hallucination time interval concept defined by [Minnen 2003]. The method can handle low-level critical errors, but not finer errors such as the ones produced by a person's shadow.

Artificial Intelligence (AI) techniques are also proposed. Tran and Davis [Tran 2008] use Markov Logic Networks (MLNs) to infer events in a parking

lot. This is a 2-layered approach, which can describe uncertainties of the human activities. The drawback of the defined MLNs is the assumption that an identical sub-activity occurs only once. Recently, Gupta et al. [Gupta 2009] present a description-based approach for a probabilistic activity understanding. Unlike other description-based approaches designed to recognize complex activities, their approach aims at recognizing primitive-level actions more reliably by modeling causality among the actions. A tree structured AND-OR graph is defined to represent a storyline of a baseball game. Each action is manually labeled and the system iteratively searches for the best match of AND-OR graph structures. Other logic-based approaches include the Situation Graph Trees (SGT) originally propose [Haag 2000] to represent explicitly the combination of temporal, and semantic specializations of Fuzzy Metric Temporal Logic predicates. Also, Siskind [Siskind 2011] proposed a hierarchical approach that is able to represent and recognize high-level activities with more than three levels. They use the force dynamics for the recognition of simple primitive-level actions, and define a method named "event logic" to recognize high-level activities. The approach recognizes the time interval of an activity by calculating the union and intersection of sub-activity time intervals. The idea suggests that a recognized activity can be used as a sub-activity descriptor of another activity, but this is exploited only once in the approach. In our work we exploit this idea explicitly allowing the re-usage of an recognized activity as a descriptive sub-activity of another one at a higher-level.

### 2.3.3.1 Hierarchical Comparison

Hierarchical approaches can recognize high-level activities which can be decomposed into simpler sub-activities. They can incorporate human knowledge into the systems more easily due to its composite nature. In general, they require less training data to successfully recognize activity. Statistical and syntactic approaches provide a probabilistic framework for reliable recognition with noisy inputs. However, they have difficulties representing and recognizing activities with concurrent sub-activities.

In general the main drawback is their fully supervised top-down construction. Even when it can recognize high-order activities, the user needs to define almost all of the expected situations to recognize an activity, such a thing has the following problems for loosely constraint activities.

- The user cannot model all possible variants of the same semantical activity. The deterministic nature of this method makes of updating new instances of a same activity a hard task.
- Due to the disjoint relationships between what the system sees (perceptual information) and what the user thinks that the system sees, the activity models are in many cases not capable of being associated to perceptual data. There-

fore the users need to tune the models considering the low level errors (e.g. person blob shifts due to shadow). Such a task is avoided when the models are learnt.

- The users can define rules only for high-order activities, but when the high order activity is mostly defined by local motion ("eating is described by the arm motion towards the person's mouth") it becomes a hard task for the user to describe such a thing.

# Perceptual Information

---

## Contents

|   |           |
|---|-----------|
| <b>3.1 Global Tracks</b> . . . . .                                      | <b>44</b> |
| 3.1.1 General tracker . . . . .   | 45        |
| 3.1.2 Depth tracker . . . . .   | 46        |
| 3.1.3 Simple tracker, <i>our ad-hoc single person tracker</i> . . . . . | 47        |
| <b>3.2 Braking a video into chunks</b> . . . . .                        | <b>50</b> |
| <b>3.3 Pixel Tracklets (PT)</b> . . . . .                               | <b>51</b> |
| <b>3.4 Perceptual Feature Chunks</b> . . . . .                          | <b>51</b> |
| <b>3.5 Summary</b> . . . . .  | <b>52</b> |

---

In this chapter we describe the blocks of low-level features that we build for our system. The proposed low-level features are extracted directly from visual information. Other low-level features could be added without affecting the proposed framework. The type of features chosen to characterize video depends strictly on the environment and the final application of the method.

For understanding long-term activities, the global position of an agent<sup>1</sup> at each time is crucial information. At each video frame the position is usually represented by a 3D point<sup>2</sup> which allows to understand the relative location of the agent with respect to fixed objects and other agents. The representation of the global position of an agent over time is a trajectory. Tracking agents during long periods of time is a challenging task which is still open. The accurate calculation of agent trajectories is an important task for the proposed framework. Trajectories are the core information used in this work.

The tracking of an agent is usually composed of a vertical chain of successive stages which have to face different challenges. The understanding of the scene configurations (e.g. lighting) can make a tracking method fail or succeed. We propose and use 3 different agent tracking methods depending on the scene configurations and the final application of the system. We explain these methods

---

<sup>1</sup>An agent is the tagged element that we are interested to analyze, e.g. a human, a cluster of coherent motion.

<sup>2</sup>For example, the coordinates of the center of mass of the object silhouette.



Figure 3.1: Examples of classical vision problems for people detection.

in the next section named "Global Tracks".

The understanding of long activities can be treated as a divide and conquer algorithm, where the understanding of small chunks (small sequences of images) of video can explain videos of long duration. We break a video into chunks. In most of the literature approaches, the chunks have a fixed size and do not aim at representing a semantical video property. We propose dynamic size video chunks, where each chunk aims at describing a short action of an agent. Ideally, the video chunks could be described semantically. We explain the details of extracting "Video Chunks" in its correspondent section.

The global tracks of an agent are weak descriptors of the motion that can characterize finer activities. For example, the movement of a human arm is not described by its center of mass. We propose the extraction of a set of pixel-based tracklets<sup>1</sup>. The methodology for extracting the local motion is explained in the section "Pixel Tracklets".

Finally, we build a representation which describes formally a block of information that we use in future chapters to construct our recognition system. Each block is a set of global and local descriptive features representing a semantic action. A block is named "Perceptual Feature Chunk" (PFC) and is detailed in its corresponding section.

### 3.1 Global Tracks

The tracking of an agent using a non wearable sensor such as a fixed camera, carries a set of vision related challenges. In most of the cases, an agent is classified frame by frame and spatio-temporal links of the re-identified agents are established. The vision related problems vary according to the scene. Most of the problems occur in the frame-to-frame tracking of the detected agent. In most of the cases the detection

<sup>1</sup>A tracklet is a filtered, strong, pixel-based trajectory of short duration.

challenges are related to occlusions (complete or partial), illumination changes and cluttered environments (see fig. 3.1 -a, b-). Further, many approaches detect agents using the difference of pixels in time (foreground and background), build clusters of connected pixels and group the clusters into objects. These last methods have temporal problems such as background integration (i.e. an object becomes part of the background due to the lack of motion) and foreground miss-detections (e.g. objects that are moved such as a chair are detected and tracked for long time).

Designing a detection and tracking system requires a priori knowledge of the scene configuration. For example, in outdoor scenes weather conditions (see fig. 3.1 b) need to be considered. In indoor scenes the illumination plays an important role. It is almost sure that for long-term (hours or days) recordings the illumination is going to change radically. Not only the type of methods will define the quality of the tracking system, also the type of sensors and their disposition in the scene. The designer of the system needs to consider configurations that can improve the accuracy of the system. For example, it could be interesting to know that a fixed camera at a certain height can help to prevent occlusion problems, or that strong backlight conditions interfere with the perception of color (see fig. 3.1 c). In the following subsections we explore different tracking systems for different scene configurations.

We use 3 different tracking systems depending on the configuration of the scene and on the type of activities that we aim at understanding. We describe first, a monocular camera segmentation-based system that we call "General track". Second, we describe a depth map based, people and joints tracking system that we call "Depth track". Third, we propose an *ad-hoc* method named "Simple tracker" which is designed to track a single person in an indoor environment during long periods of time. In each of the following subsections we describe the benefits and drawbacks of each method.

### 3.1.1 General tracker

As the name indicates, the general tracker is a multi purpose tracking system. The system is developed under the *Scene Understanding Platform* (SUP) at the *Spatio-Temporal Activity Recognition Systems*<sup>1</sup> research group. It is a foreground-background segmentation-based system. The segmentation has been described by Anh-Tuan et al. [Nghiem 2009]. The tracker [Zúñiga 2011] is probabilistic, and can follow multiple objects at the same time. It contains an object classifier which associates the blobs obtained from the segmentation procedure to a particular class. The classifier transforms 2D moving blob to a 3D parallelepiped box of an object (see fig. 3.2).

---

<sup>1</sup><http://team.inria.fr/stars/>

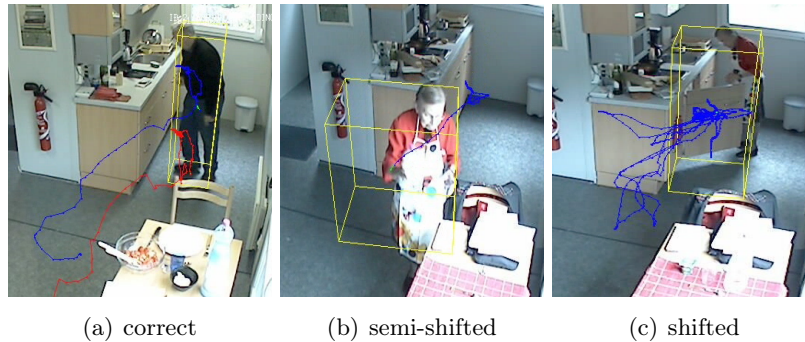


Figure 3.2: Examples of the people detection and tracking in our home-care apartment, under different light conditions. In (a) the artificial light smooths the person shadow allowing a correct location of the person, in (b) and (c) natural light incoming from the window produces detection shifts.

The benefits of a "classical" segmentation-based classification and tracking method is its generality. It can be applied to many scenes using the right configuration to track multiple objects. Nevertheless, the methods are strongly dependent on the segmentation stage. In most of recent approaches the segmentation adapts over time. In long indoor videos, the sudden changes of artificial light (e.g. turn on and off) in combination with the natural light changes (e.g. clouds) and the object shadows introduce unrecoverable failure situations for the tracker. In figure 3.2 we illustrate outputs of the tracking system under different lighting conditions in the same scene. Nevertheless, with the correct parametric configuration the tracking system is able to perform accurately in most of the cases.

### 3.1.2 Depth tracker

Using Depth cameras it is possible to track multiple objects regardless the lighting conditions. The depth camera we use, is composed of an active infrared laser projector (called emitter) combined with monochrome CMOS sensor (called receiver) which captures video data in 3D. The sensor has an angular field of view of 57 degrees horizontally and 43 degrees vertically. The figure 3.3 illustrates examples of the images acquired with the Depth sensor.

The human detector and tracker used in this work is proposed by Shotton et al. [Shotton 2011]. The method has the capability of detecting and tracking also the person joints (see fig. 3.4 a).

The benefits of using a Depth camera is the real 3D human localization, and also the possibility of tracking in low lighting conditions. The people detector algorithm is capable of tracking multiple persons partially occluded, and is robust in long term videos. However, the sensor has difficulties to work in strong lighting



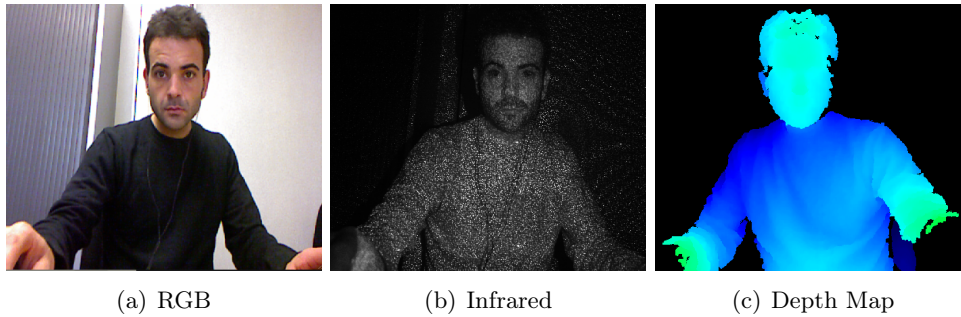


Figure 3.3: Examples of Depth camera images. (a) is a RGB image, (b) is the infrared camera acquisition, the dots are the laser rays which describes particular patterns (shapes), (c) is the depth map colored by distance (depth) to the camera

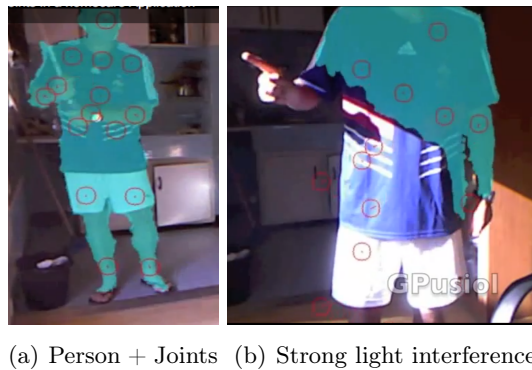


Figure 3.4: Examples of the [Shotton 2011] people detector. (a) A person and 16 joints are detected and tracked. (b) An error occurs when the strong natural illumination interferes with the laser frequency producing a lost of information.

conditions due to interference with the frequency of the laser. (see fig. 3.4 b). The people detector algorithm is hard to train and currently can only detect persons up to 5 meters of distance.

### 3.1.3 Simple tracker, *our ad-hoc single person tracker*

We develop a global motion tracker for home-care applications. The tracker aims at calculating the position of an agent considering the constraint: *"a single agent or non, is present in the scene at any time"*. We assume that the main motion in the scene is produced by the agent most of the time. Therefore, we cluster pixel-based frame-to-frame motion (F2F), to locate globally the main motion in the scene. Unexpected situations can occur (e.g. sudden changes of illumination), they are reflected as multiple spots of important motion in the scene. These situations tend to have a short duration and are handled by the tracking system. Nevertheless, they could be long enough to confuse the global position of the agent. Therefore,

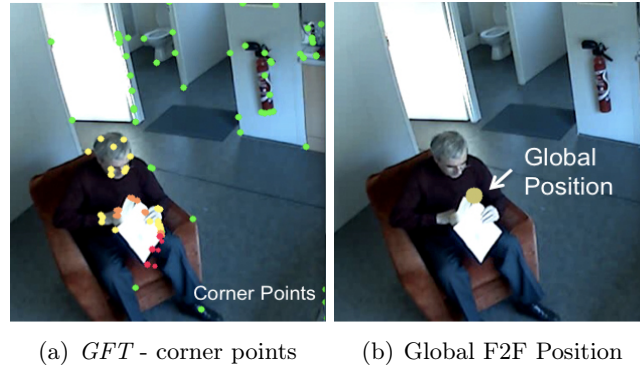


Figure 3.5: Example of the global position of a person computed by the simple tracker. In (a) the points represent the computed *GFTs* and a point color represents its membership to one of the speed clusters. The speed clusters are represented by their speed as: static (green), slow (yellow), medium (orange), fast (red). In (b) the person position is computed by the average of the centroid of the slow, medium and fast clusters.

we use a Kalman filter over a temporal window. The filter smooths the global position of the agent helping to recover the system from failures produced by long unexpected situations.

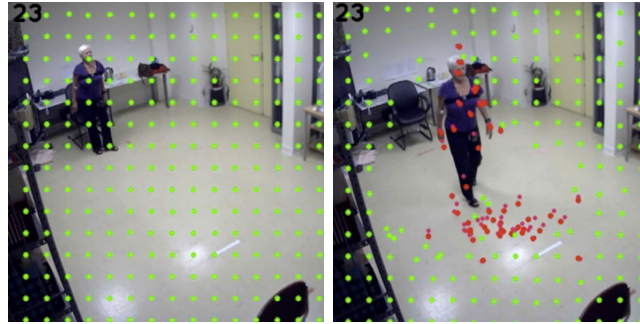
The global position of the agent is computed in two stages: 1) Global F2F Position (*GF2FP*), 2) Global Smooth Position (*GSP*).

### 3.1.3.1 Global F2F Position (*GF2FP*)

The global position of an agent is computed by clustering coherent frame to frame motion. Motion is computed by tracking particular corner points in subsequent images. Initially, we start by locating 500<sup>1</sup> Good Features to Track (*GFT*) as described in [Shi 1994]. Second, we track the *GFTs* over time using a pyramidal implementation of KLT tracking [Bouguet 2000]. Third, at each frame, we compute the speed of each *GFT* by calculating the position translation with respect to the previous frame. Fourth, at each frame, we compute 4 clusters of the points with respect to their speed. The clusters aim at representing static, slow, medium and fast motion. Finally, the *GF2FP* of an agent is the average of the centroids of the 3 fastest point clusters (i.e. slow, medium and fast motion). In the figure 3.5 we illustrate these calculations through the computed *GFTs*, clusters and Global Position (or *GF2FP*).

Other descriptors have been tried. Initially we deploy a grid of points over the image (to cover the whole scene) as initial pixels to track. We apply the tracking algorithm and several mistaken points appear mostly induced by the shadows of

<sup>1</sup>The parameter "500" is scene dependent



(a) Initial grid of points      (b) Pyramidal KLT errors

Figure 3.6: Example of points tracking [Bouguet 2000] where the initial points are random pixels extracted from a grid of fixed size. In (a), the initial grid. In (b), the tracked moved points appear in red color. The person’s shadow makes that some points appear like moving but they do not correspond to the silhouette of the person which can be considered as an error for our purposes.

the agents while moving. In figure 3.6 we show examples of the grid and the tracking errors. Also, besides KLT, we have tried the SIFT and SURF strong pixel descriptors, they perform similarly to KLT but with much slower computational speed, while with KLT we process the video in real time.

We propose the frequently **re-initialization** of the tracking system (i.e. features and tracking) to minimize errors in the  $GF2FP$  estimation. Principally two types of error are avoided. First, in some situations, tracked points are confused with the background. The confusions over long periods of time can harm the consistency of the  $GF2FP$ . Second, the  $GFTs$  are mostly concentrated in region with high variation of texture. Agents appearing in a low texture region are not tagged by initialized  $GFTs$ . When the system is re-initialized and an agent is in the low texture regions new  $GFTs$  are stucked to the borders of the person’s silhouette improving the final  $GF2FP$ . The re-initialization is performed by setting a fixed amount of frames, or by an external signal that measures some agent properties (e.g. changes in the agent speed). The re-initialization calculates a new set of initial  $GFTs$  and links the old agent’s global position with the new frame calculations.

Other ambiguous situation is when the high motion occurs far from an agent. For example, a person (agent) reading in an armchair (low motion), and reflexions in a window are perceived (high motion). In our experience, the erroneous sources of high motion occur during short periods of time and are generally due to illumination changes. Smoothing the  $GF2FP$  into  $GSP$  allows the filtering process of isolate illumination noise. The smoothing is performed using a Kalman filter and it is described as follows.

### 3.1.3.2 Global Smooth Position (*GSP*)

The *GF2FP* of an agent is smoothed using a linear Kalman filter  $K_1$ . At each new frame  $f$ , the prediction of  $K_1$  is averaged with the new observed position  $o_f$ , obtaining a smoothed trajectory of positions  $\{p_f\}$ :

$$p_f = \frac{1}{2}(o_f + K_1(p_{f-1}))$$

In general, the Kalman filter is configured to hold a buffer of 10 position values which is equivalent to 10 frames, and therefore in the case of tracking sharp changes of direction are smoothed with a small tracking delay.

## 3.2 Braking a video into chunks

We brake the video into dynamic or static video chunks. A video chunk is a short sequence of images (frames). A video can be braked into fixed-length chunks by simply parametrizing the chunk length (i.e. number of frames). The problem of using fixed length chunks for long term activities is the redundant stored information. For example, the interesting activity of a person sleeping (e.g. rolling) can be detected using long (2 seconds) video chunks as well as using a shorter chunk length, nevertheless shorter length requires more storage which unnecessary increases the computational time required by the recognition process.

We propose to adapt the chunks to interesting atomic events described by the agent motion. In particular we propose to detect and use the positive and negative acceleration of an agent to brake the video into chunks. In our experience, the acceleration of the global position of an agent can provide insights its current activity (or state). Therefore, the changes in acceleration can describe the start and end of an action which are in most of the cases semantically explainable.

Regardless the method used to detect and track the global position of an agent, we detect the changes of the acceleration by measuring the speed of the agent at all frames. For each new frame  $f$ , we compute the observation speed  $o_f^s$  as the difference of the agent global position at frame  $f$  and the position in previous frame. To filter errors, the speed  $s_f$  is computed by applying a Kalman filter  $K_2$  to the observations and by averaging the prediction  $K_2$  and the observed speed  $o_f^s$ :

$$s_f = \frac{1}{2}(o_f^s + K_2(s_{f-1}))$$

Finally, we compare  $s_f$  with a threshold  $\theta^s$ . When  $s_f$  gets higher or lower than  $\theta^s$  a video chunk starts or ends.

**A remark**, the start of a new video chunk triggers a re-initialization signal to the *GF2FP* described in the previous section.

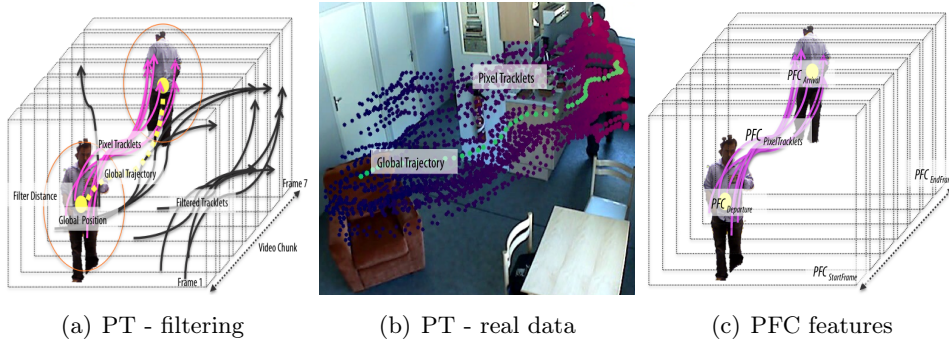


Figure 3.7: The figure illustrate the pixel tracklets and the global trajectory of the PFCs using real data and illustrations.

### 3.3 Pixel Tracklets (PT)

For each video chunk we compute a set of tracklets which aim at describing the motion of the agent mobile parts. The tracklet are pixel-based points which are tracked during the whole length of the video chunk. In a similar way as for the *GF2FP* we initialize in the first frame of a video chunk 500 good features to track. Using a pyramidal implementation of KLT [Bouguet 2000] we track the points up to the last chunk frame.

*Filtering the tracklets:* From the initial 500 tracklets only the ones related spatially to the agent remain. The other tracklets generally correspond to environmental noise. The filtering of uninteresting tracklets is performed by analyzing their proximity to the Global Position of the agent at the first and last frames of a video chunk. The proximity is set up manually as a filter distance or automatically assigned using the eroded silhouette of the agent. In figure 3.7 (a) the graphical representation of the filtering is illustrated, and in (b) the Pixel Tracklets are computed for a real video chunk.

### 3.4 Perceptual Feature Chunks

We reduce the dimensionality of a video chunk data from pixel volumes to a set of perceptual features. Each video chunk has its own set of features named Perceptual Feature Chunk. The figure 3.7 (c) illustrates the perceptual features which are described as follows:

1.  $ID_{Chunk}$ : Assigns an unique identification number to each chunk.
2.  $Video_{PFC}$ : Links the chunk with a particular piece of a video datafile.
3.  $Departure_{PFC}$ ,  $Arrival_{PFC}$ : Describes the spatial position of the person in the begin and end of a video chunk. They are represented by a Gaussian

distribution of parameters  $(\mu, \sigma)$  computed with the  $n$  first (or last) points of the global trajectory described by a tracked agent.

4.  $StartFrame_{PFC}$ ,  $EndFrame_{PFC}$ : Represents the number of the first and last frame of the chunk with respect to the whole video.
5.  $PixelTracklets_{PFC}$ : Is a vector of pixel tracklets, and each tracklet is a vector of spatial points representing the motion of the agent.

### 3.5 Summary

In this chapter we explain different methods to detect and track the **global position** of an agent in different scenes. We propose a technique to describe the agent **local motion** when object trackers fall short. The proposed techniques which are designed to work **on-line** (real time). We propose to **brake the video** into clips of fixed or **dynamic lengths** which avoid vision problems (e.g. illumination) by feed-backward initializations of other systems (i.e. trackers). Finally, we propose a **compact data representation (PFCs)** which stores only minimal but significant information. These PFCs are used in the following chapters as the main perceptual descriptors to understand long-term activity.

# Contextual Information

---

## Contents

---

|            |  |           |
|------------|--|-----------|
| <b>4.1</b> | <b>The importance of Contextual Information for activity understanding</b> | <b>53</b> |
| <b>4.2</b> | <b>Topology: A single-resolution spatio-temporal contextual model</b>      | <b>55</b> |
| 4.2.1      | Topology representation  | 56        |
| 4.2.2      | Topology training input  | 57        |
| 4.2.3      | Topology Learning, computing the scene regions                             | 57        |
| 4.2.4      | Computing the scene region parameters                                      | 59        |
| 4.2.5      | Clustering with K-Means  | 59        |
| 4.2.6      | Distance issues  | 60        |
| <b>4.3</b> | <b>Topology examples</b>   | <b>61</b> |
| <b>4.4</b> | <b>Topology Updating</b>   | <b>62</b> |
| 4.4.1      | The alignment of topologies  | 62        |
| 4.4.2      | Updating algorithm   | 70        |
| 4.4.3      | Statistical Analysis   | 71        |
| <b>4.5</b> | <b>The Scene Model, a Multi-Resolution representation</b>                  | <b>72</b> |
| 4.5.1      | Selection of the most relevant abstraction level                           | 73        |

---

In this chapter we discuss the importance of contextual information for activity analysis. We define a method to learn contextual information. We define a representation that helps to extract semantics from the learnt contextual information and we call this representation a *Topology*. We define operations between topologies to normalize and update them. Finally, we propose a multi-resolution scene model based on groups of topologies. The scene model constitutes the support to discover activities automatically.

## 4.1 The importance of Contextual Information for activity understanding

The methods that we refer below fall in the category of trajectory-based activity recognition, and are capable of discovering activities automatically or at the most with minimal supervision. This is the challenging category where the approach

proposed in this thesis can be located as well.

In the current literature, most of the trajectory-based activity recognition methods do not use prior contextual information to characterize activities. Briefly, the literature methods are composed of three stages: first, they cluster the trajectories to discover salient paths, routes or regions. Second, the interesting clusters are classified semantically and some cluster's features are used to learn activity models (e.g. speed, orientation). Finally, a recognition procedure aims at matching new information with the learnt models. The described methods are suitable to detect unfrequent activities/occurrences (e.g. "a car goes in the wrong way") in structured scenes (e.g. "traffic junction"). Nevertheless, what the methods are really doing is clustering motion without analyzing the source or underlying meaning of the motion. This brute-force way of abstracting perceptual information (single-layer) leads to different problems.

- These methods fail to model and recognize complex activities (multi-motion) in unstructured scenes (e.g. "an apartment").

The methods strongly depend on the repetition of time-constrained motion (trajectories). They assume that similar frequent motion (a cluster) is characterizing uniquely an activity, and that the cluster can be explained semantically. This is usually true in structured scenes, where the static scene objects structure the mobility of the tracked agents. For example, a road structures the possible paths that a car can take. The activities are not complex (e.g. "car turn left") and some clusters can characterize the activities correctly.

In the case of unstructured scenes the situation is different. Only few assumptions can be made on the motion of the tracked objects. For example, in an apartment there are no general rules on how to go from one place to the other nor the speed required. Even more, when the tracked objects produce high mobility (e.g. "human") the activities can become complex. In this case, multiple different motions characterize a single activity. In most of the cases, for unstructured scenes a motion cluster/path/route is too simple to model complex activities composed of several concurrent movements and consequently the recognition fails.

- In general, the methods present to the user a set of clusters of frequent or coherent motion in the scene that is hard to understand and that cannot provide semantical explanations (semantic gap problem).

One of the reasons of the above limitations is the lack of prior knowledge of the scene. We believe that such information is very important to characterize activities.



## 4.2. Topology: A single-resolution spatio-temporal contextual model 55

---

Recently some works have proposed the usage of prior contextual information to characterize complex activities. The knowledge is based on spatial scene regions that can be explained semantically. These works use the scene regions to compute semantical events that characterize an instant of a video. The events are computed from the relation of the observed perceptual features and the scene regions (i.e. object near region A). Finally the events are used to characterize different activities. The approaches use scene regions to compute events and activity models in different ways. For example, Makris et al. [Makris 2005c] learn paths among the regions; Veeraraghavan et al. [Veeraraghavan 2006b] use context free grammars (CFG); Hamid et al. [Ham 2009a] use n-grams. Bremond et al. [Vu 2003] use Finite State Automats. In general, event-based methods can deal with the semantical gap problem and are suitable for complex activity understanding. Nevertheless, most of the state of the art methods have one of the following drawbacks.

1. The scene regions are manually defined. This produces that:
  - It is time consuming and requires constant supervision over time. For example, when an object is moved, a supervised re-definition of the scene is required.
  - The user subjectivity at the moment of defining the scene regions affects the resolution of the activity that can be analyzed.
2. In most of the cases, the methods use a single resolution definition of the scene interesting regions. The single resolution disables the possibility of multi-resolution activity analysis (e.g. activity and sub-activities).

Addressing the above mentioned limitations, in this chapter we propose: 1) a method to learn automatically a set of meaningful scene regions that we name a *Topology*; 2) we propose methods to update topologies depending on the spatio-temporal changes described by the tracked agents over time; 3) we propose a multi-resolution representation of the scene capable of characterizing interesting regions and sub-regions. The scene model is used later (Chapter 6) for multi-resolution activity discovery purposes.

## 4.2 Topology: A single-resolution spatio-temporal contextual model

A topology is a single-resolution scene representation composed primarily of regions. The representation aims at characterizing interesting scene regions that could be explained semantically. For example, regions where a tracked object interacts with fixed scene objects (see Fig. 4.1). Each region can characterize spatial and temporal properties of the scene (i.e. "kitchen location", "kitchen usage"). A topology can provide a single semantical abstraction of the scene regions but it does not provide information about the sub-regions (multi-resolution).



Figure 4.1: Example of the "Casa" scene. The blue dots are the location of interesting scene objects and regions.

A topology is computed by learning the scene regions automatically. To learn these regions we use perceptual features from a set of tracked objects used for training. The main idea behind the learning procedure is to discover and characterize strong spatial and temporal scene regions that are commonly used by most of the training objects. Learning strong regions has two benefits:

1. It normalizes the scene space. This allows to abstract the perceptual features into semantical explainable object motions, filtering out the acquisition noise (e.g. a person walks from the kitchen to the armchair).
2. It allows the following assumption: When a region (e.g. "kitchen") is frequently used to perform an activity (e.g. "preparing meal") it is highly probable that a new object uses the same region to perform the same activity.

#### 4.2.1 Topology representation

A topology ( $T$ ) is a generative model represented as a vector of scene regions (SR).

$$T_{level_k} = \langle SR_1, \dots, SR_k \rangle \quad (4.1)$$

where  $k$  is the number of scene regions. The number of regions is an important parameter that describes the resolution (or semantical abstraction) in which a

## 4.2. Topology: A single-resolution spatio-temporal contextual model 57

scene is characterized. For example, the *lower levels* define a coarse resolution of the scene (e.g. "the kitchen area") while in *higher levels* the scene resolution is refined (e.g. "the sink area", "the oven area").

Each *SR* characterizes a spatial partition of the scene and is represented as a Gaussian distribution:  $SR_i \sim (\mu^i, \sigma^i)$ . The computation of the parameters  $\mu^i$  and  $\sigma^i$  is defined in (4.2.4).

### 4.2.2 Topology training input

We propose the usage of a set of *PFCs* sequences to compute a topology. Each sequence is a training dataset performed by an unique tracked object (e.g. "person A", "person B", etc.).

The type of training datasets used is defined by the final application. For example, datasets of a same person can be used to analyze the temporal evolution of an activity, while datasets of different persons can be used to analyze the divergence of performing the same activity.

There is no restriction in the temporal occurrence of a training dataset. For example,  $N$  objects can be tracked at the same time, producing  $N$  training datasets (e.g. tracking left and right hands of a person), this information can be used to understand shared spatio-temporal locations in the scene (object interactions).

Formally, the *Input* set of *PFC* sequences is defined as:

$$Input = \{Seq_1, \dots, Seq_n\}$$

where  $Seq_i = \langle PFC_1^i, \dots, PFC_k^i \rangle$  and  $i$  is the label of a tracked object.

### 4.2.3 Topology Learning, computing the scene regions

We propose to learn the scene regions by a procedure composed of two clustering stages. The two stages are designed to minimize the contribution of individual agent outliers in a global (i.e. for all tracked agents) topology characterization.

1. The **first stage** aims at discovering the interesting regions for each tracked agent individually. The procedure operates with the set *Input* of *PFC* sequences defined in 4.2.2. For each sequence (*Seq*) a clustering procedure is performed over a particular set of points:  $Points_{Seq}$ .

The set  $Points_{Seq}$  is a dense set of points characterizing the location of the changes of the agent motion. These points can describe the regions where an

interesting activity occurs (i.e. interaction with the scene). We build the set  $Points_{Seq}$  using the  $\mu$  parameter of the  $PFC_{Departure}$  and  $PFC_{Arrival}$  of all  $PFCs$  contained in a sequence  $Seq$ .

$$Points_{Seq} = \{PFC_{Departure(\mu)}\} \cup \{PFC_{Arrival(\mu)}\} \forall PFC \in Seq \quad (4.2)$$

For each  $Seq_i \in Input$  the clustering produces a set of  $k$  clusters ( $Cl$ ).

$$Cluster(Points_{Seq_i}) = \{Cl_1, \dots, Cl_k\} \quad (4.3)$$

Each of the above clusters can characterize the scene regions described by the motion of the tracked agent  $i$ .

2. The **second stage** merges the individual scene regions into a single set of regions. Each region is a new cluster of points ( $CL$ ). The  $CLs$  aims at describing the spatial regions shared by most of the training objects. The input to this stage are the centroids of the clusters  $Cl$  computed in (4.3). The usage of only the cluster centroids helps filtering out tracking noise (i.e. the trajectory of an object is noisy when the object does not move) and outliers (e.g. an object remains still most of the time in a single region).

$$Cluster\left(\bigcup_{i=1}^n Centroids(Cluster(Points_{Seq_i}))\right) = \{CL_1 \dots CL_k\} \quad (4.4)$$

For clustering we use the K-means algorithm. K-means algorithm assumes the isotropic distribution of data. The visual analysis of the input, reveals that the points surrounding an interesting region are isotropically distributed. Another property of K-means is its sensibility to outlier data, this is important to characterize regions that are not frequently visited.

In both stages the same number of clusters  $k$  is set manually. This produces a topology of  $level_k$ . In section (4.4) we propose an automatic topology updating procedure. The procedure can increment or decrement automatically the number of scene regions of a topology (i.e. change the resolution). Such a procedure can be directly applied to the second stage described above to detect automatically the number of clusters  $k$ .

In general, we prefer to control manually the number of clusters to be produced when learning for the first-time a topology. We have experimented with the automatic detection of the topology resolution. In our experience the automatic resolution detection prevent the user to keep the reference of the semantical interpretation of a scene region.

#### 4.2.4 Computing the scene region parameters

The parameters  $\mu$  and  $\sigma$  of a  $SR_i$  are obtained from the clustering information. Each cluster  $CL$  of eq. (4.4) is characterizing a scene region. The  $k^{th}$  cluster ( $CL_k$ ) is characterizing the  $k^{th}$  scene region ( $SR_k$ ). For a scene region  $SR_k$  the parameters  $\mu$  and  $\sigma$  are computed as follows.

The mean ( $\mu$ ) is the spatial coordinates of the center of  $CL_k$

$$SR_k(\mu) = Centroid(CL_k)$$

where,  $\mu \in \mathbb{R}^2$  or  $\mathbb{R}^3$ .

The standard deviation ( $\sigma$ ) is computed from the set  $\{x_1...x_n\} \in Points_{Seq}$  (see eq. 4.2). Where each point is associated to the cluster  $CL_k$  by transitivity over the two clustering stages. The set of valid points  $x_i$ , assert the following statement.

$$\{x_1...x_n\} \in Cl_i \wedge Centroid(Cl_i) \in CL_k \quad (4.5)$$

then

$$SR_k(\sigma) = Std(x_1, \dots, x_n) \quad (4.6)$$

where  $Std$  is the standard deviation of the Euclidean distance of the points  $x_i$  w.r.t the cluster center.

The usage of  $\{x_1...x_n\}$  allows a real description of the variation of the motion in a region even when the amount of agents used for training 1.

#### 4.2.5 Clustering with K-Means

K-Means was first used by MacQueen [MacQueen 1966]. It is an unsupervised learning algorithm that solves the well known clustering problem.

The procedure follows a simple way to classify a given data set through a certain number of clusters (assume  $k$  clusters) fixed a priori. The first step is to define  $k$  centroids, one for each cluster. The next step is to associate each point of the input dataset to the nearest centroid. When no point is pending, the first stage is completed and an early grouping is done. The next stage is to calculate  $k$  new centroids as the barycentres of the clusters resulting from the previous stage. After we have these  $k$  new centroids, a new binding is performed between the input data points and the nearest new centroid. A loop is generated. The loop iteratively produces a change of the location of the centroids. The loop converges to the final clusters when the  $k$  centroids do not change their position.

Briefly, the algorithm is composed of the following steps.

1. Select  $k$  data points in the input data point space. We select  $k$  points from the *Perceptual Input* set. These points represent initial group centroids.
2. Assign each input datapoint to the closest centroid, building  $k$  groups. Each group represents an intermediate scene region (SR.spatial).
3. Recalculate the positions of the  $k$  centroids for each group after step 2
4. Repeat the steps 2 and 3 until the centroids are no longer moving. This produces a separation of the objects into groups. Each group is a final SR.

Formally, given the set of of observation  $x_1 \dots x_n$ , k-means aims of partitioning the  $n$  observations into  $k$  sets  $Cl$  (clusters). At each iteration, the algorithm aims at minimizing an objective function, in this case a squared error function. The objective function

$$\arg \min_s \sum_{i=1}^k \sum_{x_j \in Cl_i} \|x_j - \mu_i\|^2 \quad (4.7)$$

where  $\mu_i$  is the mean of the points in  $Cl$ .

#### 4.2.6 Distance issues

K-Means requires the selection of a distance measurement. The selection of the distance measurement plays an important roll in clustering points. The distance measurement describe the similarity between a data point and the cluster centroid (i.e.  $\|x_j - \mu_i\|$  of eq. 4.7). We have tested different distance measurements:

1. Euclidean distance
2. City-block distance
3. Pearson correlation
4. Uncentered Pearson correlation
5. Spearman's rank correlation
6. Kendall's  $\tau$

The first two of the above distance measures are related to the Euclidean distance, while the remaining four are related to the correlation coefficient.

An extensive description of the equations for each of the above distances and experimental results, is provided in the section [A.1](#).

From the experiments, the euclidean based distances show to be appropriated to perform the spatial *SRs* characterization.

### 4.3 Topology examples

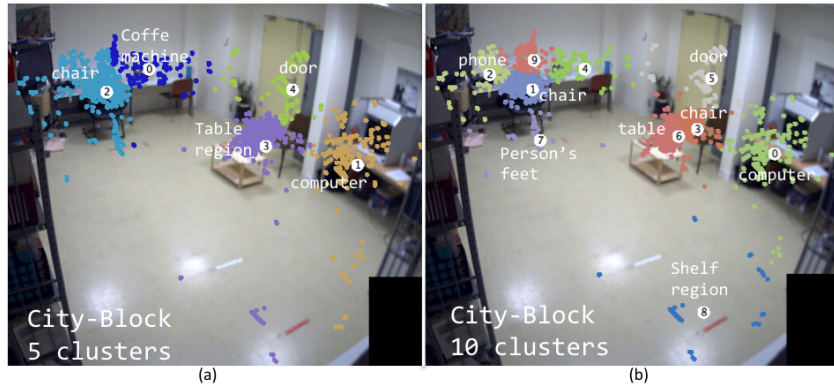


Figure 4.2: Example of k-means clustering using city-block distance measurement; a) the number of clusters is set to 5. b) the number of clusters is set to 10. Some of the discovered regions are labeled with semantical concepts. In most of the cases, the regions can be linked with the scene objects used by the agents (interactions).

Figure 4.2 displays an example of the *SRs* calculated in a hospital room (HOSPITAL dataset). The *SRs* are computed with *PFC* sequences of 4 different persons performing activities during 40 minutes each. The examples display different topology resolutions, at each level it is possible to explain semantically most of the computed regions. For each calculated region the center  $\mu$  is displayed as a labeled dot. The label is the number of the region ( $i$  of  $SR_i$ ), this label becomes important in posterior stages.

Figure 4.3 displays an example of the *SRs* calculated in an apartment scene (GERHOME dataset). The used *PFCs* correspond to data of 3 different persons performing activities for 4 hours each. The different topology levels are displayed (levels 5 and 10), explaining some regions semantically. Also, examples of the type of activities that can be observed in the regions are provided.

Figure 4.4 displays an example of a computed topology in an apartment scene (CASA dataset). The used information is extracted from RGB-D (Red-Green-Blue and Depth) image-real 3D-. The image shows the regions described by an individual performing activities for 30 minutes. The *PFCs* used have a fixed length of 1 second each.

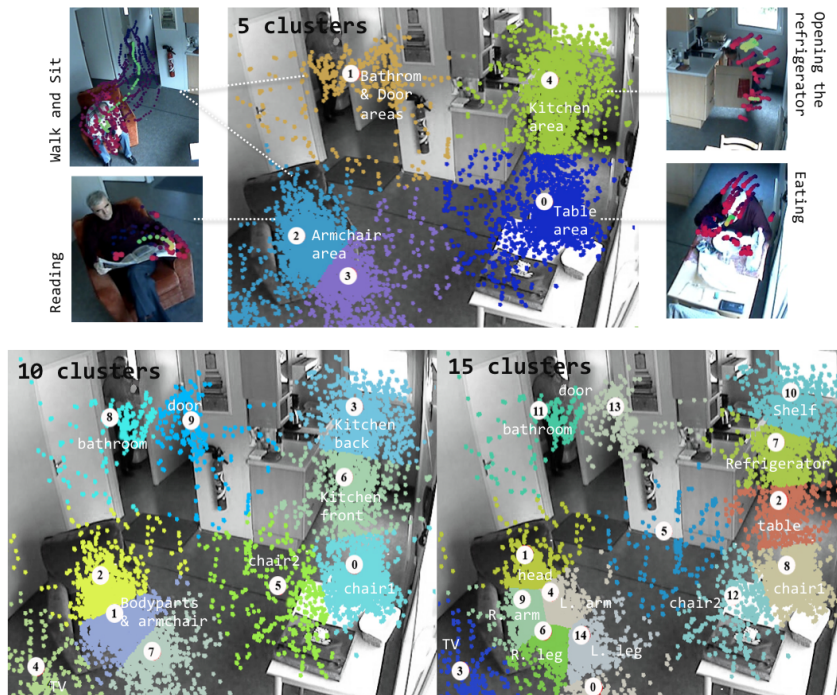


Figure 4.3: Example of k-means clustering using the euclidean distance. The scene corresponds to an apartment. The clustering procedure is set to 5, 10 and 15 clusters. The discovered regions are explained semantically and some activity examples are displayed to show the type of activities occurring at each region.

## 4.4 Topology Updating

The updating of a topology consists in adding new *SRs* and removing old ones automatically. New *SRs* are obtained of new training data. The topology updating leads to scene models which are robust to changes over time.

The automatic updating of a topology is achieved as a consequence of an alignment procedure. The alignment procedure is an operation between two topologies (e.g. the "reference" and the "updater"). The alignment procedure aims at identifying the matching and non matching *SRs* of the two topologies. The updating procedure evaluates the matching and non-matching *SRs* from the alignment to build an "updated" topology (see Fig. 4.5).

### 4.4.1 The alignment of topologies

The challenge of alignment consist in finding the matching scene regions between two topologies. At a first sight, it is possible to assume that both topologies have the same coordinate systems. Therefore, an algorithm based on the implicit-alignment





Figure 4.4: 3D regions computed in the CASA dataset. Each computed region can be semantically explained and examples of frequent activities at each region are provided.

of the  $SRs$ <sup>1</sup> could be used to find the matching  $SRs$ . Nevertheless, the assumption is not always true. The location of a  $SR$  can change due the noisy data used to learn them. In general two types of change are observed:

1. *Translation*: This is the linear shift of the  $SRs$  over the scene. It can occur due several data acquisition reasons such as: illumination changes (i.e. shifts produced by the agent shadow), modifying the object tracking algorithm, a scene object changes its location.
2. *Rotation*: This is the radial displacement of the  $SRs$ . It generally occurs when the camera is moved (e.g. cleaning the camera).

The figure 4.6 displays examples of the above mentioned problems observed in the real data that we use for experimentation.

<sup>1</sup>Most point pattern matching problems belong to this class -Rigid transformation functions-. Here the process of finding the  $SRs$  correspondences and finding the optimal alignment are performed simultaneously.

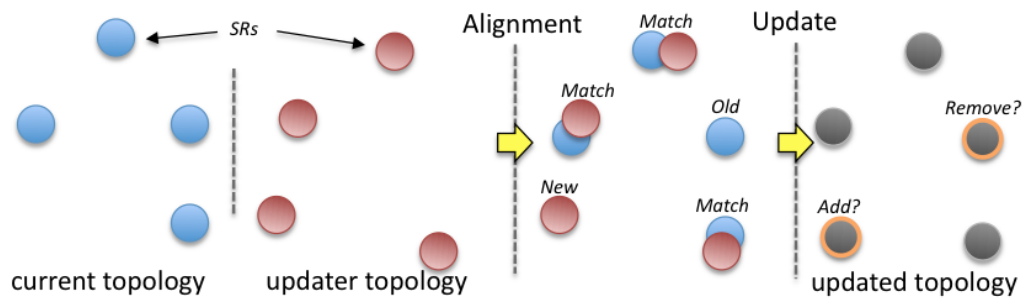


Figure 4.5: Example of the topology updating process

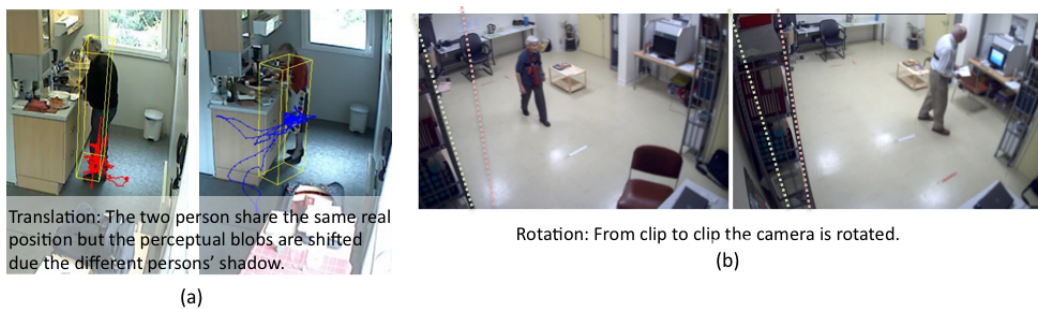


Figure 4.6: Scene alignment problems. a) The blob of the person is shifted due shadows produced by illumination changes, this produces a translation of the SRs of the persons' topology. b) The camera is slightly moved producing a rotation of the persons' topology.

Another alignment problem is the resolution (*level*) of the topologies. It is in our interest to be able to operate with topologies at different resolutions. This is, to update a "reference" topology at  $level_n$  with an "updater" topology at  $level_m$  where ( $n \neq m$ ). The multi-resolution updating has the benefit of adding *SRs* of the "updater" topology which have not been observed in the dataset used for learning the "reference" topology.

To overcome the alignment and resolution problems the usage of an explicit-alignment<sup>2</sup> based algorithm is required. In particular the algorithm needs to be robust to rotation and translation situations. We adapt a graph based algorithm [Chikkerur 2005] capable of finding correspondences in graphs of different dimensions. The algorithm was originally designed for fingerprints matching purposes. In the following subsections we explain the main stages of the adapted algorithm. To ease the reading, we consider a topology as a connected graph where each *SR* is a node.

#### 4.4.1.1 Alignment algorithm concepts

The local neighborhood of a *SR* is a finite set of other *SRs* that keeps some relations with the reference *SR*. Therefore, the local neighborhood of an *SR*  $m$  are the  $K$  nearest *SRs* of the same topology. The relations can be global or local: global structures refers to the relations between all *SRs* belonging to a topology; local structure refers to the relations between a reference *SR* and its local neighborhood.

A representation called K-plet is used to characterize the local neighborhood of a *SR*. A K-plet is invariant under translation and rotation. Also a directed graph  $G(V, E)$  is defined to represent the *SR* local relations in a formal manner. The local neighborhoods are matched using a dynamic programming based algorithm. The consolidation of the local matches is done by Coupled Breadth First Search algorithm which propagates the local matches simultaneously in both topologies. This is very similar to the way a human expert matches image features where each successive match is considered in the immediate neighborhood of previously matched feature. An important advantage of this algorithm is that, no explicit alignment of the *SR* sets is required at any stage of the matching process. Furthermore, the algorithm provides a very generic but formal framework of consolidating the local matches during the alignment. In the following section, we describe the following three stages of the alignment algorithm.

- Representation
- Local Matching

---

<sup>2</sup>In this approach, the optimal correspondence is obtained after explicitly aligning one or more corresponding *SRs*.

- Consolidation

#### 4.4.1.2 K-Plet Representation

We build a representation to capture the local structural information in the topologies called the K-plets. The K-plet representation is invariant under translation and rotation since it defines its own local coordinate system. The K-plet consists of a central scene region ( $SR$ )  $m_i$  and  $K$  other scene regions  $m_1, m_2 \dots m_K$  chosen from its local neighborhood. Each neighborhood  $SR$  is defined in terms of its local radial coordinates  $(\phi_{ij}, r_{ij})$  where  $r_{ij}$  represents the Euclidean distance between the  $SRs$   $m_i$  and  $m_j$ .  $\phi_{ij}$  is the relative orientation of  $m_j$  w.r.t the central  $SR$   $m_i$ . A K-plet example is displayed in the figure 4.7. The K-plet does not specify how the  $K$  neighbors are chosen. In the next section we outline the technique to achieve this.

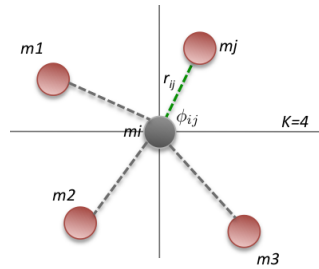


Figure 4.7: K-plet example with  $K=4$

#### 4.4.1.3 Graphical View

The local structural relationship of the K-plet is encoded formally in the form of an adjacency graph  $G(V, E)$  (see fig. 4.8). Each  $SR$  is represented by a vertex  $v$  and each neighboring  $SR$  is represented by a directed edge  $(u, v)$ . Each vertex  $u$  is colored with attributes  $(x_u, y_u, z_u)$  that represents the coordinate of a neighboring  $SR$ . Each directed edge  $(u, v)$  is labelled with the corresponding K-plet coordinates  $(\phi_{uv}, r_{uv})$ .

For alignment, the graph needs to maintain a high connectivity of all  $SRs$ . This can be achieved by selecting a particular set of neighbors at the K-plet building stage. To choose the neighbors of a  $SR$   $m_i$  we divide the space in 4 sectors and we select the nearest  $SR$   $m_j$  from each sector. The selection is performed iteratively, at each iteration a sector is visited and a single neighbor is selected (see fig. 4.9). The termination criteria is achieved when we obtain  $K$  neighbors. In the case that no  $SR$  remains in the sector, the iteration is skipped.

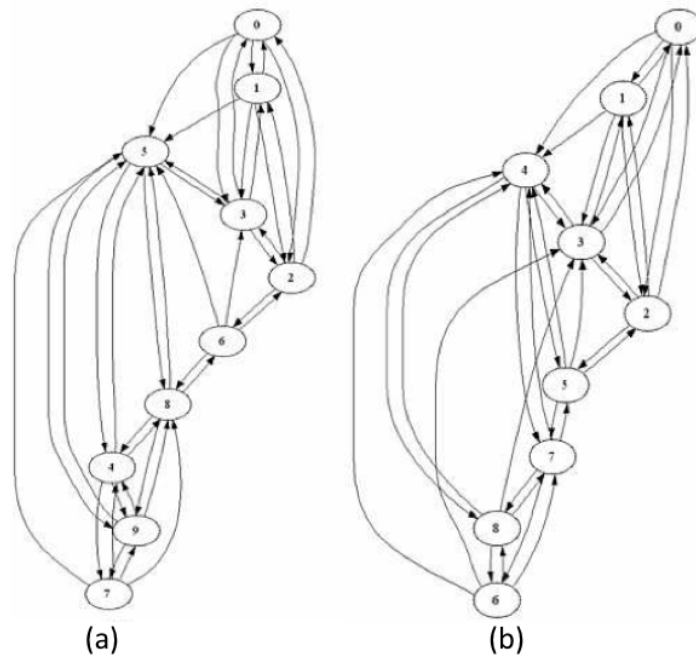


Figure 4.8: Illustration of the adjacency graph based on the K-plet representation. Each  $SR$  is connected to its 4 nearest neighbors. The graphs are built for topologies of different levels: 10 (a) and 9 (b). It is to be noted that the structure of the graphs are different due to an extra unmatched  $SR$  in the topology (a)

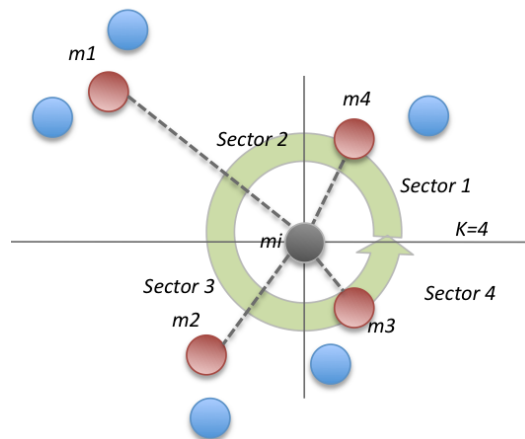


Figure 4.9: Illustration of the selection of nearest neighbors for the K-plet construction. Iteratively each sector is visited (anti-clockwise) selecting the nearest  $SR$  ( $m_j$ ) to the reference  $SR$  until  $K=4$ . The blue dots represent other  $SR$ s.

#### 4.4.1.4 Local Matching

The matching algorithm is based on matching a local neighborhood (K-plet) and propagating this match to other K-plets in the neighborhood. The accuracy of the algorithm therefore depends critically on how this local matching is performed. It is particularly important to match all neighbors simultaneously since a greedy approach of matching the closest or the best matching neighbor is sub-optimal. We convert the unordered neighbors of each K-plet into an ordered sequence by arranging them in the increasing order of the radial distance  $r_{ij}$ . The problem now is reduced to matching two ordered sequences  $S = \langle s_1, s_2, \dots, s_M \rangle$  and  $T = \langle t_1, t_2, \dots, t_N \rangle$ . Note that the sequences  $S$  and  $T$  need not necessarily be of the same length. However, in our case it is usually the case since we choose each K-plet to have a fixed number of neighbors. We use a dynamic programming approach based on string alignment algorithm. Formally, the problem of string alignment can be stated as follows: Given two strings or sequences  $S$  and  $T$ , the problem is to determine two auxiliary strings  $S'$  and  $T'$  such that

$S'$  is derived by inserting gaps ( \_ ) en  $S$

$T'$  is derived by inserting gaps ( \_ ) en  $T$

where

$|S'| = |T'|$  and  $\text{cost} \sum_{i=1}^{|S'|} \sigma(s'_i, t'_i)$  is maximized.

For example, the alignment of the sequences  $S = \{acbdb\}$ ,  $T = \{cadbd\}$  is defined as:

$$\begin{aligned} S' &= ac\_ \_ bcd b \\ T' &= \_ cad b \_ d \_ \end{aligned}$$

A trivial solution would be to list all possible sequences  $S'$  and  $T'$  and select the pair with the least/most alignment cost. However, this would require exponential time. Instead we can solve this using dynamic programming in  $O(MN)$  time as follows. We define  $D[i, j] = (i \in \{0, 1, \dots, M\}, j \in \{0, 1, \dots, N\})$  as the cost of aligning substrings  $S(1..i)$  and  $T(1..j)$ . The cost of aligning  $S$  and  $T$  is therefore given by  $D[M, N]$ . Dynamic programming uses a recurrence relation between  $D[i, j]$  and already computed values to reduce the run-time substantially. It is assumed of course that  $D[k, l]$  is optimal and given that the previous sub-problems have been optimally defined, we can match  $s_i$  and  $t_j$  in three ways

- the elements  $s[i]$  and  $t[j]$  match with cost  $\sigma(s[i], t[j])$ .
- a gap is inserted in  $t$  ( $s[i]$  match with a gap) with cost  $\sigma(s[i], \_)$ .
- a gap is inserted in  $s$  ( $t[j]$  match with a gap) with cost  $\sigma(\_, t[j])$ .

$$D[i, j] = \max \left\{ \begin{array}{l} D[i-1, j-1] + \sigma(s[j], t[i]) \\ D[i-1, j] + \sigma(s[j], \_) \\ D[i, j-1] + \sigma(\_, t[j]) \end{array} \right\}$$

The initial conditions are given as follows.

$$\begin{aligned} D[0, 0] &= 0 \\ D[0, j] &= D[0, j-1] + \sigma(\_, t[j]) \\ D[i, 0] &= D[i-1, 0] + \sigma(s[j], \_) \end{aligned}$$

While this process produces the cost of aligning the two sequences, determining the subsequence takes additional steps of keeping track of previous choices in each step of the process. After the  $D[M, N]$  has been determined, the aligned sequence can be generated in  $O(|S|)$  time.

#### 4.4.1.5 Consolidation - Coupled Breadth First Search (CBFS)

The third stage and perhaps the most important aspect of the matching algorithm corresponds to a formal approach for consolidating all the local matches between the two topologies without requiring explicit alignment. An algorithm called Coupled BFS algorithm (CBFS) is used for this purpose. CBFS is a modification of the regular breadth first algorithm except for two special modifications.

- The graph traversal occurs in two directed graphs  $G$  and  $H$  corresponding to the "reference" and "updater" topologies simultaneously.
- While the regular BFS algorithm visits each vertex  $v$  in the adjacency list of neighbors, CBFS visits only the the vertices  $v_G \in V$  and  $v_H \in H$  such that  $v_G$  and  $v_H$  are locally matched vertices.

An overview of CBFS is depicted as follows.

Inputs: Graphs  $G(V, E)$  and  $H(V, E)$  from the "reference" and "updater" topologies.

i: source node in graph  $G$

j: source node in graph  $H$

Outputs: Number of vertex that can be matched from the given sources.

1. Let  $G(V, E)$  and  $H(V, E)$  represent the graphs corresponding to the two topologies.
2. Let  $G_C$  and  $H_C$  represent a FIFO queue.
3. Let  $M$  represents a set of matched vertex pairs  $\langle g, h \rangle$ .
- 4 Initialize
  - a. For each vertex  $g$  in  $G(V, E)$  and  $h$  in  $H(V, E)$ 
    - i.  $color[g] = WHITE$  // unvisited node
    - ii.  $color[h] = WHITE$
  - b.  $color[i] = GRAY$
  - c.  $color[j] = GRAY$

- ```

d. M = M + <g[i],h[j]>
e. ENQUEUE(GC,g[i])
f. ENQUEUE(HC,h[j])
5. While (GC is not empty and HQ is not empty)
  a. gu= DEQUEUE(GC)
  b. hu= DEQUEUE(HC)
  c. Find matching neighbors of gu and hu using dynamic programming
  d. For each matching neighbors gv (of gu) and hv (of hu)
  e. If (color[gv]== WHITE and color[hv] == WHITE)
    i. M = M + <gv,hv>
    ii. ENQUEUE(GC,gv)
    iii. ENQUEUE(HQ,hv)
6. Return M (the size of M gives the matching count)

```

#### 4.4.1.6 Global Matching

It is to be noted that the CBFS algorithm requires us to specify two vertices as the source nodes from which to begin the traversal. Since the point correspondences are not known a priori, we execute the CBFS algorithm for all possible correspondence pairs  $(g[i], h[j])$ . We finally consider the initialization pair which returns the maximum number of matched nodes to compute a global matching score. The global score is calculated using

$$s = \frac{m^2}{M_R M_T} \quad (4.8)$$

Here  $m$  represents the number of matched *SRs* and  $M_R$  and  $M_T$  represent the total number of *SRs* in the "reference" and "updater" topologies respectively. We consider that the best topology alignment returns the  $arg_{max}(s)$  of all possible initializations in eq. 4.8.

#### 4.4.2 Updating algorithm

The updating procedure is a set of heuristic decisions to operate with two aligned topologies. The operations can produce a new "updated" topology.

Assuming that the alignment is performed, the list ( $M$ ) of matching pairs of *SRs* can be obtained from the consolidation algorithm (see sec. 4.4.1.5). Also, two lists of not matching *SRs* can be extracted. The first list ( $R$ ) contains the not matching *SRs* belonging to the "reference" topology. The second list ( $U$ ) contains the *SRs* of the "updater" topology.

The updating is performed in an incremental fashion (none *SR* is removed from the "reference" to obtain the "updated" topology). The decision of only increasing is due to observation of different datasets. We observe that rarely used regions are characteristics of particular activities. The updating is performed by the following



stages.

1. This step smooths small scene region variations.  
 For all pair  $(SR_i, SR'_j) \in M$ ;  
 A) estimate the average  $SR$ ;  
 B) append it to the "updated" topology.
2. Add  $R$  to the "updated" topology.
3. For each  $SR'_q \in U$ ;  
 A) rename it  $SR_{n+i}$ , where  $i \in 1 \dots (U)_{length}$  and  $n$  is  $argmax_i SR_i \in \text{"updated"}$ ;  
 B) append it to the "updated".  
 Keeping the reference  $SRs$  labels is possible to use old activity models (built with the reference topology) in the "updated" topology

The updating procedure is dependent on the final application and the updating frequency setup. It is easy to modify the updating procedure to satisfy particular user requirements.

#### 4.4.3 Statistical Analysis

In previous sections we describe the way a topology can explain spatially the scene. But in fact a topology could explain temporal properties of the scene as well. This type of explanation can be achieved by performing intra and inter statistical analysis of the scene regions. The range of analysis is wide and depends on the user interest. A description of "which regions are most frequently visited" and "the average time elapsed of an object in a region" are some examples of the analysis that can be done.

In this work we cannot reproduce all scene analysis, but we consider important the evaluation of an inter  $SRs$  property. The property could be described as "the amount of usage of a region". We consider the situation when a region is occupied for longer or shorter times is important since it can describe salient parts of a video.

Formally, a topology could be seen as a Mixture of Gaussians:  $\langle SR_1, \dots, SR_n \rangle$  where the "the amount of usage of a region" is represented by the mixture parameter ( $\pi$ ) of each  $SR_k$ . The parameter  $\pi$  of  $SR_k$  is computed as:

$$SR_k.\pi = \frac{N_k}{\sum_{i=1}^n N_i} \quad (4.9)$$

where  $N_k$  is the amount of points in the set  $\{x_1, \dots, x_n\}$  defined in eq. 4.5.

An example of the analysis that can be made using the mixture parameter is displayed in the figure 4.10. The figure display two topologies (a) and (b), computed

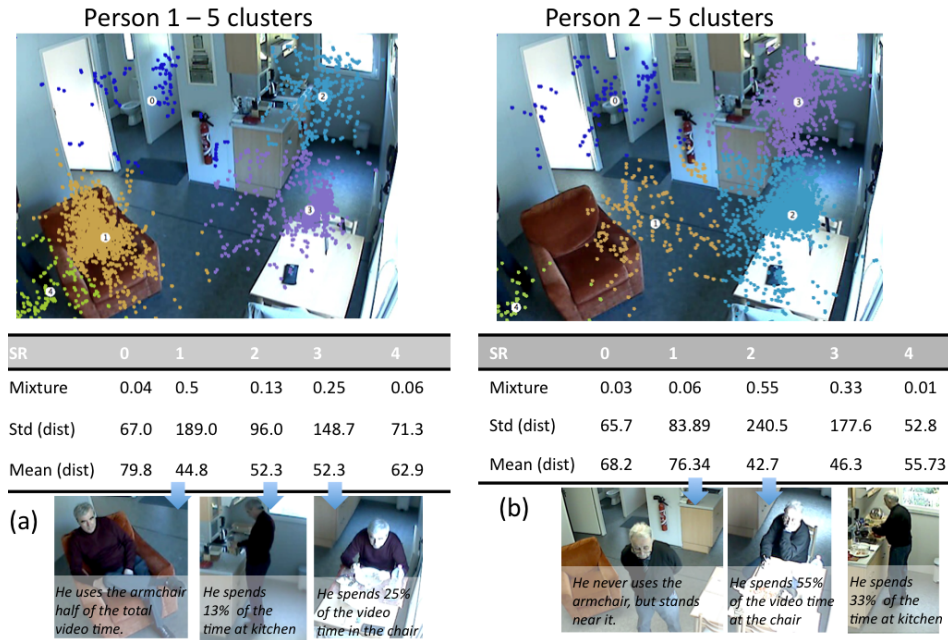


Figure 4.10: Topologies computed individually of person 1 and 2. Each person performs activity during 4 hours in the GERHOME dataset. The tables display values of spatial and temporal properties of the scene regions. A brief analysis is performed for each person of the usage of the discovered regions.

for individual persons (1 and 2). Each person performs activities during 4 hours in the GERHOME apartment. With the images (a) and (b) a table of values is presented with three values. 1) Mixture is the  $\pi$  parameter described above; 2) *Std (dist)* is the  $\sigma$  described in the equation (4.6) which represents the dispersion of the data in a scene region. 3) *Mean (dist)* aims at representing the average radius of the *SRs*. Below the tables, it is displayed a brief description of the analysis that can be performed about the usage of each computed *SR*.

## 4.5 The Scene Model, a Multi-Resolution representation

A topology can describe spatial properties of a scene in a single resolution. The problem is that the semantical explanations of the scene regions has different degrees of abstraction (i.e. regions and sub-regions). We propose to represent scene model as a vector of topologies of different resolutions. We propose to calculate 3 topologies for building a model, these topologies aims at describing high, medium and low semantical abstraction degrees. Generally, we use topologies with 5, 10, and 15 *SRs* because they can represent most of the interesting regions for long term activities

in the experimental indoor scenes.

$$SceneModel = \langle Topology_{level\ high}, Topology_{level\ medium}, Topology_{level\ low} \rangle \quad (4.10)$$

#### 4.5.1 Selection of the most relevant abstraction level

We would like to describe how the learnt  $SRs$  are automatically ordered by relevance in the different abstraction levels. The mechanism can be seen through an example.

**First**, it is necessary to observe the  $SR_1$  of the images (a) and (b) of Fig. 4.10. It can be noticed that while (a) describes the "armchair", (b) describes a region in the apartment "corridor".

**Second**, concerning the topology of level 5 displayed in Fig. 4.11, the topology is computed using more data. The "corridor" region disappears (red circle) while the "armchair" region does not.

**Third**, it is interesting to observe that the "corridor" region not only reappears in a topology of level 10 (see Fig. 4.11), but has also similar parametric values as the one described in (b) of Fig. 4.10.

The above situation is observed in most of the cases, and supports the multi-resolution scene model since it expresses that the regions are not lost, but shifted from level to level accordingly to their relevance.

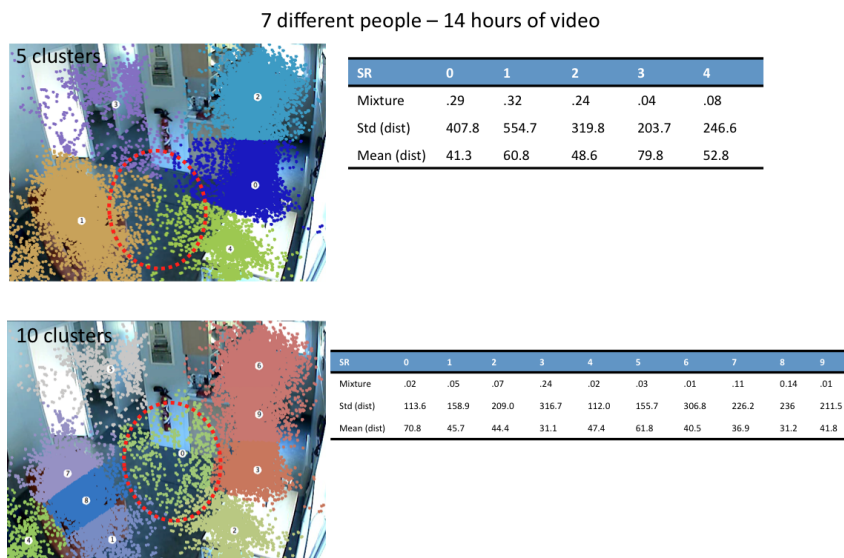


Figure 4.11: Examples of topologies of level 5 and 10 of the GERHOME dataset. The topologies are computed using data of 7 different people, performing activities in an apartment during 4 hours each.

# Primitive Events

---

## Contents

---

|            |                                                            |           |
|------------|------------------------------------------------------------|-----------|
| <b>5.1</b> | <b>Problem Definition</b>                                  | <b>75</b> |
| <b>5.2</b> | <b>Primitive Events</b>                                    | <b>76</b> |
| <b>5.3</b> | <b>Building a Primitive Event</b>                          | <b>76</b> |
| <b>5.4</b> | <b>The Duration and <math>ID_{Chunk}</math> attributes</b> | <b>77</b> |
| <b>5.5</b> | <b>The Type attribute</b>                                  | <b>77</b> |
| 5.5.1      | Type representation                                        | 78        |
| 5.5.2      | Type computation                                           | 78        |
| 5.5.3      | Type Example                                               | 78        |
| 5.5.4      | Type Ambiguity                                             | 79        |
| 5.5.5      | Type Quantity                                              | 79        |
| 5.5.6      | Discussion                                                 | 79        |
| <b>5.6</b> | <b>The Local Dynamics Attribute</b>                        | <b>80</b> |
| 5.6.1      | Local Dynamics representation                              | 81        |
| 5.6.2      | Local Dynamics Abstraction                                 | 81        |
| <b>5.7</b> | <b>The additional Logic attribute</b>                      | <b>87</b> |
| <b>5.8</b> | <b>Primitive events sequence</b>                           | <b>87</b> |

---

In this chapter we address the problem of uncertainty due to the *semantic gap* described in 1.1.1 by proposing an intermediate representation that can be used as construction blocks for learning activity models.

## 5.1 Problem Definition

Most of the unsupervised literature approaches bridge the *semantic gap* in a single step. This is achieved by building activity models directly from perceptual features (*feature-based models*)<sup>1</sup>. Several of these approaches can recognize occurrences of actions of interest in a video sequence but their models are rigid and difficult to understand by humans. The main problems are described by the following items.

---

<sup>1</sup>In a similar way as Histogram of Oriented Gradients (HOGs) are learned for human detection purposes, and HAAR for face detection.

- The models are hard to update and modify manually.
- The models are uniquely designed to handle a specific type of training data. When the input data slightly changes, new models have to be trained for the same activity.
- The models are complex for the human interpretation, making hard to explain the system failures (i.e. the reasons of an activity miss-recognition). Consequently, it is hard to measure the models' reliability.

## 5.2 Primitive Events

Our view, is that the abstraction of perceptual features to conceptual information does not occur in a single step, but is gradual. We address the semantic gap building an intermediate representation. We propose such representation and we name it **Primitive Event** (*PE*). The *PE* representation aims at satisfying the following:

- It acts as the link between the perceptual features and the semantical information. This is, a representation that is meaningful for humans and is capable of describing complex occurrences of an activity of interest in the video. The system failures can be explained through the representation.
- It is composed of modular independent information units. It enables the construction of flexible activity models and applications on top. The models are capable of being updated/modified by parts when new primitive events are learned.
- It aims at describing the first possible abstraction from perceptual features to conceptual information (**Primitive** abstractions).
- It describes the interesting **events** during a period of time. The definition of an event is discussed in [Lavee 2009].

## 5.3 Building a Primitive Event

A Primitive Event is built from a single Perceptual Feature Chunk (*PFC*) described in chapter 3. This way, a *PE* represents an event in the video, that takes place in a small period of time.

A *PE* is composed of **attributes**. The attributes are calculated by abstracting the perceptual features of a *PFC*. Each attribute has a particular purpose and is independent from the others. The attributes are calculated automatically by dedicated abstraction procedures.

The abstraction procedures add semantical interpretation to the perceptual features by using contextual information. The contextual information can be

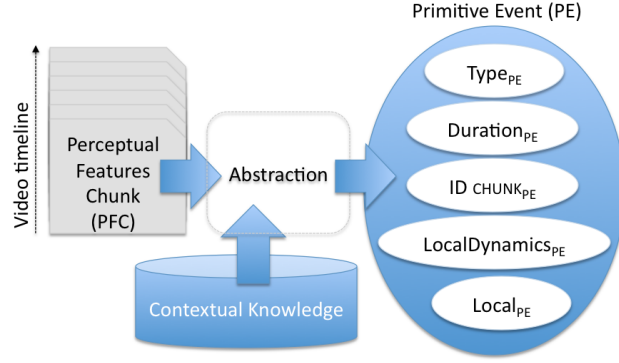


Figure 5.1: Example of the Primitive Event building sequence.

learned or added as human knowledge of the features.

The image 5.1 illustrates a diagram of the information flow for building a Primitive Event. A  $PE$  can be composed by several attributes. We define a set of attributes that is useful for general activity understanding purposes:  $ID_{Chunk}$ ;  $Duration$ ;  $Type$ ;  $LocalDynamics$ ;  $Logic$ . In the following sections we explain each of the attributes.

## 5.4 The Duration and $ID_{Chunk}$ attributes

The  $Duration_{PE}$  and the  $ID_{Chunk}$  are directly inferred from the  $PFC$ .

The duration is the amount of video seconds covered by the  $PFC$ :

$$Duration_{PE} = \frac{EndFrame_{PFC} - StartFrame_{PFC}}{fps}$$

where  $fps$  is the video frame rate<sup>2</sup>. The  $Chunk_{ID}$  attribute is an identification key to link the  $PE$  with the video timeline.

$$ID_{Chunk_{PE}} = (Video_{PFC}, StartFrame_{PFC}, EndFrame_{PFC})$$

## 5.5 The Type attribute

The **Type** attribute describes the global movement of an agent (or tracked object) over the scene. The abstraction is achieved by the classification of perceptual features into meaningful agent transitions over the scene. The transitions are obtained by fusing the  $PFC$  global tracklets (perceptual feature) and a learned *Topology* (contextual knowledge). The abstraction accomplishes the following:

<sup>2</sup>It denotes the amount of frames per second (fps) that a video is encoded.

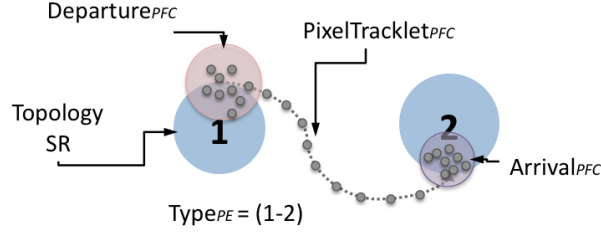


Figure 5.2: Example of the computation of a  $Type_{PE}$  (1-2) assuming that the  $SR_1$  and  $SR_2$  are the closest regions to the  $Departure_{PFC}$  and  $Arrival_{PFC}$  mean values.

1. It filters noisy agent trajectories by normalizing them to a learnt global motion.
2. It adds a semantical interpretation by the classification of the motion into human understandable transitions.

### 5.5.1 Type representation

The  $Type_{PE}$  is represented as a couple and a direction operator ( $Start \rightarrow End$ ).

$$Type_{PE} = (Start \rightarrow End)$$

### 5.5.2 Type computation

Let a  $PFC$  and a topology  $T$  where  $SR_i$  is the  $i^{th}$  *SceneRegion*  $\in T$ , then  $Start$  and  $End$  are the labels of the nearest  $SR_i$  to  $Departure|Arrival_{PFC}$  respectively. The nearest  $SR$  is computed by a distance measurement ( $\nabla$ ); which compares the distributions of the  $SRs$  and the  $Departure_{PFC}|Arrival_{PFC}$ :

$$Start = \arg \min_i (\nabla (SR_i, PFC_{Departure}))$$

$$End = \arg \min_i (\nabla (SR_i, PFC_{Arrival}))$$

where  $SR_i \sim (\mu_i, \sigma_i)$  and  $Departure|Arrival_{PFC} \sim (\mu', \sigma')$ . The Euclidean distance is adopted as the default measurement. We have also compared Bhattacharya and Correlation measurements in several experiments. Nevertheless, we did not find significant differences among them.

### 5.5.3 Type Example

The figure 5.3 displays an example of two  $Type_{PE}$  computed in an apartment scene. In (a) the initial position of the person and 3 Scene Regions of a learnt topology are displayed. In (b) a  $Type_{PE}$  is built, abstracting the translation of the person from the table to the bottom of the kitchen. In (c) the person moves towards a different part of the kitchen to interact with an object.



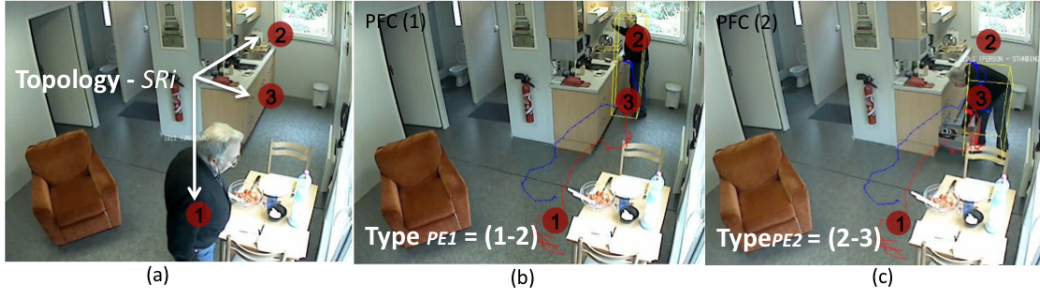


Figure 5.3: Examples of two  $Type_{PE}$  computed from consecutive  $PFCs$ . In (a) are displayed the initial position of the person and 3 learnt  $SRs$  of a  $Topology$ . In (b) and in (c) the computed  $Type_{PE}$  are displayed.

#### 5.5.4 Type Ambiguity

An ambiguity occurs when a  $PFC_{Departure}$  or  $PFC_{Arrival}$  is significantly close to more than one  $SR$ . We use the  $SR$  mixture proportions to compute the  $Start$  and  $End$  labels. To address to the problem we propose two strategies as follows:

1. **Push to strongest:** Considers that the best association is the  $SR$  with higher density of points.

$$Start|End = \arg \max_i (\nabla(SR_i, Departure|Arrival_{PFC}) * \pi(SR_i))$$

where  $\pi$  is the number of elements associated to the  $SR$ .

2. **Push to weakest:** Is analogous.

#### 5.5.5 Type Quantity

The number of different  $Type_{PE}$  is fixed. It can be deduced from the number of possible transitions between the  $SRs$  of an input topology  $T = \langle SR_1 \dots SR_i \rangle$ .

$$\# Type = (\arg \max_i (SR_i))^2$$

The image 5.4 displays an example of the possible  $Type_{PE}$  using a  $Topology$  of 3 scene regions (i.e. level 3).

#### 5.5.6 Discussion

At this point, it is interesting to compare the benefits and drawbacks of the representation of the agent global trajectory in a  $PFC$  and of the topology representation. The comparison allows to understand conceptually the benefits of combining both type of information to build a  $Type_{PE}$ .

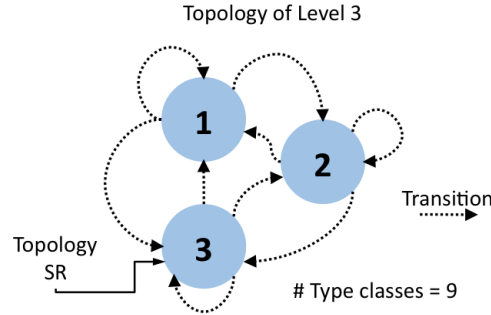


Figure 5.4: Example of the number of unique  $Type_{PE}$  using a *Topology* of level 3 to build the primitive events.

- One benefit of the vector representation of the trajectories of a PFC is its generality and compact representation of a snapshot of the event of interest in the video sequence. The drawback of this representation: it does not allow for straightforward semantic interpretation. In other words, the abstraction of the video input is meaningless for a human without being processed first by an appropriate model.
- The *Topology* representation has the benefit of describing the semantical information learned in an unsupervised manner (i.e. regions of interaction with scene objects). The drawback: the semantical information is a macro model of the context, which by itself does not allow the interpretation of interesting events a video snapshot.

From the above observation it can be deduced that the object-based (agent) global trajectories and a topology contain complementary information. The perceptual information characterizes the events in a video snapshot (low-level information) and a *Topology* characterizes semantic descriptions (high-level information). The fusion of the two information into a  $Type_{PE}$  can describe the global motion of the object in a semantical way (i.e. the person moves from A to B).

## 5.6 The Local Dynamics Attribute

In section 5.5 we define the  $Type_{PE}$  as an attribute that describes the global motion of an object (or agent) over the scene (e.g. "person stays in the armchair"). The limitation of the  $Type_{PE}$  is that due to its global nature in some cases is too coarse to describe finer activities (i.e. "person reading in the armchair"). To work around such limitation we propose to enrich a  $PE$  with finer information by adding the object local dynamics. The local dynamics is the motion described by an object parts. For example, when the object is a person, the hands, arm, torso, etc. are

the object parts.

The  $LocalDynamics_{PE}$  attribute aims at describing the **principal** dynamics of the parts of an object (sub-objects) in a  $PFC$ . We represent these dynamics with point tracklets.

Computing the tracklets of the motion of the parts is not straight forward. The problem relies in the limitation of the current state-of-the-art methods to automatically classify and track sub-objects (i.e. long training; scene dependence; changes of appearance; shape morphisms). To bypass such limitations, we propose to avoid the object parts detection and to work directly at the pixel level.

With the current methods it is possible to compute accurate pixel-based tracklets during short periods of time. The tracklets obtained from the moving pixels of an object can be abstracted revealing the underlying object part dynamics.

In particular the  $PixelTracklets_{PFC}$  is a dense set of pixel-based tracklets describing the moving pixels of the object parts. We propose to cluster the  $PixelTracklets_{PFC}$  to compute the main local dynamics.

The figure 5.5 illustrates five examples of real  $PixelTracklets_{PFC}$  (pink) and of the computed  $LocalDynamics_{PE}$  (green). It can be noticed that the  $LocalDynamics_{PE}$  can characterize different local motions while the agent global motion (i.e. position) is the same.

### 5.6.1 Local Dynamics representation

The  $PE_{LocalDynamics}$  is represented as a bag of pixel tracklets:

$$LocalDynamics_{PE} = \{Dyn_1, \dots, Dyn_n\}$$

where  $Dyn_i$  is a vector of points

$$Dyn_i = \langle p_1, \dots, p_k \rangle$$

where  $k = StartFrame_{PFC} - EndFrame_{PFC}$ .  $Dyn_i$  has the same length as its associated  $PFC$ .

### 5.6.2 Local Dynamics Abstraction

The abstraction of  $PixelTracklets_{PFC}$  into  $LocalDynamics_{PE}$  is achieved by clustering.

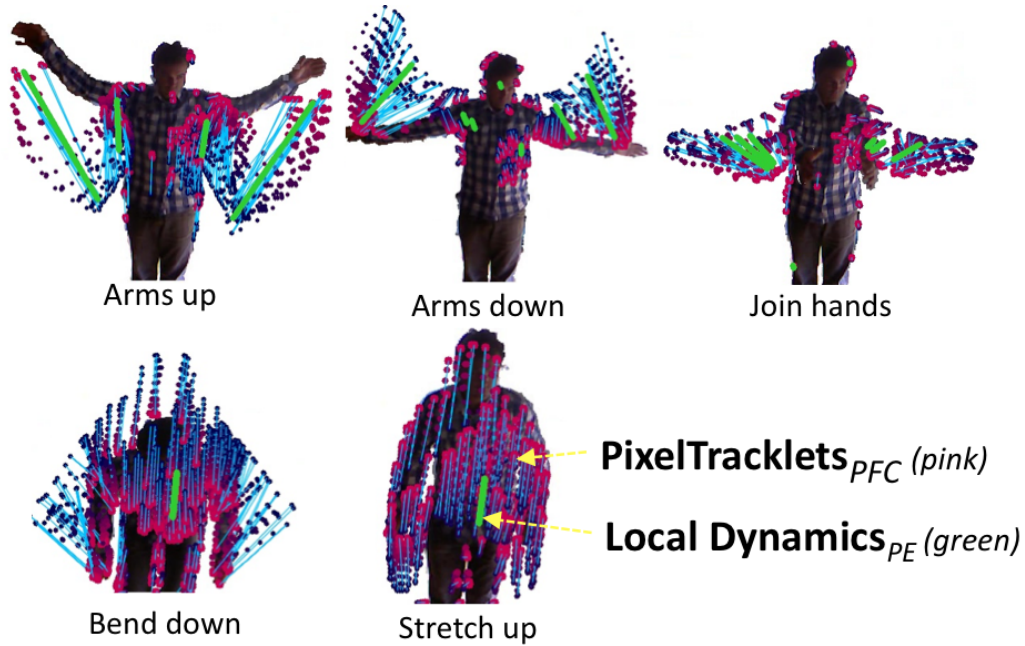


Figure 5.5: Example of the abstraction of  $PixelTracklets_{PFC}$  (pink) into  $LocalDynamics_{PE}$  (green). Each,  $LocalDynamics_{PE}$  is displayed as a straight line corresponding to the start and end points of an abstracted tracklet (blue line).

### 5.6.2.1 Feature Space

Independently from the adopted clustering technique, the selection of a feature space structures the meaning of the resulting clusters. In general, the usage of a multi-feature space produces clusters that are difficult to understand by humans. The multi-feature approaches [Faisal I Bashir 2007, Li 2006b, Pusiol 2008] has to deal with complex normalization, optimization, and convergence problems. The usage of multiple features is a convenient technique when the goal is to find outlier clusters (e.g. unfrequent trajectories). In our case, the goal is the contrary. We aim at finding a set of dense clusters and filter outlier clusters generally produced by noisy data (e.g. points confused with the background). We use an one dimensional feature space defined by the set of coordinates of the  $PixelTracklets_{PFC}$ ' data points.

### 5.6.2.2 Clustering

To cluster the  $PixelTracklets_{PFC}$  we adapt the Mean Shift clustering algorithm.

Mean Shift represents a general non-parametric mode-seeking/clustering procedure. Mean Shift was first proposed by Fukunaga & Hosteler [Fukunaga 1975] and

extended by Comaniciu, Meer and Ramesh [Comaniciu 2002] to low-level vision problems. It is used by Anjum and Cavallaro [Anjum 2008] to cluster multi-feature object-based trajectories. We implement the approach proposed by Subbarao and Meer [Subbarao 2009] which extends the conventional Mean Shift to data points lying on Riemannian manifolds. The approach can handle multiple non-vector feature spaces; nevertheless we use the euclidean geometry (base case) to model our vectorial feature space.

The idea behind Mean Shift is to treat the points in a  $d$ -dimensional feature space as an empirical probability density function (*pdf*). The dense regions in the feature space correspond to the modes (local maxima) of the underlying distribution. For some data points in the feature space, a gradient ascendent procedure is performed iteratively. At each iteration the procedure shifts a kernel window until convergence. The iterative procedure converges when the stationary points are found. The stationary points are the modes or hilltops on the virtual terrain defined by the kernels. For a proof of convergence, see [Comaniciu 2002].

Here we briefly describe the derivation of mean-shift as in [Comaniciu 2002] adapted to the tracklets notation. Let a tracklet  $\mathbf{x}_i$ ,  $\mathbf{x}_i \in \mathbb{R}^{d * \text{tracklet length}}$ ,  $d = \text{point dimension}$ ,  $i = 1, \dots, n$  be  $n$  independent, identically distributed tracklets generated by an unknown distribution  $f$ . For a tracklet  $\mathbf{y}$  the *kernel density estimate*

$$\hat{f}(\mathbf{y}) = \frac{c_{k,h}}{n} \sum_{i=1}^n k\left(\frac{\|\mathbf{y} - \mathbf{x}_i\|^2}{h^2}\right) \quad (5.1)$$

based on a *profile function*  $k$  satisfying  $k(z) \geq 0$  for  $z \geq 0$ , is a nonparametric estimator of the density  $f(\mathbf{y})$  at  $\mathbf{y}$ . The value  $h$  (termed the *bandwidth* parameter) defines the radius of the kernel. The constant  $c_{k,h}$  is chosen to ensure that  $\hat{f}_k$  integrates to one (normalization). Defining  $g(\cdot) = -k'(\cdot)$  and taking the gradient of defined in the equation 5.1 we obtain:

$$\mathbf{m}_h(\mathbf{y}) = C \frac{\nabla \hat{f}_k(\mathbf{y})}{\hat{f}_g(\mathbf{y})} = \frac{\sum_{i=1}^n \mathbf{x}_i g(\|\mathbf{y} - \mathbf{x}_i\|^2/h^2)}{\sum_{i=1}^n g(\|\mathbf{y} - \mathbf{x}_i\|^2/h^2)} \quad (5.2)$$

where,  $C$  is a positive constant and  $\mathbf{m}_h(\mathbf{y})$  is the *mean shift vector*. The *mean shift vector* is a shifted tracklet towards the direction of the maximum increase of density. The expression (5.2) can show that the mean shift vector is proportional to a normalized density gradient estimate at the tracklet  $\mathbf{y}$  obtained with kernel  $k$ .

The mean shift iterative procedure for a given tracklet  $\mathbf{y}_j$  is as follows.

1. Compute the *mean shift vector*:  $\mathbf{m}_h(\mathbf{y}_j)$
2. Shift the density estimation window:  $\mathbf{y}_{j+1} = \mathbf{m}_h(\mathbf{y}_j) + \mathbf{y}_j$

3. Iterate steps 1. and 2. until convergence (discovery of the stationary points).

The step (2.) is a gradient ascent technique converging to a stationary point of the density. Similar points can be detected and removed, to obtain only the modes.

The figure (5.9) displays the computed  $LocalDynamics_{PE}$  (modes) using 2-dimensional<sup>1</sup> real data. The clustering is computed using all points of the  $PFC_{Tracklets}$ . The  $LocalDynamics_{PE}$  of fig. (5.9) are describing the local motion of hands, body rotations, and balance.

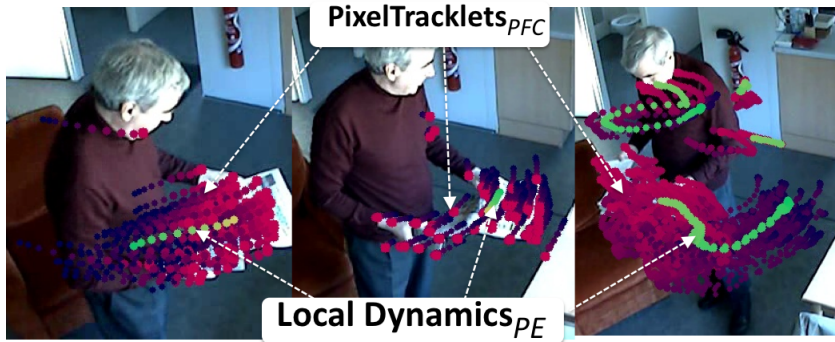


Figure 5.6: Example of the abstraction of  $PixelTracklets_{PFC}$  to  $PC_{LocalDynamics}$

### 5.6.2.3 Bandwidth

As with most of the so called *non – parametric* approaches there are parameters to be set. In the case of Mean Shift the value of the bandwidth (or kernel size) is unspecified. The choice of the kernel size plays an important role in Mean Shift clustering. Selecting small sizes will produce clusters with a single tracklet, while with large sizes produce a cluster with all tracklets grouped together.

Some efforts have been reported to locally vary the bandwidth. Singh et al. [Singh 2003] determine local bandwidths using Parzen windows to mimic local density. Wang et al. [Wang 2004] presents an anisotropic kernel in which the shape, scale, and orientation of the kernels adapt to the local structure of images. It is also possible to adapt the bandwidth locally for each datapoint of a trajectory with the distance to its  $k^{th}$  nearest neighbor (kNN). Most of these approaches improve the quality of the clusters structure for general clustering purposes, but they are not adapted to our particular interest. We are interested into producing a finer description of the local motion (more clusters) only when the amount of global motion is low. In the other case the local motion can become noisy and the  $Type_{PE}$  can describe accurately the agent global motion. We adapt the kernel's bandwidth in function to the object's global motion as follows.

<sup>1</sup>The  $x$  and  $y$  point coordinates.

$$h = \|PFC_{Departure} \cdot \mu - PFC_{Arrival} \cdot \mu\| * C \quad (5.3)$$

where  $C$  is a factorization constant.  $C$  is set depending on the camera visual field of view. Figure 5.7 displays an example of the clustering results using the defined adaptive bandwidth.

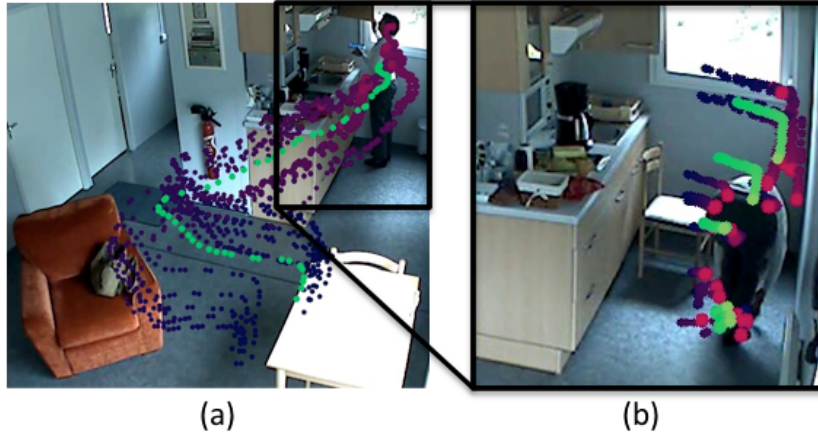


Figure 5.7: Example of the clustering results with the defined adaptive bandwidth. The green tracklets are the computed Mean Shift modes. In (a) the person walks to the kitchen; the long motion is captured by a single cluster (i.e. long green tracklet). In (b) the person moves laterally, bends and moves the hands; this complex motion is captured by 5 clusters (i.e. short green tracklets).

#### 5.6.2.4 Kernel selection

Other decision to be made is to choose the kernel shape. We have tried the Gaussian and Epanechnikov kernels. Both kernels have interesting optimization properties. Flashing and Tomasi [Flashing 2005] show that for the Epanechnikov kernels, the mean shift vector lies along the gradient and is in fact a Newton step. Recently, Carreira and Perpinan [Carreira-Perpinan 2007] show that for Gaussian kernels, the mean shift step is the same as Expectation-Maximization.

In practice, we do not find significant differences by changing kernels. Using both type of kernels the procedure works in real-time. We adopt as default the Gaussian kernel.

#### 5.6.2.5 Practical Clustering

The usage of complete tracklets for clustering, can capture the flow of the object local motion. Nevertheless, when the flows are complex it is hard to accurately measure

the tracklets inter-distance. We have proposed a compact structure [Pusiol 2008] which can produce accurate clusters in noisy situations. In practice, we use the start and end points of each tracklet for clustering. The resulting clusters can describe interesting motion (see image 5.8), reduce the computational time (about 90%), and filter out noise. The image 5.9 displays examples of the clustering approach under noisy situations. In (a) several tracklet points are confused when the person "stands up" the  $LocalDynamics_{PE}$  are correctly computed. In (b) we intentionally add noisy data to the  $PixelTracklets_{PFC}$ , and algorithm still makes a good approximation of the expected local motion. In general the algorithm converges to the main dynamics of the agent parts. In some cases (e.g. heavy scene light changes) the algorithm can fail, but in practice the failures are temporarily short and do not affect the whole system.

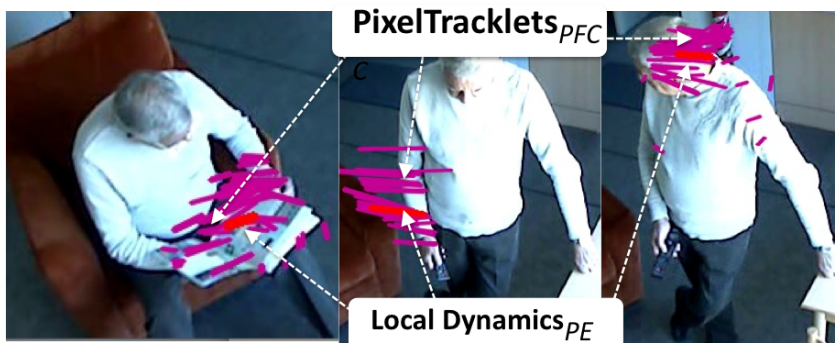


Figure 5.8: Example of the abstraction of  $PixelTracklets_{PFC}$  to  $LocalDynamics_{PE}$ . The clustering is performed using the first and last points of the  $PixelTracklets_{PFC}$ .

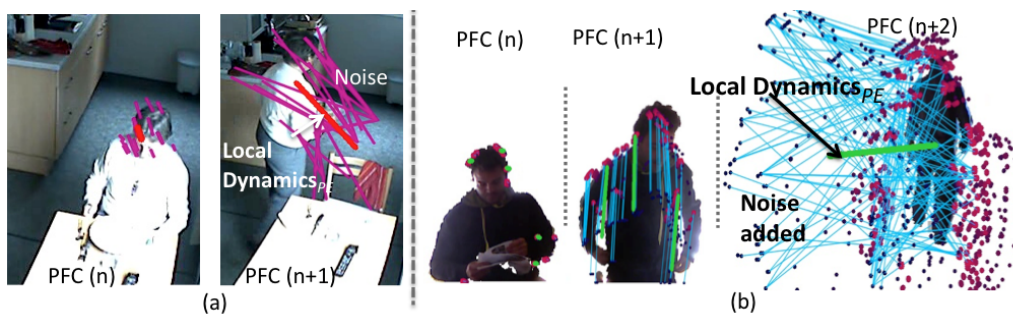


Figure 5.9: Example of the  $LocalDynamics$  computed under noisy conditions. (a) real data, in  $PFC(n+1)$  some tracklets are confused (i.e. first and last points) with the background. In (b) noise is added to the  $PFC(n+2)$ . The algorithm converges to the expected  $LocalDynamics$  in both cases.



### 5.6.2.6 Discussion

There is a wide range of clustering techniques that can be used to group the  $PixelTracklets_{PFC}$ . An interesting survey of clustering techniques can be found in [JAIN 2011]. We can compare some properties of Mean-Shift with K-Means which is a popular clustering algorithm. K-Means has the advantage of being simple, fast, and efficient. One of the most important difference is that K-means makes different broad assumptions.

- In K-means the number of clusters is already known. Mean Shift, being a non parametric algorithm, does not assume anything about number of clusters. The number of modes gives the number of clusters.
- In K-means the clusters are shaped spherically (or elliptically). Mean shift is based on density estimation. Therefore it can handle arbitrarily shaped clusters.
- K-means is very sensitive to initialization. A wrong initialization can delay convergence or some times even result in wrong clusters. Mean Shift is robust to initialization. In our case we run mean shift for each point from the feature space. This produces the same modes when using the same input data.
- K-means is sensitive to outliers while Mean Shift is not.

## 5.7 The additional Logic attribute

The  $Logic_{PE}$  holds a boolean value. The value is aims at describing some property of a video snapshot (e.g. "is the person facing the camera").

An usage of the attribute, is as a  $PFC$  noise filter. Using a noise evaluator of perceptual features is possible to enable or disable the  $PE$  (e.g. light change). Disabled  $PEs$  are not used as building blocks of an activity model.

Other usage, is to describe the presence of an event of interest in a frame. For example, image-based algorithms such as for face detection, can accurately find faces in some video frames. Nevertheless, due to rotations they cannot hold the face to build a trajectory over all frames. The  $Logic_{PE}$  can describe that a face is detected in the  $PFC$ . Such descriptor could be of particular importance to build an activity model.

## 5.8 Primitive events sequence

For each new  $PFCs$  in the video dataset, we compute a  $PE$  using a learnt topology  $T$ . The process produces a sequence of primitive events ( $PE_{seq}$ ). A primitive event

sequence is represented by a couple.

$$PE_{seq} = (\langle PE_1, \dots, PE_n \rangle, T)$$

At this point, it is important to analyze different primitive event sequences to explain the type of information obtained. The display of sequences of  $PE_{Local\ Dynamics}$  are used for refinement. The sequences can be complex to understand in long datasets, nevertheless in some occasions they reveal interesting information about the video activities.

We display an example of two sequences of  $PEs$  in figure 5.10. The sequences are presented as a chart representing the  $Type_{PE}$ ,  $Duration_{PE}$ ,  $ID_{Chunk_{PE}}$  attributes.

The sequences are obtained from 2 videos of 8 hours of length in total. Each video contains a person living in an apartment. The sequences of  $PEs$  are computed using a shared topology which is learned by the trajectories of the two involved persons. Using a shared topology it is possible to align the persons spatial location over the scene. The used topology is composed of 8 Scene Regions (level 8) and it is learned as it is described in the Chapter 4.

In figure 5.10, for each sequence, the  $PEs$  are ordered from left to right by their time of appearance in the video. Each color represents a  $Type_{PE}$ . The same color represents the same  $Type_{PE}$ . The graph vertical axis displays the  $PE_{Chunk_{ID}}$  as an instance number. The crescent shape of the sequence over the vertical axis helps to differentiate long sub-sequences of  $PEs$  of the same type -e.g. (a) (b)- and a single  $PE$  of a long duration -e.g. (i)-.

Also in figure 5.10, it can be noticed that similar sub-sequences are describing the same long term activity. For example:

- Long sub-sequences of  $PEs$  of the same type such as: (b) (d) (e) red color, are describing the activity "Eating". For the persons A and D, the sub-sequences of the same  $Type_{PE}$  are describing the same semantical activity.
- Short sub-sequences of particular  $PEs$  -blue color- describe the activity "Preparing meal", the same sub-sequences are found in both datasets. These sub-sequences describe the transitions between regions in the kitchen.
- A long duration  $PE$  such as: (i) and (j) for person D is describing a low-motion activity in a fixed place (e.g. "Sitting in the armchair").
- Long sub-sequences of  $PEs$  of the same type describe the activity "Sitting at the front of the table" (a). It occurs only in the dataset of person A. As it is expected, no sub-sequence of the same  $Type_{PE}$  as (a) is found in the  $PEs$  sequence of Person D, since he never performs the activity of (a).

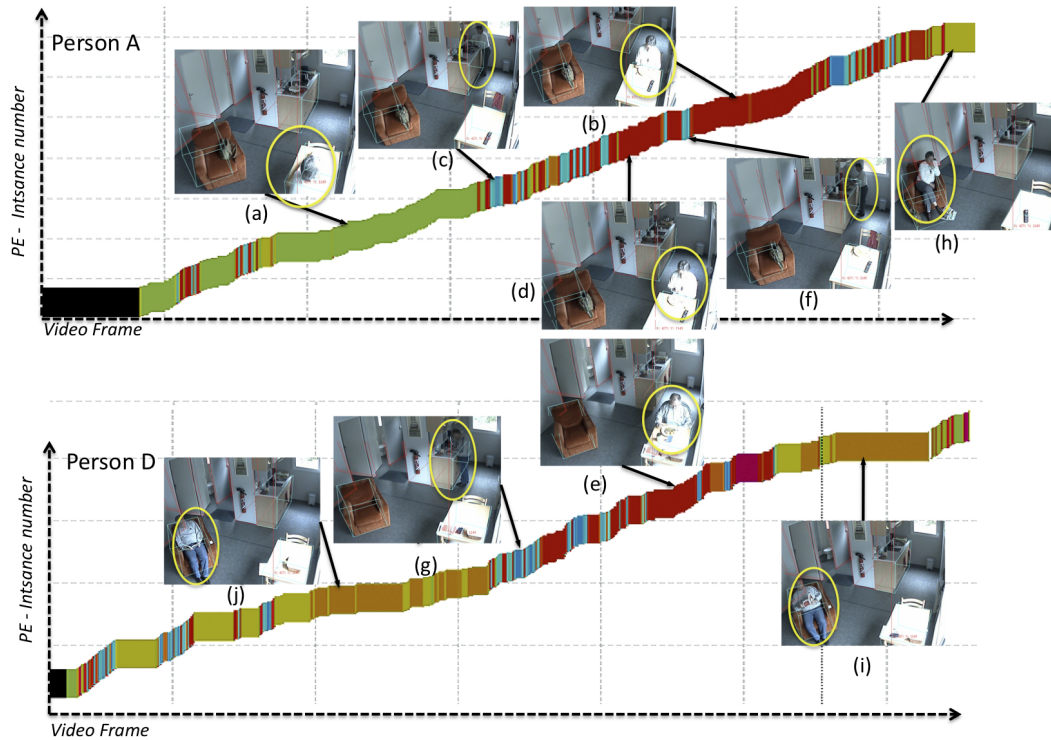


Figure 5.10: Example of two sequences of Primitive Events of two persons. Each sequence represents 4 hours of video. The global trajectory is obtained by the center of mass of the person. The  $PEs$  are the colored segments, where the same color represents the same  $Type_{PE}$ , and the segment width is the temporal length of the  $PE$ . From left to right the primitive events are ordered by time of appearance in the video. From bottom to top it is displayed the instance number of the  $PEs$  as they appear in the sequence. It can be noticed that similar colored sub-sequences of  $PEs$  represent similar activities in both videos.

A sequence of *PEs* is very informative about the underlying structures of the long-term activities appearing in the video. The structures are contained in sub-sequences of *PEs*. The sub-sequences describe particular patterns. These patterns can be recognized automatically. The recognition of such patterns lead to the automatic discovery of long term activities. The discovery of activities is the issue discussed in the next chapter.

# Activities: Discovery, Modeling and Recognition

---

As humans we interact by communication. The communication takes place at different abstraction layers. The layers can be seen as snapshots of reality at particular resolutions. For example, a good broadcaster is someone who can adapt his speech to the "conceptual world" of his audience. A clear communication between several people occurs when the involved peers are interacting at similar resolutions. Humans are constantly tuning their resolution to common spaces to achieve effective communications<sup>1</sup>.

The notion of resolution can also be found when we aim at describing activities semantically. For a given activity we are always capable of naming a sub-activity.

We want activity recognition systems which can adapt to different resolutions providing meaningful information on demand. To achieve such a goal, two items need to be satisfied: 1) The activity models need to capture the notion of multi-resolution layers which is an hierarchical structure of activities and sub-activities. This way the models can characterize multiple activities at the same instance of time. 2) The systems need to be able to navigate between the resolution layers to retrieve relevant information to the user, in similar way as humans do when there is a communication.

An inherent difficulty for the systems is to find automatically the period of time in which an interesting activity occurs (i.e. start and end of an activity). We call such a process: **activity discovery**.

Marking the start/end of an activity (discovery) is a hard task even for humans. For example, we let 3 persons to freely mark and label activities in a video (15 minutes of regular activities of a person in an office). We find out that the same label is found in less than 10% of the cases. And that the start/end of the matched activities diverged of 26.6%. The problem is again the resolution. In most of the cases, humans tend to pay attention to one resolution at a time. The switch between resolutions depends on changes that become interesting. For example, a person that is labeled as "*sitting*" suddenly moves an arm. The label becomes

---

<sup>1</sup>A professor addresses in different ways to preschool and university students.

*"moving arm"* forgetting that the person is still *"sitting"*.

The discovery procedure is performed at different resolution levels to find the start/end of an activity. The method is based on grouping patterns of semantically rich primitive events. The patterns representing activities and sub-activities are packaged in activity models for later use at a recognition stage. The activity recognition stage describes a procedure which aims at automatically detecting modeled activities in unseen videos.

Putting all together, this chapter addresses three parts of this work. First, we propose a method to automatically discover activities. Second, we propose methods to build multi-resolution models that uniquely characterize an activity by describing the relationships with its sub-activities. By labeling the models, semantical interpretation of the data can be achieved. Third, we propose a mechanism to recognize modeled activities in new videos.

## 6.1 Input: Primitive Event sequences

The sequences of primitive events are very descriptive about the activities occurring in a video. In each input sequence, activities are described by spatio-temporal sub-sequences of primitive events. Seeking and extracting meaningful sub-sequences is the process of discovering activities. The sub-sequences are extracted when they match to pre-defined patterns of primitive events. As input, we use three sequences of primitive events for each video dataset. The sequences aim at characterizing fine, intermediate and coarse semantical activities and each sequence is computed using a scene model composed of dedicated topologies.

## 6.2 Approach: Activity Discovery

The discovery of activities is an automatic process. The process is performed for different resolutions at the same time retrieving a coarse-to-fine qualitative resume of the activities in the video timeline. The procedure has two principal goals:

1. **Find and mark the start and end** video frames of interesting activities. This can be seen as splitting and grouping the sequences of primitive events into sub-sequences that represent meaningful activities.
2. Perform a **naive classification** of the discovered activities to identify the instances of the same activity in the timeline.

The two goals are achieved by seeking and classifying primitive event sub-sequences that correspond to particular patterns (**activity patterns**).

### 6.2.1 Seeking for activity patterns

We observe a tight relationship between an object's transitions among the scene regions and a semantic activity. Usually, the interesting transitions are the primitive events which sequentially describe a closed loop between the regions. We refer to a loop as a sub-sequence of *PEs* which involves a fixed number of scene regions. For example, when a person is "Preparing Meal" several movements between the kitchen "sink" and the "table" appear sequentially. Then, when the "sink" and "table" are scene regions a loop involving these 2 regions can characterize the activity "Preparing Meal".

We define as activities to the patterns which capture sub-sequences of primitive events that can describe a semantic activity (i.e. "Eating"). The patterns aim at capturing and characterizing the basic constructors of the transition loops. Considering a topology and a primitive event sequence as a state diagram the basic loop constructors correspond to either: 1) the staying in a current state "*Stay*" or 2) the change of state "*Change*". The benefits of using basic loop constructors as the patterns used to discover activity is discussed later in this chapter.

Using a primitive event sequence as input, we seek for sub-sequences following two patterns: *Stay* and *Change*. Each pattern is a loop constructor and describes semantically an activity. Formally,

A *Stay* pattern describes an activity occurring within a single topology region (e.g. "Sitting in the Armchair"). A *Stay* is composed of a sub-sequence of *PEs* of the same type.

A *Change* pattern describes an activity occurring between two topology regions (e.g. "from Bathroom to Table"). A *Change* is composed of a single *PE*.

Using regular expressions:

A  $Stay_{A-A}$  pattern is a maximal sub-sequence of the input *PE* sequence of the type:

$$Stay_{A-A} = (A \rightarrow A)^+ \quad (6.1)$$

A  $Change_{A-B}$  pattern is a single *PE* of the type:

$$Change_{A-B} = (A \rightarrow B), A \neq B \quad (6.2)$$

The process of activity discovery takes as input 3 sequences of primitive events at different resolutions<sup>1</sup>. For each sequence the *Stay* and *Change* sub-sequences are marked and extracted corresponding to the discovered activities.

---

<sup>1</sup>Each resolution can be seen as different interpretation of the same video chunks.

**A discovered activity:** We define a discovered activity (**DA**) as an extracted  $Stay_{A-A}$  or  $Change_{A-B}$  primitive event sub-sequence. Formally:

$$DA_{P-Q} \in \{Stay_{P-Q} \cup Change_{P-Q}\} \quad (6.3)$$

The discovered activities can take place at different resolutions at the same time. The resolution of each discovered activity is inherited from the resolution of the input primitive event sequence from where it is extracted.

Each discovered activity is coarsely classified by 3 attributes:  $type_{DA}$ ,  $start\ frame_{DA}$  and  $end\ frame_{DA}$ . The  $type$  of a discovered activity is the sub-index (e.g.  $A-B$ ) attached to it (i.e.  $DA_{A-B}$ ):

$$(type \sim A - B) \Leftrightarrow (type_{DA} == DA_{A-B}) \quad (6.4)$$

The  $start\ frame_{DA}$  and  $end\ frame_{DA}$  are temporal marks (frames) obtained from the first and last primitive event composing the sub-sequence of  $DA$ .

The discovered activities appear sequentially ordered and cover the whole length of the input video. All primitive events at all resolutions are linked to some discovered activity. It means that all activities of the video are discovered and that all video frames are linked to a discovered activity (no empty spaces). With respect to the video frames the  $start\ frame$  attribute of a discovered activity is the immediate next frame of the  $end\ frame$  attribute of the previous discovered activity in the timeline (see fig. 6.1).

The figure 6.1 illustrates the discovery procedure of 3 input  $PE$  sequences at different resolution levels (i.e.  $Level : 5, 10, 15$ ). Each discovered activity is colored by its  $type$  (i.e. subindex (A-B)). The colored segments could be used as content retrieval. The  $start\ frame$  and  $end\ frame$  of each discovered activity are represented with red and blue arrows. It is interesting to see how hierarchical tree structures can be automatically build, where **each discovered activity is a "sub-activity" of another discovered activity at a coarser resolution.**



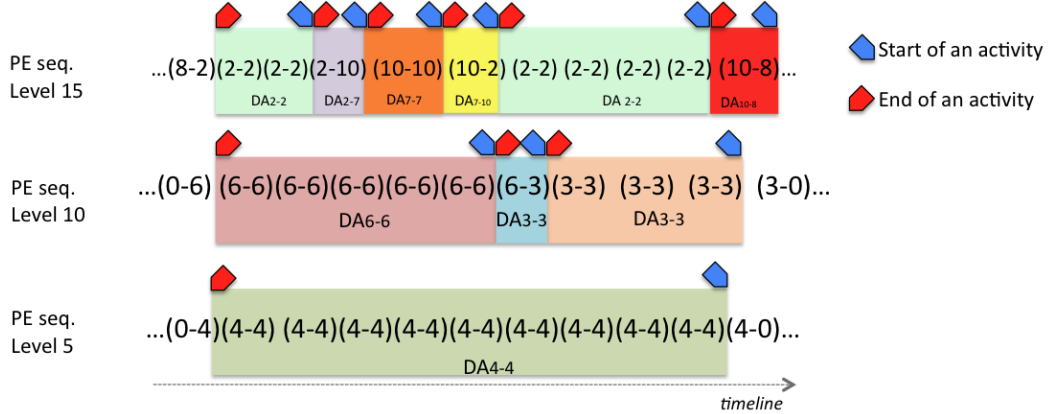


Figure 6.1: Example of discovered activities -DAs- (colored segments) extracted with the *Change* and *Stay* patterns at multiple resolutions. In red and blue arrows are marked the  $start\_frame_{DA}$  and  $end\_frame_{DA}$  of each *DA* corresponding to their occurrence in the video timeline.

The discovered activities (**DA**) are not the models of the activity, they are breaking the video into meaningful parts where each part is a semantic activity. We propose the modeling of a target activity as a hierarchical characterization of the structure defined by its sub-activities. The activity modeling is a subject discussed in the next chapter.

### 6.2.2 Behind the *Stay* and *Change* activity patterns

In our previous work [Guido 2010], we use complex patterns to discover activity based on trajectory analysis. The patterns characterize the interactions between several scene regions at once (i.e. complex loops<sup>1</sup>). We have evaluated different types of patterns for activity discovery converging to the basics and previously defined *Stay* and *Change* patterns.

The motivation of the basic *Stay* and *Change* patterns come across the observation of the activities and their relationships with the primitive event sequences. Analyzing the structure of the activities (loosely constrained) we find that the interesting sub-sequences are built of closed sequences of transitions of multiple scene regions. The two structures that can generate loops are the previously defined *Stay* and *Change* patterns. The basic patterns can be composed to describe complex interactions (or loops) between the scene regions.

Our statement is: *The "Stay" and "Change" patterns are sufficient and necessary to discover complex activities*. Following, the benefits of using the defined

<sup>1</sup>We refer as complex loops to those which involve 2+ scene regions.

patterns are shown by comparing them with other complex ones and summarizing their differences:

**1. The basic patterns are easy to understand by humans.** Semantically the classification of *Stay* and *Change* sub-sequences can be directly related with the objects motion or non motion between regions. The classification of more complex structures require the definition of different classes for the classification of a single discovered activity.

**2. The activity discovery is non parametric and deterministic.** Patterns that build loops of 2+ scene regions allow the competition of the self contained sub-sequences. The competition makes hard to take a deterministic decision of which sub-sequences represent an interesting activity (see figure 6.2). The problem could be addressed by defining sub-sequences weighting rules. For example, long sub-sequences involving two scene regions are more important that the sub-sequences involving three regions. The definition of rules and parameters is avoided when using the *Stay* and *Change* patterns.

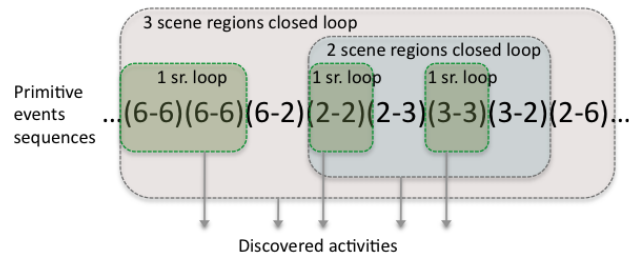


Figure 6.2: Illustration of the activity recognition competition when complex loop patterns are extracted.

**3. The basic patterns can describe complex ones.** A problem that can be noticed is that the *Stay* and *Change* are too simple to characterize complex interactions of multiple scene regions (e.g. loops between 3+ regions). Such statement can be true for a single resolution *PE* sequence. Nevertheless, by working at multiple resolutions, the basic patterns are grouped automatically composing complex interactions. The grouping is automatically achieved by the alignment of the 3 primitive event sequences in a timeline. A pattern at a coarse resolution contains the complex interactions described at a finer resolution.

For example, the figure 6.3 illustrates the situation of the hierarchical construction of closed loops and the structure of scene regions. Where, a basic pattern ( $Stay_{1-1}$ ) contains complex interactions between regions (2 and 3) at a finer resolution.

For a multiple resolution approach, the use of the defined patterns (i.e. *Stay* and *Change*) is sufficient to describe multi-region interactions as other complex patterns would do.

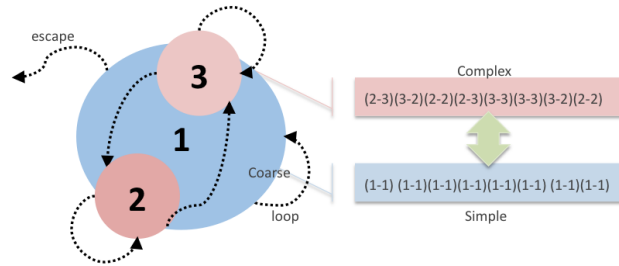


Figure 6.3: Example of "complex" relationships between scene regions (2 and 3) at a finer resolution, described by a "simple" pattern  $Stay_{1-1}$  at a coarser resolution. Therefore, a basic pattern can describe complex interactions.

(2) means that complex models (n-grams, histograms) need to be build characterize and cluster the discovered activities. The problem of building such models is the parametrization and the models complexity for human understanding (n-gram of 10 different primitive event types).

#### 4. The basic pattern instances are used to learn activity models.

All discovered activities are a sub-activity of an activity at a coarser resolution. Therefore, any discovered activity characterizes an other activity. Such recursive and hierarchical property allows the use of the discovered activities as the descriptive features for characterizing the model of a coarser activity. The modeling of an activity is an issue discussed later in this chapter. *The model of an activity is a multi-resolution characterization of the hierarchical structure defined by its sub-activities.* The model of an activity is learned using the discovered sub-activities (features) of finer resolutions.

If the sub-activities were discovered through complex patterns allowing interactions of more than 2 scene regions at the time, complex representations were required, otherwise different sub-activities are clustered together (i.e. underfitting).

For example, let a primitive event be a letter "a" or "b", and a discovered activity a *word*. Suppose that a pattern allows sub-sequences between 2 scene regions, and the words "baaaab" and "abbbba" are extracted. Both words contain the same letters, but they are not representing the same activity due to the different frequency and ordinary of the letters. In such a case, an uni-gram (see A.2.3) could better characterize the distribution of the frequency of the letters. Now, suppose also that the order of the letters is important. In that case, an even more

complex representation should be used, for example a n-gram. Therefore, while more letters (primitive events) are allowed (pattern) in a word (discovered activity) more complex is the representation required.

The discovered sub-activities are used as features for learning activity models. We want features as compact and as descriptive as possible. They need to be descriptive to discriminate similar models. They also need to be compact to be friendly understandable capable of providing explanations in the case of a system failure (e.g. an activity miss-recognition). The *Stay* and *Change* are compact words composed of a single letter, the words are as equally descriptive as complex patterns due to the use of a multi-resolution scheme as it is explained in the previous item **3**. Concluding, the proposed patterns represent a necessary and sufficient descriptor of sub-activities that can be used as a support to build activity models.

### 6.2.3 Two main achievements

- **First**, a discovered activity -DA- defines a time segment marked by the *start frame<sub>DA</sub>* and *end frame<sub>DA</sub>* attributes. Therefore, each activity (token) has a temporal length (i.e. duration of *Stay* and *Change*) that represents an activity. Other methods usually understand activity using a sliding window over the video timeline to build a token over a set of letters (*PEs* in our case). For example, the use of *n - grams* (e.g. [Ham 2009b]) requires the definition of *n* which is the length of a sliding token (window). More recently, [Emonet 2011] uses non parametric bayesian networks with words composed of sequences of letters of a fixed length. The methods require the set up of several parameters in the discovery stage, and build overlapping words of sequences of primitive events. The usage of a sliding window is time consuming and requires its size parametrization. The main advantage of our discovery process is to cluster letters into interesting words where the length of a token is adapted to the perceptual information (i.e. a single token captures that a person is sitting for 10 minutes). The discovery procedure reduces the unnecessary storage of useless information, storing only the information that has a semantic meaning.
- **Second**, each discovered activity is coarsely classified by its *type<sub>DA</sub>*. The classification occurs at multiple resolutions providing to the user a snapshot of the video. Most of literature approaches (e.g. [Jouneau 2011]) propose a single resolution data representation of the results of a supervised activity recognition method. Our classification is obtained automatically and is used as an interface that provides insights of the occurrences of the interesting activities and sub-activities. In the next section (6.3), we provide results of the activity discovery procedure which show the type of inferences that can be made automatically using the coarse-classification.

## 6.3 Activity discovery examples and HCI

The sequences of discovered activities are presented to the user in a multi-resolution graphic. The discovered activities are colored by type. For example, all  $DA_{A-B}$  corresponding to a certain resolution level, have the same color. The color is assigned using the *type* attribute of the discovered activities.

The figure 6.1 illustrates an example of the graphical representation of the activity discovery method. Where the discovered activities are colored at different resolutions describing sequences and sub-sequences of discovered activities.

The display of sequences of discovered activities is a first automatic classification of the activities and sub-activities occurring in the video. The classification provides insights for the rapid analysis in seconds of hours of video. Following, we present results of the activity discovery procedure for different types of videos and applications.

### 6.3.1 Homecare Gerhome

Figure 6.4 displays the discovery results for a person living in an apartment during 4 hours. The apartment scene is displayed in the figure 4.3. The video is categorized as a home-care application.

*The parameters:* The video is recorded using a monocular camera from 10 am to 2 pm. The length of the perceptual feature chunks (*PFCs*) is adapted dynamically (4 hours of video are reduced to 786 KB on disk). The global and local features are computed using 2D information (segmentation, classification and tracking errors occur and are taking care by the system). The primitive event sequences are built using topologies of levels 4, 7 and 10.

*Analysis:* In the figure 6.4 we add semantic labels to 4 of the discovered activities.

- Globally, three of the four selected activities cover above 83% of the video time. These activities are: "*Preparing Meal*", "*Eating*" and, "*Sitting in the armchair*". The remaining activity "*In the bathroom*" occupies less than 7% of the time. The remaining video time (non labeled) mostly corresponds to short movements of the person between the scene regions.
- Analyzing the segments of "*Sitting in the armchair*". In the resolution of Level 4: all instances (*a*, *b*, *c*, *n*, *p*) have the same color (same *PE* sub-sequence). In the Levels 6 and 11: sub-activities of *b*, *c* are found as color stripes (meaning

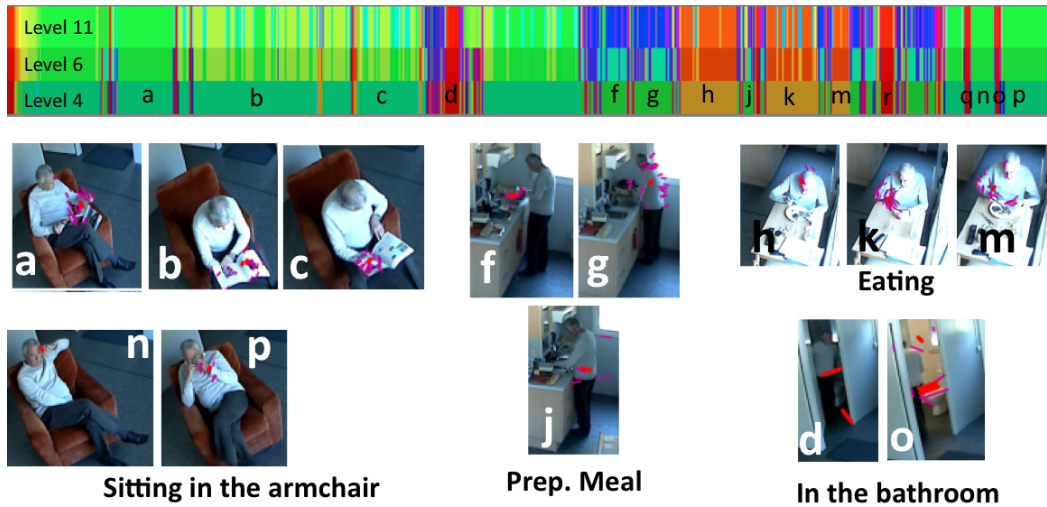


Figure 6.4: Example of the activity discovery results. All activities are automatically colored. We extract, for illustration purposes, 4 activities which cover the 90% of the video time. Snapshots of the activities and their occurrence in the video timeline are marked with letters.

motion changes). The stripes are related to the person laying back and forward while reading a magazine. In *a*, *n*, and *p* the person is mostly static.

- Analyzing the segments of "Preparing Meal" and "Eating". Between *f* and *g* the person "sets the table"; he eats in *h*; he prepares another meal in *j* and, eats again in *k* and *m*. Finding "Preparing Meal" after "Eating" is an example of how unstructured the activities are, even from a behavioral point of view.
- The segments *q*, *o*, *r*, and *d* can be considered as unfrequent. The segments correspond to the "In the bathroom" activity.

To evaluate the errors of the automatically discovered activity segments (i.e.  $start\ frame_{DA}$  and  $end\ frame_{DA}$  marks), 2 persons annotated manually the selected activities in the video<sup>1</sup>. We compute an *error* metric as the percentage of non intersecting segments. **First**, we compare the error between the two manually annotated ground truths to describe the human error: it correspond to the 5% of the video duration. **Second**, we compare the proposed discovery procedure and the average ground truth: the *error* of the discovery method and the average annotation is 4%.

<sup>1</sup>Two ground truths of the same video are built describing the  $start\ frame_{DA}$  and  $end\ frame_{DA}$  of each instance of the 4 targeted activities.

### 6.3.2 Hospital

Figure 6.5 illustrates the discovery results of a person performing activities in a hospital room during 1 hour. The hospital scene is displayed in the figure 4.2. The video is categorized as an healthcare application.

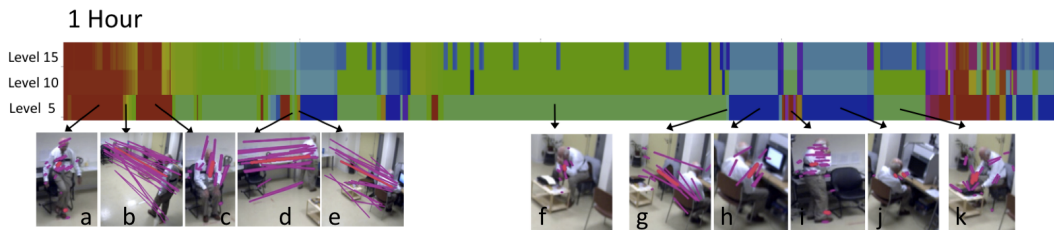


Figure 6.5: Illustration of the activity discovery results of a Hospital scene dataset. The colored stripes are the discovered activities in the video timeline, similar colors are similar activities. Also it is displayed the snapshots of the processed video linked to the discovered activities.

*The parameters:* The video is captured using a monocular camera. The length of the perceptual feature chunks is adapted dynamically. The global and local features are computed using 2D information (segmentation, classification and tracking errors are present). The primitive event sequences are built using topologies of levels 5, 10 and 15.

*Analysis:* It is interesting to analyze the colors and the spatial location where the activities occur. For example, *a* and *c* occur at the "chair" area. The segments: *b*, *i*, *d*, *e*, and *g*, are displacements over the scene. The segments: *k* and *f* are activities at the table region. Finally, the segments: *j* and *h* represent activities at the computer region.

### 6.3.3 Sleeping Monitor

The goal of this application is to discover and characterize "activities" of a person while he/she sleeps. The results can be used by doctors to analyze disorders that involve abnormal and unnatural movements in connection with sleep (e.g. sleep terror, sleep walking, epileptic attacks, etc.). Also it is possible to analyze the evolution of medical treatments or psychiatric conditions that produce sleep disorders (e.g. mood disorders, panic, psychosis *such as schizophrenia*, etc.).

The contribution of vision in this field is limited. A challenge is to detect humans under low-light conditions and with minimal motion. We propose the use of RGB-D information to detect and track the 3D silhouette of a sleeping person.

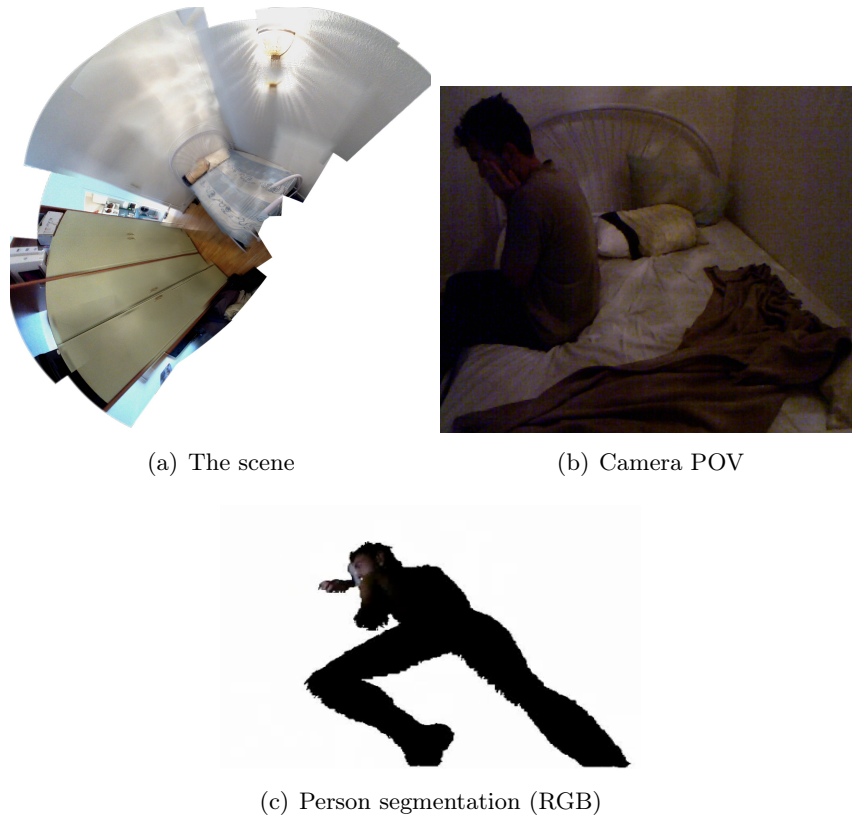


Figure 6.6: Example of the setup of the sleeping monitor scene.

Figure 6.6 illustrates the setup of the sleeping monitor scene. It is a real home bedroom (a) with a bed (b). The segmentation of the sleeping person (c) is a RGB image (it appears dark when the lights are off).

Figure 6.7 illustrates the discovery results of a person sleeping during 6 hours.

*The parameters:* The perceptual feature chunks have a fixed length of 1.2 seconds each. The full video is reduced to a file of 2.4 MB on disc. The PFCs contain only global information. The information used is minimal. This is, the position of the center of mass of the tracked person in 3D (depth camera). The primitive event sequences are built using topologies of levels 4, 6 and 8.

*Analysis:* The semantic labeling of the discovered activities is ambiguous. Nevertheless, it is interesting to analyze the posture and motion between the discovered segments. In most of the cases, the usage of a single 3D point seems to be enough to differentiate and characterize postures<sup>1</sup>. This posture is used to visually evaluate the discovered activities.

<sup>1</sup>The center of mass (3D) changes with the posture changes. For example, extending an arm makes that the center of mass slightly shifts its position towards the extended arm.



- The interval segments *a* and *g*, are similar at "Level 4" and different at "Levels 4, 6".
- The interval segments *b* and *h*, are similar at all resolutions. Such a similarity is coherent with the posture images.
- The interval segments *c* and *f*, are similar, the similarity is coherent with the posture images, where the person is bending the legs.
- The interval segments *j*, appears as an unfrequent color. It is the only instance where the person sleeps upside down.
- The interval segments *d* and *e*, show at "Level 6" the motion of the arms of the individual towards his head without modifying the global body position.

We evaluate the discovered activities by marking the changes of the global posture of the person. When the motions produce a change of the position of all the body parts. We compare the marks with the automatically detected ones (e.g. changes of color at "Level 4"). We are able to discover all posture changes. Even more, we discovered 2 true changes that were missing in the ground truth. The temporal errors between the matched changes are  $< 1\%$ . This indicates that the difference between the manual (GT) and automatic marks is lower than 1 minute in 6 hours of video.

In our future work we aim at exploring in deeper detail the sleeping monitoring application. The presented results have been already evaluated by doctors asserting the visualization of the sleeping cycles. Next steps include the comparison of the proposed method with currently used techniques (i.e. accelerometers).

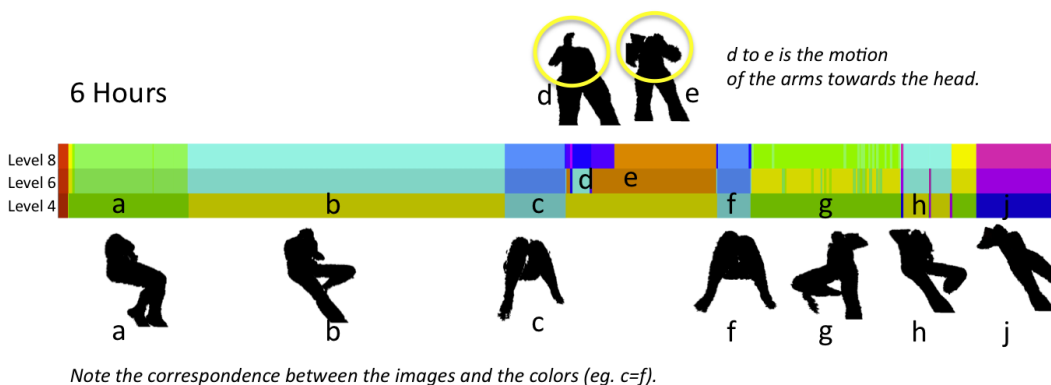


Figure 6.7: Illustration of the activity discovery method applied to the center of mass (3D) of a tracked sleeping person.

### 6.3.4 The Office

The office is a 12 minutes dataset where classical office activities occur. The discovery results are illustrated in the figure 6.8.

*The parameters:* The perceptual feature chunks contain only global information. The information used is the position of the center of mass of the tracked person. The primitive event sequences are built using topologies of levels 5, 8 and 10.

We evaluate the procedure by comparison of the discovered segments and a ground truth. The ground truth is built by two persons. The ground truth contains the *star frame<sub>DA</sub>* and *end frame<sub>DA</sub>* marks of a set of target activities. We provide a set of possible activity labels: "Standing" (a)(b)(d), "walking", "sitting at the desk chair" (X), "sitting at table" (c), "laying back at the chair" (g), "laying forward at the chair" (f). Comparing the ground truth and the discovered segments the temporal difference (*error*) is 2% (0.24 minutes).

An interesting event takes place when the ground truth characterizes the activities at the desktop chair. The manual annotations are not describing the multi-resolution nature of the activities "sitting at the desk chair" (X) and "laying back/forward at chair" (g)(f). The discovery method is capable of describing such a situation and characterizing the sub-activities (g)(f) occurring when the person "balances" in the chair.

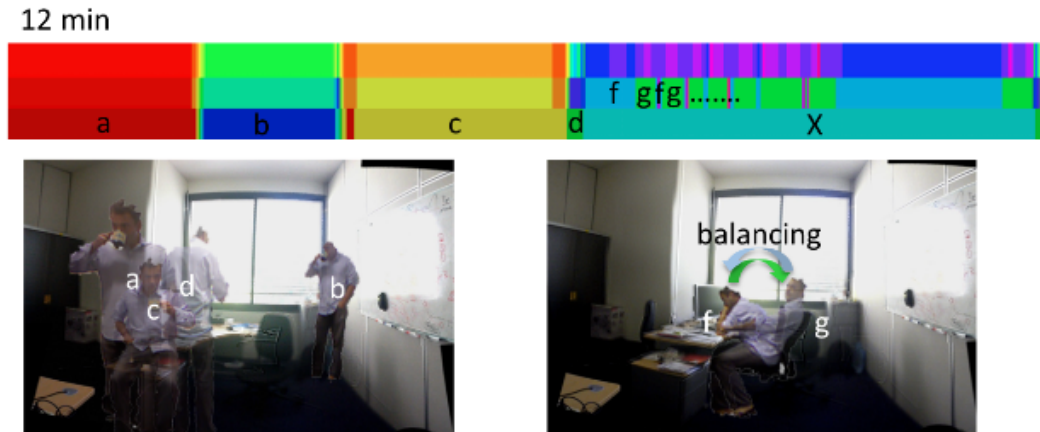


Figure 6.8: Illustration of the activity discovery method applied to the center of mass (3D) of a tracked person in an office scene.

## 6.4 Activity Models

The model of an activity is its description using a formal language. The model can be seen as a package of descriptors that can uniquely characterize a type of activity. The descriptors are obtained by abstracting different activity features. The set of possible features is large and may include temporal, spatial, local and global characterizations of a scenario. The selection of the features depends mainly on three aspects:

1. The application. For example, if the users interest is to analyze the variation of the length of an activity performed by different people, the duration of the activity needs to be represented in the model.
2. The feature discriminative strength. This is, the capability of a descriptor to characterize uniquely an activity. For example, a feature indicating the presence of a face can characterize an activity such as "the person is **facing** the camera" in a better way than his center of mass.
3. The reliability of the features. While less erroneous data is characterized in the features, then better is the model of the activity.

When the application is well defined, the selection of the features becomes a trade-off between its discriminative strength and the amount of noise that they can carry.

The descriptors we use for learning the model of an activity are its sub-activities and the attributes of the sub-activities (e.g. *LocalDynamics*). The learning process is achieved automatically using a set of **discovered activities (DA)** of several finer resolutions. The set of DAs represents the sub-activities and it is computed automatically.

*The benefit of learning the activity models is to allow the management of noise.* In most of the cases, it can be assumed that the data noise is repeated under similar environmental conditions (when the same activity is perceived). Using real data (i.e. signal) the presence of noise is assumed, the noise is included as a part of the model. The noise is hard to synthesize making that most of the manually specified systems cannot understand nor handle the presence of noise. For example, the presence of a window can produce a shadow which shifts the position of a tracked person. Such a shift is hard to specify manually, nevertheless it can be learned.

Learning models allows the computation of the similarity between activities. Measuring the similarity enables also: 1) the automatic **recognition** of an activity in an unseen dataset; 2) the measurement of the variance of "the way" an activity is perceived; 3) the detection of unfrequent activities. In this work we evaluate the framework to recognize target activities. Nevertheless, the framework can detect unfrequent activities by comparing the frequency of activity models which are built

automatically.

The modeling of frequent activities leads to a generative model of the scenario. The scenario model is a macroscopic description of the expected activities and it can be used to achieve complex goals such as people profiling. The profiling is achieved performing statistical comparisons among the perceived activities of different persons. For example, persons that perform low motion activities for long periods of time (e.g. watch TV) can be considered more sedentary than others.

The activity models proposed in this chapter, capture the hierarchical structure of an activity and its sub-activities. The model of a target activity is learned based on the correspondent discovered sub-activities. For each activity, the set of sub-activities is defined by a neighborhood. The neighborhood defines a hierarchical and multi-resolution relationship of activities and sub-activities.

#### 6.4.1 Activity neighborhood

The neighborhood of an activity  $A$  at some resolution level ( $l$ ) is the recursive representation of the links between  $A$  and its sub-activities  $B_i$  at the next finer resolution ( $l + 1$ )<sup>1</sup>. Formally, it is represented as a tree of multiple layers.

$$A_{neighborhood} = ((B1, B1_{neighborhood}), \dots, (Bn, Bn_{neighborhood})) \quad (6.5)$$

where,  $B1 \dots Bn$  are the sub-activities of  $A$  at the immediate finer resolution.

In practice, we compute the neighborhood for all discovered activities  $DAs$  at the coarser resolution.

Let  $A$  and  $B$  be  $DAs$ , and  $resolution_{coarse}(A) < resolution_{finer}(B)$ <sup>2</sup>. Then,  $B$  is a sub-activity of  $A$  when the temporal interval of  $B$  intersects with the temporal interval of  $A$  in the same video timeline.  $B$  is a sub-activity of  $A$  if the following statement is true:

$$\begin{aligned} & ((start\ frame_A \leq start\ frame_B) \wedge (end\ frame_A \geq start\ frame_B)) \\ & \quad \| ((start\ frame_A \leq end\ frame_B) \wedge (end\ frame_A \geq end\ frame_B)) \\ & \quad \| ((start\ frame_A \leq start\ frame_B) \wedge (end\ frame_A \geq end\ frame_B)) \\ & \quad \| ((start\ frame_A \geq start\ frame_B) \wedge (end\ frame_A \leq end\ frame_B)) \end{aligned} \quad (6.6)$$

The statement 6.6 implies that the same  $B$  can be a sub-activity of more than a single activity. The situation occurs when a sub-activity takes place at the border

<sup>1</sup>We use  $l + 1$  to describe the next finer resolution to  $l$ , meaning that  $l + 1$  is built using more scene regions than  $l$ . For example, in figure 6.9 let  $l = Level5$ , then  $l + 1 = Level10$ .

<sup>2</sup>A finer resolution is equivalent to high resolution and conceptually represents less abstraction, therefore it is capable of characterizing finer activities. An activity ( $B$ ) at a finer (or high) resolution is a sub-activity of other activity ( $A$ ) at a coarse (or low) resolution.

of two adjacent scene regions.

The figure 6.9 illustrates a simple schema of the construction of the neighborhood of an activity  $A$  and its sub-activities at 2 finer resolutions.

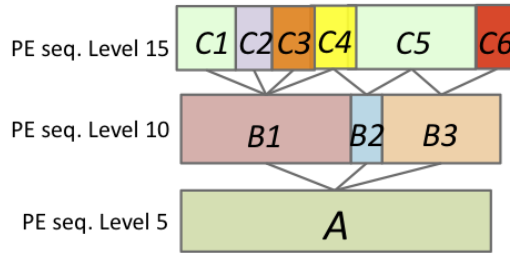


Figure 6.9: Illustration of the neighborhood of  $A$ :  $A_{neighborhood} = (B1, (C1, C2, C3, C4), B2, (C4, C5), B3, (C5, C6))$ .

The figure 6.9 illustrates a compact representation of a neighborhood. A more complete representation of the same neighborhood is displayed in figure 6.10. The different  $DAs$  are coarsely classified and labeled (colored). Also, each  $DA$  contains attached its sub-sequence of primitive events.

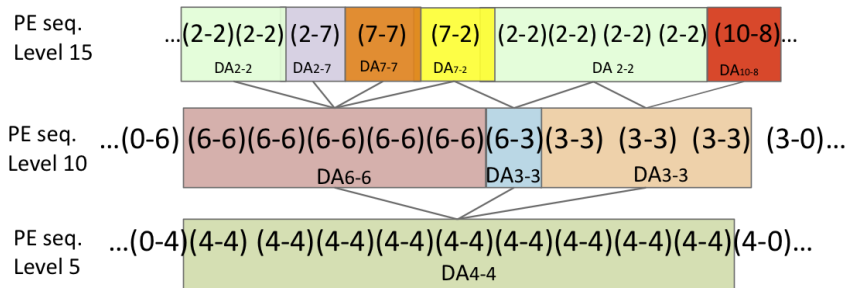


Figure 6.10: Extension of the illustration of figure 6.9 into a complete representation of how the system understands the primitive events, discovered activities and activity neighborhoods.

### 6.4.2 Human Interaction

The combination of an activity neighborhood and the sub-sequence of primitive events contained in each discovered activity is an automatic retrieval from the system to the user. The retrieval process provides to the user the insights of the activities occurring in the video. The figure 6.10 illustrates the retrieval of information. In practice, the user analyses visually the percentage of each sub-activity

contributing to an activity. Such analysis is in many cases enough to identify the instances of an interesting activity in the video. Also, a semantical analysis referring to the objects motion (high, low, repetitive, etc.) can be performed. At this point, the real label of the activity (e.g. "preparing meal") is lacking. The label is necessary to claim real semantical interpretation of the video data. The activity label can be set manually and this is the only human interaction of this work.

The activity labeling is a supervised interaction step and it is optional. Labeling an activity enables that with minimal human effort the system becomes capable of retrieving natural language semantics.

*The process of labeling* is illustrated in the figure 6.11. 1) The sequences of discovered activities are presented to the user. 2) The user analyzes the percentages of sub-activities or observe the real video to set a label for each interesting discovered activity. The semantic label is considered by the system as a target activity. Then, several *DAs* with the same label representing the same target activity.

The labeling can be performed at any resolution level and can be propagated though out. Nevertheless, for simplicity in this work we label only activities at the coarser resolution.

The figure 6.11 illustrates an example of the labeling procedure where the  $DA_{4-4}$  is labeled as "*Preparing Meal*" and a model is learnt using its neighborhood.

The model of a target activity (labeled) is automatically learnt. In the case that two discovered activities are labeled with the same semantical meaning, the user has to decide to learn independent models or to use both as **training prototypes** to learn a single activity model. The figure 6.12 illustrates an schema of the modeling procedure where 3 training prototypes of the same target activity (i.e. "*PreparingMeal*") can be used to learn a reinforced activity model<sup>1</sup>.

In practice, the human intervention is minimal and adds important semantical capabilities to the system. The labeling is a procedure where the framework guides to the user by suggesting to him/her with a set of interesting activities (*DAs*) which can be detected. Therefore, the labeling takes only few seconds allowing the description of scenarios with minimal human effort.

### 6.4.3 The Modeling Procedures

We aim at defining general models capable of characterizing a wide range of activities. The models are learnt with 2D or 3D global and local information. The

---

<sup>1</sup>If several training prototypes are used to learn an activity model, the model is called a **reinforced** model.

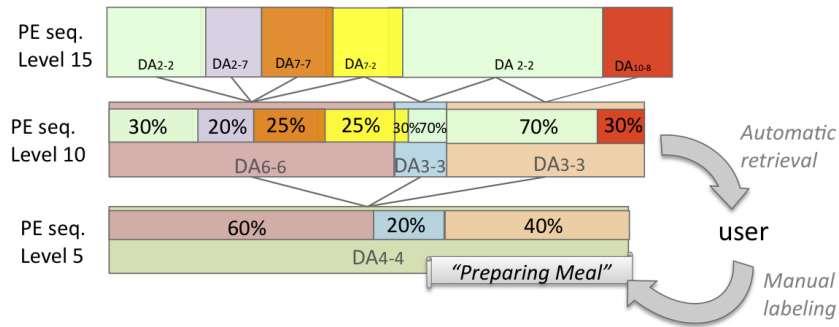


Figure 6.11: Illustration of the process of labeling. It is the only weakly supervised part of this work and it is optional. The user chooses a discovered activity which is displayed graphically. The selection triggers a pop up video which the user can interpret semantically. The semantical interpretation is a label which is attached to the selected discovered activity.

models need to be formal structures flexible enough to be modified with minimal effort.

The image 6.12 illustrates the general schema for learning the model of an activity. The training data used for learning a model are few (or one) training prototypes (reference instances) and their correspondent neighborhoods of a target activity (e.g. "Preparing Meal"). The prototypes share the same label, therefore ideally they have the same semantic meaning which is chosen by the user. We represent the model as an empty box which contains the description of a target activity once it is learned.

In this work, we use **single** model to denote models learnt using one training prototype, and **reinforced** model when it is learnt using several prototypes.

The model of an activity can be defined in several ways and many ideas can be borrowed from the natural language modeling paradigm. In A.2 we present the analogies of the language and activity modeling paradigms. Also, Manning and Schutze [Manning 2005] propose an interesting description of to the foundations of statistical natural language processing.

One type of models that can be borrowed from the language processing and used for activity modeling is based on the frequency of appearance, ordering, and relevance of the sub-activities. These models are known as **frequency models**. A description of classical frequency models is presented in A.2.2. Such as: Tf-Idf (A.2.2.1); a metric of word relevance (A.2.2.2); Successor-Predecessor (A.2.2.3); Successor-Predecessor Quotient (A.2.2.4); n-grams (A.2.3). The previously enumerated methods cannot handle multi-resolution structures (activity neighborhood)

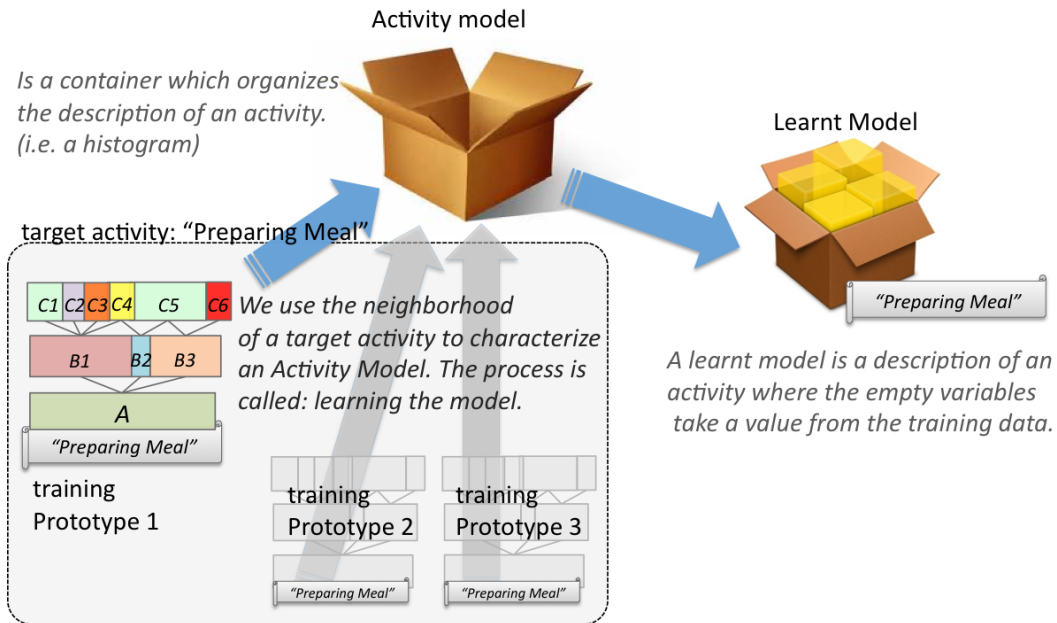


Figure 6.12: General schema of the process of learning of an activity model. Few training prototypes are used to learn a model. The prototypes share the same label chosen by the user. A learnt model is used later for activity recognition purposes.

directly. The models we propose are inspired by the measurement of the frequency of the training data as the previous methods, but adding multi-resolution capabilities.

Another type of models that can be borrowed from the language processing are **topic models**. When writing a text document, it is common practice to first sketch down the main ideas/topics that we want to write. Then, we express those ideas with concrete sentences and paragraphs. Inspired in this two step process, **topic models** are formal probabilistic generative models that learn some topics characterized by histograms over a set of words. The topics correspond to abstract latent components, however in many situations they are interpreted as real topics. Understanding the different variants of topic models is not in the scope of this work but we briefly refer to two of the most popular approaches:

1. **LDA-Latent Dirichlet Allocation:** Given a set of text documents, and knowing that certain topics characterize the documents, an interesting problem is to classify the words of the document into topics, which indicate their possible meaning. In activity modeling, a topic can be seen as a cluster of sub-activities that characterize a target activity. LDA is a popular model based on Bayesian networks to infer the underlying topics of text data. LDA is introduced by Blei et al [Blei 2003b]. An instructive overview of LDA can



be found in: [Blei 2011]. There are several variants of LDA, including a hierarchical version [Blei 2003a]. Nevertheless, the main limitation of LDA is that the number of topics needs to be set a-priori.

2. **HDP-Hierarchical Dirichlet Process:** HDP is described by [Teh 2005], it is a model derived from LDA, where the number of topics is not assumed to be given. HDP learns the number of topics from the sampling data. Since the number of topics is unknown a priori and could be large, the finite topic mixture of each document can be replaced by an infinite topic mixture with a Dirichlet process as a prior. The main difference between HDP and LDA is that LDA assumes a fixed number of topics to be set a priori. Recently, a variant of HDP is proposed to cluster structured activities using as input optical flow [Emonet 2011]. Nevertheless, the approach can only detect unfrequent activities of short duration.

Using natural language approaches out of the box (*frequency* and *topic*) for non-structured long-term activities has three main problems:

**First**, the approaches consider the ordering of the words (sub-activities). Beyond language modeling exists the notion of "building a sentence", which is possible due to the rules established by the language grammar. In our case, when activities are loosely constrained (e.g. "preparing meal") nothing guarantees that a person is going to "use the refrigerator" before/after "using the sink".

**Second**, the hierarchical learning of the models assumes the data has been collected at a single resolution. The models are not prepared for multi-resolution data representation. Even more, the input data for document modeling is simple (words), while for activities more quantitative information needs to be represented (e.g. location, duration, motion).

**Third**, the models are mainly designed to work with low-level features as single-layer approaches. Therefore, the models are complex and hard to modify manually. We aim at flexible models capable of describing the high-level discovered activities which hold semantics.

In this work we propose two methods for modeling activities that follow the classical deterministic and probabilistic paradigms. The models are described in the following sections 6.4.4 and 6.4.5.

### 6.4.4 Hierarchical Activity Models (HAM)

We get inspired by the Hierarchical Dirichlet Process to propose a hierarchical model that captures the structure of an activity and its sub-activities at once. The model is represented as a tree of nodes (defined in section 6.4.4.1) which initially have a similar structure to an activity neighborhood (defined in section 6.4.1).

Each level of the tree of nodes represents a resolution level of an activity neighborhood. In practice, we use 3 resolution levels. The training of a model takes as input a set of activity neighborhoods. The root of the neighborhoods describes the same semantical activity which is the target activity to be modeled. Each input neighborhood is considered as a training prototype of the same target activity.

#### *A pre-clustering stage*

At each resolution level of the input neighborhoods we perform a clustering stage over the set of discovered activities ( $DA$ ). The clustering aims at grouping similar  $DA$ s to build nodes<sup>1</sup> ( $\mathbf{N}$ ) which are the building blocks of the model. Several  $DA$  attributes can be considered for the clustering stage (e.g. "duration", "local dynamics", etc). In practice, we cluster the  $DA$ s that share the same type ( $type_{DA}$ ) -which are graphically illustrated by the same color-. The clustering is performed recursively and the scope is local to the neighborhood of a  $DA$ ; this means that the information of a single  $DA$  can be repeated in two different nodes. An example of the clustering stage is illustrated in the figure 6.13, where a tree of nodes is learnt pre-clustering the inputs:  $A1_{neighborhood}$  and  $A2_{neighborhood}$ .  $A1$  and  $A2$  are semantically equivalent (i.e. correspond to the same activity: "Preparing Meal"). The clustering of the  $DA$ s ( $B_i, C_j, D_k, E_l, F_m$ )<sup>2</sup> is performed at each resolution level. The clustering results are nodes which are built based on the type of the composing  $DA$ s (i.e. cluster members). Graphically, we illustrate the correspondence between a node and its composing  $DA$ s using the same color.

The **links** between the nodes represent the hierarchical structure of the model. The link structure is based on the structure of the input activity neighborhoods. For example, in the figure 6.13 a sub-node of  $\mathbf{N}_0$  is a cluster composed of sub-activities of  $A$ , where  $A$  is an activity composing the node  $\mathbf{N}_0$ .

A **node** is not only a cluster of activities but it holds also the abstraction of the properties of the contained activities and the linkage with finer resolution sub-activities. In the next sections we explain the notion of a node and we define a set of procedures to learn the model of an activity.

<sup>1</sup>A node and its properties are defined in the next sub-section 6.4.4.1.

<sup>2</sup>The  $DA$ s:  $B_i, C_j, D_k, E_l, F_m$  are the discovered sub-activities of "Preparing Meal" at different resolutions.

#### 6.4.4.1 A Node

A node represents a cluster of discovered activities ( $DA$ ) that share similar properties (e.g. type, duration) and similar sub-activities. A node  $\mathbf{N}$  is composed of a set of discovered activities  $\{DA_1, DA_2, \dots, DA_n\}$  where all  $DAs$  are at the same resolution level. A node is defined based on the resolution level of its composing  $DAs$ <sup>1</sup>.

Each node  $\mathbf{N}$  has two attached elements: *attributes* and *sub-attributes* (see 6.14). The *attributes* of  $\mathbf{N}$  are a set of measurements over the  $DAs$  self contained in the node (i.e. "The  $DAs$  average duration"). The *sub-attributes* of  $\mathbf{N}$  are models characterizing the *attributes* of the sub-nodes linked to  $\mathbf{N}$ . The *sub-attributes* aim at describing the relationships between the sub-nodes (i.e. "The percentage of time occupied by a sub-activity").

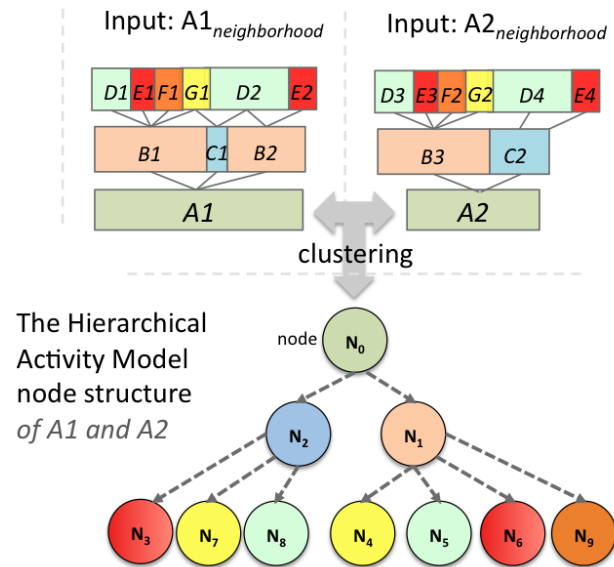


Figure 6.13: Example of the construction of a reinforced model resulting of 2 input training prototypes. The discovered activities  $A_1$  and  $A_2$  have the same semantical meaning which corresponds to the activity we are interested to learn (i.e. target activity). Each  $N_i$  is a computed node obtained from clustering the sub-activities  $B_i, C_j, D_k, E_l, F_m$  at the different resolutions of the input neighborhoods.

#### The Node Attributes

Many attributes could be calculated for describing temporal and spatial activity properties of the activities contained in a node. For a node  $\mathbf{N}$  we define 4 attributes. The attributes aim at characterizing: 1) global and local spatial properties of the

<sup>1</sup>A node is at resolution  $l$  when it is build of  $DAs$  at resolution  $l$ .

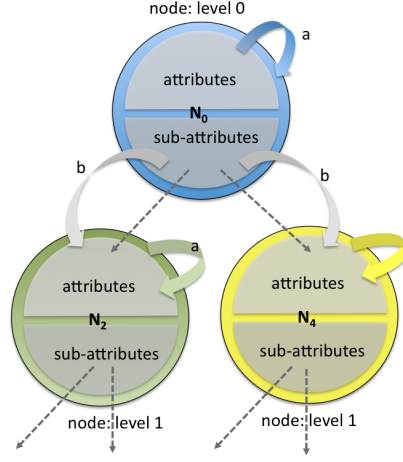


Figure 6.14: Example of the architecture of the model. Each node ( $\mathbf{N}_{0,2,4}$ ) is composed of attributes and sub-attributes. The attributes characterize the  $DA$ s of a node and the sub-attributes characterize the sub- $DA$ s of a lower tree level. With respect to the activities,  $N_{2,4}$  represent two different sub-activity clusters of  $N_0$ .

$DA$ s of  $\mathbf{N}$ , 2) temporal properties of the  $DA$ s of  $\mathbf{N}$  (e.g. duration). The attributes are:

1. *type*: it is adopted from the  $DA$ s composing a node (usually represented by a color). For a node  $\mathbf{N}$ ,  $type_{\mathbf{N}} = type_{DA_q}, \forall DA_q \in \mathbf{N}$ .
2. *Instances*: is the amount of  $DA$ s  $\in \mathbf{N}$ . Note that all  $DA_i \in \mathbf{N}$  have the same type.
3. *Duration*: is a gaussian distribution  $Duration(\mu_d, \sigma_d^2)$  which describes the temporal length of the set of activities in a node  $\mathbf{N}$ . In practice,  $Duration_{\mathbf{N}}$  measures the amount of frames used by the  $DA$ s composing  $\mathbf{N}$ . For the set of discovered activities  $\{DA_1, DA_2, \dots, DA_n\} \in \mathbf{N}$ :

$$\mu_d = \sum_{i=1}^n \frac{end\ frame_{DA_i} - start\ frame_{DA_i}}{n}$$

$$\sigma_d^2 = E[((end\ frame_{DA_i} - start\ frame_{DA_i}) - \mu_d)^2]$$

4. *Local Dynamic Distribution (LDD)*: is a discrete probability distribution that describes the main angles and lengths of the local motion tracklets of the  $DA$ s inside a node. The attribute is calculated from information of the primitive events that compose the  $DA$ s of a node. The figure 6.15 sketches the type of information used to build the histogram of a discovered activity  $DA_1$ . The

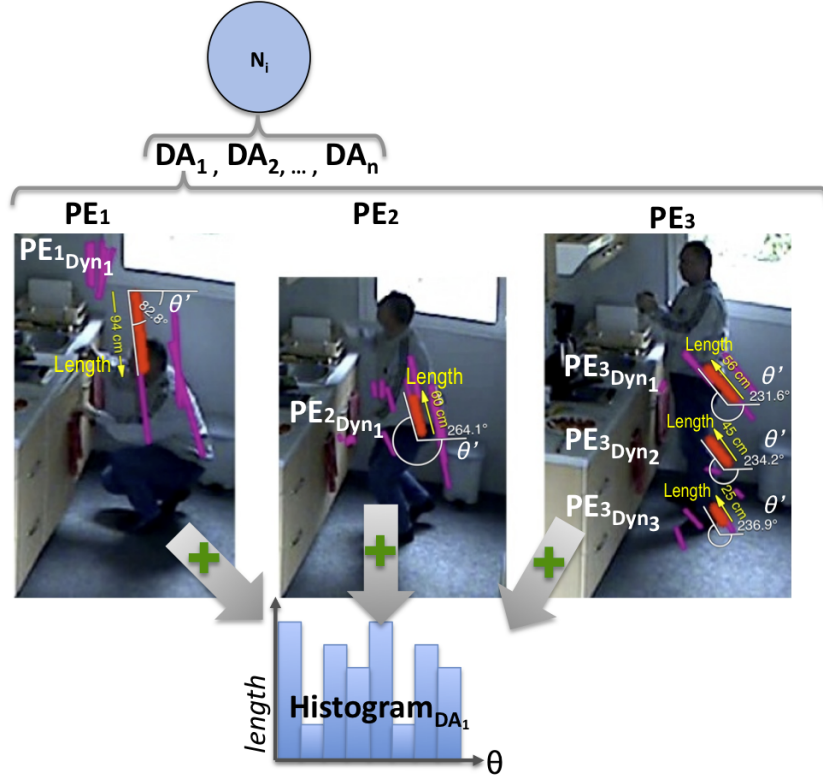


Figure 6.15: Illustration of the construction of a histogram ( $Histogram_{DA_1}$ ) characterizing the local dynamics contained in the discovered activity  $DA_1$ . The histogram is computed by the combination of the angles  $\theta$  and  $lengths$  of all the primitive events contained in  $DA_1$ .

attribute is calculated from all the histograms of the  $DAs$  composing the node. For a discovered activity  $DA_i \in \mathbf{N}$  the histogram  $Histogram_{DA_i}$  is computed using the set of primitive events ( $\{PE_q\}$ ) composing  $DA_i$ :

$$\{PE_q\} | PE_q \in DA_i \quad (6.7)$$

A  $Histogram_{DA_i}$  has 8 bins ( $\theta$ ), each bin is the normalization of the angle described by a local motion tracklet.  $\theta$  can take one of the following values: 22.5, 67.5, 112.5, 157.5, 202.5, 247.5, 292.5, 337.5. The count is the accumulation of the length of the  $PE$  tracklets. A  $Histogram_{DA_i}$  is computed as:

$$Histogram_{DA_i}(\theta) = \sum_{m,q | PE_m \in DA_i \wedge Dyn_q \in PE_m} PE_{m.Dyn_q.Length} \Leftrightarrow PE_{m.Dyn_q.\theta'=\theta} \quad (6.8)$$

The  $PE_{m.Dyn_q.Length}$  is the length of the local motion tracklet  $q$ . The tracklet  $q$  is included in the primitive event  $PE_m$ . The length  $PE_{m.Dyn_q.Length}$  is computed as a straight line between the first  $[0]$  and last  $[n]$  points of the tracklet

$q$ .

$$PE_{m.Dyn_{q.Length}} = PE_{m.Dyn[n]} - PE_{m.Dyn[0]} \quad (6.9)$$

The  $PE_{m.Dyn_{q,\theta'}}$  is the normalization result of the angle described by the tracklet  $q$ . The normalization is achieved by assigning the closest pre-defined angle ( $\theta$ ) to the real calculation. The angle is calculated using the first [0] and last [n] points of the tracklet  $q$ :

$$PE_{m.Dyn_{i,\theta'}} = Norm(angle(PE_{m.Dyn[n]}, PE_{m.Dyn[0]})) \quad (6.10)$$

where  $Norm$  is the angle normalization<sup>1</sup>, and

$$angle(A, B) = (180/PI) * \begin{cases} value(A, B), & \text{if } value(A, B) > 0 \\ 2 * PI + value(A, B), & \text{otherwise} \end{cases}$$

where

$$value(A, B) = \begin{cases} PI/2, & \text{if } A.x \equiv B.x \wedge B.y \leq A.y \\ 3 * PI/2, & \text{if } A.x \equiv B.x \wedge A.y \leq B.y \\ 0, & \text{if } A.y \equiv B.y \wedge A.x \leq B.x \\ 2 * PI, & \text{if } A.y \equiv B.y \wedge B.x \leq A.x \\ atan2((B.y - A.y), (A.x - B.x)), & \text{otherwise} \end{cases}$$

where  $A.x$  and  $A.y$  are the spatial coordinates of a tracklet point in 2D.

Finally, the attribute *Local Dynamic Distribution* of a node  $\mathbf{N}$  ( $LDD_{\mathbf{N}}$ ) is learned from the set  $\{DA_i.histogram \mid \forall i DA_i \in N\}$ .  $N_{LDD}$  is a discrete distribution where a random variable  $X$  can take 8 possible values  $\theta$ .

$$LDD_{\mathbf{N}}(\theta_p) = p\{X = \theta_p\} = \sum_{i=0}^{max(i)|DA_i \in N} \frac{histogram_{DA_i}(\theta_p)}{\sum_{q=0}^8 histogram_{DA_i}(\theta_q)} \quad (6.11)$$

In our experiments the usage of a combination of the length and angle of the local dynamic tracklets provides a stable descriptor. In previous attempts we have also introduced the number of tracklets per  $PE$ . Nevertheless, due to the pixel-based nature of the local dynamics it is hard to assure that a similar movement produces the same number of tracklets.

### The Node Sub-Attributes

The set of attributes defined in the previous section is learned from the  $DA$ s contained on a single node implying that there is no description of the relationships

<sup>1</sup>The normalization maps the computed angle value to the nearest value in the set  $\{22.5, 67.5, 112.5, 157.5, 202.5, 247.5, 292.5, 337.5\}$ .

between different nodes. The relationships between nodes is a crucial part of the activity model, informally it describes for an activity the composition of its sub-activities.

The *sub-attributes* are a mean to share information among groups of data (nodes). They are latent variables which are learnt from the attributes of the sub-nodes.

For a node  $(\mathbf{N}^l)^1$  we define two sub-attributes named *mixture<sub>subactivity</sub>* and *timelapse<sub>subactivity</sub>* which aim at describing two properties of the sub-nodes  $(\mathbf{N}_i^{l+1})$  linked to  $\mathbf{N}^l$ .

1. *mixture<sub>subactivity</sub>*: Represents the contribution of each sub-activity *type* in the composition of sub-activities.

The *mixture<sub>subactivity</sub>* is calculated using the *type<sub>N<sup>l+1</sup></sub>* attribute of a set of sub-nodes  $(\mathbf{N}_q^{l+1})$ . The sub-nodes are contained in a set of nodes which have the same *type<sub>N<sup>l</sup></sub>* = ... = *type<sub>N<sub>n</sub><sup>l</sup></sub>*. The set of nodes correspond to different training prototypes of a target activity.

The number of instances of the sub-nodes with the same *type* is modeled with a Gaussian distribution  $(\Theta_{type}^{mixture})$ . The Gaussian distribution characterizes the appearance of the sub-nodes (*type<sub>N<sup>l+1</sup></sub>*) in the nodes  $\mathbf{N}_1^l \dots \mathbf{N}_n^l$ . A Gaussian distribution  $\Theta_{type_i}^{mixture}$  corresponds to a *type<sub>N<sub>q</sub><sup>l+1</sup></sub>* of the sub-nodes of  $\mathbf{N}_1^l \dots \mathbf{N}_n^l$ . The number of different distributions  $(\Theta_{type}^{mixture})$  is the same as the number of unique sub-node *types* at the resolution  $l + 1$  linked to the  $\mathbf{N}_1^l \dots \mathbf{N}_n^l$ .

For a node  $\mathbf{N}_1^l$ , the *mixture<sub>subactivity<sub>N<sub>1</sub><sup>l</sup></sub></sub>* is modeled as a mixture of gaussians (MOG) of  $(\Theta_{type_i}^{mixture})$  with the following parameters:

$K$  = number of mixture components (Gaussians).

$O$  = number of observations (sub-activity instances).

$w_{i=1 \dots K}$  = prior probability of a particular component  $i$ , (the mixture weight of each node type).

$\Theta_{type_{i=1 \dots K}}^{mixture}$  = parameter of the distribution of the observations associated with the component  $i$ . In our case  $\Theta_{type_i}^{mixture} \sim (\mu_i, \sigma_i)$ .

---

<sup>1</sup>The super-index  $l$  denotes the resolution level of a node, where  $l + 1$  corresponds to a finer resolution than  $l$ . Therefore, a node  $\mathbf{N}^{l+1}$  is a sub-node of a node  $(\mathbf{N}^l)$  when they are hierarchically linked.

For two nodes  $\mathbf{N}_1^1$  and  $\mathbf{N}_2^1$ . Let a set  $\{\mathbf{N}_1^{1+1}, \dots, \mathbf{N}_n^{1+1}\}$  of sub-nodes of  $\mathbf{N}_1^1$  and a set  $\{\mathbf{N}_{n+1}^{1+1}, \dots, \mathbf{N}_m^{1+1}\}$  of sub-nodes of  $\mathbf{N}_2^1$ . Let  $type_{\mathbf{N}_1^1} = type_{\mathbf{N}_2^1}$ , then the parameters of  $mixture_{subactivity_{\mathbf{N}_1^1}}(K, O, w, \Theta_{type}^{mixture})$  are calculated as follows:

$$K = |\{type_{\mathbf{N}_1^{1+1}}, \dots, type_{\mathbf{N}_n^{1+1}}\} \cup \{type_{\mathbf{N}_{n+1}^{1+1}}, \dots, type_{\mathbf{N}_m^{1+1}}\}| \quad (6.12)$$

$K$  is the cardinality of uniques *type* of the nodes.

$$\mu_i = \frac{\sum_{p=1}^m Instances_{\mathbf{N}_p^{1+1}} * \delta_{(type_{\mathbf{N}_p^{1+1}}, type_{\mathbf{N}_i^{1+1}})}}{\sum_{p=1}^m \delta_{(type_{\mathbf{N}_p^{1+1}}, type_{\mathbf{N}_i^{1+1}})}} \quad i \leq m \quad (6.13)$$

$$w_i = \frac{\sum_{p=1}^m \delta_{(type_{\mathbf{N}_p^{1+1}}, type_{\mathbf{N}_i^{1+1}})}}{O} \quad (6.14)$$

where  $\delta$  is the Kronecker delta.

2. *timelapse<sub>subactivity</sub>*: represents the distribution of the temporal duration of the sub-nodes. In other words, *timelapse* characterizes the temporal duration of the sub-activities of the same *type*.

The *timelapse<sub>subactivity</sub>* is learned using the  $Duration_{\mathbf{N}^{1+1}}$  attribute of a set of sub-nodes ( $\mathbf{N}_q^{1+1}$ ). The sub-nodes are linked to nodes ( $\mathbf{N}_{1, \dots, m}^1$ ) which have the same *type* and are extracted from different prototypes of a an activity.

The  $Duration_{\mathbf{N}^{1+1}}$  attribute is a Gaussian distribution which characterizes the temporal length of the activities of the sub-nodes  $\mathbf{N}_{1, \dots, n}^{1+1}$  where these sub-nodes of the same type ( $type_{\mathbf{N}_q^{1+1}}^i$ ). We compute a new Gaussian distribution  $\Theta_{type_i}^{timelapse}$  for each set of sub-nodes with the same type ( $type_{\mathbf{N}_{1, \dots, n}}^i$ ) which characterizes the average  $Duration$  of  $\mathbf{N}_1^{1+1} \dots \mathbf{N}_n^{1+1}$ .

For a node  $\mathbf{N}_1^1$ , the  $timelapse_{subactivity_{\mathbf{N}_1^1}}$  is modeled as a mixture of gaussians of ( $\Theta_{type_i}^{timelapse}$ ) with the following parameters:

$K$  = number of mixture components.

$O$  = number of observations (# of sub-nodes).

$w_{i=1 \dots K}$  = prior probability of a particular component  $i$ , (the mixture weight of each node type).

$\Theta_{type_{i=1 \dots K}}^{timelapse}$  = parameter of distribution of observation associated with component  $i$ . In our case  $\Theta_{type_i}^{timelapse} \sim (\mu_i, \sigma_i)$ .



For two nodes  $\mathbf{N}_1^l$  and  $\mathbf{N}_2^l$ . Let a set  $\{\mathbf{N}_1^{l+1}, \dots, \mathbf{N}_n^{l+1}\}$  of sub-nodes of  $\mathbf{N}_1^l$  and a set  $\{\mathbf{N}_{n+1}^{l+1}, \dots, \mathbf{N}_m^{l+1}\}$  of sub-nodes of  $\mathbf{N}_2^l$ . Let  $type_{\mathbf{N}_1^l} = type_{\mathbf{N}_2^l}$ , then the parameters of  $timelapse_{subactivity_{\mathbf{N}_1^l}}(K, O, w, \Theta_{type}^{timelapse})$  are calculated:

$$K = |\{type_{\mathbf{N}_1^{l+1}}, \dots, type_{\mathbf{N}_n^{l+1}}\} \cup \{type_{\mathbf{N}_{n+1}^{l+1}}, \dots, type_{\mathbf{N}_m^{l+1}}\}| \quad (6.15)$$

$K$  is the cardinality of uniques *type* of the nodes.

$$\mu_i = \frac{\sum_{p=1}^m Duration_{\mathbf{N}_p^{l+1}}(\mu) * \delta_{(type_{\mathbf{N}_p^{l+1}}, type_{\mathbf{N}_i^{l+1}})})}{\sum_{p=1}^m \delta_{(type_{\mathbf{N}_p^{l+1}}, type_{\mathbf{N}_i^{l+1}})})} \quad i \leq m \quad (6.16)$$

$$w_i = \frac{\sum_{p=1}^m Duration_{\mathbf{N}_p^{l+1}}(\mu) * \delta_{(type_{\mathbf{N}_p^{l+1}}, type_{\mathbf{N}_i^{l+1}})})}{\sum_{q=1}^m Duration_{\mathbf{N}_q^{l+1}}(\mu)} \quad i \leq m \quad (6.17)$$

where  $\delta$  is the Kronecker delta.

#### 6.4.4.2 The HAM recursive property

In the previous section, we define a hierarchical structure of nodes to model a target activity. The definition supposes a hierarchical structure of activities and sub-activities of two resolutions (i.e.  $l$  and  $l + 1$ ). In practice, we use three resolutions to learn an activity model. The resolutions correspond to the three resolutions described in the neighborhood of the input training prototypes. The previously defined calculations are sufficient to build models involving several levels of resolution. It is enough to calculate recursively the *attributes* and the *sub-attributes*, starting from a set of leaf nodes and considering that the father is the root node. The process can be repeated iteratively until reaching the real root of the tree of nodes at built of the training prototypes.

#### 6.4.5 Multiresolution-Histograms models (MH)

The MH approach can be seen as a brute force method which aims at capturing local and global information of the an activity in a set of multi-dimensional **histograms**. Each histogram is computed from one resolution level of the neighborhood of an input activity.

For an activity, at each of the 3 different resolutions of a training neighborhood, we compute 3 histograms:  $H_1, H_2, H_3$ . Where  $H_i$  captures the information of the PEs sequence of *DAs* at a resolution  $l^1$  inside the neighborhoods of the training prototypes of a target activity.

---

<sup>1</sup>For example,  $H_1$  corresponds to the resolution  $l$ ,  $H_2$  corresponds to resolution  $l + 1$  and  $H_3$  corresponds to resolution  $l + 2$ .

For different samples of the target activity, the *pre – clustering* stage is performed as described in the previous section. The histograms are computed from the resulting tree of nodes that represent the hierarchical structure of an activity and its sub-activities.

For each resolution  $l$  of a tree, a histogram  $H_i$  is computed.  $H_i$  is a histogram of 2 dimensions that characterizes the local and global agent motion described with the primitive events ( $PEs$ ) contained in the input  $DAs$  of all the nodes at the resolution  $l$ .

- A first histogram bin (*global feature*) is a  $type_{PE}$ <sup>1</sup> which is equivalent to one of the node type ( $Type_N$ ).
- A second bin (*local feature*) is an angle value  $\theta \in \{22.5, \dots, 337.5\}$  (8 possible values, which aims at characterizing the agent's local motion).

The histogram count is defined as the accumulation of:

$$H_i(type_{PE}, \theta) = \sum LDD_N(\theta) \quad : \quad \{\forall N \mid type_N = type_{PE}\} \quad (6.18)$$

Where  $LDD$  is the *Local Dynamic Distribution* attribute of the nodes of  $type_N$  (see 6.11), and the nodes  $N$  are at a resolution  $l$  correspondent to  $H_i$ .

The figure 6.16 illustrates an example of a histogram ( $H_1$ ) learned using a set of  $PEs$  contained in an input prototype neighborhood. The histogram  $H_i$  is computed at a resolution  $l$  which in this case is represented as a resolution of Level  $10^2$ .

There are two main advantages of modeling activities based on histograms. 1) Both, global and local person motions are packaged together in an unique histogram. An off the shelf histogram similarity measurement can characterize the distance between different activities. 2) The structure of the model is general and flexible, a new feature can be encoded in a histogram by simply adding a new dimension and computing the count as the relationship between all dimensions.

The drawback of this method is that it requires the set up of numerous parameters (i.e. add weights to each feature) to measure the similarity 1:1 between models. Also, due to the the simplicity of the model, all nodes of a resolution  $l$  are all characterized together without considering the hierarchical links with the sub-nodes at a finer resolution  $l + 1$ .

<sup>1</sup>Recall that the  $type_{PE}$  describes a global transition of the agent between two scene regions.

<sup>2</sup>Recall that the resolution  $l$  is corresponds to a *Level n* where  $n$  describes the number of scene regions of a topology. Then, Level 10 denotes that the  $PEs$  are computed using a topology of 10 regions.

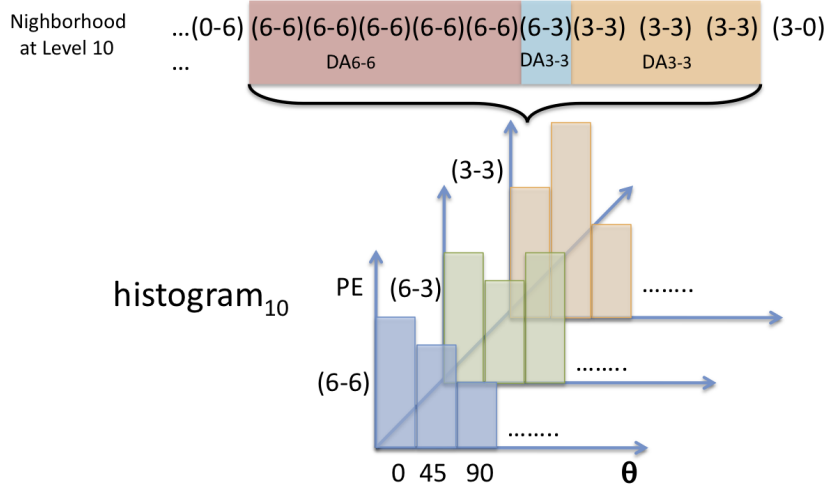


Figure 6.16: Example of an histogram  $H_{10}$ . Note that the real model contains 3 of these histograms.

## 6.5 Activity Similarity

In this section we define general measurements of the similarity between activity models. We describe a 1:1 comparison between two activity models. Due to the previous definitions of 2 procedures to model activities: HAM and MH, we define 2 methods to compute the similarity between activity models.

The selection of a similarity metric depends on the application of the framework where user preferences need to be considered. For example, the user can consider more important to stress the similarity of the duration of an activity over the local motion (i.e. *LocalDynamics*). Following, we explain the metrics proposed to compute the similarity between activity models. These metrics are used later to recognize activities automatically in unseen videos.

### 6.5.1 The similarity of HAM - Hierarchical Activity Models

We define a metric that computes a similarity *score* between two activity models in a recursive manner. The recursive procedure assumes that the root node of the models to compare have the same *type*. Let  $\mathbf{N}_A^1$  and  $\mathbf{N}_B^1$  be the root nodes of two HAMs. Let  $\{\mathbf{N}_{A_1}^{1+1}, \dots, \mathbf{N}_{A_n}^{1+1}\}$  and  $\{\mathbf{N}_{B_1}^{1+1}, \dots, \mathbf{N}_{B_m}^{1+1}\}$  be the sub-nodes linked to  $\mathbf{N}_A^1$  and  $\mathbf{N}_B^1$  respectively.

To compute the *score* we define a recursive function *Compare* as follows:

$$score = Compare_{\mathbf{N}_A^1, \mathbf{N}_B^1} \quad (6.19)$$

$$Compare_{\mathbf{N}_A^1, \mathbf{N}_B^1} = \sum \begin{cases} 1 - \Delta_1(LDD_{\mathbf{N}_A^1}, LDD_{\mathbf{N}_B^1}) \\ 1 - \Delta_2(Instances_{\mathbf{N}_A^1}, Instances_{\mathbf{N}_B^1}) \\ 1 - \Delta_3(Duration_{\mathbf{N}_A^1}, Duration_{\mathbf{N}_B^1}) \\ 1 - \Delta_4(mixture_{\mathbf{N}_A^1}(w), mixture_{\mathbf{N}_B^1}(w)) \\ 1 - \Delta_5(timelapse_{\mathbf{N}_A^1}(w), timelapse_{\mathbf{N}_B^1}(w)) \end{cases} + Recursion_{\mathbf{N}_A^1, \mathbf{N}_B^1}$$

$$Recursion_{\mathbf{N}_A^1, \mathbf{N}_B^1} = \sum_{i,j=1}^{m,n} \delta(\cdot) * Compare(\mathbf{N}_{A_i}^{l+1}, \mathbf{N}_{B_j}^{l+1})$$

$$\delta(\cdot) = \delta_{(type_{\mathbf{N}_{A_i}^{l+1}}, type_{\mathbf{N}_{B_j}^{l+1}})}$$

where  $\Delta_i$  is a normalized euclidean distance and  $w$  is the mixture wight of the sub-nodes<sup>1</sup>; the  $\delta(\cdot)$  Kronecker delta assuring that the pair of nodes to compare share the same *type*. *Recursion* denotes the recursive step which compares the sub-nodes in case of existence. When a pair of nodes to compare are leaves of the HAMs the *Recursion* is skipped.

### 6.5.2 The similarity of MH - Multiresolution Histograms model

An activity is modeled by  $n$  (e.g. 3) 2-dimensional histograms, where each histogram corresponds to a hierarchical resolution  $l$ . We propose the usage of a distance that measures similarity of all (local and global) activity descriptors at once. Since the activities are modeled by a set of 2-dimensional histograms, we use the Earth Movers distance (EMD) [Levina 2001] to compare histograms.

We compute a *score* that measures the similarity of the activities at the different resolutions (capturing the sub-activity similarity).

Supposing three resolutions ( $n = 3$ ) as it is defined in the section 6.4.5. The *score* is computed by comparing the  $n$  histograms of one model: *Activity* ( $H_l, H_{l+1}, H_{l+2}$ ) with the 3 histograms of a second model: *Activity'* ( $H'_l, H'_{l+1}, H'_{l+2}$ ) adding the result values:

$$score = \sum_{i=l}^{l+2} EMD(H_i, H'_i) \quad (6.20)$$

---

<sup>1</sup>The *mixture* and *timelapse* are abbreviations of *mixture<sub>subactivity</sub>* and *timelapse<sub>subactivity</sub>* sub-attributes of a node.

## 6.6 Activity Recognition

For a new unseen video dataset, we aim at recognizing target activities in an unsupervised way. The task is achieved by measuring the similarity between learnt activity models (i.e. template) and candidate activity models (i.e. test) where the candidate activity models are built automatically.

For a *new*<sup>1</sup> video, the recognition process, starts by searching and building models of candidate activities. The candidates are *new* discovered activities which correspond spatially with a target activity (i.e. the root node has the same *type*). Then, the recognition process measures the similarity between a target activity model and the candidate models to compute a similarity *score* which allows the deterministic activity recognition.

Suppose  $Activity_{model}$ <sup>2</sup> is the learnt model of a target activity, and  $Scene_{model}$  is the training scene model used for learning the  $Activity_{model}$ .

The recognition of a target activity in a *new* video is an unsupervised procedure computed in five stages:

**First**, the *new* sequence of perceptual features chunks (*PFCs*) is computed.

**Second**, the *new* Primitive Event sequences<sup>3</sup> are computed using the previously learnt  $Scene_{model}$ . This way, the *PEs* of the new video match spatially ( $type_{PE}$ ) with the *PEs* used for learning the target activity.

**Third**, the activity discovery process is performed with the *new PEs*. From the *new* computed sequence of test *DAs*, those that contain the same *type* (color) as the one used for labeling the target activity are called candidate activities. For example, in figure 6.11 we label  $DA_{4-4}$  to model "Preparing Meal". For the *new* video, all  $DA_{4-4}$  appearing at the resolution of  $Level_5$  are selected.

**Fourth**, the algorithm builds a model ( $Activity_{model}^{candidate}$ ) for each candidate activity selected in the previous step.

**Fifth**, due to the previous steps, the models:  $Activity_{model}$  and  $Activity_{model}^{candidate}$  have a global spatial correspondence. To assure that both activities are semantically the same, we compute a *score* between the models using a similarity measurement (defined in previous sections). Finally, the computed score is thresholded to achieve a deterministic activity recognition. The threshold can be configured manually or

<sup>1</sup>*new* denotes unseen or testing data.

<sup>2</sup>Usually the  $Activity_{model}$  learnt from training data is a reinforced model.

<sup>3</sup>3 sequences are computed, each sequence corresponds to a *Topology* resolution of the training  $Scene_{model}$ .

learnt by computing the average *score* of the training prototypes used to build an *Activity<sub>model</sub>*.

The activity recognition process can work in an on-line<sup>1</sup> fashion. The procedure of seeking for candidate activities speeds up the whole recognition process. It can be seen as a coarse-recognition stage which considers the global position as the main activity descriptor. The procedure is refined by computing a similarity *score* which takes into account finer activity descriptors for the final recognition. In our off-line experiments the algorithm is able to recognize all instances of any target activity used for evaluation in less than 5 seconds (the *new* data to analyze is a video of 4 hours).

---

<sup>1</sup>The recognition has a small delay corresponding to the inevitable completion of a characterizing part of the target activity in the *new* video.

# Evaluation

---

This chapter aims at demonstrating the efficacy of the whole chain of algorithms proposed in this thesis to understand long term activities.

We divide the evaluation of this work in two parts:

1. **Recognition:** We evaluate the performance of the framework for seeking and recognizing targeted activities in unseen videos. For example, once an activity (e.g. "Preparing Meal") has been learnt from a training dataset we evaluate the recognition of all of the instances of the same activity appearing in a new video.

To achieve the recognition, the model of a target activity is learnt from few training prototypes<sup>1</sup> selected by the user. The evaluation of the recognition procedure is sub-divided into 3 system configurations:

- (a) **Scene & Sensor:** Three scenes are evaluated. Also, we variate the acquisition sensor and compute Perceptual Feature Chunks of monocular cameras (2D -sensor) and of RGB-Depth map sensor (3D -sensor) depending on the scene.
  - (b) **Resolutions:** We compare the hierarchical activity representation (multi-resolution), with the same techniques applied to a single-resolution activity representation.
  - (c) **Modeling technique:** We learn models of target activities using Hierarchical Activity Models (HAM) and Multiresolution-Histogram models (MH). We compare the recognition results using the two types of modeling techniques for different datasets.
2. **Ranking:** Measures the performance of the framework for helping the user to find similar instances of a selected activity. The user selects a discovered activity and the algorithm seeks similar activities and ranks them. The process is completely unsupervised and can be considered as a video understanding tool for statistical analysis.

---

<sup>1</sup>A reinforced model.

## 7.1 Evaluation Scenes

The proposed activity understanding method is applied to different system configurations. A first configuration is to change the data source. We divide the evaluation scenes depending in the sensor type. We use two types of sensors: a classical video camera (such as a webcam) and a depth-map camera. Therefore, we have two type of scenes:

1. Monocular Video Camera Scenes
2. Depth Camera Scenes

In the following subsections we explain the composition of the scenes (i.e. fixed objects, regions); the registered videos; the participants (also called from a perceptual perspective: actors, agents, mobiles, or objects); and the type of targeted activities for evaluating the activity recognition method.

### 7.1.1 Monocular Video Camera Scenes -MCS-

We capture and analyze a collection of videos (datasets) corresponding to two different *MCSs*. The datasets are named HOME-CARE and HOSPITAL, their name corresponds to the type of applications they are intended to serve. The two scenes are described as follows:

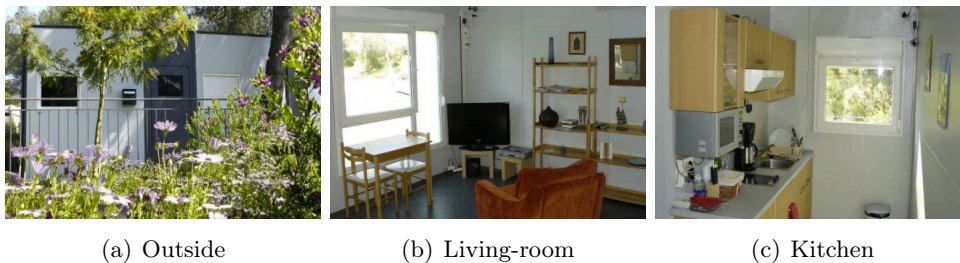


Figure 7.1: The images are from the Gerhome laboratory, which we use to evaluate our method for HOME-CARE applications.

- HOME-CARE, *the description*:

To develop and test the proposed methods applied to home-care applications, we have set up an experimental laboratory under the E.U. project Gerhome<sup>1</sup>. The laboratory is located at the Scientific Center of Technical Buildings at Sophia Antipolis in France. The laboratory looks like a typical apartment of an elderly person:  $42m^2$  with an entrance, a living-room, a bedroom, a

<sup>1</sup><http://gerhome.cstb.fr>





Figure 7.2: Coarse description of the important regions in the scenes corresponding to the (a) HOSPITAL and (b) HOME-CARE datasets.

bathroom, and a kitchen. The kitchen is equipped with a stove, a microwave, a fridge, cupboards and drawers. The living-room contains a table, an armchair, a TV and chairs. The apartment is equipped with: 4 monocular cameras; 12 contact sensors (e.g. "open/closed door"); 4 pressure sensors (i.e. "on the chair"). From the setup, we use only minimal visual information (a video camera), but we compare with multi-modal approaches in later sections. Some images the of the Gerhome laboratory environment are displayed in the figure 7.1. The used camera point of view and the meaningful scene regions are described in the figure 7.2 (a).

- HOSPITAL, *the description:*

The scene is built under the framework of the French ANR project SweetHome. The scene is an equipped room designed to understand the triggers, the impact of treatments, and the evolution of elders diseases. The overall aim of the project is to develop a technological approach for behavioral assessment in early and moderate stage of Alzheimer's Disease. The room has  $32 m^2$ , and it is located at the Memory Centre University hospital (Geriatric pole). It is equipped with various objects for the agents interaction such as: a table and a chair for sitting and reading; reading material; a TV; a coffee corner; a chair to perform exercises. The camera point of view and some objects of the scene are illustrated in the figure 7.2 (b).

#### 7.1.1.1 The -MCS- videos

Each video contains a single agent and is recorded using a monocular video camera of  $640 \times 480$  pixels of resolution, streaming at 8 fps.

- HOME-CARE, *the videos:*

The dataset contains 7 elderly people performing non-guided activities in an apartment. In total **24 hours** of video are **evaluated**. The agents are aged from 60 to 85 years. The videos are captures from 10 am. to 2 pm. The participants are recruited by advertisements for a research study, and they are instructed to behave as natural as possible. The participants are alone during the observation. The videos are available under demand<sup>2</sup>.

After the observations the volunteers are interviewed separately, an study of the questionnaires indicates that the sensors do not impact in their everyday behavior. The fact that the instrumented home is not the volunteers real home have some effect in their natural behavior. For example, in some cases the volunteers interact with several kitchen doors before executing an activity. The situation is reported as problematic in previous work [Zouba 2010] where the activity models are defined manually, which is not our case.

- HOSPITAL, *the videos*:

The HOSPITAL dataset contains 4 videos of patients performing guided and non guided activities in the room (**3 hours evaluated**). Some of the patients are diagnosed with early stages of Alzheimer. The dataset is currently being used to study Alzheimer's disease symptoms [Romdhane 2010]. The patients are instructed to perform two exercices:

1. Follow a protocol of guided activities defined by doctors. The protocol last about 20 minutes, and is surveilled by a nurse outside the camera scope.
2. Act freely in the room during 40 minutes, in this case the person is completely alone.

### 7.1.2 Depth Camera Scene -DCS-

The scene and recordings are digitalized using a RGB-D camera. We capture and evaluate different datasets for different configurations. Our *DCS* is named CASA standing for "Common Apartment for System Analysis" and its described as follows.

- CASA, *the description*:

The CASA scene is a real apartment located in the city of Nice, France. The apartment has 50  $m^2$  of surface and the perceptive area includes regions such as a table, a kitchen and a bathroom. Each region contains typical fixed objects such as a fridge, tables, a stove, shelves, etc. Other type of objects are mobile such as the kitchen utensils, the shelf doors, the chairs, etc. The scene is full of reflexions (due to the window glass); partial and complete occlusions; and illumination variations. These situations make of the scene an ideal place for field-testing the proposed system. The interesting regions and objects are illustrated in the figure 7.3. In the apartment two datasets are recorded:

<sup>2</sup><http://www-sop.inria.fr/members/Francois.Bremond/topicsText/GerhomeProject>

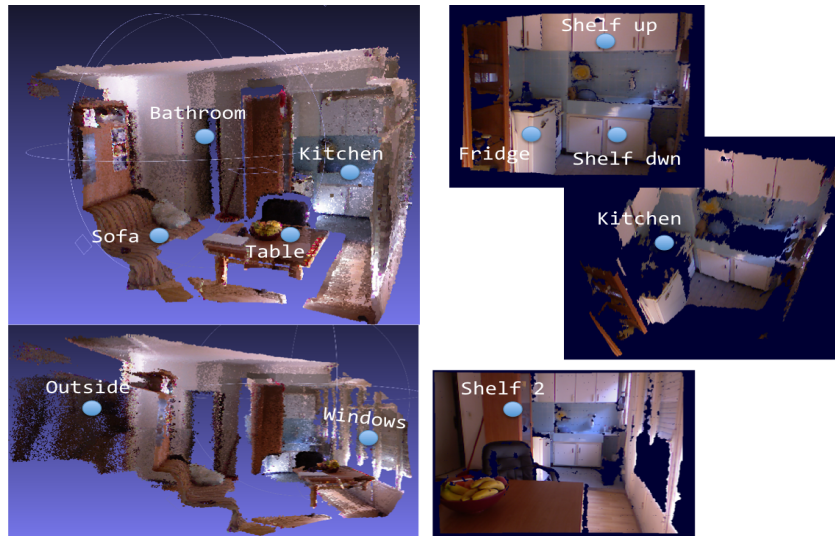


Figure 7.3: 3D characterization of the CASA scene, built with a VSLAM to describe the real scene depths. The labeled dots represent interesting regions and objects for the agents interaction.

*single-CASA* and *multi-CASA*. The datasets contain information of multiple objects tracked at the same time (i.e. person, joints). The datasets are named (or prefixed) according to the number of agents present in each one.

### 7.1.3 The *-DCS-* datasets

The recordings are captured at 30 fps. The depth camera resolution is of 640 x 480 px. and 16 meters of depth. The RGB resolution is 640x480 (only used for illustrations). For the *DCS* datasets, the local dynamics are not computed, the real 3D global information is accurate enough to recognize most of the activities. Also, in one dataset the finer descriptions obtained with the local dynamics are replaced with the tracking of the person joints.

CASA represents a real world situation because the agents are well familiarized with the environment. The agents know the location of everything (e.g. the cooking utensils) in advance.

- *single-CASA*, the dataset:

The *single-CASA* dataset is composed of 2 depth videos with one agent, 30 years old, performing activities during 10 minutes at 2 p.m. in the CASA apartment. The particularity of this dataset is the availability of the skeleton joint tracks (e.g. head, hands, etc.). The dataset is composed of several objects: 15 joints and the center of the person are tracked at each frame. We use independently each object to recognize composite (or concurrent) activities

recognition. The composite activities are concurrences of different activities at the same time, which merged describe a richer semantic. For example, the moving hands over a table is recognized as "Interaction with the table" which can be merged with an activity "Sitting on a chair" detected using the center of the person silhouette to build the composite activity: "Working+Sitting". An example of the type of tracked objects for *single-CASA* is illustrated in the figure 7.4 (a).

- *multi-CASA*, the dataset:

The *multi-CASA* dataset is composed of two videos of 3 hours of length each (about 30 Gb of RGB storage). In each video, two persons (out of 3 persons in total, a person is the same one in 2 datasets) are present most of the time. The persons are aged between 24 and 30 years old (younger than in previous datasets). The recording takes place between 21:00h and 24:00h of 2 different days. An example of the multiple persons tracked in the dataset is illustrated in the figure 7.4 (b).



(a) Silhouette and Joints (*single-CASA*)      (b) Two Silhouettes (*multi-CASA*)

Figure 7.4: Example of the tracked objects in the two *-DCS-* datasets. (a) Represents *single-CASA* dataset, where a single person is tracked (blue silhouette). Also, the joints: head, hands, neck, waist points, and other articulations are marked with a red circle. (b) Represents the information of the *multi-CASA* dataset, where 2 agents are tracked at all time.

#### 7.1.4 The dataset summary and highlights

In the previous sections we presented 4 different datasets. The proposed evaluation datasets variate in several aspects:

- *Sensor*: The digitalization of the data is obtained from RGB and Depth cameras working at different resolutions and fps. The trajectories of the objects contain numerous perceptual errors such as occlusions and confused IDs, which

are handled by this work. We aim at showing that the source of the data is not really important as long as an object can be tracked and therefore the proposed method is sensor independent. The method could be applied on other sources such as GPS trajectories for a macroscopic activity recognition system (i.e. vehicles such as trucks, UAVs, tracked during days can get semantical activity descriptions).

- *Agents*: We propose datasets of seniors (60-85 ys.), young-adults (24-30 ys.). The datasets include similar activities performed in different ways, the variations are due to the difference of age principally. For example, in most of the cases, the seniors are more structured to perform an activity. Also, the speed of performing an activity changes due to the age.
- *Configuration Flexibility*: We propose different configurations. For the *MCSs* the local dynamics are computed to have finer activity characterizations. In *DCSs* we avoid the computation of local dynamics but we add real 3D point representation. Also, the combination of different objects is a possible configuration to achieve composite activities. The flexibility is such that in future sections we will demonstrate that the inclusion (or not) of descriptors (i.e. local dynamics) to our models, is not a blocking or restrictive aspect.
- *Applications*: We evaluate the method for assisted living and medical applications. With a similar configuration, the method could be applied to security purposes. It is not in the scope of this work to try all possible configurations for different applications, clearly we focus on activity recognition at home.

## 7.2 The target activities

To evaluate the accuracy of the proposed approach we target a set of activities which we aim at recognizing.

*Which and how to select target activities*: The selection of activities for evaluation should not depend on the perceptual features but on an application requirement. For example, for security, the early recognition of a person intention of producing a crime is a requirement. Nevertheless, it makes no sense to evaluate our method to recognize the intentions of a person. The target of the activities is a trade off between the perceptual features used and the real application requirements.

For each dataset, we target a set of activities. The activities can be discovered at the same scene resolution to avoid extensive set-up explanations. In most of the cases, we select activities which their recognition is required (e.g. physical exercises [Romdhane 2010]). These activities are usually the typical interesting activities in the home-care application literature, which makes possible the comparison of our method with others.

Other important aspect of the target activities is the challenges they present. For example, the recognition of the activity "in the kitchen" evaluates mostly the performance of the tracking method and its recognition is less challenging than a finer activity such as "Preparing Meal" (which is composed of particular complex motions). In most cases, the recognition challenges are two:

- *Discrimination power*: The correct recognition and discrimination of different but similar activities occurring at the same spatial location. These activities are complex to be described perceptually.
- *Handling low-level errors*: The recognition under typical lower-level computer vision problems such as: miss detected objects, occlusions, light changes, object ID confusions, background integration, lack of motion, etc.

Following, we describe the target activities ordered by dataset and the inherent challenges to recognize them. In most cases we present illustrations of the activities and its local motion to represent its flow of movements. Also, we show some possible confusing situations, where most of the recognition systems usually fail.

1. HOME-CARE, *the activities*:

- "Eating": Examples of persons eating are illustrated in the figure 7.6 -from (a) to (d)-. The challenge is the potential confusions with activities at similar spatial 2D locations such as "Preparing Meal" illustrated in fig. 7.6 -from (e) to (h)- and with other activities such as the ones corresponding to the items (i) to (l) of the same figure.
- "Preparing Meal": The activity is composed of several movements in the kitchen, mostly defined by the person's interaction with the kitchen table, refrigerator, and sink. The figure 7.6 -from (e) to (h)- illustrates examples of instances of the activity. Possible confusions are "Eating" and other activities taking place at the kitchen such as "Washing the dishes".
- "Standing at Armchair": The activity usually takes place when the person stands in front of a TV which is outside of the camera FOV (see fig. 7.7 (a)). The activity is challenging due to its short duration and that can be confused with "Reading/sitting at the armchair" illustrated in fig. 7.7 (b and c).
- "Sitting at the eating place": The activity takes place on one of the chairs of the table. Examples of the activity are illustrated in 7.6 (k and l).

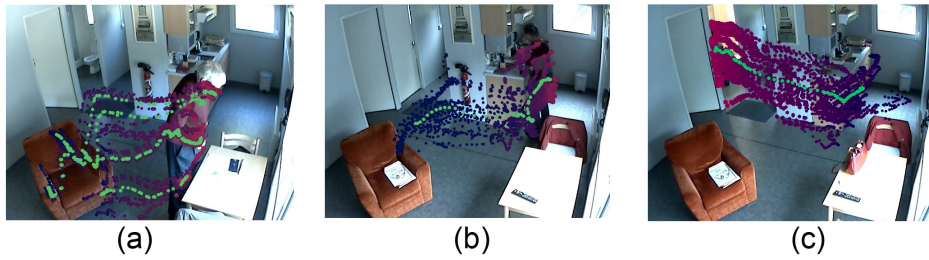


Figure 7.5: Examples HOME-CARE activities characterized by the change of scene region. (a) "Armchair to Table", (b) "Armchair to Kitchen", (c) "Kitchen to bathroom". The activities described in (a) and (b) are potential confusions due to the similar ending scene region.

- "Reading in the Armchair": The activity is illustrated in the figure 7.7 (b) - usually is a long duration activity. The motion of the person arm while reading can be detected thanks to the local motion descriptors. Therefore we aim at differentiating "Reading in the Armchair" of other activities at the same region such as: "Sitting in the armchair" or "Standing at Armchair". The purpose of recognizing this activity is to provide insights of the importance of the local dynamics to improve recognition by detecting finer activities.
- "Inside the bathroom": Activity that takes place in the bathroom. The challenge of the activity is related to low-quality perceptual features (high noise) that can be placed at the bathroom area due to the distance to the camera.
- "Armchair to Table": Is an activity characterized by the displacement of the person between two scene regions. The figure 7.5 (a) illustrates an instance of the activity. The problem is that "Armchair to Table" can be confused with "Armchair to Kitchen" which is illustrated in fig. 7.5 (b).
- "Armchair to Kitchen": Describes the displacement of the person among two regions: the armchair and kitchen. An example is displayed in fig. 7.5 (b). The activity can be confused with "Armchair to Table" due to the similar perceptual ending region.

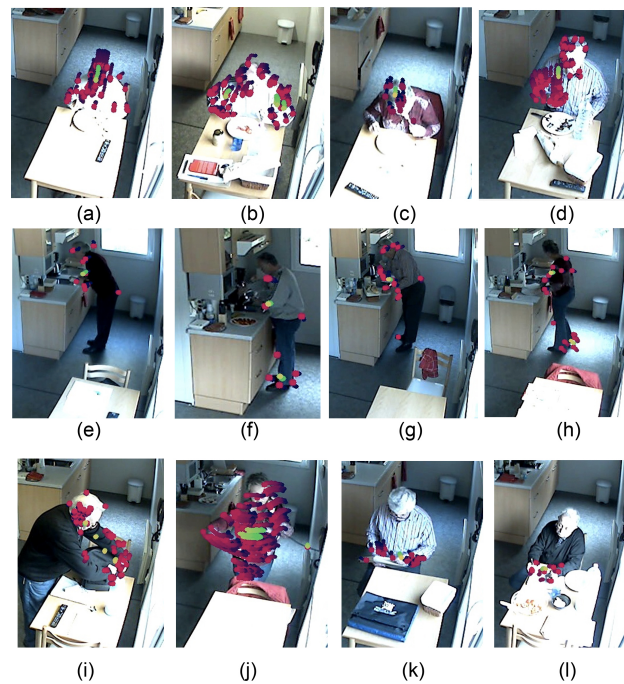


Figure 7.6: Example of HOME-CARE activities and potential activity recognition confusions. The activity "Eating" is displayed from (a) to (d). The activity "Preparing Meal" is illustrated from (e) to (h). Other activities: (i) "Working at the table", (j) "Using the wall phone", (k) and (l) "Sitting at the eating place". All activities share a similar spatial 2D location due to the camera perspective. Such a thing means that they can be confused at the recognition step. The challenge for the recognition method is to differentiate them correctly.



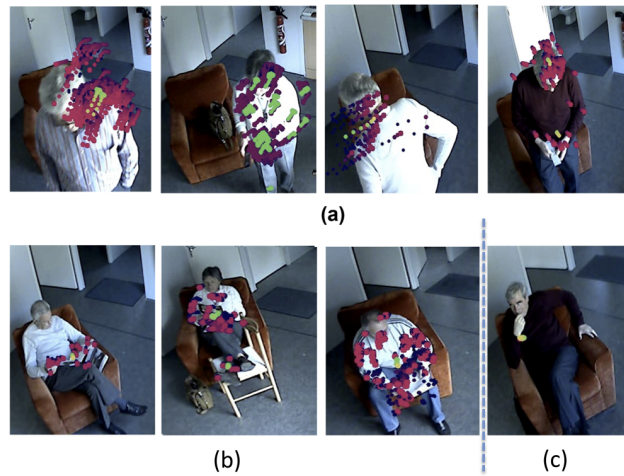


Figure 7.7: Example of HOMECARE activities around the armchair area. (a) "Standing in Armchair", (b) "Reading in Armchair", "Sitting at Armchair". The 3 activities are potential confusions due to the 2D point of view.

## 2. HOSPITAL, *the activities*:

- "Balance": Is an exercise designed to measure the person stability by standing on one foot at a time. An example is illustrated in figure 7.8 (a). The activity occurs at the chair area (see fig. 7.2) and can be confused with other activities at the same place such as "Up/Down".
- "Up/Down": Is an exercise where the person stands up and sits down without stopping. The exercise aims at measuring the speed of the person to react to these opposite movements. An example is illustrated in the figure 7.8 (b). The activity can be confused with others in the area.
- "Reading at the table": The activity is not part of the protocol, but is performed by many patients. An illustration is displayed in fig. 7.8 (c).
- "At the Computer ": The person uses the computer or stays sitting at such location. An example is illustrated in the figure 7.8 (d).
- "Preparing Coffee": The person uses the coffee machine at the back of the room (see fig. 7.2) to prepare a cup of coffee. An example can be found in figure 7.8 (e).
- "Exercice 1 ": An exercise that is characterized by the change of scene regions from the chair to the mark (see 7.2), The exercise is designed to

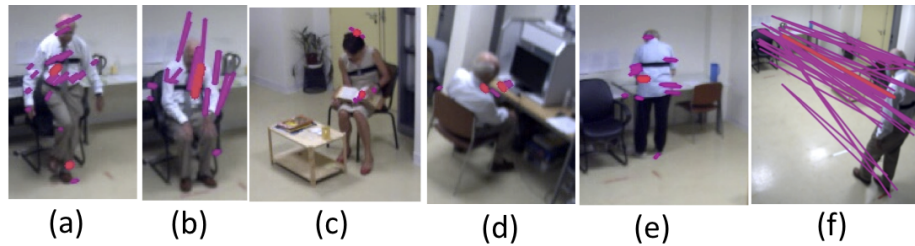


Figure 7.8: Example of HOSPITAL target activities. (a) "Balance", (b) "Up/Down", (c) "Reading at the table", (d) "At the computer", (e) "Preparing coffee", (f) "Exercise 1".

measure the walking speed of the patient. An example is illustrated in the figure 7.8 (f).

- "Exercice 2 ": Is a walking exercise, which is similar to "Exercise 1" but in the opposite sense (from the mark to the chair).

### 3. *Single-CASA*, the activities:

We target 5 activities. Two activities are mostly linked with the position of the person (prefixed "At."); One activity is characterized by the posture and location ("Sitting.near"). Finally, two activities are composed of the combination of a posture or position and other activity recognized at the same instance of time.

- "Sitting.Near": The person is sitting on a chair, with a relaxed posture, and facing the table -see figure 7.9 (a)-. It is recognized using the center of the silhouette. It can be confused with other activities at the same location -see figure 7.9 (f)-.
- "At.Kitchen": The person is at the kitchen, recognized using the center of the silhouette -see figure 7.9 (b)-.
- "At.Bathroom": The person is at the bathroom, recognized using the center of the silhouette -see figure 7.9 (c)-.
- "Working + Sitting.near": Recognized when the person is "Sitting.Near" (silhouette) and the activity "Working" is recognized using the hands joints -see figure 7.9 (d)-.

- "Interaction.Fridge": Recognized when the person is performing "At.Kitchen" (silhouette) and when the activity "Hands.on.Fridge" is recognized using the hand joints -see figure 7.9 (e)-.

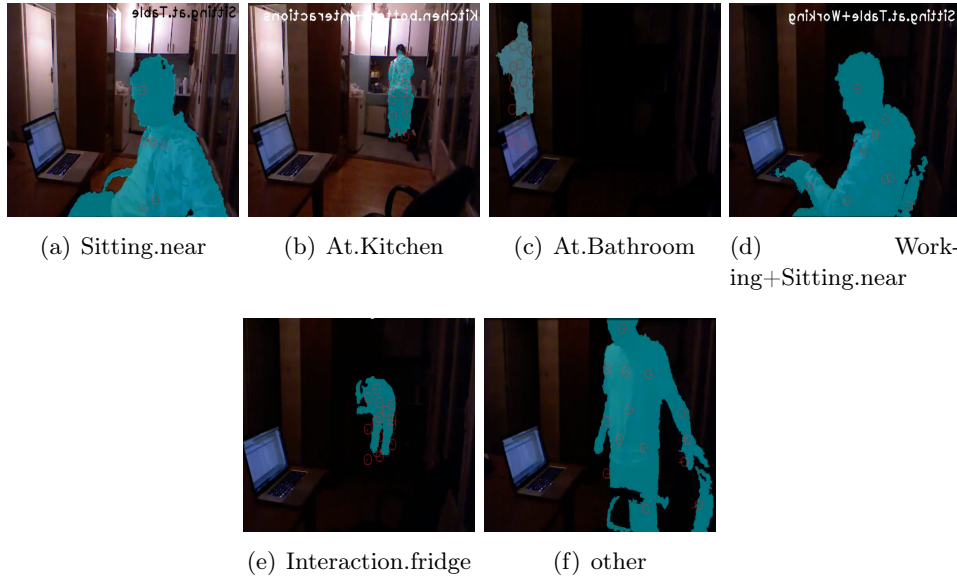


Figure 7.9: Examples of six activities of the *Single-CASA* dataset (a). The colored silhouette is a tracked person and the red circles are the tracked joints (e.g. head). The activities (b)(c) can be confused and, the activities (a)(d)(f) as well.

4. *Multi-CASA*, the activities: We target 4 activities. Two activities occur in the Kitchen area and other two in the table area. The activities can be easily confused due to the numerous vision errors.

- "Sitting.near": The person is sitting in a chair near to the camera, and facing the table. It is recognized using the center of the silhouette. Examples of the activity are illustrated in the figure 7.10 (a). In the other images of the same figure different variations of the activity are shown. Also, possible errors such as ID confusions -see fig. 7.10 (c) and (d)-.
- "Sitting.far": The person is sitting in a chair near to the camera, and facing the table. It is recognized using the center of the silhouette. Examples of the activity are illustrated in the figure 7.10 (i). In the other images of the same figure different variations of the activity are shown. Also, possible errors such as object miss detections -see fig. 7.10 (g)-.

- "Preparing.Meal": The person prepares meal when several interactions with the kitchen equipment (e.g. stove) occur. Examples of the activity are illustrated in the figure 7.11. Also, we illustrate examples of errors that could lead to confusions in fig. 7.11 (b to d).
- "Interacting.TrashCan": Recognized when a person uses the trash can. An example is illustrated in fig. 7.11 (a) -white silhouette-.



Figure 7.10: Examples of "Sitting.near" and "Sitting.far", including different postures and errors such as ID changes and miss recognized objects.

### 7.3 Recognition

The recognition of activities is directly related to the activity discovery method (i.e. a failure of the discovery method implies a failure of the recognition method). **The evaluation of the activity recognition method reflects the accuracy**

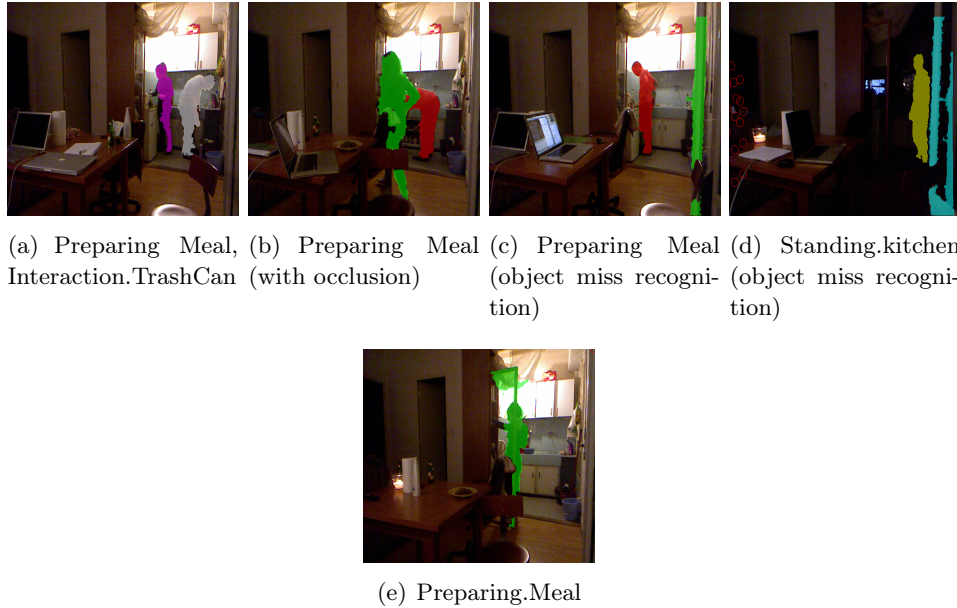


Figure 7.11: Examples of activities of the *multi-CASA* dataset. The colored silhouette is supposed to be a tracked person where the color is the ID.

of the discovery procedure.

### 7.3.1 Learning activity models, evaluation procedure

We evaluate the activity recognition method using cross validation technique. The evaluation is performed by learning the scene and activity models from the training videos and by recognizing activities in a test video. The procedure can be described in 3 stages:

1. We learn the scene model and activity models from the training persons. For example, in HOME-CARE which is composed of 6 videos, to recognize activities of person G (test), the scene and activity models are learned using the videos of the training persons A,B,C,D,E,F as it is illustrated in fig. 7.12 (a).
2. The activity recognition is performed in a new dataset (person G) using as input learned activity and scene models -see fig. 7.12 (b)-.The procedure returns a set of intervals of time where the target activity is located as it is illustrated in fig. 7.12 (c).
3. A manually annotated ground truth (**GT**) is used to evaluate the whole recognition system. The **GT** describes the intervals of time when an activity begins

and ends. There exist one **GT** for each test video. The recognized activity instances are compared with the **GT** instances as it is illustrated in the figure 7.12 (d). The comparison is achieved calculating a set of performance measures which are described in the next section.

The stages 1) 2) and 3) described above, represent an experiment over a test video. For the evaluation of the system, one experiment is performed for each test video accumulating the performance results (e.g. for HOME-CARE 6 experiments are performed).

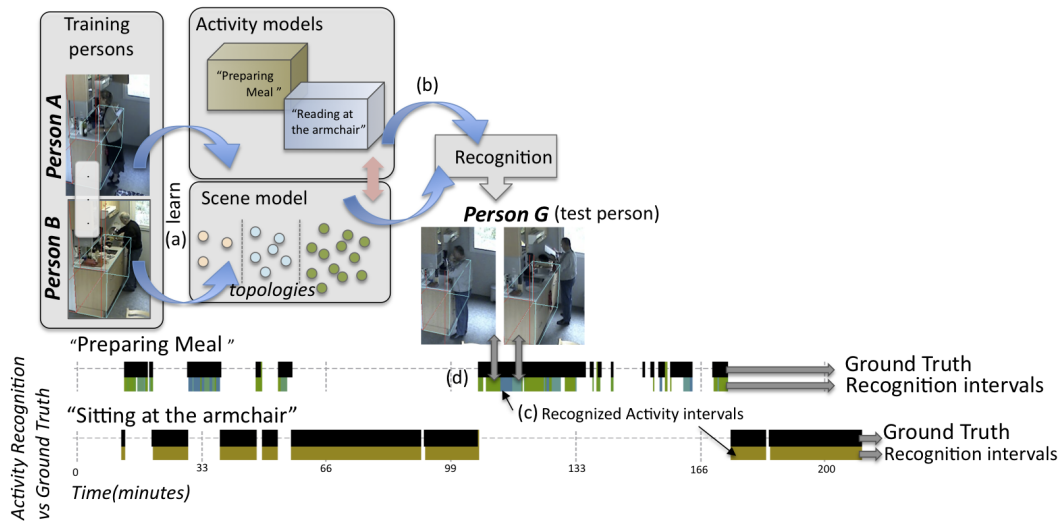


Figure 7.12: Example of the stages a) b) c) d) involved in the evaluation procedure.

### 7.3.2 Performance measurements:

For each video, a manually labeled **GT** describes the intervals of time when an activity begins and ends. The Activity Recognition method returns the intervals of time where an activity is recognized. Each recognized activity instance is compared to the **GT** and a set of measures are extracted. The figure 7.13 describes intuitively the variables used to calculate the performance measures which are defined as follows:

True Positive (TP): Number of activity instances correctly recognized.

$$TP = \# \{true\ positive_i\} \quad (7.1)$$

False Positive (FP): Number of recognized instances not appearing in the **GT**.

$$FP = \# \{false\ positive_i\} \quad (7.2)$$

False Negative (FN): Number of instances appearing in the **GT** but not recognized.

$$FN = \# \{false\ negative_i\} \quad (7.3)$$

Recognition Time (RT): Percentage of time the activity is correctly recognized, over the **GT** duration of the activity.

$$RT = \frac{\sum_p recognition\ time_p}{\sum_q ground\ truth\ time_q} \quad (7.4)$$

False Recognition Time (FT): Percentage of time the activity is wrongly recognized (i.e. while it is not occurring in the **GT**), over the time the activity is recognized.

$$FT = \frac{\sum_p false\ recognition\ time_p}{video\ length - \sum_q ground\ truth\ time_q} \quad (7.5)$$

The above defined measures can evaluate the performance of the recognition system in two ways: 1) The hit/miss activity instances are characterized by the TP, FP, FN, in a classical way. 2) The temporal length of the activity instances are characterized by the RT and FT measures. The combination of both provides good insights on the performance of the recognition process.

Other metrics can be used to compare different configurations of the system (e.g. the usage of single and multi resolution). Also the metrics can be used to compare the recognition results using other methods of the literature. These metrics are metrics over metrics, normalizing the amount of samples used to obtain the first metrics. The metrics are defined:

TPR: true positive rate (also called recall rate or sensitivity in some publications) measures the proportion of actual positives which are correctly identified as such, it is defined as:

$$TPR = TP/(TP + FN) \quad -\ higher\ is\ better- \quad (7.6)$$

FDR: false discovery rate, is analogous to the TPR, it is defined as:

$$FDR = FP/(FP + TP) \quad -\ lower\ is\ better- \quad (7.7)$$

PPV: positive predictive value (equivalent to precision), it is defined as:

$$PPV = TP/(TP + FP) \quad -\ higher\ is\ better- \quad (7.8)$$

The measures PPV and FDR are complementary, appearing both in the literature, we present both measurements in the evaluation.

### 7.3.3 Configuration

The most important parameter is the configuration of the multiple resolutions, which is the targetted number of clusters for each topology. We use the same

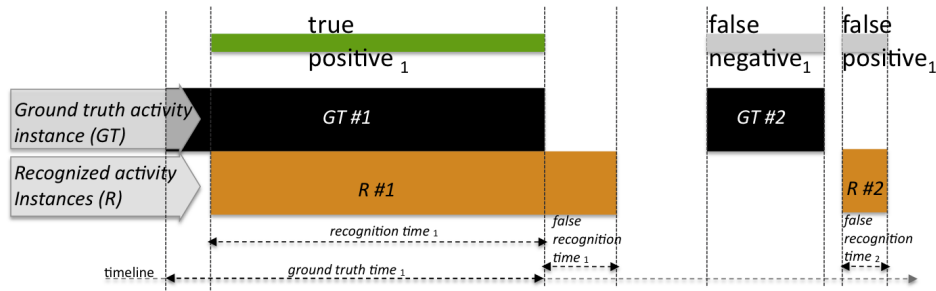


Figure 7.13: Example of the variables used to compute the performance measures extracted from the comparison of the recognized and the ground truth activity instances. A recognized activity can be TP, FN, FP instance or duration.

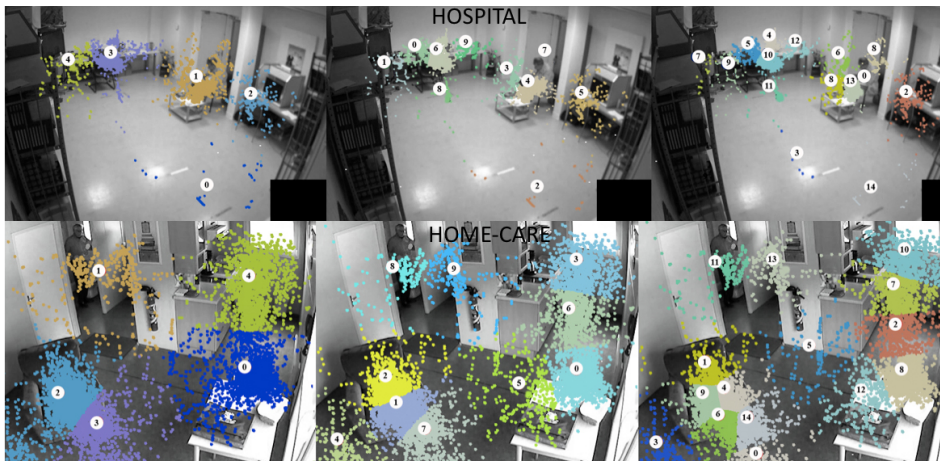


Figure 7.14: Learned topologies for the datasets: HOSPITAL (top) and HOME-CARE (bottom).

configuration 5, 10, and 15 clusters for the 3 scene topologies. Examples of the topologies are displayed in the figures 7.14 and 7.15.

Another important parameter is the usage of Perceptual Feature Chunks (PFCs) of fixed or dynamic lengths. For the *MCS* we use dynamics PFCs. For the *CASA* datasets the PFCs have a fixed length. In *single-CASA*, the PFCs have a length of 0.3 seconds. In *multi-CASA*, the PFCs have a length of 1.5 seconds.

Other parameters are as described along this thesis.

### 7.3.4 Recognition results, monocular camera dataset

Tables 7.1 and 7.2 display the recognition results in the HOSPITAL and HOME-CARE datasets respectively. The tables show the results using the Hierarchical Activity Models **HAM** and the Histogram Models **HM**.



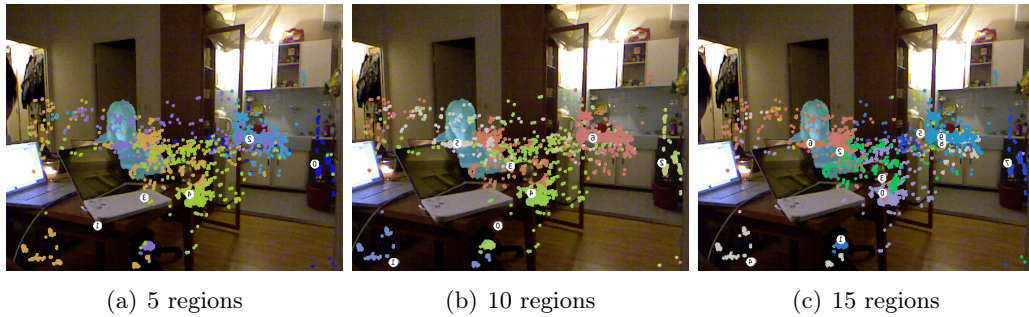


Figure 7.15: Example of the topologies of level (regions) 5,10, and 15, computed for the *multi-CASA* dataset.

In both datasets and for the two types of model, the recognition results show a high rate of True Positives (TP) and a low rate of False Negatives (FN). In the overall, the target activities are detected 99% of the time. Also, the false positive rate is low (lower than 1% in the overall). The evaluation of TP, FP and FN demonstrates that the system can accurately recognize most of the instances of the target activities with a very low rate of errors. The duration of the recognized activities is matching in all cases above the 80% of the ground truth activities, which means that the system can not only count the amount of activity instances but also have a pretty accurate description of their duration.

The occurrences of FN and FP can be explained. In most of the cases, the FP occur when the motion of the people is similar while doing different activities, meaning that the descriptors are not discriminative enough. The FN because of the lack of the person motion. The FP occurs because a person stops an activity without changing of place (i.e. at the end of Eating the person stays still for a while). The problem can be addressed by using a more refined topology (more clusters).

To illustrate the complexity of the recognized activities we display some results graphically in fig. 7.16. The examples show the recognition results of two videos of 1 and 4 hours of length each. The first video corresponds to the HOSPITAL dataset, and the second one to the HOME-CARE dataset. In fig. 7.16 the recognition true positives are marked (\*) colored segment where the length of the segment is the detected duration. The segments are located in the video timeline providing a representation of the real ordering and distribution of the occurrences of the target activities. In the image, we also display in red color the segments where the person is at the spatial location where a target activity takes place but the person is not performing such activity (potential confusion).

For example, the activities "Balance" and "Up/Down" take place at the same 2D position and due to the usage of local dynamics they are not confused

| <b>HAM</b>           |    |    |    |      |    |
|----------------------|----|----|----|------|----|
|                      | TP | FP | FN | RT   | FT |
| Balance              | 3  | 0  | 0  | 100% | 1% |
| Up/Down              | 3  | 0  | 0  | 100% | 4% |
| Reading at the table | 10 | 0  | 1  | 95%  | 2% |
| Preparing Coffe      | 7  | 1  | 0  | 88%  | 5% |
| At the Computer      | 6  | 1  | 0  | 91%  | 4% |
| Exercice 1           | 3  | 0  | 0  | 99%  | 2% |
| Exercice 2           | 3  | 0  | 0  | 99%  | 1% |

| <b>HM</b>            |    |    |    |      |    |
|----------------------|----|----|----|------|----|
|                      | TP | FP | FN | RT   | FT |
| Balance              | 3  | 0  | 0  | 100% | 1% |
| Up/Down              | 3  | 0  | 0  | 100% | 4% |
| Reading at the table | 10 | 1  | 1  | 95%  | 3% |
| Preparing Coffe      | 7  | 1  | 0  | 88%  | 5% |
| At the Computer      | 6  | 1  | 0  | 91%  | 4% |
| Exercice 1           | 3  | 0  | 0  | 99%  | 2% |
| Exercice 2           | 3  | 0  | 0  | 99%  | 1% |

Table 7.1: Recognition results for the HOSPITAL dataset using **HAM** and **HM** models to characterize the target activities.

between them nor with other different activities. Similar is the case of "Eating" and "Preparing Meal" which could be confused due to the camera perspective. Finally, "Reading" is a challenging activity that can only be differentiated from "Sitting at the armchair" by the subtle motion of the person while managing reading material, the system is able to detect the activity in most of the cases.

Also in fig. 7.16, we provide examples (marked A, B, C) of false positives and false negatives, which provide insights of the importance of combining local and global features.

| <b>HAM</b>              |    |    |    |     |    |
|-------------------------|----|----|----|-----|----|
|                         | TP | FP | FN | RT  | FT |
| Eating                  | 31 | 1  | 0  | 97% | 7% |
| Reading in the Armchair | 24 | 5  | 0  | 94% | 8% |
| Preparing Meal          | 54 | 2  | 1  | 95% | 4% |
| Standing at Armchair    | 11 | 2  | 0  | 95% | 5% |
| Sitting at Eating place | 8  | 0  | 1  | 99% | 2% |
| Inside the bathroom     | 14 | 2  | 0  | 82% | 4% |
| Armchair to Table       | 32 | 2  | 0  | 97% | 1% |
| Armchair to Kitchen     | 15 | 1  | 0  | 98% | 3% |

| <b>HM</b>               |    |    |    |     |     |
|-------------------------|----|----|----|-----|-----|
|                         | TP | FP | FN | RT  | FT  |
| Eating                  | 31 | 1  | 0  | 97% | 7%  |
| Reading in the Armchair | 24 | 4  | 0  | 92% | 11% |
| Preparing Meal          | 52 | 6  | 3  | 83% | 6%  |
| Standing at Armchair    | 11 | 2  | 0  | 95% | 5%  |
| Sitting at Eating place | 8  | 0  | 1  | 99% | 2%  |
| Inside the bathroom     | 14 | 4  | 0  | 82% | 7%  |
| Armchair to Table       | 32 | 4  | 0  | 96% | 1%  |
| Armchair to Kitchen     | 15 | 1  | 0  | 98% | 3%  |

Table 7.2: Recognition results for the HOME-CARE dataset using **HAM** and **HM** models to characterize the target activities.

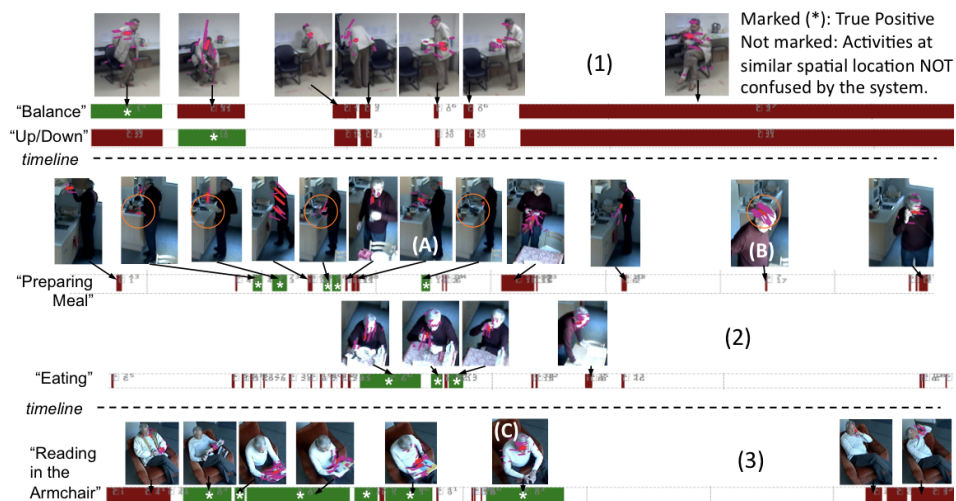


Figure 7.16: Marked with (\*) are the recognized segments (TP) of the activities: (1) "Balance" and "Up/Down" from GERHOME; (2) "Preparing Meal" and "Eating" from HOME-CARE; (3) "Reading at the Armchair" from HOME-CARE. The activities are aligned in time. Not marked segments are other -different- activities occurring at the same spatial location not matching with the model. At the top, images representing characteristic actions of the activities. (A) is a False Negative due to lack of motion; (B) is an example of how local motion occurs at the "Preparing Meal" location, but there is no global position matching and a possible FP is avoided; (C) is a False positive due to similar motion and global position with the activity model.

| <i>Tracked object:</i> silhouette center |    |    |    |
|------------------------------------------|----|----|----|
|                                          | TP | FP | FN |
| Sitting.near                             | 8  | 0  | 0  |
| At.Kitchen                               | 3  | 0  | 0  |
| At.Bathroom                              | 3  | 0  | 0  |

| <i>Tracked object:</i>        |  | Right hand joint |    |    |
|-------------------------------|--|------------------|----|----|
|                               |  | TP               | FP | FN |
| Working+Sitting.near          |  | 2                | 0  | 1  |
| Interaction.Fridge+At.Kitchen |  | 2                | 1  | 0  |

Table 7.3: Recognition results of the *single-CASA* datasets using the combination of different tracked objects. Top table: the person center of mass of the tracked object. Bottom: the hand joints are used and the recognition results are accumulated in a single value. The usage of the joints helps recognizing finer activities with only few errors produced by the challenging tracking of the joints.

### 7.3.5 Recognition results, depth camera datasets

- *single-CASA, results.*

The recognition results of *single-CASA* are displayed in table 7.3. The top table shows perfect results for the recognition of activities characterized by the location and posture of the person. The bottom part of the table 7.3, displays the results of composite activities.

The challenge of recognizing composite activities is linked to the performance of the weakest tracker, in this case the hand tracker. Tracking the hands is difficult due to the multiple shapes and occlusions. The FP and FN of table 7.3 can be explained. The FN occur due to a miss detection of the person, which leads to a miss detection of the hands, the situation is captured and displayed in fig. 7.17 (a). The FN occur due to the similar motion of the person at the exact position where the interaction with the fridge occurs. The situation is displayed in 7.17 (b), and it should be compared with a TP displayed in fig. 7.17 (c).

- *multi-CASA, results.*

The recognition results of *multi-CASA* are displayed in the table 7.4. It needs to be remembered that the videos contain several object detection errors as it is illustrated in fig. 7.11 and fig. 7.10. For the activity recognition, the PFCs are built using only the center of the person. Nevertheless, the results show an excellent performance of the proposed approach for multiple persons in real situations.

| <i>Tracked object:</i> | silhouette center |    |    |
|------------------------|-------------------|----|----|
|                        | TP                | FP | FN |
| Preparing.Meal         | 10                | 0  | 0  |
| Interacting.Trashcan   | 2                 | 1  | 1  |
| Sitting.near           | 3                 | 0  | 0  |
| Sitting.Far            | 3                 | 0  | 1  |

Table 7.4: Recognition results of the *multi-CASA* datasets using the person silhouette center as the main feature of the tracked object.

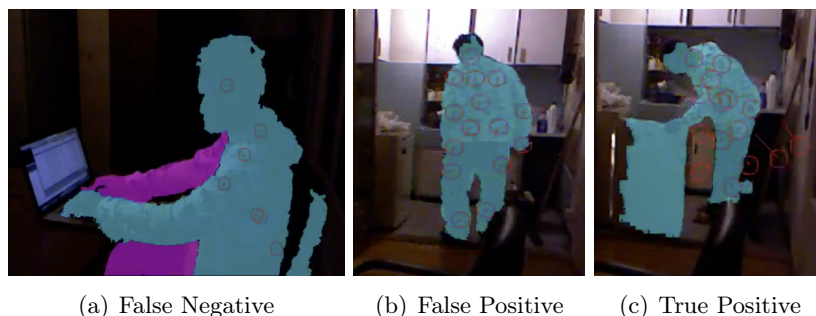


Figure 7.17: Example of FN, FP, TP in the *single-CASA* dataset.

### 7.3.6 Comparison: Multi-Resolution Histograms vs. Hierarchical Activity Models

The results presented in previous section are analyzed to evaluate the activity modeling with the two proposed methods: **HAM** and **HM**. The analysis is done by comparing the metrics TPR, FDR and PPV, presented in the table 7.5.

Improving the recognition results of **HMs** is challenging due to its high accuracy. Nevertheless, comparing side by side the recognition results using **HM** and **HAM** it is possible to assert the better performance of **HAM** over **HM** as it can be noticed in the table 7.5. The improvement is more notorious in the HOME-CARE dataset due to the bigger amount of activity instances. In most of the cases the usage of **HAM** decreases the false detection rate without increasing the TPR, such a thing is equivalent to decrease the FPs without decreasing the TPs.

### 7.3.7 Comparison: Multi-Resolution vs. Single-resolution approach **HM-basic**

We aim at measuring the contribution of the local dynamics and the multiple-resolution configuration for the overall system. We compare the results of learning the activity models using **HMs** and a similar previous method [Pusiol 2010], which we call **HM-basic**. The main differences between **HM-basic** and **HM** are:

|                         | <b>HAM</b>  |             |             | <b>HM</b> |      |      |
|-------------------------|-------------|-------------|-------------|-----------|------|------|
|                         | TPR         | FDR         | PPV         | TPR       | FDR  | PPV  |
| Balance                 | 1           | 0           | 1           | 1         | 0    | 1    |
| Up/Down                 | 1           | 0           | 1           | 1         | 0    | 1    |
| Reading at the table    | 0.9         | <b>0</b>    | <b>1</b>    | 0.9       | 0.09 | 0.9  |
| Preparing Coffe         | 1           | 0.12        | 0.87        | 1         | 0.12 | 0.87 |
| At the Computer         | 1           | 0.14        | 0.85        | 1         | 0.14 | 0.85 |
| Excercise 1             | 1           | 0           | 1           | 1         | 0    | 1    |
| Excercise 2             | 1           | 0           | 1           | 1         | 0    | 1    |
| Eating                  | 1           | 0.3         | 0.96        | 1         | 0.3  | 0.96 |
| Reading in the Armchair | 1           | <b>0.17</b> | 0.82        | 1         | 0.14 | 0.85 |
| Preparing Meal          | <b>0.98</b> | <b>0.03</b> | <b>0.96</b> | 0.94      | 0.10 | 0.89 |
| Standing at Armchair    | 1           | 0.15        | 0.84        | 1         | 0.15 | 0.84 |
| Sitting at Eating place | 0.8         | 0           | 1           | 0.8       | 0    | 1    |
| Inside the bathroom     | 1           | <b>0.12</b> | <b>0.87</b> | 1         | 0.25 | 0.77 |
| Armchair to Table       | 1           | <b>0.05</b> | <b>0.94</b> | 1         | 0.11 | 0.88 |
| Armchair to Kitchen     | 1           | 0.06        | 0.93        | 1         | 0.06 | 0.93 |

Table 7.5: Comparison of the recognition results: TPR (high is better), FDR (low is better), PPV (high is better). The activities correspond to the GERHOME and HOME-CARE datasets. The system configuration is the same than previously except for the modeling method which are **HM** or **HAM**.

- In **HM-basic** the approach uses only the global information to build primitive events, while in **HMs** it combines global and local features.
- In **HM-basic** the activity model is a single-dimensional histogram, while in **HMs** the model is multi-dimensional containing a normalization of the person local motion.
- In **HM-basic** the approach is single-resolution: the discovery of activities is performed at a single resolution level, while in **HMs** is multi-resolution (5, 10 and 15 clusters).

We compare the metrics TPR, FDR and PPV to provide insights into the improvements that are achieved with the method proposed in this work.

Table 7.6 show the results of the recognition using **HM-single**. It is evaluated for 5 activities where 2 are evaluated in previous experiments: "Preparing Meal" (or *cooking*), and "Eating". For both activities, the results of table 7.6 can be compared with the table 7.5. The side by side comparison shows an improvement of over 10% for TPR; and over 13 % for FDR and PPV, while using our proposed method (HM modeling, multi-resolution and local dynamics). Similarly, the usage of **HAM** have have also a grate importance.

In Table 7.6, the activity "Kitchen to bathroom" describes the *change* of scene regions. The recognition of a *change* activity can be compared with similar activities "Armchair to kitchen/Table" of the table 7.5, where the better results are obtained using multi-resolution HM.

The lack of local dynamics in [Pusiol 2010] makes hard to characterize activities such as "Reading in the Armchair", in that case the **HM-basic** is limited to the activity "Sitting in the armchair" which is more vague. The results of the recognition of "Reading in the Armchair" using **HM** are better than the results of the more general (easier to recognize) activity "Sitting at the armchair" of [Pusiol 2010], which highlights the importance of the person local dynamics for the recognition of finer activities.

To conclude, the usage of multiple resolutions and local dynamics allows the recognition of finer activities (more precise activities) and also improves significantly the activity recognition accuracy.

### 7.3.8 Comparison: Our proposed -HAM and HM - vs. state of the art -SA1- [Zouba 2010], in HOME-CARE

The literature approach [Zouba 2010] (called **SA1**) evaluate the recognition of activities with the same HOME-CARE video dataset. **SA1** uses information of the videos but also of other sensors (i.e. pressure, contact) to recognize activities.



|                         | HM-basic |      |      |
|-------------------------|----------|------|------|
|                         | TPR      | FDR  | PPV  |
| Preparing Meal          | 0.88     | 0.14 | 0.85 |
| Eating                  | 0.81     | 0.23 | 0.76 |
| Sitting at the armchair | 0.91     | 0.26 | 0.73 |
| Kitchen to bathroom     | 0.75     | 0.25 | 0.75 |

Table 7.6: Recognition metrics of the activities recognized in [Pusiol 2010]. TPR (high is better), FDR (low is better), PPV (high is better). The values presented in this table can be compared with the ones of the table 7.5 to evaluate the contribution of the usage of local dynamics and multiple resolutions.

The models of the activities proposed in **SA1** are defined manually. The user has to describe with a modeling language [Vu 2003] the expected activities that may happen. For example, the activity "Preparing Meal" has been described by: 1) The fridge is opened (using the contact sensor), 2) the person is bending in the kitchen (using the posture and people tracker).

#### HAM vs. SA1:

Target activities are shared between the proposed method **HAM** and the start of the art **SA1**. For example, the activity "*Preparing meal*"<sup>1</sup> is directly comparable. In [Zouba 2010] the activity "*Sitting on chair*" is reported<sup>2</sup>. In this work, we can differentiate two finer activities at the chair "*Sitting at eating place*" and "*Eating*". The accumulation of the detection of both activities can be compared<sup>3</sup> with the recognition results of "*Sitting on chair*" reported in [Zouba 2010]. The **SA1** is evaluated with a 5 persons while our approach is evaluated using 7 persons. The length of the videos of each person is the same (4 hours each). The reported results of **SA1** are compared with the recognition results of **HAM** in the table 7.7.

#### HM vs. SA1:

**SA1** has been evaluated using cross-validation for 5 persons for the activity "Sitting in the armchair". In general due to the use of local motion, we are capable of recognizing finer activity using **HAM** (e.g. "Reading at the Armchair"). In previous work **HM-basic** was used to recognize "Sitting in the armchair". The comparison is displayed in the table 7.8.

The errors of **SA1** are due to different reasons: 1) The use of a segmentation based approach for the human detection in long-term videos; 2) The lack of local dynamics (only coarse activities can be detected); 3) The activity model is

<sup>1</sup>In [Zouba 2010] the activity "*Preparing meal*" is called "*Prepare lunch*".

<sup>2</sup>The lack of local motion descriptors in **SA1** disables the accurate recognition of finer activity at the chair.

<sup>3</sup>There is no third activity that can be observed at the chair. Therefore the person is either "*Eating*" or "*Sitting at eating place*", and the combination of both is "*Sitting on chair*" of **SA1**.

| Our <b>HAM</b>          |    |    |    |      |      |
|-------------------------|----|----|----|------|------|
|                         | TP | FP | FN | TPR  | PPV  |
| Eating                  | 31 | 1  | 0  | 1    | 0.96 |
| Sitting at Eating Place | 8  | 0  | 1  | 0.8  | 1    |
| Preparing Meal          | 54 | 2  | 1  | 0.98 | 0.96 |

| [Zouba 2010]     |    |    |    |      |      |
|------------------|----|----|----|------|------|
|                  | TP | FP | FN | TPR  | PPV  |
| Sitting on Chair | 54 | 15 | 12 | 0.78 | 0.82 |
| Preparing Meal   | 4  | 3  | 1  | 0.57 | 0.8  |

Table 7.7: Comparison, reported values for **HAM** (top) and the state of the art **SA1** (bottom).

| Our <b>HM-basic</b>     |    |    |    |      |      |
|-------------------------|----|----|----|------|------|
|                         | TP | FP | FN | TPR  | PPV  |
| Sitting in the Armchair | -  | -  | -  | 0.91 | 0.73 |

| [Zouba 2010]            |    |    |    |     |      |
|-------------------------|----|----|----|-----|------|
|                         | TP | FP | FN | TPR | PPV  |
| Sitting in the Armchair | 49 | 12 | 8  | 0.8 | 0.86 |

Table 7.8: Comparison, reported values of **HM-basic** (top) and the state of the art **SA1** (bottom). The TP, FP and FN are not available.

manually built not considering perceptual errors in the model. 4) pressure and contact sensors can produce erroneous data (e.g. the system detects a person sitting when a heavy bag is over a chair). Nevertheless, the use of other sensors enables to reinforce the activity models with non visual information. It can be seen that **HAM** considers more visual features than **SA1**, but **SA1** is more flexible to the introduction of non-visual information for the creation of activity models.

In most of the cases the performance of **HAM** for recognizing long-term activities improves the results of **SA1**<sup>1</sup>. Nevertheless, in [Zouba 2010] it is also reported the detection of short activities such as "use stove" and "use fridge". These activities are detected using the information of contact sensors which is supported by **SA1** but not by **HAM**.

### 7.3.9 Ranking

In previous section we show that **HAM** provides more descriptive activity models. Now, we aim at evaluating the capabilities of **HAM**.

<sup>1</sup>This includes the recognition of finer activities such as "Eating" and "Sitting at Eating Place" of **HAM** compared with the enclosing activity "Sitting on Chair" of **SA1**.

Our statement: *A good discriminative model would be able to differentiate between the targeted activities and other -different- activities using minimal training prototypes.*

We propose a mechanism to provide insights into the **HAM** descriptiveness in 4 stages:

- First, learn the model of a target activity using a single training prototype<sup>1</sup> (e.g. "Eating" for person A).
- Second, measure the similarity (compare) of the modeled activity with all other activities sharing the same spatial location (i.e. "Eating" for person A, is compared with all activities discovered in the table area for the same person).
- Third, order the activities by similarity from most similar to less similar. Ideally, all the targeted activities should appear in the first places of the ordering.
- Fourth, compare the obtained activity ordering with the activities contained in the manually annotated ground truth. From the comparison a rank measurement (**R**) -defined below- is computed.
- Fifth, repeat the previous steps using other prototype of the target activity to learn a model. Calculate **R** learning the model for all instances of a target activity for different persons.

### 7.3.9.1 The rank measurement

The rank (**R**) measurement is a comparison between a set of ordered recognized activities and a ground truth. By our statement, all true positives (TP) of an activity should appear in the first places of the set of ordered activities computed with the mechanism described before. The **R** metric aims at describing the amount of false positives (FP) ordered as "more similar" than the last ordered TP.

Suppose  $\langle X_i, X_{i+1}, \dots, TP_{i+q} \dots \rangle$  is an ordered list where  $X$  can be a  $TP$  or a  $FP$  and  $TP_{i+q}$  is the last ordered true positive appearing in the list, **R** is defined as:

$$\mathbf{R} = \frac{\#(FP_i)}{q} \quad i \leq q \quad (7.9)$$

The value  $R \in [0..1]$  represents a perfect result when is 1, otherwise it shows that false positives are being confused. In practice, the use of an accurate people tracker provides sufficient data to recognize all activity instances, therefore FN do not occur.

---

<sup>1</sup>A single model.

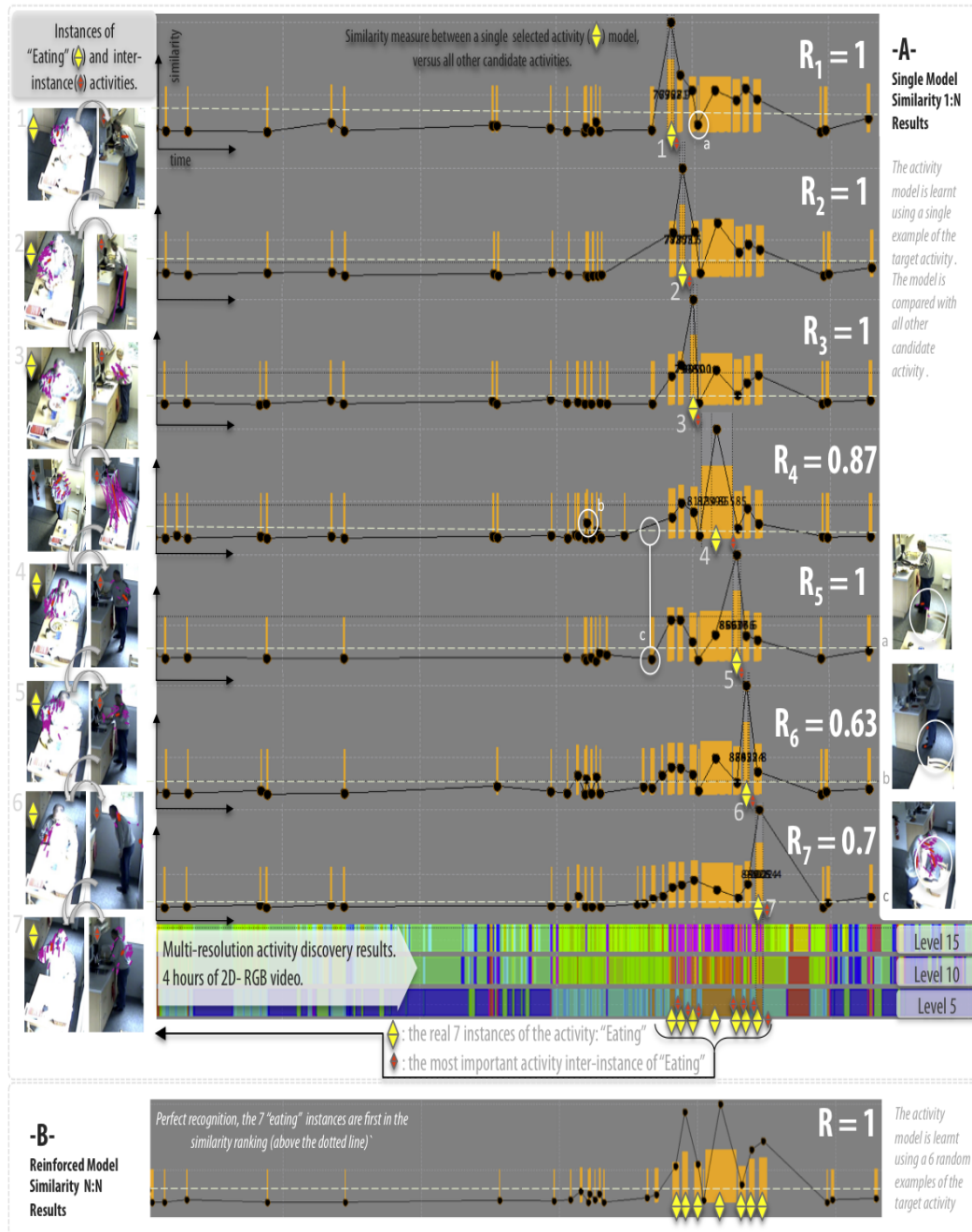


Figure 7.18: Person A:In (A) results of computing the  $\mathbf{R}$  score for the 7 ground truth instances of "Eating". The model is learned using the discovered activity marked with the yellow triangle. The black dots correspond to the distances between the modeled instance and the candidate activities. Left, the sequence of snapshots of the real video is represented. Right, different situations: (a) "Eating" correctly filtered out even when the local motion of the person foot could confuse the system; (b) a confusion occurs due to the motion of the foot of the person and the 2D perception; (c) when the similarity is low we do not display the candidate activity. In (B)  $\mathbf{R}$  is computed for the same person with a richer activity model. The activity model is learnt using 6 random prototypes of different persons instead of person A. Notice, that increasing the number of prototypes used for learning the model increases  $\mathbf{R}$  as it would be expected.

### 7.3.9.2 The rank experiments - *single-instance training*

We perform experiments in the HOME-CARE scene. We target the activity "Eating" due its complex variations among persons (e.g. different length, number of instances, etc.). Also, "Eating" can be easily confused with other activities at the table area and due to the camera perspective, with activities occurring at the kitchen area.

From the HOME-CARE dataset we select 3 random persons, named persons A, B and, C. For each person, we perform as many experiments as "Eating" instances contained in the activity ground truth. We perform the activity discovery procedure and we learn the activity model of one of the discovered instances which correspond to an instance in the ground truth. For each experiment we compute the  $\mathbf{R}$  score.

All experiments are displayed in the figures 7.18, 7.19 and, 7.20 for the persons A, B and, C respectively. Each image shows at the bottom the full activity discovery results colored type of Discovered Activity (DA). The ground truth of targeted instances are marked with yellow triangles. Over the illustration of the discovered activities, a set of layers represents the experiments (7, 3 and 5 experiments). Each layer corresponds to an experiment. For each experiment layer, the activity model is learned using the DA marked by the user - yellow triangle -. The colored segments (same color) correspond to candidate activities occurring at the same place as the modeled one. The black dots correspond to the similarity of the modeled instance (e.g. "Eating" person A) with all other candidate activities. The dashed white line appearing in each experiment layer represents the limit between the last ordered  $TP$  and the  $FPs$  with higher order position. All candidate activities which distance is over the dashed line are used to calculate  $\mathbf{R}$ .

In the left side of the figures, it is displayed a snapshot of the real video. The yellow triangle marks an "Eating" snapshot and the red triangles are snapshots of other activities occurring between the instances of "Eating". In the left side of the images we aim at explaining the rank failures.

The results are summarized in the table 7.9, where in most of the cases  $\mathbf{R}$  is maximum.



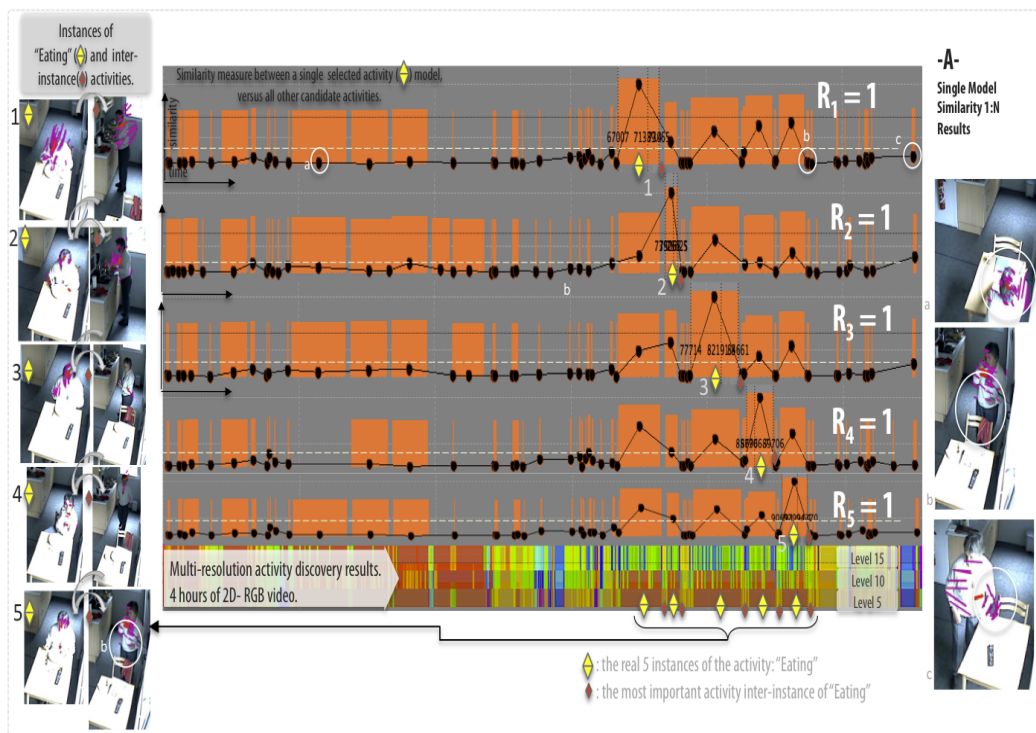


Figure 7.20: Person C: In (A), the results of computing the  $\mathbf{R}$  score for the 4 ground truth instances of "Eating". It is interesting to remark that while in figs. 7.18 and 7.19 the activity "Eating" appear as the longest (duration) for this person, other activities at the table area have similar duration and are not confused with "Eating". Among the potential confusions: (a) Long duration activity at the table. (b) The person is "Using the fixed phone". (c) The person interacts with the table.

### 7.3.9.3 The rank experiments - *multi-instance training*

The results of 7.9 show the system capability to recognize activities using minimal information (a single training activity) to learn an activity model. We also aim at showing that learning the model of an activity using multiple prototypes of a target activity<sup>1</sup> can improve the results. The experiment is illustrated in figure 7.18 (B) where for the person A we learn the model of the activity "Eating" using 6 training instances. The 6 instances are random samples from different persons, also different from person A. The result is  $\mathbf{R} = 1$ , which improves the average of the individual experiments for person A (table 7.9). The results provide a good insight of the benefits of learning a rich model using several activity examples rather activity models learnt with a single training instance.

---

<sup>1</sup>A reinforced model is more descriptive than a single model.



# Conclusions

---

## Contents

---

|            |                        |            |
|------------|------------------------|------------|
| <b>8.1</b> | <b>Conclusions</b>     | <b>159</b> |
| 8.1.1      | Closing the loop       | 160        |
| 8.1.2      | Summary                | 160        |
| 8.1.3      | Take it home message   | 161        |
| <b>8.2</b> | <b>Future Work</b>     | <b>162</b> |
| 8.2.1      | In the short term      | 162        |
| 8.2.2      | In the long term       | 163        |
| <b>8.3</b> | <b>A final thought</b> | <b>163</b> |

---

## 8.1 Conclusions

In this work we present a complete framework for activity understanding (i.e. discovery, modeling and recognition). To achieve the proposed framework we have combined and adapted techniques of different research fields and solved problems inherent to the techniques deployment as an unified system. We have explored and built over *computer vision* techniques to extract and dynamically compute perceptual descriptors (i.e. PFCs). We have adapted and extended *machine learning* techniques for clustering and data reduction to discover the interesting regions of a scene (i.e. topology) and to extract the main local motions of object parts (i.e. local dynamics). We borrow *graph theory* techniques of graph representation (i.e. K-plet) and matching (i.e. CBFS) to characterize, update, and align topologies. We have explored and deployed *pattern recognition* techniques to discover, model, and recognize activities in new datasets. We have used *human computer interface* techniques to retrieve analytical information of the discovered activities allowing the user to easily interact with the system (e.g. adding semantical interpretation). The combination of the above described techniques have a main objective which is to deal with the challenge of bridging the semantic gap limitation present in most of the literature approaches.

### 8.1.1 Closing the loop

To bridge the semantic gap we have proposed a hybrid method which is able to find, characterize and detect long-term activities. The method is strongly based on an intermediate layer of **primitive events** which are the responsible of linking semantics with perceptual information in a comprehensible manner. Thanks to the intermediate layer, the proposed method overcomes the problems of manually describing the target activities by using learning techniques to build automatically the activity models. In the other hand, the limitation of the learning techniques (e.g. lack of semantical interpretation) is addressed using weakly supervision. In other words, the proposed method uses the benefit of a supervised paradigm to deal with the drawbacks of the unsupervised one, and vice-versa. Conceptually, the main idea is to let the system to guide the user of which activities it is able to recognize automatically, and let the user to provide a simple feedback to the system about labels of the activities to recognize in a generative fashion.

A point that needs to be stress is the architecture of the framework. This framework presents a vertical integration strategy. The integration connects each processing layer from raw data acquisition to its semantical interpretation. The integration is possible due to the deep understanding of the methods deployed at each stage. The understanding of each stage allows to identify the environmental conditions of particular failures (e.g. object miss detection) and to evaluate its impact on the whole system. As a result, the proposed architecture is flexible, the modification of a single processing stage does not critically affect the whole system (e.g. changing the people detector). And therefore, none of the parts is worth more than the sum of them in the whole system.

### 8.1.2 Summary

The framework is composed of several parts (divided by chapters) which are logically ordered. A brief synthesize of the parts can help to understand their unique role in the whole framework.

- The definition and building of **Perceptual Feature Chunks (PFCs)** provide a description of a video snapshot with the particularity of combining in a single package the global and local movements of an object. Such a thing, allows to achieve abstract and precise descriptions of an ongoing activity at once. The PFCs by themselves can be used to feed directly single-layer complex activity models which bridge in one step the semantic gap. Towards avoiding such a bridge we propose to link the PFCs with contextual information. Such a link does not only explain "what" is happening but also "where and how", allowing users to understand activity models.
- We learn the interesting regions of the scene and build **topologies**. The topologies are at multiple resolutions to avoid configuration problems (i.e.

number of clusters). The topologies provide macro models of the interactions of an agent (i.e. person) with fixed objects (e.g. table) and regions (e.g. kitchen). In this work we see the contextual information corresponding to a scene as a "living" entity which can change over time and can be different among agents. Such a thing differs with previous approaches where the contextual information is defined once for all agents and it is updated only when significant changes occur (e.g. a table has moved from one place to another). Also, it can be seen that the stage of learning the contextual information could be replaced by a manual procedure without affecting the framework.

- The combination of descriptive video snapshots (PFCs) and macro models of the scene (Topology) allows the creation of **Primitive Events** (PE) located at the middle layer of the semantic gap. The PEs play the role of linking perceptive information with semantic interpretation. Building PEs is only possible through the previous stages. The sequences of PEs contains the characteristic structures of the long-term activities which still need to be revealed.
- We discover activities using a set of defined patterns which characterize mainly the movement over the scene at different resolutions. The types of pattern may be changed or extended without affecting the rest of the system. An interesting field to be explored in deeper detail is the way the activities are automatically retrieved and presented to the user. We proposed a sequence of colored intervals which can represent dependencies (sub-activities) when they are aligned on a same timeline. The graphical representation allows an intuitive and easy interaction of non expert users with the system. Also, on top of the activity discovery we propose two techniques to **model hierarchically** the activities of interest. The particularity of both models is that they are fed with multi-resolution information. Finally, we propose a way to link all the previous parts together to achieve the recognition of target activities in new datasets.
- We test the system in home-care applications. We process massive amounts of data to claim that the method is valid for long-term activities. The obtained results show the capability of the system of being used in real world problems and open new doors for future extensions.

### 8.1.3 Take it home message

This work aims at being applied off the shelf in regular homes without the need of parametric configuration. Most of the state-of-the-art methods do not overpass the status of "proof of concept" due to the requirement of configuring numerous parameters and/or modules (e.g. people detector). Nowadays, activity understanding needs to solve real world problems (e.g. Alzheimer disease) and to do that, it requires extensive testing in real world conditions. To push research forward we need to pay attention to the complexity of the systems which are being provided to non

expert users. It is them who are going to validate the methods by providing testing data and feedbacks to the research community. To take into account the interest of non expert users, a good practice is to minimize the parametric configuration of the system. Nevertheless, minimizing the parametrization should not degrade the system performance in any way. Such a concept is followed in this work where we design an activity recognition system which **can be configured in 20 seconds** to start tracking people in real time and detecting activities such as "Eating" in an off-line mode. This system is mainly possible due to two techniques: 1) the usage of multi-resolution (i.e. topologies, activity models), which means that many configurations are used and only the meaningful ones prevail for the recognition of activities; 2) adapting parameters dynamically using other perceptual information (i.e. mean-shift clustering is adapted to the amount of motion to compute the *Local Dynamics* of a person).

## 8.2 Future Work

### 8.2.1 In the short term

The proposed work describes the know-how of a complete and tested loop for long-term activity understanding. We see this loop as a milestone which can be applied to indoor physical activity applications to help improving the well-being of people. The framework can be applied of the shelf. Nevertheless, there is a lot of room for improvement which could make this work to evolve. At this point we can describe 4 key aspects which can be improved in the short term.

**First**, we are interested into exploring in deeper detail, the activity modeling process. We observe that for long-term activities no sharp temporal constraints can be assumed. Nevertheless, further research could help us to clarify this point. We are taken the initial steps to compare our modeling techniques and recent topic modeling techniques to decide in which situations the temporal constraints (if any) play an important role at the moment of recognizing activities. The comparison could lead to hybrid models which consider the temporal constraints only when they are needed.

**Second**, only the global information of an object is used to discover new activities. In some cases, an activity can change without being noticed by the global information (e.g. reading and eating at the table). Currently the situation is being handled by increasing the number of scene clusters and assuming that a change in the activity is reflected in the global information when the contextual information is finer. Nevertheless, in some cases the solution is not optimal and can lead to errors directly produced by the object tracker. Therefore, we need to explore the salient changes of the local motion which can trigger the beginning of new activities even when such an event is not noticed by the global information.

**Third**, other types of data (e.g. audio, gyroscope) which also can be massively collected can be hooked to the framework. In particular, exploring how the primitive event representation can be extended to handle multi-sensorial information (e.g. natural language processing provided through microphones). Also, understanding up to which point (and how) the primitive events can be overloaded with richer information without losing the benefit of easily describing semantics.

**Fourth**, A current limitation of long-term activity understanding systems is the complexity of the evaluation procedures. The evaluation of the systems requires of ground truths (annotations) of long duration. The construction of these ground truths require of large amount of time and of human resources. We have faced such a limitation during the evaluation of this work. Therefore, we are interested into elaborate new evaluation procedures to better assess the proposed framework and to enable the possibility of understanding activities which can be revealed analyzing longer periods of time (i.e. months).

### 8.2.2 In the long term

We consider that the proposed work is settling solid basis for other research works in the long run. At this point we can identify 2 points which could be addressed in the future.

**First**, for semi-supervised approaches the way the information is presented to the non expert users takes a big responsibility for the success of a system. The information needs to be adapted to the needs and expertise of the regular user, only then meaningful user feed-back can be collected to improve the learned knowledge of the hybrid system. Currently, the information is presented in an authoritative manner assuming that the users have the same experience as the developer to synthesize complex information. We aim at exploring the domain of Human Machine Interfaces (HMI) and psychology to understand the good practices of displaying information in a way that a regular user naturally gets interested to help the system with feed-back.

**Second**, this work can contribute to the creation and extension of new ontologies. In particular, semi-supervised approaches such as the one proposed in this work in conjunction with intuitive HMI tools can help to align ontologies and can improve in faster ways the discovery of activities.

## 8.3 A final thought

Another way to see this work is as a **thought manner to cluster trajectory information**. This last statement opens a world of numerous applications. The applications are those where objects can be tracked for long periods of time. For

example, we are currently testing the system to recognize activities and perform analytics of persons tracked (GPS) during extensive workouts (i.e. training for marathon) to guide coaches in the creation of personalized training plans.

Finally, having explained the know-how of the proposed approach the reader should have understood the limitations and benefits of the proposed method. At this point we aim at challenging the reader to imagine the possible applications of this work, addressing new problems in their own fields of interest.

# Appendix

---

## A.1 K-Means Clustering, Distances

### A.1.1 Euclidean distance

The Euclidean distance is a true metric that satisfies the triangle inequality. We define the Euclidean distance as

$$d = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

Only those terms are included in the summation for which both  $x_i$  and  $y_i$  are present. The denominator  $n$  is chosen accordingly. In this formula, the feature space  $x_i$  and  $y_i$  are subtracted directly from each other. We should therefore make sure that the feature space is properly normalized when using the Euclidean distance. In our case, the feature space is the euclidean space, therefore no normalization is required. Unlike the correlation-based distance functions, the Euclidean distance takes the magnitude of the feature space into account. It therefore preserves more information about the data and may be preferable. The image [A.1](#) displays an example of clustering (i.e. computing a topology) using the euclidean distance.

### A.1.2 City-block distance

The city-block distance, alternatively known as the Manhattan distance and  $L_1$  norm, is related to the Euclidean distance. Whereas the Euclidean distance corresponds to the length of the shortest path between two points, the city-block distance is the sum of distances along each dimension.

$$d = \frac{1}{n} \sum_{i=1}^n |x_i - y_i|$$

This is equal to the distance you would have to walk between two points in a city, where you have to walk along city blocks. The city-block distance is a metric, as it satisfies the triangle inequality. As for the Euclidean distance, the expression data are subtracted directly from each other, and we should therefore make sure that they are properly normalized.



Figure A.1: Example of k-means clustering - Euclidean Distance

### A.1.3 The Pearson correlation coefficient

The Pearson correlation coefficient is defined as

$$r = \frac{1}{n} \sum_{i=1}^n \left( \frac{x_i - \bar{x}}{\sigma_x} \right) \left( \frac{y_i - \bar{y}}{\sigma_y} \right)$$

in which  $\bar{x}$ ;  $\bar{y}$  are the sample mean of  $x$  and  $y$  respectively;  $\sigma_x$  and  $\sigma_y$  are the sample standard deviation of  $x$  and  $y$ . The Pearson correlation coefficient is a measure for how well a straight line can be fitted to a scatterplot of  $x$  and  $y$ . If all the points in the scatterplot lie on a straight line, the Pearson correlation coefficient is either  $+1$  or  $-1$  depending on whether the slope of line is positive or negative. If the Pearson correlation coefficient is equal to zero, there is no correlation between  $x$  and  $y$ . The Pearson *distance* is defined as

$$d_p \equiv 1 - r$$

As the Pearson correlation coefficient lies between  $-1$  and  $1$ , the Pearson distance lies between  $0$  and  $2$ . Note that the Pearson correlation automatically centers the data by subtracting the mean, and normalizes them by dividing by the standard deviation. While such normalization may be useful in some situations information is being lost in this step. In particular, the magnitude of changes is being ignored. This is in fact the reason why the Pearson distance does not satisfy the triangle inequality.





Figure A.2: Example of k-means clustering using the Pearson's Correlation distance.

#### A.1.4 Uncentered correlation (cosine of the angle)

In some cases, it may be preferable to use the uncentered correlation instead of the regular Pearson correlation coefficient. The uncentered correlation is defined as

$$r_u = \frac{1}{n} \sum_{i=1}^n \left( \frac{x_i}{\sigma_x^{(0)}} \right) \left( \frac{y_i}{\sigma_y^{(0)}} \right)$$

where

$$\sigma_x^{(0)} = \frac{1}{n} \sqrt{\sum_{i=1}^n x_i^2}$$

$$\sigma_y^{(0)} = \frac{1}{n} \sqrt{\sum_{i=1}^n y_i^2}$$

This is the same expression as for the regular Pearson correlation coefficient, except that the sample means  $\bar{x}$  and  $\bar{y}$  are set equal to zero. The uncentered correlation may be appropriate if there is a zero reference state. The distance corresponding to the uncentered correlation coefficient is defined as

$$d_U \equiv 1 - r_u$$

where  $r_u$  is the uncentered correlation. As the uncentered correlation coefficient lies between -1 and 1, the corresponding distance lies between 0 and 2. The uncentered correlation is equal to the cosine of the angle of the two data vectors in



Figure A.3: Example of k-means clustering using the uncentered Pearson’s correlation.

n-dimensional space, and is often referred to as such. (From this viewpoint, it would make more sense to define the distance as the arc cosine of the uncentered correlation coefficient).

### A.1.5 Spearman rank correlation

The Spearman rank correlation is an example of a non-parametric similarity measure. It is useful because it is more robust against outliers than the Pearson correlation. To calculate the Spearman rank correlation, we replace each data value in a vector by their position rank (from smaller to greater), building an ordered rank vector. We then calculate the Pearson correlation between the two rank vectors instead of the data vectors. As in the case of the Pearson correlation, we can define a distance measure corresponding to the Spearman rank correlation as

$$d_s \equiv 1 - r_s,$$

where  $r_s$  is the Spearman rank correlation.

### A.1.6 Kendall’s $\tau$

Kendall’s  $\tau$  is another example of a non-parametric similarity measure. It is similar to the Spearman rank correlation, but instead of the ranks themselves only the relative rank are used to calculate  $\tau$ .

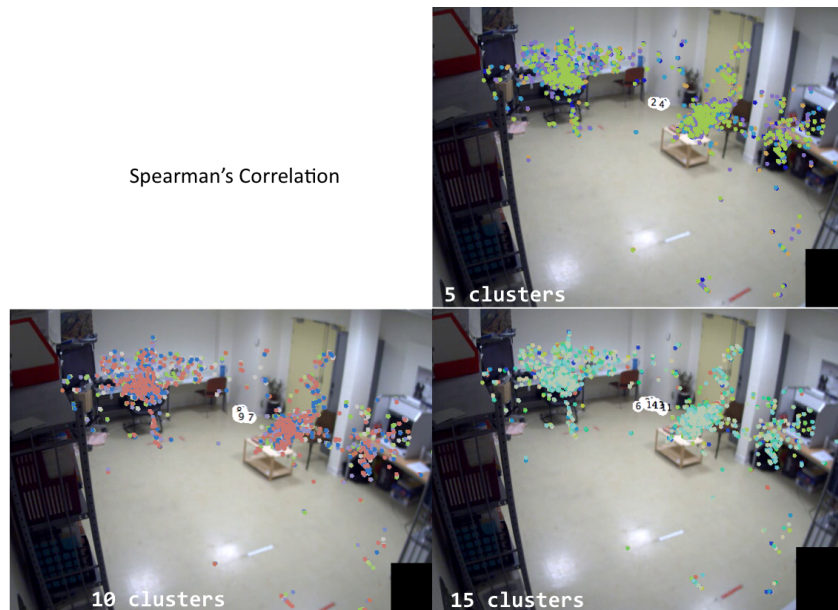


Figure A.4: Example of k-means clustering

$$d_K \equiv 1 - \tau,$$

A Kendall's  $\tau$  is defined such that it lies between -1 and 1, the corresponding distance is between 0 and 2.



Figure A.5: Example of k-means clustering

## A.2 Natural Language Processing for Activity Modeling

Language modeling can be used for speech recognition, handwriting recognition, spelling correction, foreign language reading/writing aid, machine translation and so on.

A language model is defined as a probability distribution over word sequences. In "Foundations of Statistical Natural Language Processing" [Manning 2005], Manning and Schutze point out that words in a corpus can be thought as random discrete data, which is generated according to some unknown probability distribution. For example, if we have a string "I like Chinese food" denoted as  $w = w_0w_1w_2w_3$ , the language model is  $P(w)$ . In other words, the language model attempts to reflect how frequently a string  $w$  occurs as a sentence.

Nowadays, a popular class of models is based on frequency. This class of models uses term occurrence to weight each word while ignoring any relationship between words. Since words do not occur in a random order, another class of language models consists of n-grams, such as *uni-gram*, *bi-gram* and , which predict each word given the  $n - 1$  previous words. The parameter  $n - 1$  indicates how many previous words are used to model the next word. However, the n-gram models only focus on the likelihood of short sequences of words, they ignore how words are semantically related to each other. Recently the attention turned into a particular class of language models named of topic models. These models start to be widely

used for activity recognition.

Topic models are based on the "bag-of-words" assumption, which means that words can be thought as sampled from a set (i.e. bag) of words without any order. For example, the vector representation of "Lucy is taller than Sam" is the same when the vector representation of "Sam is taller than Lucy". Topic models also include latent components which are often referred to as topics since they tend to cluster together words related to the same topic. Such latent topics can serve as a basis to measure how closely related words are by meaning. While there are many ways of modeling topics, Bayesian approaches provide a principled statistical framework to do this. Moreover, a hierarchical non parameter data model can provide more precise results. Due to their relevance, we introduce two of the most well known topic models in the main stream of this work. In section 6.4.3 we discuss: Latent Dirichlet Allocation (LDA) and Hierarchical Dirichlet Processes (HDP).

### A.2.1 A discovered activity and the underlying relationship with language models

The patterns: *Change* and *Stay* defined in Chapter 6, can be seen as words in language modeling. A word is generally used as a descriptive unit of a document. In our case, we aim at describing activities by its sub-activities hence a recursive definition appear. An activity is itself a sub-activity of another activity at a coarser resolution. When an activity is used as a descriptive sub-activity we consider it as a word. When an activity is our object of interest (the one we aim at modeling) we consider its corresponding video as document.

#### A.2.1.1 Language and Activity modeling notation

We assume that there is a corpus (a set of documents or activities) and that each document consists of a set of words, which is equivalent to say that each activity consists of a set of sub-activities. The following notation is be used. Upper case letters denote variables:  $T$  for topics,  $W$  for the sequence of words or sub-activities, and  $D$  for documents or activities. The calligraphic versions of the same letters denote their domain:  $\mathcal{T} = \text{dom}(T)$ ,  $\mathcal{W} = \text{dom}(W)$ , and  $\mathcal{D} = \text{dom}(D)$ . Lower case letters indicate values for the variables:  $t \in \mathcal{T}$ ;  $w \in \mathcal{W}$ , and  $d \in \mathcal{D}$ . Sets are denoted by bold letters:  $\mathbf{T} = \{T_0, T_1, \dots, T_{K-1}\}$  is a set of variables and  $\mathbf{t} = \{t_0, t_1, \dots, t_{K-1}\}$  is a joint assignment of values for  $\mathbf{T}$ . We reserve  $K$  to indicate the number of topics and  $N$  to indicate the number of words in each document. A set subscripted by a negative index denotes all items in the set except the one subscripted:  $T_{-k} = \{T_1, T_1, \dots, T_K\}$ . Some specific lower case letters are used as indices. For example,  $i$  is used as an index for words in a document (or sequence);  $d$  is used as an index for the documents in a corpus or a cluster; and  $k$  is used as an index for topics.

## A.2.2 Frequency models

### A.2.2.1 Tf-Idf

In information retrieval (IR), researchers developed many techniques to extract relevant keywords. A popular approach is called  $Tf - Idf$ . There are many forms for term frequency  $Tf$ , the basic one is using raw term frequency of each word  $w_i$  in each document  $d$  as follows:

$$Tf_{w_i,d} = n_{w_i,d}, \quad (\text{A.1})$$

where  $n_{w_i,d}$  is the number of occurrences of word  $w_i$  in document  $d$ . In some situations, it would be preferable for  $Tf$  to grow at a slower rate than linear, so Salton created another form which is logarithmic:

$$Tf_{w_i,d} = \begin{cases} \log(n_{w_i,d}) + 1, & \text{if } n_{w_i,d} < 0 \\ 0, & \text{otherwise} \end{cases} \quad (\text{A.2})$$

By this equation, if a word has higher frequency in a specific document, that word should have higher  $Tf$  weight.

Purely relying on term frequency is still limited because query terms are different in their ability to discriminate documents. A query term is not a good discriminator if it occurs in many documents. We should give it less weight than one occurring in few documents. For examples, when querying "information retrieval", it is unlikely that documents containing "information" might be more relevant than documents containing "retrieval". In 1972, Sparck Jones introduced a measure of term specificity (discriminative power) called Inverse Document Frequency ( $Idf$ ).

The classic equation of  $Idf$  is:

$$Idf_{w_i} = \log_{10}(|\mathcal{D}|/|\{d : w_i \in d, d \in \mathcal{D}\}|); \quad (\text{A.3})$$

where  $|\mathcal{D}|$  is the number of documents in corpus  $\mathcal{D}$  and  $|\{d : w_i \in d, d \in \mathcal{D}\}|$  is the number of documents that include the word  $w_i$ . For instance, the word "the" happens in every document in corpus  $\mathcal{D}$ . If there are 1,000,000 documents in  $\mathcal{D}$ , and  $|\{d : w_i \in d, d \in \mathcal{D}\}| = 1,000,000$  too, then the  $Idf$  value of "the" is 0. On the other hand, if a word occurs only in some of documents, the  $Idf$  value of that word should be higher. In the end, the  $Tf - Idf$  value is obtained as follows:

$$Tf - Idf = Tf * Idf \quad (\text{A.4})$$

This method uses  $Tf$  value to find more frequent words in a document, and  $Idf$  value to prune common words that are meaningless in the corpus. However, if the corpus is related to only one topic, then the most relevant word may happen in every document, but it is pruned by  $Idf$ . As a result,  $Tf - Idf$  is not suitable to label a cluster of documents.

### A.2.2.2 A metric of word relevance

A metric of word relevance is proposed by Zhou and Slater. This method estimates the relevance of uni-grams (or single words). Since meaningful words tend to cluster in certain parts of the text instead of being randomly distributed throughout the text, this method computes a measure of the tightness of the clusters of each word. Words that tend to cluster more frequently are given a higher relevance score. The approach starts by building a list of positions  $l_w = -1, t_1, t_2, \dots, t_m, n$  where  $t_i$  denotes the position of the  $i^{\text{th}}$  occurrence of word  $w$  and  $n$  is the total number of words in the document. Then we compute the average distance  $\mu$  between successive occurrences of  $w$  as follows:

$$\mu = \frac{n+1}{m+1}, \quad (\text{A.5})$$

where  $m$  is the number of occurrences of word  $w$ . We can compare this global average distance to a local average distance  $d(t_i)$  for each position  $t_i$ :

$$d(t_i) = \frac{t_{i+1} + t_{i-1}}{2}, \quad i = 1, \dots, m. \quad (\text{A.6})$$

Occurrences of  $w$  where  $d(t_i) \leq \mu$  are said to be part of a cluster since they are closer to their neighbors than average. More precisely, we denote by  $\nu(t_i)$  a function that verifies whether position  $t_i$  is part of a cluster as follows:

$$\nu(t_i) = \frac{\mu - d(t_i)}{\mu} \quad (\text{A.7})$$

Finally, we estimate the relevance of a word by computing the average reduction in average distance for the cluster:

$$\Gamma(w) = 1/m * \sum_1^m \sigma(t_i) * \nu(t_i), \quad (\text{A.8})$$

While this method is simple and efficient it tends to give low scores to relevant words that are frequent or rare because they tend not to cluster.

### A.2.2.3 Successor-Predecessor

Since relevant words tend to co-occur with a small set of predecessor and successor words, the Score function estimates the relevance of a word by measuring the variation of the frequency of its immediate neighbors. This is done by first defining the successor Score function  $SC_{suc}(w)$  for the words that immediately follow  $w$ :

$$SC_{suc}(w) = \sqrt{\frac{1}{\|\gamma\| - 1} \sum_{y_i \in \gamma} \left( \frac{p(w, y_i) - p(w, \cdot)}{p(w, \cdot)} \right)^2} \quad (\text{A.9})$$

Where  $\gamma$  is the set of distinct words in the corpus and  $\|\gamma\|$  denotes the size of  $\gamma$ . Also,  $p(w, y_i)$  is the probability that  $y_i$  follows  $w$  and  $p(w, \cdot)$  is the average probability of the successor words of  $w$ . The joint term  $p(w, y_i)$  is defined as follows:

$$p(w, y_i) = \frac{f(w, y_i)}{N} \quad (\text{A.10})$$

and

$$p(w, \cdot) = \frac{1}{\|\gamma\| - 1} \sum_{y_i \in \gamma} p(w, y_i) \quad (\text{A.11})$$

Here,  $N$  denotes the number of occurrences of word  $w$  in the corpus and  $f(w, y_i)$  is the frequency of bi-gram  $(w, y_i)$ . Similarly, we can define a predecessor score function:

$$SC_{pred}(w) = \sqrt{\frac{1}{\|\gamma\| - 1} \sum_{y_i \in \gamma} \left( \frac{p(w, y_i) - p(\cdot, w)}{p(\cdot, w)} \right)^2} \quad (\text{A.12})$$

By taking the average of the predecessor and successor scores, we obtain an overall score that can be used to estimate the relevance of words:

$$SC(w) = \frac{SC_{suc}(w) + SC_{pred}(w)}{2} \quad (\text{A.13})$$

According to this method, if the word  $w$  is followed or following a lot of different words, then the score of this word is pretty low; if the word is frequent and has a small set of successors and predecessors, then the score will be high.

#### A.2.2.4 Successor Predecessor Quotient scoring

The Successor-Predecessor Quotient ( $SPQ$ ) measure is another statistical metric that measures the importance of the word  $w$  based on the quotient between its number of distinct successors and its number of distinct predecessors. The  $SPQ$  value can be obtained by the equation:

$$SPQ(w) = \frac{N_{suc}(w)}{N_{pred}(w)}, \quad (\text{A.14})$$

where  $N_{suc}(w)$  and  $N_{pred}(w)$  represent the number of distinct successors and predecessors of word  $w$  in the corpus. Experimental [da Silva 2011] results show that  $SPQ$  is better than  $SC$

#### A.2.2.5 The Islands score

The Islands method extracts a word if it is more relevant than its neighbor words. In this approach we compute the average relevance of the predecessors and successors as follows:

$$Avg_{pre}(w) = \sum_{y_i \in \{pred\} f w} p(y_i, w) * r(y_i), \quad (\text{A.15})$$

$$Avg_{suc}(w) = \sum_{y_i \in \{suc\} f w} p(w, y_i) * r(y_i), \quad (\text{A.16})$$

where  $p(y_i, w)$  means the probability of occurrence of bi-gram  $(y_i, w)$  and  $r(y_i)$  is the relevance value given by some generic  $r(\cdot)$  metric, which could be the Score function or  $SQP$  function. A word is considered relevant if it satisfies:

$$r(w) \leq 0.9 * \max(Avg_{pre}(w), Avg_{suc}(w)). \quad (\text{A.17})$$



The above techniques identify potentially relevant words based on different properties than frequency. While these properties are interesting, it is not clear that they extract good words for the labels. Ideally, the computer should understand the meaning of the words, sentences and documents to extract good labels. To that effect, topic models take a step in this direction.

### A.2.3 n-gram Models

A  $n$ -gram is a subsequence of  $n$  words  $(w_0, \dots, w_{n-1})$  from a given sequence  $w = w_0, \dots, w_{N-1}$ . It is generally difficult to model  $P(w)$  directly, nevertheless the following chain rule can be applied:

$$P(w) = P(w_0)P(w_1|w_0)\dots P(w_{N-1}|w_{N-2}). \quad (\text{A.18})$$

When we condition each word on at most  $n-1$  previous words, we obtain an  $n$ -gram model.

### A.2.4 uni-gram model

If  $n$  equals one, it is called a uni-gram model. When we get a word sequence  $w$  with  $N$  tokens, we can denote it as  $w = w_0, \dots, w_{N-1}$ . Then the likelihood of this sequence is assumed to be the product of the probabilities of each word separately:

$$P(w) = \prod_{i=0}^{N-1} P(w_i) \quad (\text{A.19})$$

It is common to approximate  $P(w)$  by the relative frequency of  $w_i$ :

$$P(w) = \frac{N_{w_i}}{N}, \quad (\text{A.20})$$

Here,  $N_{w_i}$  is the frequency of  $w_i$  in  $\mathcal{D}$ .

The uni-gram models makes the assumption that each word is sampled independently and identically, ignoring the relationship between words. Such assumption is suitable to represent chunks of loosely constrained activities where the sampling order is relaxed.

### A.2.5 bi-gram model

When  $n$  equals two, we get a bi-gram model. Consider a sequence of words  $w$  of length  $N$ . The probability of this sequence is computed as follows for the bi-gram model:

$$P(w) = P(w_0)P(w_1|w_0)\dots P(w_{N-1}|w_{N-2}). \quad (\text{A.21})$$

$$P(w) = \prod_{i=0}^{N-1} P(w_i|w_{i-1}), \quad (\text{A.22})$$

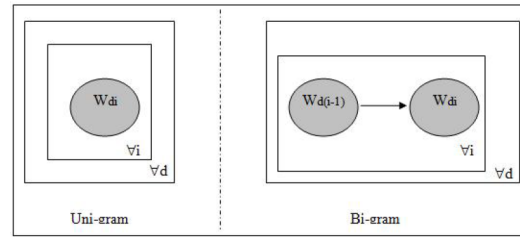


Figure A.6: The uni-gram and bi-gram representation

Let  $N_{w_i}$  be the frequency of word  $w_i$  in  $w$ , and let  $N_{w_i|w_{i-1}}$  be the number of occurrences of the word  $w_i$  following word  $w_{i-1}$  in  $w$ . When training by maximum likelihood  $P(w_i|w_{i-1}) = N_{w_i|w_{i-1}}/N_{w_{i-1}}$ .

The bi-gram models assume a relationship between a word and its predecessor, it can be used to describe in a relaxed manner the relationship of the sub-activities composing a loosely constrained activity.

# Bibliography

- [Aggarwal 2011] J K Aggarwal and M S Ryoo. *Human activity analysis*. ACM Computing Surveys, vol. 43, no. 3, pages 1–43, apr 2011. (Cited on page 13.)
- [Albanese 2008] Massimiliano Albanese, Rama Chellappa, Vincenzo Moscato, Antonio Picariello, V S Subrahmanian, Pavan Turaga and Octavian Udrea. *A Constrained Probabilistic Petri Net Framework for Human Activity Detection in Video*. IEEE Trans. Multimedia, vol. 10, no. 6, pages 982–996, jan 2008. (Cited on page 40.)
- [ALLEN 1983] JAMES F ALLEN. *Maintaining knowledge about temporal intervals*. Communications of the ACM, vol. 26, no. 11, pages 832–843, nov 1983. (Cited on page 39.)
- [ALLEN 1994] JAMES F ALLEN and GEORGE FERGUSON. *Actions and Events in Interval Temporal Logic*. Journal of Logic and Computation, vol. 4, no. 5, pages 531–579, 1994. (Cited on page 39.)
- [Anjum 2008] N. Anjum and A. Cavallaro. *Multi-feature object trajectory clustering for video analysis*. In IEEE Transactions on Circuits for Video Technology, pages 1555–1564, 2008. (Cited on page 83.)
- [Antonini 2004] G. Antonini and J. Thiran. *Trajectories clustering in ICA space: an application to automatic counting of pedestrians in video sequences*. In ACIVS 2004, EPFL, 2004. EPFL, IEEE. (Cited on page 28.)
- [Basharat 2008] Arslan Basharat, Alexei Gritai and Mubarak Shah. *Learning object motion patterns for anomaly detection and improved object detection*. In 2008 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 1–8. IEEE, jan 2008. (Cited on pages 28 and 29.)
- [Blank 2005] M Blank, L Gorelick, E Shechtman, M Irani and R Basri. *Actions as space-time shapes*. In Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1, pages 1395–1402 Vol. 2. IEEE, feb 2005. (Cited on page 21.)
- [Blei 2003a] David M Blei, Michael I Jordan, Thomas L Griffiths and Joshua B Tenenbaum. *Hierarchical Topic Models and the Nested Chinese Restaurant Process*. Advances in Neural Information Processing Systems, pages 1–8, nov 2003. (Cited on page 111.)
- [Blei 2003b] David M Blei, Andrew Y Ng and Michael I Jordan. *BleiNgJordan2003*. Journal of Machine Learning Research, pages 1–30, nov 2003. (Cited on page 110.)

- [Blei 2011] David M Blei. *Introduction to Probabilistic Topic Models*. Communications of the ACM, pages 1–16, jun 2011. (Cited on page 111.)
- [Bobick 1997] Aaron F. Bobick and Andrew D. Wilson. *A State-Based Approach to the Representation and Recognition of Gesture*. IEEE Trans. Pattern Anal. Mach. Intell., vol. 19, no. 12, pages 1325–1337, 1997. (Cited on page 29.)
- [Bobick 2001] A F Bobick and J.W. Davis. *The recognition of human movement using temporal templates*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 23, no. 3, pages 257–267, mar 2001. (Cited on page 17.)
- [Borzin 2007] Artyom Borzin, Ehud Rivlin and Michael Rudzsky. *Borzin:kx*. In 2007 International Workshop on Content-Based Multimedia Indexing, pages 33–39. IEEE, jan 2007. (Cited on page 40.)
- [Bouguet 2000] Jean-Yves Bouguet. *Pyramidal Implementation of the Lucas Kanade Feature Tracker Description of the algorithm*. Rapport technique, 2000. (Cited on pages 48, 49 and 51.)
- [Brand 1996] M Brand. *Understanding manipulation in video*. In Second International Conference on Automatic Face and Gesture Recognition, pages 94–99. IEEE Comput. Soc. Press, jan 1996. (Cited on page 37.)
- [Brand 2000] M Brand and V Kettner. *Discovery and segmentation of activities in video*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 8, pages 844–851, 2000. (Cited on page 32.)
- [Brdiczka 2009] O Brdiczka, M Langet, J Maisonnasse and J.L. Crowley. *Detecting Human Behavior Models From Multimodal Observation in a Smart Home*. IEEE Transactions on Automation Science and Engineering, vol. 6, no. 4, pages 588–597, 2009. (Cited on page 2.)
- [Bregonzio 2009] M Bregonzio, Shaogang Gong and Tao Xiang. *Recognising action as clouds of space-time interest points*. In 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPR Workshops), pages 1948–1955. IEEE, feb 2009. (Cited on page 22.)
- [Calderara 2007] S Calderara, R Cucchiara and A Prati. *Detection of Abnormal Behaviors using a Mixture of Von Mises Distributions*. In IEEE AVSS 2007, page 1, 1, jul 2007. (Cited on page 29.)
- [Campbell 1995] L W Campbell and A F Bobick. *Recognition of human body motion using phase space constraints*. In IEEE International Conference on Computer Vision, pages 624–630. IEEE Comput. Soc. Press, feb 1995. (Cited on page 19.)
- [Carreira-Perpinan 2007] Miguel A Carreira-Perpinan. *Gaussian Mean-Shift Is an EM Algorithm*. IEEE Trans. Pattern Anal. Mach. Intell., vol. 29, no. 5, pages 767–776, 2007. (Cited on page 85.)

- [Chikkerur 2005] Sharat Chikkerur. *K-plet and CBFS: A Graph based Fingerprint Representation and Matching Algorithm*. pages 1–15, nov 2005. (Cited on page 65.)
- [Chomat 1999] O. Chomat and J.L. Crowley. *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*. In Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pages 104–109. IEEE Comput. Soc, 1999. (Cited on page 20.)
- [Comaniciu 2002] D Comaniciu and P Meer. *Mean shift: a robust approach toward feature space analysis*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 5, pages 603–619, may 2002. Este paper es el que define un poco mean-shift. (Cited on page 83.)
- [da Silva 2011] Joao Ventura da Silva and Joaquim Ferreira. *Ranking and Extraction of Relevant Single Words in Text*. pages 1–20, oct 2011. (Cited on page 174.)
- [Dai 2008] Peng Dai, Huijun Di, Ligeng Dong, Linmi Tao and Guangyou Xu. *Group Interaction Analysis in Dynamic Context*. Systems, Man, and Cybernetics, Part B, IEEE Transactions on, vol. 38, no. 1, pages 275–282, feb 2008. (Cited on page 36.)
- [Damen 2009] D Damen and D Hogg. *Recognizing linked events: Searching the space of feasible explanations*. In 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPR Workshops), pages 927–934. IEEE, jan 2009. (Cited on page 36.)
- [Darnell Moore 2002] Irfan Essa Darnell Moore. *Recognizing Multitasked Activities from Video Using Stochastic Context-Free Grammar*. 2002. (Cited on page 38.)
- [Dollar 2005] P Dollar, V Rabaud, G Cottrell and S Belongie. *Behavior Recognition via Sparse Spatio-Temporal Features*. In 2005 IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance, pages 65–72. IEEE, feb 2005. (Cited on page 21.)
- [Duong 2005] T V Duong, H H Bui, D Q Phung and S Venkatesh. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. In 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), pages 838–845. IEEE, jan 2005. (Cited on page 36.)
- [Efros 2003] Efros, Berg, Mori and Malik. *Recognizing action at a distance*. In ICCV 2003: 9th International Conference on Computer Vision, pages 726–733 vol.2. IEEE, jan 2003. (Cited on page 29.)

- [Emonet 2011] Remi Emonet, Jagannadan Varadarajan and Jean-Marc Odobez. *EmonetVaradarajanOdobez-CVPR-2011*. Computer Vision and Pattern Recognition, pages 1–8, apr 2011. (Cited on pages 24, 25, 98 and 111.)
- [Faisal I Bashir 2007] Student Member Ashfaq A Khokhar Senior Member Dan Schonfeld Senior Member Faisal I Bashir. *Real-time motion trajectory-based indexing and retrieval of video sequences*. IEEE Trans. Multimedia, vol. 9, pages 58–65, 2007. @ARTICLEBashir07real-timemotion, author = Faisal I. Bashir and Student Member and Ashfaq A. Khokhar and Senior Member and Dan Schonfeld and Senior Member, title = Real-time motion trajectory-based indexing and retrieval of video sequences, journal = IEEE Trans. Multimedia, year = 2007, volume = 9, pages = 58–65 . (Cited on page 82.)
- [Fashing 2005] M Fashing. *Mean shift is a bound optimization*. Pattern Analysis and Machine . . . , 2005. (Cited on page 85.)
- [Fukunaga 1975] K Fukunaga and L Hostetler. *The estimation of the gradient of a density function, with applications in pattern recognition*. IEEE Transactions on Information Theory, vol. 21, no. 1, pages 32–40, jan 1975. (Cited on page 82.)
- [Galata 2001] Aphrodite Galata, Neil Johnson and David Hogg. *Learning Variable-Length Markov Models of Behavior*. Computer Vision and Image Understanding, vol. 81, no. 3, pages 398–413, mar 2001. (Cited on page 31.)
- [Ghanem 2004] N. Ghanem, D. DeMenthon, D. Doermann and L. Davis. *Representation and Recognition of Events in Surveillance Video Using Petri Nets*. pages 112– 112, jan 2004. (Cited on page 40.)
- [Gong 2003] Shaogang Gong and Tao Xiang. *Recognition of group activities using dynamic probabilistic networks*. In ICCV 2003: 9th International Conference on Computer Vision, pages 742–749 vol.2. IEEE, jan 2003. (Cited on page 36.)
- [Guido 2010] Pusiolo Guido, Bramond Francois and Thonnat Monique. *AVSS10-PBT*. International Conference on Advanced Video and Signal-Based Surveillance, pages 1–8, jul 2010. (Cited on page 95.)
- [Gupta 2009] A Gupta, P Srinivasan, Jianbo Shi and L S Davis. *Understanding videos, constructing plots learning a visually grounded storyline model from annotated videos*. In 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPR Workshops), pages 2012–2019. IEEE, jan 2009. (Cited on page 41.)
- [Haag 2000] M Haag. *Incremental recognition of traffic situations from video image sequences*. Image and Vision Computing, vol. 18, no. 2, pages 137–153, jan 2000. (Cited on page 41.)

- [Hakeem 2004] Asaad Hakeem, Yaser Sheikh and Mubarak Shah. *A hierarchical event representation for the analysis of videos*. jan 2004. (Cited on page 40.)
- [Ham 2009a] A Novel Sequence Representation for Unsupervised Analysis of Human Activities , A.I Journal, 2009. (Cited on pages 34 and 55.)
- [Ham 2009b] A Novel Sequence Representation for Unsupervised Analysis of Human Activities , Artificial Intelligence Journal, Accepted Paper, 2009. (Cited on page 98.)
- [Hospedales 2009] Timothy Hospedales, Shaogang Gong and Tao Xiang. *A markov clustering topic model for mining behaviour in video*. In 2009 IEEE 12th International Conference on Computer Vision (ICCV), pages 1165–1172. IEEE, jan 2009. (Cited on page 24.)
- [Hu 2006] Weiming Hu, Xuejuan Xiao, Zhouyu Fu, D Xie, Tieniu Tan and S Maybank. *A system for learning statistical motion patterns*. IEEE Trans. Pattern Anal. Mach. Intell., vol. 28, no. 9, pages 1450–1464, feb 2006. (Cited on pages 27 and 28.)
- [Ivanov 2000] Y A Ivanov and A F Bobick. *Recognition of visual activities and interactions by stochastic parsing*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 8, pages 852–872, 2000. (Cited on page 38.)
- [JAIN 2011] A K JAIN. *Data Clustering: A Review*. pages 1–60, may 2011. (Cited on page 87.)
- [Jiang 2009] Fan Jiang, Ying Wu and A K Katsaggelos. *A Dynamic Hierarchical Clustering Method for Trajectory-Based Unusual Video Event Detection*. IEEE Transactions on Image Processing, vol. 18, no. 4, pages 907–913, feb 2009. (Cited on page 31.)
- [Johnson 1996] N Johnson. *Learning the distribution of object trajectories for event recognition*. Image and Vision Computing, vol. 14, no. 8, pages 609–615, aug 1996. (Cited on page 27.)
- [Jonathan H Fernyhough 1996] Anthony G. Cohn David C Hogg Jonathan H Fernyhough. *Generation of Semantic Regions from Image Sequences*. 1996. (Cited on page 27.)
- [Joo 2006] Seong-Wook Joo and R Chellappa. *2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'06)*. In 2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'06), pages 107–107. IEEE, jan 2006. (Cited on page 38.)
- [Jouneau 2011] Erwan Jouneau and Cyril Carincotte. *jouneau.dvi*. International Conference on Image Processing, pages 1–4, may 2011. (Cited on page 98.)

- [Katz 1963] S Katz, A B Ford, R W Moskowitz, B A Jackson and M W Jaffe. *Studies of Illness in the Aged: The Index of ADL: A Standardized Measure of Biological and Psychosocial Function*. JAMA: The Journal of the American Medical Association, vol. 185, no. 12, pages 914–919, sep 1963. (Cited on page 13.)
- [Ke 2007] Yan Ke, Rahul Sukthankar and Martial Hebert. *Spatio-temporal Shape and Flow Correlation for Action Recognition*. In 2007 IEEE Conference on Computer Vision and Pattern Recognition, pages 1–8. IEEE, feb 2007. (Cited on page 17.)
- [Kitani 2007] Kris Kitani, Yoichi Sato and Akihiro Sugimoto. *Recovering the Basic Structure of Human Activities from a Video-Based Symbol String*. In 2007 IEEE Workshop on Motion and Video Computing (WMVC'07), pages 9–9. IEEE, feb 2007. (Cited on page 38.)
- [Kuettel 2010] Daniel Kuettel, Michael D Breitenstein, Luc Van Gool and Vittorio Ferrari. *What's going on? Discovering spatio-temporal dependencies in dynamic scenes*. In 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 1951–1958. IEEE, 2010. (Cited on page 24.)
- [Laptev 2003] Ivan Laptev and Tony Lindeberg. *Space-time Interest Points*. In ICCV, pages 432–439, 2003. (Cited on page 20.)
- [Laptev 2005] Ivan Laptev. *On Space-Time Interest Points*. International Journal of Computer Vision, vol. 64, no. 2-3, pages 107–123, sep 2005. (Cited on page 20.)
- [Laptev 2008] Ivan Laptev, Marcin Marszalek, Cordelia Schmid and Benjamin Rozenfeld. *Learning realistic human actions from movies*. In 2008 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 1–8. IEEE, jan 2008. (Cited on page 23.)
- [Lavee 2009] Gal Lavee, Ehud Rivlin and Michael Rudzsky. *Understanding Video Events: A Survey of Methods for Automatic Interpretation of Semantic Occurrences in Video*. IEEE Transactions on Systems, Man, and Cybernetics, vol. 39, pages 489–504, 2009. (Cited on page 76.)
- [Levina 2001] E Levina and P Bickel. *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*. In Eighth IEEE International Conference on Computer Vision, pages 251–256. IEEE Comput. Soc, 2001. (Cited on page 122.)
- [Li 2006a] Xi Li, Weiming Hu and Wei Hu. *A Coarse-to-Fine Strategy for Vehicle Motion Trajectory Clustering*. In ICPR '06:, pages 591–594, 2006. (Cited on page 28.)



- [Li 2006b] Xi Li, Weiming Hu and Wei Hu. *A Coarse-to-Fine Strategy for Vehicle Motion Trajectory Clustering*. In ICPR '06: Proceedings of the 18th International Conference on Pattern Recognition, pages 591–594, 2006. (Cited on page 82.)
- [Liu 2008a] Jingen Liu, Saad Ali and Mubarak Shah. *Recognizing human actions using multiple features*. In 2008 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 1–8. IEEE, feb 2008. (Cited on page 22.)
- [Liu 2008b] Jingen Liu and Mubarak Shah. *Learning human actions via information maximization*. In 2008 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 1–8. IEEE, feb 2008. (Cited on page 23.)
- [Lowe 1999] D G Lowe. *Object recognition from local scale-invariant features*. In Proceedings of the Seventh IEEE International Conference on Computer Vision, pages 1150–1157 vol.2. IEEE, feb 1999. (Cited on page 21.)
- [Lublinerman 2006] R Lublinerman, M Sznaier and O Camps. *Dynamics Based Robust Motion Segmentation*. In 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 1 (CVPR'06), pages 1176–1184. IEEE, jan 2006. (Cited on page 29.)
- [Lv 2007] Fengjun Lv and Ramakant Nevatia. *Single View Human Action Recognition using Key Pose Matching and Viterbi Path Searching*. In 2007 IEEE Conference on Computer Vision and Pattern Recognition, pages 1–8. IEEE, 2007. (Cited on page 32.)
- [MacQueen 1966] James B MacQueen and WESTERN MANAGEMENT SCIENCE INST UNIV OF CALIFORNIA LOS ANGELES. *SOME METHODS FOR CLASSIFICATION AND ANALYSIS OF MULTIVARIATE OBSERVATIONS*, 1966. (Cited on page 59.)
- [Makris 2005a] D. Makris and T. Ellis. *Learning Semantic Scene Models From Observing Activity in Visual Surveillance*. Systems, Man, and Cybernetics, Part B, IEEE Transactions on, vol. 35, no. 3, pages 397–408, jun 2005. (Cited on pages 27, 28 and 34.)
- [Makris 2005b] D. Makris and T. Ellis. *Learning Semantic Scene Models From Observing Activity in Visual Surveillance*. SMC-B, vol. 35, no. 3, pages 397–408, jun 2005. (Cited on page 28.)
- [Makris 2005c] D. Makris and T. Ellis. *Learning semantic scene models from observing activity in visual surveillance*. Systems, Man, and Cybernetics, Part B, IEEE Transactions on, vol. 35, no. 3, pages 397–408, jun 2005. (Cited on page 55.)

- [Manning 2005] Christopher D Manning and Hinrich Schutze. *Foundations of Statistical Natural Language Processing*. pages 1–704, may 2005. (Cited on pages 109 and 170.)
- [Messing 2009] Ross Messing, Chris Pal and Henry Kautz. *Activity recognition using the velocity histories of tracked keypoints*. In 2009 IEEE 12th International Conference on Computer Vision (ICCV), pages 104–111. IEEE, jan 2009. (Cited on page 23.)
- [Minnen 2003] D. Minnen, I. Essa and T Starner. *Expectation grammars: leveraging high-level expectations for activity recognition*. In CVPR 2003: Computer Vision and Pattern Recognition Conference, pages II–626–II–632. IEEE Comput. Soc, jan 2003. (Cited on pages 38 and 40.)
- [Morris 2008a] B T Morris and M M Trivedi. *A Survey of Vision-Based Trajectory Learning and Analysis for Surveillance*. IEEE Transactions on Circuits and Systems for Video Technology, vol. 18, no. 8, pages 1114–1127, aug 2008. (Cited on page 29.)
- [Morris 2008b] Brendan T Morris and Mohan M Trivedi. *Learning and Classification of Trajectories in Dynamic Scenes: A General Framework for Live Video Analysis*. Advanced Video and Signal Based Surveillance, 2008. (Cited on page 34.)
- [MP 1969] Lawton MP and Brody EM. *Assessment of older people: self-maintaining and instrumental activities of daily living*. feb 1969. (Cited on page 13.)
- [Nam 1999] Yanghee Nam, Nwangyun Wohn and Hyung Lee-Kwang. *Modeling and recognition of hand gesture using colored Petri nets*. IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans, vol. 29, no. 5, pages 514–521, 1999. (Cited on page 40.)
- [Natarajan 2007] Pradeep Natarajan and Ramakant Nevatia. *Coupled Hidden Semi Markov Models for Activity Recognition*. In 2007 IEEE Workshop on Motion and Video Computing (WMVC'07), pages 10–10. IEEE, jan 2007. (Cited on pages 32 and 33.)
- [Nevatia 2003] Ram Nevatia, Tao Zhao and Somboon Hongeng. *Hierarchical Language-based Representation of Events in Video Streams*. In 2003 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW), pages 39–39. IEEE, jan 2003. (Cited on page 39.)
- [Nghiem 2009] Anh-Tuan Nghiem, François Brémond and Monique Thonnat. *Controlling Background Subtraction Algorithms for Robust Object Detection*. dec 2009. (Cited on page 45.)

- [Nguyen 2005] N T Nguyen, D Q Phung, S Venkatesh and H Bui. *Learning and detecting activities from movement trajectories using the hierarchical hidden Markov model*. In 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), pages 955–960. IEEE, jan 2005. (Cited on page 36.)
- [Niebles 2008] Juan Carlos Niebles, Hongcheng Wang and Li Fei-Fei. *Unsupervised Learning of Human Action Categories Using Spatial-Temporal Words*. International Journal of Computer Vision, vol. 79, no. 3, pages 299–318, mar 2008. (Cited on pages 21 and 23.)
- [Oikonomopoulos 2009] Antonios Oikonomopoulos, Maja Pantic and Ioannis Patras. *Sparse B-spline polynomial descriptors for human activity recognition*. Image and Vision Computing, vol. 27, no. 12, pages 1814–1825, nov 2009. (Cited on page 23.)
- [Oliver 2000] N.M. Oliver, B. Rosario and A.P. Pentland. *A Bayesian computer vision system for modeling human interactions*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 8, pages 831–843, 2000. (Cited on pages 32 and 33.)
- [Oliver 2002] N Oliver, E Horvitz and A Garg. *Proceedings. Fourth IEEE International Conference on Multimodal Interfaces*. In Fourth IEEE International Conference on Multimodal Interfaces, pages 3–8. IEEE Comput. Soc, 2002. (Cited on page 36.)
- [Park 2004] Sangho Park and J K Aggarwal. *A hierarchical Bayesian network for event recognition of human actions and interactions*. Multimedia Systems, vol. 10, no. 2, pages 164–179, aug 2004. (Cited on page 32.)
- [Patino 2008] L. Patino, H. Benhadda, E. Corvee, F Bremond and M Thonnat. *Extraction of activity patterns on large video recordings*. Computer Vision, IET, vol. 2, no. 2, pages 108–128, jun 2008. (Cited on page 28.)
- [Piciarelli 2006a] C. Piciarelli and G FORESTI. *On-line trajectory clustering for anomalous events detection*. Pattern Recognition Letters, vol. 27, no. 15, pages 1835–1842, nov 2006. (Cited on page 28.)
- [Piciarelli 2006b] C. Piciarelli and G.L. Foresti. *On-line trajectory clustering for anomalous events detection*. Pattern Recogn. Lett., vol. 27, no. 15, pages 1835–1842, 2006. (Cited on page 28.)
- [Pinhanez 1998] C S Pinhanez and A F Bobick. *Human action detection using PNF propagation of temporal constraints*. In 1998 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pages 898–904. IEEE Comput. Soc, jan 1998. (Cited on page 39.)

- [Porikli 2004] F. Porikli. *Learning object trajectory patterns by spectral clustering*. Multimedia and Expo, 2004. ICME '04. 2004 IEEE International Conference on, vol. 2, pages 1171–1174 Vol.2, jun 2004. (Cited on page 34.)
- [Pusiol 2008] G Pusiol, L. Patino, F Bremond and M Thonnant. *Optimizing Trajectories Clustering for Activity Recognition*. 1st International workshop on machine learning for vision-based motion analysis (MLVMA 08) in association with ECCV 2008, 2008. (Cited on pages 82 and 86.)
- [Pusiol 2010] Guido Pusiol, François Brémond and Monique Thonnat. *Trajectory Based Activity Discovery*. IEEE AVSS'10, vol. 0, pages 270–277, 2010. (Cited on pages 148, 150 and 151.)
- [Rao 2001] C. Rao and M Shah. *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*. In 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001, pages II–316–II–322. IEEE Comput. Soc, 2001. (Cited on page 19.)
- [Rapantzikos 2009] K Rapantzikos, Y Avrithis and S Kollias. *Dense saliency-based spatiotemporal feature points for action recognition*. In 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPR Workshops), pages 1454–1461. IEEE, feb 2009. (Cited on page 22.)
- [Rodriguez 2012] Mikel D Rodriguez, Javed Ahmed and Mubarak Shah. *2008 IEEE Conference on Computer Vision and Pattern Recognition*. In 2008 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 1–8. IEEE, feb 2012. (Cited on page 18.)
- [Romdhane 2010] R Romdhane, E Mulin, N. Zouba A Derreumeaux, J Piano, L Lee, I Leroi, P Mallea, R David, M Thonnat, F Bremond and P H Robert. *Automatic Video Monitoring system for assessment of Alzheimer's Disease symptoms*. JNHA - The Journal of Nutrition, Health and Aging Ms. No. JNHA-D-11-00004R1 - 2011., may 2010. (Cited on pages 128 and 131.)
- [Ross 2004] P E Ross. *Managing care through the air [remote health monitoring]*. IEEE Spectrum, vol. 41, no. 12, pages 26–31, 2004. (Cited on page 2.)
- [Ryoo 2008] M S Ryoo and J K Aggarwal. *Semantic Representation and Recognition of Continued and Recursive Human Activities*. International Journal of Computer Vision, vol. 82, no. 1, pages 1–24, nov 2008. (Cited on page 40.)
- [Savarese 2008] Silvio Savarese, Andrey DelPozo, Juan Carlos Niebles and Li Fei-Fei. *Spatial-Temporal correlatons for unsupervised action classification*. In 2008 IEEE Workshop on Motion and video Computing (WMVC), pages 1–8. IEEE, feb 2008. (Cited on page 23.)

- [Schuldt 2004] C. Schuldt, I. Laptev and B. Caputo. *Recognizing human actions: a local SVM approach*. In Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004., pages 32–36 Vol.3. IEEE, feb 2004. (Cited on pages 18 and 21.)
- [Scovanner 2007] Paul Scovanner, Saad Ali and Mubarak Shah. *Proceedings of the 15th international conference on Multimedia - MULTIMEDIA '07*. In the 15th international conference, pages 357–360, New York, New York, USA, 2007. ACM Press. (Cited on page 21.)
- [Shechtman 2005] E Shechtman and M Irani. *Space-time behavior based correlation*. In 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), pages 405–412. IEEE, feb 2005. (Cited on page 17.)
- [Sheikh 2005] Y Sheikh, M Sheikh and M Shah. *Exploring the space of a human action*. In Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1, pages 144–149 Vol. 1. IEEE, feb 2005. (Cited on page 18.)
- [Shi 1994] Jianbo Shi and Carlo Tomasi. *Good Features to Track*. In IEEE CVPR'94, pages 593–600, 1994. (Cited on page 48.)
- [Shi 2004] Yifan Shi, Yan Huang, D. Minnen, A. Bobick and I. Essa. *Propagation networks for recognition of partially ordered sequential action*. In Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004., pages 862–869. IEEE, jan 2004. (Cited on page 37.)
- [Shotton 2011] Jamie Shotton, Andrew Fitzgibbon, Mat Cook, Toby Sharp, Mark Finocchio, Richard Moore, Alex Kipman and Andrew Blake. *Real-time human pose recognition in parts from single depth images*. In 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 1297–1304. IEEE, feb 2011. (Cited on pages 19, 46 and 47.)
- [Singh 2003] Singh and Ahuja. *Proceedings Ninth IEEE International Conference on Computer Vision*. In ICCV 2003: 9th International Conference on Computer Vision, pages 2–9 vol.1. IEEE, oct 2003. (Cited on page 84.)
- [Siskind 2011] J. M. Siskind. *Grounding the Lexical Semantics of Verbs in Visual Perception using Force Dynamics and Event Logic*. jun 2011. (Cited on page 41.)
- [Smeulders 2000] A W M Smeulders, M Worring, S Santini, A Gupta and R Jain. *Content-based image retrieval at the end of the early years*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 12, pages 1349–1380, 2000. (Cited on page 3.)

- [Sminchisescu 2005] C Sminchisescu, A Kanaujia, Zhiguo Li and D Metaxas. *Conditional models for contextual human motion recognition*. In Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1, pages 1808–1815 Vol. 2. IEEE, jan 2005. (Cited on page 33.)
- [Starner 1995] T Starner and A. Pentland. *Proceedings of International Symposium on Computer Vision - ISCV*. In International Symposium on Computer Vision - ISCV, pages 265–270. IEEE Comput. Soc. Press, feb 1995. (Cited on page 31.)
- [Stikic 2008] Maja Stikic, Tam Huynh, Kristof Van Laerhoven and Bernt Schiele. *ADL recognition based on the combination of RFID and accelerometer sensing*. In 2008 Second International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth), pages 258–263. IEEE, feb 2008. (Cited on page 13.)
- [Subbarao 2009] Raghav Subbarao and Peter Meer. *Nonlinear Mean Shift over Riemannian Manifolds*. International Journal of Computer Vision, vol. 84, no. 1, pages 1–20, mar 2009. (Cited on page 83.)
- [Sun 2009] Ju Sun, Xiao Wu, Shuicheng Yan, Loong-Fah Cheong, Tat-Seng Chua and Jintao Li. *Hierarchical spatio-temporal context modeling for action recognition*. In 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPR Workshops), pages 2004–2011. IEEE, jan 2009. (Cited on page 23.)
- [Teh 2005] Yee Whye Teh, Michael I Jordan, Matthew J Beal and David M Blei. *Sharing Clusters Among Related Groups hierarchical Dirichlet Processes*. pages 1–8, jan 2005. (Cited on page 111.)
- [Tran 2008] Son D Tran and Larry S Davis. Event Modeling and Recognition Using Markov Logic Networks, volume 5303 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008. (Cited on page 40.)
- [Turaga 2008] P Turaga, R Chellappa, V S Subrahmanian and O Udrea. *Machine Recognition of Human Activities: A Survey*. IEEE Transactions on Circuits and Systems for Video Technology, vol. 18, no. 11, pages 1473–1488, 2008. (Cited on page 13.)
- [Veeraraghavan 2006a] A Veeraraghavan and A K Roy-Chowdhury. *The Function Space of an Activity*. In 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 1 (CVPR'06), pages 959–968. IEEE, jan 2006. (Cited on page 30.)
- [Veeraraghavan 2006b] Harini Veeraraghavan, Nikolaos Papanikolopoulos and Paul Schrater. *Learning Dynamic Event Descriptions in Image Sequences*. In 2007 IEEE Conference on Computer Vision and Pattern Recognition, pages 1–6. IEEE, jan 2006. (Cited on page 55.)

- [Viterbi 1967] A Viterbi. *Error bounds for convolutional codes and an asymptotically optimum decoding algorithm*. IEEE Transactions on Information Theory, vol. 13, no. 2, pages 260–269, jan 1967. (Cited on page 31.)
- [Vu 2003] Thinh Vu, François Brémond and Monique Thonnat. *Automatic Video Interpretation: A Novel Algorithm for Temporal Scenario Recognition*. International Joint Conference on Artificial Intelligence (IJCAI), pages 1295–1302, jan 2003. (Cited on pages 39, 55 and 151.)
- [Wang 2004] Jue Wang, Bo Thiesson, Yingqing Xu and Michael Cohen. Lecture Notes in Computer Science, volume 3022 of *Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Dough Tygar, Moshe Y. Vardi, Gerhard Weikum* *Lecture Notes in Computer Science 3022-9743 1611-3349*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004. (Cited on page 84.)
- [Wang 2007] Liang Wang and David Suter. *Recognizing Human Activities from Silhouettes: Motion Subspace and Factorial Discriminative Graphical Model*. In 2007 IEEE Conference on Computer Vision and Pattern Recognition, pages 1–8. IEEE, jan 2007. (Cited on page 33.)
- [Wang 2009a] Xiaogang Wang, Xiaoxu Ma and W E L Grimson. *Unsupervised Activity Perception in Crowded and Complicated Scenes Using Hierarchical Bayesian Models*. IEEE Trans. Pattern Anal. Mach. Intell., vol. 31, no. 3, pages 539–555, 2009. (Cited on page 24.)
- [Wang 2009b] Yang Wang and G Mori. *Human Action Recognition by Semilattent Topic Models*. IEEE Trans. Pattern Anal. Mach. Intell., vol. 31, no. 10, pages 1762–1774, oct 2009. (Cited on page 24.)
- [Wong 2007] Shu-Fai Wong and Roberto Cipolla. *Extracting Spatiotemporal Interest Points using Global Information*. In 2007 IEEE 11th International Conference on Computer Vision, pages 1–8. IEEE, jan 2007. (Cited on page 23.)
- [Yamato 1992] J Yamato, J Ohya and K Ishii. *Recognizing human action in time-sequential images using hidden Markov model*. In 1992 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pages 379–385. IEEE Comput. Soc. Press, jan 1992. (Cited on page 31.)
- [Yao 2009] Benjamin Yao and Song-Chun Zhu. *Learning deformable action templates from cluttered videos*. In 2009 IEEE 12th International Conference on Computer Vision (ICCV), pages 1507–1514. IEEE, jan 2009. (Cited on page 23.)
- [Yilma 2005] A Yilma and M Shah. *Recognizing human actions in videos acquired by uncalibrated moving cameras*. In Tenth IEEE International Conference

- on Computer Vision (ICCV'05) Volume 1, pages 150–157 Vol. 1. IEEE, jan 2005. (Cited on page 19.)
- [Yu 2006] E Yu and J K Aggarwal. *Detection of fence climbing from monocular video*. In 18th International Conference on Pattern Recognition (ICPR'06), pages 375–378. IEEE, jan 2006. (Cited on page 36.)
- [Zaidi 1999] A K Zaidi. *On temporal logic programming using Petri nets*. IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans, vol. 29, no. 3, pages 245–254, may 1999. (Cited on page 40.)
- [Zelnik-Manor 2006] L Zelnik-Manor and M Irani. *Statistical analysis of dynamic actions*. IEEE Trans. Pattern Anal. Mach. Intell., vol. 28, no. 9, pages 1530–1535, dec 2006. (Cited on page 20.)
- [Zhang 2006] Dong Zhang, D Gatica-Perez, S Bengio and I McCowan. *Modeling individual and group actions in meetings with layered HMMs*. IEEE Trans. Multimedia, vol. 8, no. 3, pages 509–520, jan 2006. (Cited on page 36.)
- [Zouba 2009] N Zouba, F Bremond and M Thonnat. *Multisensor Fusion for Monitoring Elderly Activities at Home*. In AVSS'09., Genoa, Italy., sep 2009. (Cited on page 13.)
- [Zouba 2010] Nadia Zouba, François Brémond and Monique Thonnat. *2010 7th IEEE International Conference on Advanced Video and Signal Based Surveillance*. In 2010 7th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), pages 278–285. IEEE, sep 2010. (Cited on pages vii, 128, 150, 151 and 152.)
- [Zúñiga 2011] Marcos D Zúñiga, François Brémond and Monique Thonnat. *Real-time reliability measure-driven multi-hypothesis tracking using 2D and 3D features*. EURASIP Journal on Advances in Signal Processing, vol. 2011, no. 1, page 142, 2011. (Cited on page 45.)



---

## Discovery of human activities in video

**Abstract:** The main objective of this thesis is to propose a complete framework for activity discovery, modeling and recognition using video information. The framework uses perceptual information (e.g. trajectories) as input and goes up to activities (semantics). The framework is divided into five main parts:

First, we break the video into chunks to characterize activities. We propose different techniques to extract perceptual features from the chunks. This way, we build packages of perceptual features capable of describing activity occurring in small periods of time.

Second, we propose to learn the video contextual information. We build scene models by learning salient perceptual features. The models end up containing interesting scene regions capable of describing basic semantics (i.e. region where interactions occur).

Third, we propose to reduce the gap between low-level vision information and semantic interpretation, by building an intermediate layer composed of Primitive Events. The proposed representation for primitive events aims at describing the meaningful motions over the scene. This is achieved by abstracting perceptual features using contextual information in an unsupervised manner.

Fourth, we propose a pattern -based method to discover activities at multiple resolutions (i.e. activities and sub-activities). Also, we propose a generative method to model multi-resolution activities. The models are built as a flexible probabilistic framework easy to update.

Finally, we propose an activity recognition method that finds in a deterministic manner the occurrences of modeled activities in unseen datasets. Semantics are provided by the method under user interaction.

All this research work has been evaluated using real datasets of people living in an apartment (home-care application) and elder patients in a hospital.

**Keywords:** Activity Discovery, Activity Models, Activity Recognition, learning, unsupervised, video retrieval.

---