Human activity recognition for the use in intelligent spaces

Meinders, M.J.

Award date:
2012

Graduation Project DDSS
Architecture, Building and Planning
Eindhoven University of Technology

Human Activity Recognition for the use in Intelligent Spaces

Student
M.J. Meinders (0531671)
m.j.meinders@student.tue.nl

Graduation committee
prof. dr. ir. B. de Vries
dr. O.D. Amft
ir. A.J. Jessurun

Version
June 2012

Abstract

The aim of this Graduation Project is to develop a generic biological inspired activity recognition system for the use in intelligent spaces. Intelligent spaces form the context for this project. The goal is to develop a working prototype that can learn and recognize human activities from a limited training set in all kinds of spaces and situations. For testing purposes, the office environment is chosen as subject for the intelligent space.

The purpose of the intelligent space, in this case the office, is left out of the scope of the project. The scope is limited to the perceptive system of the intelligent space. The notion is that the prototype should not be bound to a specific space, but it should be a generic perceptive system able to cope in any given space within the build environment. The fact that no space is the same, developing a prototype without any domain knowledge in which it can learn and recognize activities, is the main challenge of this project.

In al layers of the prototype, the data processing is kept as abstract and low level as possible to keep it as generic as possible. This is done by using local features, scale invariant descriptors and by using hidden Markov models for pattern recognition.

The novel approach of the prototype is that it combines structure as well as motion features in one system making it able to train and recognize a variety of activities in a variety of situations. From rhythmic expressive actions with a simple cyclic pattern to activities where the movement is subtle and complex like typing and reading, can all be trained and recognized.

The prototype has been tested on two very different data sets. The first set in which the videos are shot in a controlled environment in which simple actions were performed. The second set in which videos are shot in a normal office where daily office activities are captured and categorized afterwards. The prototype has given some promising results proving it can cope with very different spaces, actions and activities.

Table of contents

1. Introduction

1.1 Motivation

As part of my master, I studied half a year abroad at the University of Sydney. One of the projects I worked on involved video tracking. I developed a simple background subtraction and blob detection model that could follow people around on a square. The purpose was to visualize the paths taken by people walking over the square and that resulted in some interesting results. The project opened a new door for me called computer vision.

When Bauke de Vries proposed that I would do my final project on intelligent spaces and that I would focus on the perceptive system, I got excited. Not only could I dive deeper into the field of computer vision, I could also get my hands dirty on computer intelligence.
The latter has always fascinated me. Not only are the possibilities of artificial intelligence immense, it can also give insight in to how the human brain works, a marble of evolution.

The focus of the intelligent space will be on the recognition of activities of people in that space. This field of computer vision is that of human action/activity recognition. Inspired by the human brain, I wanted to develop a system in which (some of) the techniques are based on the workings of the human perceptive system.

1.2 Research objectives and questions

Human activity recognition simply means transforming (signal or pixel) data into useful activity related information. The purpose of a human activity recognition system, its environment and the representational type and format of the information produced by it, the "useful information", determines the model chain, the techniques used and thus its complexity.

This project will focus on the whole model chain. Although the purpose of the system will be the use for intelligent spaces, the purpose of the space itself will not be part of the project scope, but it will merely give direction to the development of the model chain.

The objective of this project is to develop a working human activity recognition prototype. Although human activity recognition is a hot topic in computer vision and a lot of research has already been done, its use in every day live is still limited to specific scenario's and use cases. In those cases, it has proven to be a helpful tool, but due to the specific purposes in each case, the same systems are not easy to implement in other cases. Human action/activity recognition still has a long way to mature and in my view misses "plug and play" capabilities.

To approach the problem of human activity recognition for intelligent spaces I chose for a practical angle. The focus will be on the generic accent of the model, because that is where the main challenge lies. The more generic the model is, the less knowledge about the prototype's environment and expected observations can be build into the model. This means that the model has to deal a lot more with uncertainties, making the problem a lot more complex.

Although the prototype has to deal with *human* activity recognition it will not deal with the concept of a human being or its parts at all. Nor will it grasp the concepts of human movement. It will merely match structure and motion patterns to patterns it already knows to be a specific action / activity. The latter

however is also a part of the problem, because the prototype must have the ability to learn those patterns first.

The ability to process the video data real time is not part of the objectives. In principle computational complexity is left out of the scope. The priority lies in the recognition of activities. However, to keep to development process practical, in some areas decisions are made to reduce computational complexity to speed up development and testing. These are situations where small code optimizations significantly reduce computing time, while the loss in quality is minor.

Objective / main goal

To develop a biological inspired generic prototype for robust human activity recognition for in the use in intelligent spaces.

Problem definition

How to develop a method that is generic, context independent, that can learn and recognize human activities and provide a robust ground to be build upon for further complex recognition of higher level activities.

Sub problems

What defines a generic method and how can it be realized?

- Which assumptions can be incorporated in the model without limiting its generality?
- How can the context dependency be minimized?

How can motion and structure information be linked synergetic for activity recognition?

- Which features should be extracted to capture motion?
- Which features should be extracted to capture structure?

How does 'the intelligent space' play part in the scope of the project?

- How can the intelligent space be defined?
- Does the 'intelligent space' add to the problems complexity?
- How can the scope of 'the intelligent space' help in the reduction of the problems complexity?

Which learning and classification techniques should be applied for activity recognition?

1.3 Research approach

The approach for this project is twofold:

1. The biological inspired approach, where I will try to borrow techniques that are derived from research of the human perceptive system.

2. The generic low-level approach whereby the classes, with other words the activities that have to be recognized, are kept abstract. No context or preliminary assumptions may be hardcoded into the model.

1.4 Research relevance and contributions

As stated before, human action/activity recognition is largely in a research phase. There are real life uses for it, but for most of those cases the system is so specifically build for its purpose that using the system for other

purposes is not possible without making fundamental changes. This makes these systems expensive and not flexible.

By keeping the scope of activities and environments as open as possible, a system can be developed that can be used in far more situations. This is where the biggest challenge lies within human activity recognition, trying to cope with variations and the unforeseen.

Human activity recognition is an unexplored field at DDSS and therefore this project could help future projects related to intelligent spaces.

1.5 Report Outline

In chapter two “Theory” a wide scope of background information is given on several aspects of the project. Short descriptions will be given about intelligent spaces and human activity recognition. The general model outline and several techniques used in the prototype are discussed as well.

In the chapter “Prototype” the model outline and the techniques used are discussed following the model outline previously discussed in the chapter “Theory”. It discusses how the techniques are implemented into the prototype.

In the chapter “Validation” the prototype is tested. The chapter discusses how the prototype was tested, which datasets were used and what the results are.

Finally, the chapter “Conclusion” discusses the results; it reflects back on the prototype and discusses some future improvements.

Each chapter consists of numbered sections and they may contain multiple sub-sections that are unnumbered.

2. Theory

2.1 Introduction

In this chapter I would like to discuss some background information on this project. Intelligent spaces will be discussed first because it forms the context for this project. Different types of intelligent spaces are covered. In addition, because the office environment is the test case for the prototype, the section will elaborate a little on intelligent offices as well.

Human activity recognition is discussed outside the context of this project to give a general summary of this branch within computer sciences. The section discusses the general purpose of human activity recognition and will have some overlap with the section about intelligent spaces.

A model outline is presented for human activity recognition in general. Although the research on HAR as sprouted many different approaches, the presented model chain gives a rough outline to which most of the approaches can be reduced. The prototype will also follow this model chain.

An important step in the prototype is called multi scale image processing and it is discussed in section 2.6. The technique is used as part of the feature detection in the prototype allowing it to process video data in a scale invariant manner. The section tries to explain the concept of scale space and how it works. The actual implementation is discussed in the chapter of the prototype.

Scale space operators closely resemble receptive field profiles found in the mammalian retina. This is one of the similarities with biological vision. Because biological vision inspired some more techniques used in the prototype, section 2.7 highlights some parts of the human perspective system, which have found their way into computer vision and into the prototype.

Another important technique used a lot in the recognition of temporal / sequential patterns are hidden Markov models. At first HMMs where used especially in speech processing, but eventually it found its way in human action and activity recognition and has proven to be well suited for the job. A large part of the approaches use HMMs and a lot of research is still being done on different variations of HMMs. HMMs are used in the prototype as well and a section is devoted to explain what HMMs are and how they work.

2.2 Intelligent spaces

Introduction

Intelligent spaces are spaces of the build environment that are equipped with sensors, which enable the space to perceive and understand what is happening within its confinement. With that knowledge the space can interact with the people using it. There are different levels of interaction possible.

There are various concepts of an intelligent space. According to Emile Aarts and Stefano Marzano, a smart environment should not only know what is happening in its space, but it should also know who is in the space along with the accompanying properties and attributes. Emile Aarts and Stefano Marzano have laid the base for the vision of future consumer electronics called ‘Ambient Intelligence’. [1]

A more scientific approach to the concept of the intelligent space is MIT’s Intelligent Room. It is an experimental space where different setups and techniques can be tried to develop a user-centric space where the room is considered an interface.

Input

The purpose of the intelligent space determines more or less the type of input the space requires or perhaps to which the space is limited. Video often is the source of input of the perceptive system in intelligent spaces. In general, no big measures have to be taken for spaces or for the people / devices inside to install video capturing devices, with other words they are easy to install. Besides that, video captures a broad amount of information giving you a lot to work with (which can be a disadvantage as well). Video information can be supplemented with audio to broaden its information scope, but audio can be used individually as well.

With the use of multiple video's 3D data can be extracted to give the data an extra dimension. Another variation of the use of video van be infrared, it is another great way for input because the heat signatures make it easier to distinguish people from background data.

Other techniques are also possible: think of embedded sensors in all kinds of devices or perhaps (motion-) sensors on the people themselves. The huge advantage of using (multiple) embedded sensors is that the choice for sensors allows you to target and extract just the information you need. In that perspective sensors are definitely more *efficient* that video capture, but if they are more *effective* is a whole other point of discussion.

Interaction levels

Intelligent spaces can be categorized by their interaction level. The categorization partially overlaps the application areas of human activity recognition described in section 2.3.

Passive intelligent spaces only analyze the space being used over a given period. This information can be used to enhance space utilization or perhaps improve human performance or comfort. Information of analyzed spaces can also have great value for the design of new related spaces.

Reactive intelligent spaces monitor the space and signal predefined people or systems if specific situations occur. Think of intelligent surveillance spaces for security or health care applications.

Intelligent spaces can also be *interactive*. They can interact with the people using the space, having a two-way information stream from human to computer and vice versa; communicating.

Finally, spaces can be *proactive*. They can adjust or change in anticipation of predicted situations.

Ambient Intelligence

Environments of ambient intelligence are intelligent spaces in a personalized experience context. They predict future spaces that will enrich and ease everyday live with the help of technology. Ambient Intelligence emphasizes on greater user-friendliness, support for human interactions, more efficient services support, and user-empowerment.

Aarts and Stefano state five layers of intelligence to be incorporated in ambient intelligence [1]:
Embedded: Devices are invisible and integrated into the environment.
Context aware: Recognizes you and your situational context.
Personalized: The space can be tailored towards your needs and desires.
Adaptive: The space can change in response to you.
Anticipatory: The space anticipates your desires without conscious mediation.

The application environments for ambient intelligence are environments of daily activities. A typical context for an ambient intelligence environment is the home environment, but office and especially health care environments ('smart hospitals') are also the subjects of research for ambient intelligence.

The Intelligent Room

Another interesting approach of intelligent spaces, and like ambient intelligence is user centric, is MIT's Intelligent Room. The overall idea is to make computation ready-at-hand. Computer intelligence should be available without the user having to shift his or her mode of thinking or interaction. Brooks *et al.* reject the idea of building special spaces in which such intelligent interaction can occur. The computation should adapt to the environment and to the people using it. [8]

The idea of the intelligent room has a lot in common with the vision of ambient intelligence, but it emphasizes more on interfacing; the human – computer interaction.

Intelligent offices

Complex buildings with a lot of people management such as hospitals, schools and large offices have challenging functional requirements. Intelligent buildings can help with way finding, space use and crime prevention. Offices are often the subject of intelligent spaces because it can help increase production and/or comfort and reduce the risk of functional obsolescence, hereby raising the overall value of the workspace.

A nice example is the research project "User Simulation of Space Utilization" done by Ph.D. Tabak done at the TU/e. The aim of his research project was to develop a model for the simulation of human movement and utilization of space capacity in office buildings using the organization workflow and building design as input.

Criticism

Beside the promising possibilities of intelligent spaces, it is a subject of criticism as well. Criticism is mostly targeted at the visions of personalized intelligent spaces, but is applicable to the whole spectrum of intelligent spaces. Because the more intelligent a space gets, the more human ethics and morals start to play a role. Terms like 'big brother' are then part of the discussion.

The biggest concern is about privacy and how it is affected in several ways. Information about people can be misused in the wrong hands or can be leaked to the public. Privacy in the sense of dignity is affected; a system that knows who you are, but you do not know anything about that system; an imbalance of information equilibrium. In this perspective is does not even matter how anonymous the information gathered, the feeling of an unknown system infiltrating a person's territory, can threaten their peace of mind. The same counts for privacy as a utility: that you cannot to be 'left alone'.

Beside privacy implications, social implications of intelligent spaces are a major topic of discussion as well. Will people at some stage become depended on intelligent spaces for their way of living? Will intelligent spaces furthermore stimulate the individualism of society? These questions derive from future visions of intelligent spaces, and form another research field not discussed in this report.

2.3 Human activity recognition

Introduction

The purpose of human activity recognition is to transform pixel data into useful information. The type of information delivered by the model depends on its purpose.

Action vs. Activity

The literature is not in agreement on the definitions of action and activity and the terms are used interchangingly. Different definitions have been proposed and in this report the definitions are partly derived from Moeslund *et al.* [29]. Moeslund *et al.* see an action as a hierarchal step towards an activity. An *action* describes a whole-body - possibly cyclic - movement or posture. An *activity* contains a single or number of subsequent actions, and gives context to the action(s) being performed. Example of an action is a forehand; the overall activity would then be playing tennis. In this report, although the focus is on *activity* recognition, both terms are used regularly.

Application area's

Mimicking the way humans perceive the actions or activities of others allows for a wide range of applications. Moeslund *et al.* [28] considers the following three major application areas: *surveillance*, *analysis* and *control*.

Surveillance

The area of surveillance could certainly need automation seeing the number of surveillance cameras increasing. The surveillance area covers applications for tracking one or several subjects and detecting unusual behavior. This type of observation task is not well suited to humans because of the concentration required over long periods of time. It is easy to miss essential indications for misplaced behavior if most of the time nothing happens, what can have serious consequences.
Surveillance is usually for security purposes but it can also be seen in a more positive light. Monitoring activities of daily living (ADL's) is gaining interest as well because of the growing population of elderly people and their need for care. A system that contributes to the safety of elderly at home is therefore more than needed. Medical professionals believe that one of the best ways to detect emerging physical and mental health problems, before they become critical – particularly for the elderly – is analyzing the human behavior and looking for changes in the activities of daily living. (Zouba *et al.* [47])

Analysis

Detailed observation tasks with the goal of performance improvement could also use the help of intelligent systems that know where to look for and keep track of it. Human performance analysis in office settings can contribute to a more efficient work process. Oliver *et al.* [32, 33, 34] has proposed multi-modal systems for human performance analysis for the office environment en research the utility of such a system in relation to its computational costs.

Analysis of human motion may also be used in clinical studies or it may help athletes improve their performances.

Another area for analysis that is applicable for intelligent spaces is performance improvement of space utilization. By analyzing how and when the space is used, spaces can be adapted to optimize their usage. The results can also be used for designing new spaces, better adapted for their intentional use.

Control

The control area of human activity recognition relates to applications where specific human actions or gestures function as input for control or interfacing in a computer environment. In many papers it is also referred to as human – computer interaction.

Research in this area in relation to intelligent spaces is also part of a vision for future consumer products, commonly referred to as Ambient Intelligence.

Control systems are systems where the participants often consciously choose to use it for a certain purpose, whereas that is not always the case in analysis and surveillance systems. Participants consciously choose to shift their mode of thinking and handling to be able to use that system. Using active sensing like RFID tags, accelerators or the awareness of (other) intrusive sensors is more easily accepted to capture motion than when you are the subject of human activity recognition without your consent. Therefore, the range of input technologies applicable for the control area -and to some degree the analysis area- is far larger than in the other areas. Thus, these systems are more flexible in their approach of recognition.

For most surveillance applications, passive sensing is used for capturing motion. Passive sensing or non-intrusive sensing is based on natural signal sources like visual light.

Each area has an interaction level between system and subject(s). The interaction level of surveillance systems is very low, most of the times it not even applicable. In the analysis area, the interaction starts to play a role, usually the one of feedback: The system analyzes the subject(s) and shows the results at the end of a session. The subject uses the feedback to improve his or hers performance. In control systems, the interaction plays a very important role between system and subject. In fact, the interaction is the primary goal of a control system.

2.4 General model outline

Introduction

What is a model in the first place in the context of this report? A model is system or chain of algorithms or (sub-) models that has data as input and gives information as output. Not to be confused with hidden Markov models. The output given is usually in the form of *a human understandable description* or *a decision* and its focus depends on the context / purpose of the model.

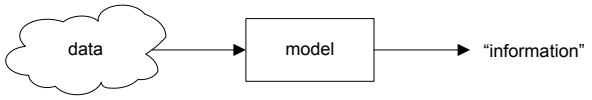


Diagram 1: Simplified model

Although the model chain of human activity recognition can heavily vary on the techniques used, a general model outline can be derived. The general flow of a human activity recognition model is shown in the following diagram. Each of these steps will be further explained in the next sections.

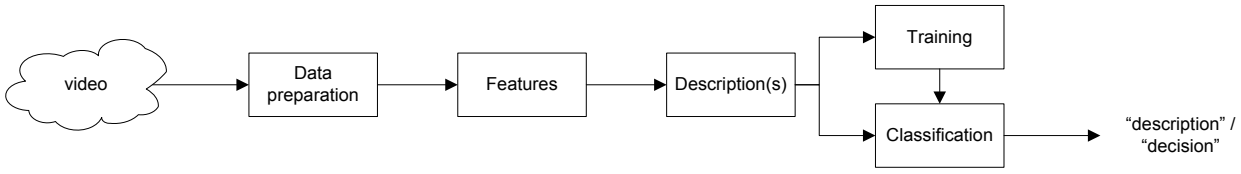


Diagram 2: General flow of a human action/activity recognition model

Data preparation

In most situations, data has to be preprocessed first in order to make feature detection possible. Features are relevant pieces of information, which have the power to represent larger pieces of data.

Depending on the type of the features, they can be extracted directly from the pixel data or indirectly. In the first case this step can be skipped. In the second case, the data has to be preprocessed first. To take the case of this prototype, a multi-scale representation of the data has to be calculated from the data in order to make feature detection possible. This is such a computational intensive and complex part of the whole model, that it simply cannot be seen as part of the feature detection step and is tackled on its own. This is stated because in most literature the data preparation is seen as part of the feature detection.

In most cases, video input is gray scaled from color video input. The primarily reason is that it simplifies the problem at hand without throwing away to much critical information contained in the data. This is off course not the case in situations where color is used intentionally for the video processing, think of green screens or colored tracking spots etc. In these situations, the colors and their behaviors are known in advance.

Background subtraction or segmentation is a commonly used data preparation technique for basic motion or blob detection. It can be calculated in several ways. It can be calculated by taking the difference of two consecutive frames. This is a straightforward method, but it is not capable of detecting subtle motion.

$$I_{diff}(x, y) = I_t(x, y) - I_{t-1}(x, y)$$

Background segmentation can also be calculated by having a static background observation, which can be subtracted from the data input.

$$I_{diff}(x, y) = I_t(x, y) - I_{background}(x, y)$$

The drawback of this method is that the background in video can change due to illumination variations. To rule out these unwanted effects the background image can have an adaptive behavior according to a learning rule, in most cases accompanied by a threshold to decide if the pixel is part of the foreground or background.

On the next page, you will find a sequence of data preparation steps for background segmentation (*figure 1 & 2*). As becomes clear in the sequence filtering out noise plays a significant role as well. Blobs are a lot less fragmented and the amount of blobs is reduced to only the most significant blobs.

All recording devices, either analogue or digital, are to some degree susceptible to *noise*. Therefore, noise filtering is always part of the process and in most cases can be found in the data preparation step. There are always tradeoffs to be made at noise reduction, the cost of computer power and the cost of losing significant detail (how aggressive should the noise reduction be).

Although background segmentation is a popular approach in HAR. But, because it is limited in solving complex issues and often depends on a lot of context for recognition, this approach is not suitable for the objectives stated in the previous chapter.

Figure 1: Background segmentation steps. From top left to top right: a) background image, b) video input, c) background subtraction.

Figure 2: From bottom left to bottom right: a) black – white by pixel threshold, b) noise reduction.

Noise can be filtered spatially and temporally or both. Spatial filtering takes a single frame and smoothes pixels values in relation to their neighboring pixels. Temporal filtering takes a pixel in a frame and smoothes the value in relation to values in previous frames at the same location in the frame.

A special field within computer vision is 3D computer vision. It is a branch within human action and activity recognition that tries to use the extra spatial dimension for better tracking of the body parts. The idea is to use two or more camera viewpoints to capture 3D info. Processing the data into 3D costs al lot of preprocessing and it is computational intensive, but by adding the extra spatial dimension it gives the feature selection an advantage, because the depth information makes it easier to distinguish relevant features from the ones that are not. Another 3D approach is to map/match 3D (human) models to 2D video. This is usually a less computational intensive approach. By taking the contours of a background segmented blob. The contours can be matched to the contours of a 3D model.

In the end, the data preprocessing is only there to serve the feature extraction, depending on which features need to be extracted from the pixel data.

Feature extraction

Feature extraction aims at computing abstractions of image or video information and making local decisions if the point, neighborhood or blob is relevant for the classification task. These features can be related to the structure of the image such as edges, corners or structures that are more complex. The can also be related to temporal changes. Generally a feature is an “interesting” part of an image and the feature type is highly dependent at the specific problem at hand. Features can be divided into global features and local features.

Global features are features that represent an abstraction of the whole subject or a large part of the subject. In most cases, the background segmentation is used in order to make detection of the blobs possible. The blobs

are than extracted and can be used as feature in several different ways. Commonly shape-based features are extracted from these blobs. Shape-based features can be based on a blob's silhouette, contour or skeletonization of silhouettes [2].

Besides features that are extracted from a single-frame, a lot of research is done on features that are extracted from a range of frames. Examples are optic flow and temporal templates, which include motion history images (MHI), motion energy images (MEI) and motion history volumes.

Instead of extracting features that represent the whole subject or a large part of it, a group of local features can be extracted as well. Commonly used local features are structural features such as corners, edges, t-junctions etc. Features can also be related to texture, shape, motion or color. Local features often cannot be related directly to the subject of interest but as a group, they can.

A huge disadvantage of global features is that they can be quite unstable when partial occlusion occurs. Local features can handle occlusions very well. If the subject of interest is partly occluded and some local features cannot be seen, the other local features often provide enough information to provide stable recognition.

Descriptors

If a feature cannot be used for direct matching a description of that feature is necessary in order to make it comparable. A feature has to be described in a way that its description makes it possible to *distinguish it from* other class features or to *match it with* same class features. The description tries to capture the essence of a feature so it can be compared to others. The descriptor has to satisfy several criteria:

- match same-class features
- distinguish between different features
- scale invariance
- rotation invariance
- illumination invariance

Matching and distinguishing capabilities determine the strength of the descriptor. The same class features found in different conditions should have corresponding descriptions. Corresponding feature points can be found at different scales.

Descriptors, like features, can be typed as local or global. Global descriptors commonly include contour-based and silhouette-based descriptors. General contour-based descriptors include wavelets, Fourier descriptors and Hough transform descriptors [42]. Because contour descriptors are based on the boundary of a blob, internal structure information is lost. Contour descriptors are also vulnerable to occlusions. The same counts for silhouette-based shape descriptors, they commonly include invariant moment, Zernike moment and wavelet moment descriptors [42]. The moments are computationally intensive and sensitive for disjoint shapes or shapes with noise where the silhouette information is not correct.

Classification

A pattern of features or of their descriptions allows for classification. Classification is the process of coding and organizing the patterns of features according to abstract or conceptual descriptions. There are two main approaches: 1. Classification through logic and reasoning, and 2. Classification through probabilistic reasoning.

Logic based approaches keep track of all logical consistent explanations of the observed classes. This means that all possible and consistent plans should be considered. These plans are then coded into the model. Think of decision tables or decision trees. This can be a solution for simple recognition where features, their descriptions or their behaviors, are predictable. A serious problem of logic-based approaches is their inability to represent uncertainty. They offer no mechanism for preferring one consistent approach to another and incapable of deciding whether one particular plan is more likely than another, as long as both of them can be consistent enough to explain the actions observed. There is also a lack of learning ability associated with logic-based methods.

The most used approach to classification is probabilistic reasoning. Matching is done on the probability of a class given the observations and given the patterns of known classes. The most widely used classifiers are: Neural Network, Support Vector Machines (SVM), k-nearest neighbor, Gaussian mixture model, Gaussian, naive Bayes. However, the most popular classification methods that can deal with temporal / sequential patterns are Hidden Markov models. HMMs handle activity patterns as discrete states in temporal space. The temporal evolution is modeled as a sequence of probabilistic jumps from one state to the other.

HMMs first were found in speech recognition applications in the early 1980s. In computer vision they are especially used in the action recognition stage. Over the years, several variations of the HMMs emerged trying to cope with different levels of complexity in action and activity recognition. HMMs are explained in detail in section 2.6.

Classification through probabilistic reasoning goes hand in hand with training / learning of the activity classes. Most classifiers have the possibility of learning. This is in most cases essential, because the parameters of the classes, the probabilistic models, cannot be found following logical reasoning and can only be found by learning. An additional advantage of learning models is that they can cope with unforeseen variations of the same class activity, provided that those variations are present in the training data. The ability to cope with unforeseen variations of activities is very important in real live cases.

Very interesting are unsupervised learning methods whereby the model learns to categorize unknown classes in unknown scenarios. Niebles *et al.* [31] propose a learning method for action categories using spatial-temporal words. A video sequence is represented as a collection of spatial-temporal words by extracting space-time interest points. Their algorithm automatically learns the probability distributions of the spatial-temporal words and the intermediate topics corresponding to human action categories. Unsupervised learning is complex and will not be part of the prototype's scope. The possibilities however are interesting for future development of the prototype.

2.6 Scale space and top points

Introduction

Scale space representation is the space that is formed by looking at data at a continuous range of scales simultaneously. Scale space theory forms the framework for multi-scale images. When applied to an image, the scale space of that image is a stack of different scales of that same image, thus adding an extra dimension to that 2 dimensional image: the dimension of scale. In other words, you are looking at multiple levels of detail at once, from the global structure to the smallest detail.

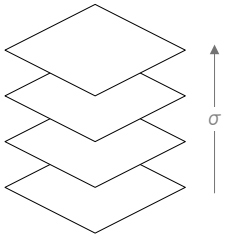


Figure 3: Multi-scale image representation

The figure above represents a multi-scale image. Starting at the bottom is the original image and as the scale increases, the image gets smoothed more and more. In *figure 4* are the associated images. You can see the fine details like leave structures vanish as stronger structures like the trunks are still easily distinguishable. The more the image is blurred, the more the higher scale structures get/stay visible. At some point, even global image structures start to appear that were not easily distinguishable of lower scale levels.

Figure 4: Different scale representations of the same image, starting with the original image on the left [21]

The notion of the scale space approach comes from the natural consequence of the progress of observation. When observing data like an image interesting features of the image structure can appear at different scales. For example, a tree -like above- can be observed at different scales: the scale of the tree structure such as the trunk and the branches, the scale of the leaves or even down to the scale of the cells.

This same example reveals two other aspects of scale space data representation:
1. The concept of *hierarchy*: data structures or features at a certain scale are part of structures of a higher-level scale.
2. Features are only meaningful over *certain ranges* of scale. Perceiving a tree at kilometer level is meaningless.

Interesting thing of scale space theory is the similarity with biological vision, scale space operators closely resemble receptive field profiles found in the mammalian retina. More on the human perceptive system in the next section (2.7).

Scale invariant

The reason scale space is important and applicable to this project is the following. In certain controlled situations, appropriate scales for analyzing data may be known a priori. However, that is not the case with image analysis. The rough data of a video is a raster of pixels in each frame. The data can be analyzed at pixel level or perhaps at a level related to the resolution of the frames. Either way, a pixel size doesn't necessary relate to dimensions of the real world it captured. Generally speaking you can't say that a pixel in the video data is equivalent to for example a meter in the real world. Even in a fixed setup, you still have to deal with perspective / depth.

Therefore, if you are searching for a certain feature and want to take into account that the representational size (e.g. pixel size) of that feature is unknown or may vary over time or data set, you should search for that feature at different scales. In that way, searching for features becomes scale invariant. It doesn't matter how big or small the feature is captured on a frame, if it's representational size is in the range of scales you are searching for, you'll most likely find it.

Scale invariant is an important aspect of keeping the prototype generic and context independent. It also relates to the build environment. As a person can vary his distance between him and the camera, thus his representational size and therefore the observation scale will vary as well. However, because the distance can only vary within the confinement of the space, the scale range is often limited. Even in big spaces the scale range can be limited by choosing an effective position for the camera.

A multi-scale representation

The basic idea of generating a multi-scale representation is to gradually smooth the data so fine scale details are successively suppressed. The longer an image structure 'survives' the smoothing, the higher scale it belongs to and often the more important that structure is.

Under general conditions on the type of computations performed for smoothing image data are Gaussian kernels and its derivatives. Gaussian operators are proven to be most suited. In addition, as stated above, Gaussian operators share similarities with the biological receptive field profiles. (*Section 2.7*)

An implementation of a scale space operator is the second order Gaussian or Laplacian of a Gaussian operator (LoG). In the *figure 5* an image is convolved with a LoG kernel at different scales. In the following example each scale step the variant of the Laplacian is multiplied by a scale factor of 1.5. By convolving with a LoG key structures get amplified. These are all the top and bottom peaks later on called critical points.

At low-level scales a lot of detail, mostly noise, is visible. The person is easily recognized in the first scales, because legs, arms, body and head are easy distinguished from the rest of the picture. As the image gets smoothed, less important structures start to fade away and eventually only a single blob survives.

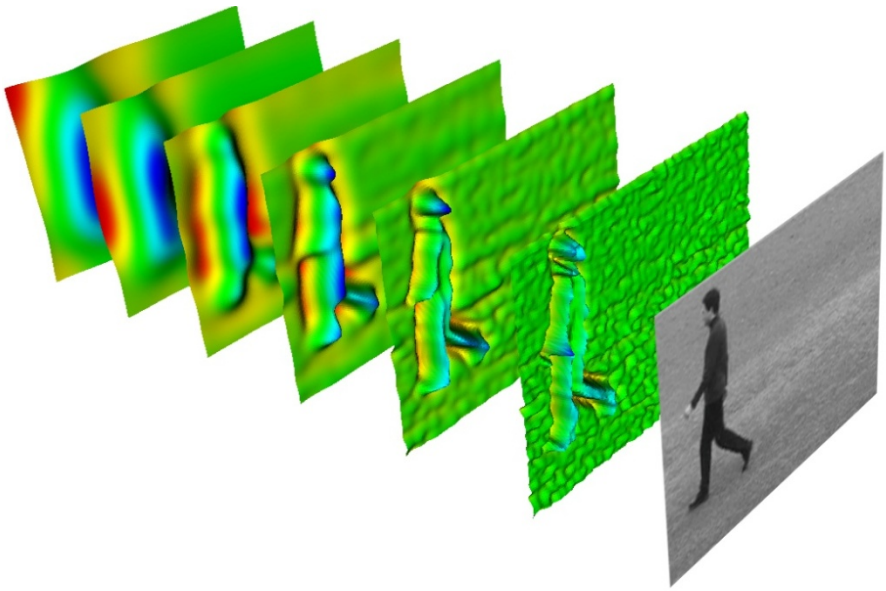


Figure 5: Multi-scale image representation of person walking

So what does this mean? It means something significant can be distinguished in a sea of data, something has been found. Ideally, perception *starts* at a high scale level to discover the most important structures. From that scale the data is zoomed into, lower scale levels are processed to find substructures in the larger structure. By constantly zooming in, an image can be processed top down. This way structures and substructures are easily grouped and place in a hierarchy. This is quite similar to the way the human perceptive system works, but instead of blurring and zooming back in, it that does this instantly with different sizes of receptive fields.

Critical points, critical paths and top points

Critical points are points at any fixed scale in which the gradient is zero. These are the top and bottom peaks; very well recognizable in *figure 5* and visualized in a 2D abstract representation in *figure 6*. Some examples are marked by blue circles.

As the scale increases, the image blurs and the critical points move along a path through the called the critical path as illustrated in *figure 6*. The points at which creation and annihilation event take place are referred to as top points. A couple of examples are marked by red circles.

Previous studies have shown that top points, also called scale space extrema, are very stable points in an image structure and can be used for content-based image retrieval [19].

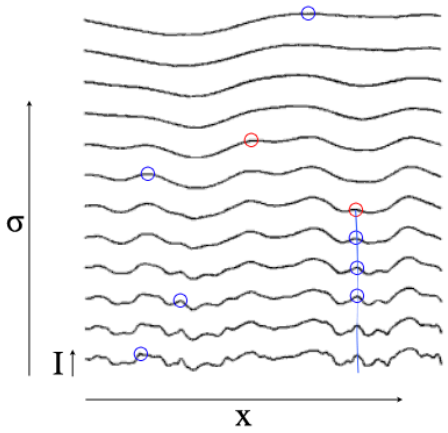


Figure 6: Multi-scale representation of 2D abstract observation (bottom graph) with critical points, paths and top points.

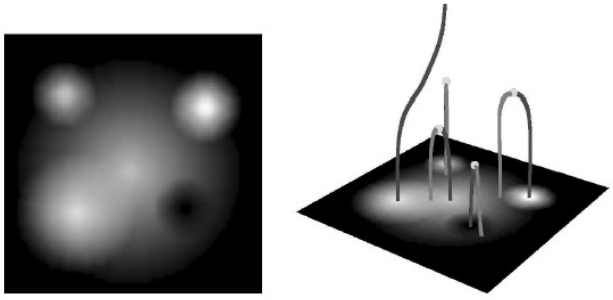


Figure 7: Left: Couple of blobs in a picture, Right: 3D representation of critical paths and top points of the blobs [19]

Scale space operators

In the *figure 5* a LoG operator is used. It has is similar to one of the most common receptive field profiles of the human retina. The LoG operator is used in this prototype as the only scale space operator, but like in receptive field profiles, there are many more operators.

The reason the LoG operator is used in the prototype is because it is rotation invariant, the computations are relatively easy compared to other operators and it responses to structures in general, unlike a lot of operators that are targeted for specific structures like for example corners, T-junctions, edges or more complex structures.

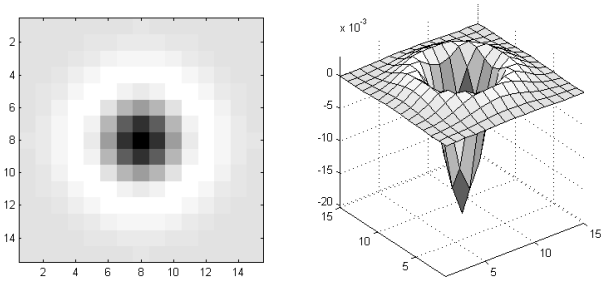


Figure 8: The Laplacian of a Gaussian kernel. Left: The 2D representation, right: The 3D representation

To give an idea of other kernels that are used in computer vision as well, below are some Gaussian derivatives up to the second order. The kernels found in the middle row -the first order Gaussian derivatives- can for example be used as simple edge detectors.

A 2-dimentional Gaussian kernel at a certain scale σ:

$$g(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x^2+y^2)}{2\sigma}}$$

Gives the following family of Gaussian derivatives up to the second order for a fixed scale σ:

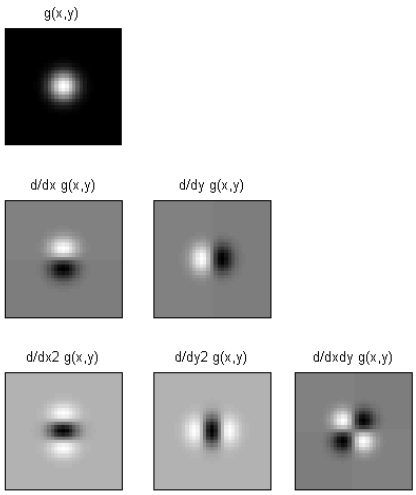


Figure 9: Family of Gaussian derivatives up to the second order.

The kernels above are still relatively simple, but by making combinations very complex operators can be made for specific tasks.

A more mathematical approach behind the concept of convolving an image with a kernel is the following: Imagine that a section of an image is taken and the intensity values along the section line are represented by a function f(x), f(x) being the intensity at pixel x.

The intensity value at each point in the function does not say anything about the local structure. In fact, that value can even be subject to noise and is therefore not reliable on its own. Only in relation with neighboring values can something be said about the local structure. By taking the derivative of f (x), you get a function that tells us a lot more about the structures in f(x). Peaks tell us there are edges at the same location of x in the original function. Convolving an image with a derivative of a Gaussian gives you the same effect; as a result, you get the derivative of the image. An additional effect of the Gaussian is that -with the appropriate scale- is flattens steep peaks, resulting in noise oppression. The larger the scale, the more noise oppression takes places. Because not the edges but the center points of structures provide the desired structural information, the image is convolved with the Laplacian of the Gaussian.

2.7 Human visual perceptive system

Introduction

The human visual perceptual system is a marble of evolution and through research, many of its secrets are revealed over the last years. Because some of the techniques and ideas used in the prototype are derived from the way the human perceptive system works, this section explains some of those aspects.

Receptive fields

The human eye or better, the human retina consist of 150 million rods and cones also called receptors, each responsible for catching a ray of light that enters the eye via the pupil. In digital camera terms, it can be compared to a 150-megapixel camera.

The rods are mainly for dim light situations and provide black-and-white vision, while cones provide us with the capability of perceiving colors, adding additional information to our perception. The cones are positioned mainly at the center of our retina, the focus area of light coming into our eyes.
In a digital camera, roughly each pixel of the sensor chip is saved to memory as a pixel in a digital image. However, this is not the case at all in the human perceptive system. Although our retina has a 150 million "pixel sensor chip", there are only 1 million neuron fibers going from the retina to the brain, to the visual cortex.

Research has shown that an important step of the data processing already takes part in the retina [Hubel and Wiesel, Nobel Prize 1981]. The retina processes the data before it reaches the brain. Receptors do not seem to be working on their own, but are part of one or more groups of (neighboring) receptors called receptive fields.

If a certain distribution of light caught by the receptive field matches a certain profile the ganglion cell triggers. If triggered, it fires rapidly.

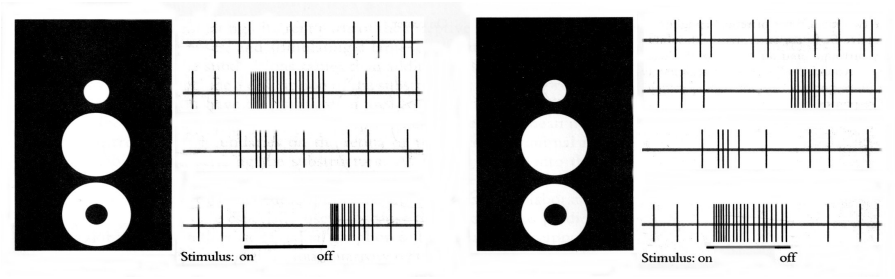


Figure 10: The discovery of receptive fields by Hubel and Wiesel

Hubel and Wiesel discovered receptive fields by stimulating a group of receptors connected to a single ganglion cell. By measuring when the ganglion cell fired signals (the vertical stripes in figure 10) to the brain they found something peculiar. A spot that was completely dark or completely illuminated had no effect. Some ganglion cells only fired rapidly if the spot had only central illumination or surround illumination. The results on the right are of in inverse profile of that to the left. These receptive field profiles are similar to the LoG operators discussed in the previous section about scale space.

The notion of scale is applicable here as well. The receptive fields vary in size; smaller receptive fields react to details, while bigger receptive fields in our retina react to larger structures.

Hubel and Wiesel even advanced the theory that these relatively simple receptive fields form input for higher-level receptive fields that can react to structures that are more complex. These complex receptive fields can be modeled as complex scale space operators all belonging to the same family of Gaussian derivatives also discussed in the previous section.

Motion detection

Besides the ganglion cells that react to structure, there are also ganglion cells that work in pairs in order to measure movement. They work as temporal coincidence detectors, like a Reichardt detector.

If a intentional delayed signal from receptive field (a) reaches the ganglion cell at the same time a signal of a neighboring receptive field (b) reaches it and both signals have the same illumination profile the ganglion cell is 'looking' for, the ganglion cell fires.

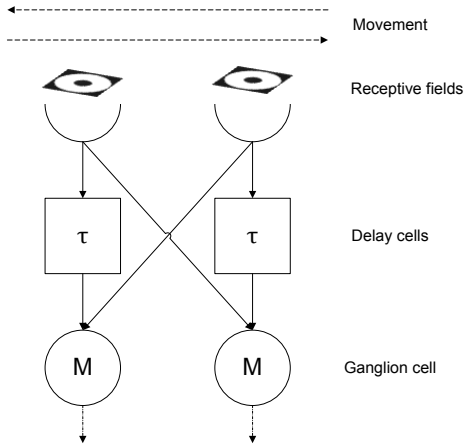


Figure 11: A motion detector (Reichardt detector) as found in the human perceptive system

This only happens if the movement's velocity matches the distance divided by the time delay. So every pair of ganglion cells is tuned to find movement for a specific velocity and direction. This means a huge of amount of ganglion pairs can be found to register movement for all kinds of speeds, directions, but also for different scales. The prototype uses the same concept for detecting motion from frame to frame.

Streams

A two-streams hypothesis supports the theory that visual processing follows two main streams: 1. The ventral stream, which is involved with structures for object recognition, and 2. The dorsal stream is responsible for processes location and motion.

Application

The notion of receptive fields is applied in a lot of research in computer vision as scale space operators or other local feature detectors. The motion detection cells have also triggered computer vision research; they can for example be found optic flow methods.

A general approach can be derived from this knowledge: The brain uses local features to (1) retrieve structure information and (2) retrieve motion information by tracking those features. The separation of structure and motion information is, although not proven yet, also an interesting aspect. The prototype applies both notions.

2.8 Hidden Markov models

Introduction

The purpose of a hidden Markov model is to model a process given a sequence of observations. The underlying process responsible for giving the observations is thereby assumed to be a Markov process. A Markov process encompasses a stochastic process whereby the probability of its next state depends only on its current state and not on its whole past.

There are three key problems where hidden Markov models are used for:

1. To compute the probability of an observation sequence O when given a Markov model represented by λ: P(O| λ). *(classification)*
2. To compute the optimal sequence of (hidden) states corresponding to a given observation sequence.
3. To compute the parameters of λ given an observation sequence to maximize P(O| λ). *(training)*

Markov chain

An example of a Markov process is the following process chain. At each moment the process can be in the state of A, B or C. The states can be concrete / meaningful, but they can also encompass something abstract and indefinable.

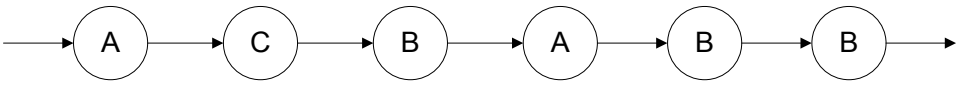


Figure 12: Schematic example of a Markov Chain

Given enough observations of the states, the transition probabilities of each state to the next can be calculated. These transition probabilities would be the defining parameters of a Markov model as seen in

figure 13. Here each state S_n corresponds to one of the states above (A, B or C) and each a_{ij} corresponds to the transition probability of one state following the other. As seen in the example and in the diagram below, it is possible that a process can stay in the same state.

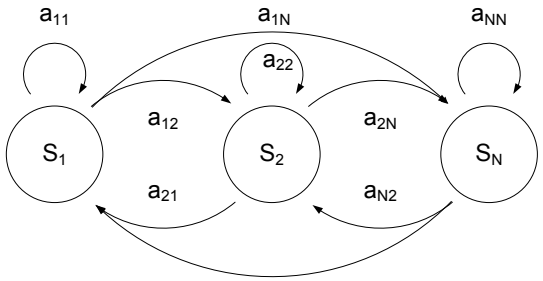


Figure 13: Schematic example of a three states Markov Model

Hidden Markov chain

In a regular Markov model, the states are directly visible to the observer, but in a hidden Markov model, the states are hidden. However, the hidden process produces observable signals. The signal is at each step of the process is correlated to the state the process is in at that moment.

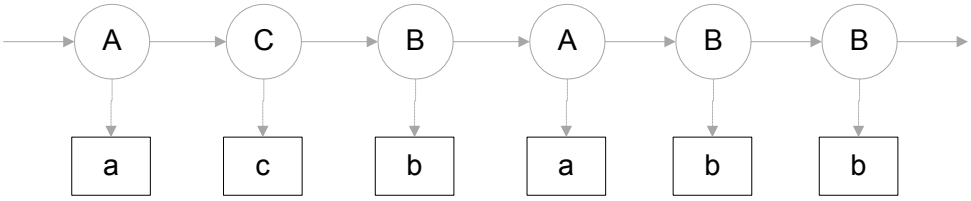


Figure 14: Schematic example of a hidden Markov Chain with observations

The next diagram shows a hidden Markov model where the model is defined by not only the transition probabilities of each state to the next, but also the observational probability output.

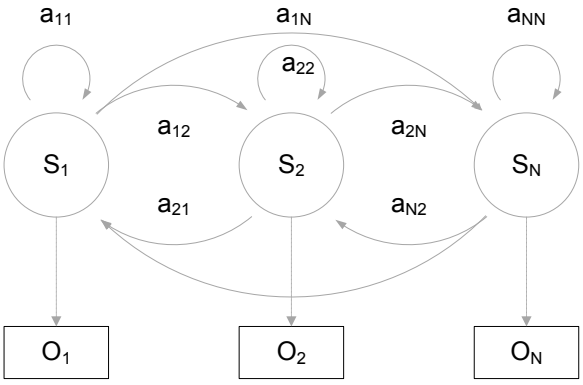


Figure 15: Schematic example of a three states hidden Markov Model

So given a sequence of observations / signals the underlying sequence of hidden states can be calculated with a certain probability. Thus, the transition probabilities of the hidden states can be computed as well.

What could these observation output probabilities hold? Observations can be either discrete or continuous. The following subsections explain these forms.

Discrete hidden Markov model

Discrete observations are observations that have a finite set of distinctive 'values'. A discrete observation can for example be 'wet' or 'dry'.

Assume the situation where the weather is a Markov process and it can be in the following three states: raining, cloudy or sunshine. Assume that these states are not directly observable, but the only thing we can observe is that the street is 'dry' or 'wet'. These observations correspond to the state the process is in at a certain moment. The following probability matrix represents an example of the observation output probabilities:

			Dry	Wet
S_1 Raining	⟶	O_1	5%	95%
S_2 Cloudy	⟶	O_2	70%	30%
S_3 Sunshine	⟶	O_3	90%	10%

Table 1: Example of three 'hidden' states S_n , their discrete observation types (dry / wet) and their observation probabilities O_n

Continuous hidden Markov model

A continuous hidden Markov model outputs observations, which have continuous values. A Gaussian distribution defines the observation probability of each state. Following the previous example, this could look something like this:

			Moisture
S_1 Raining	⟶	O_1	$\mu = 90, \sigma = 5$
S_2 Cloudy	⟶	O_2	$\mu = 70, \sigma = 15$
S_3 Sunshine	⟶	O_3	$\mu = 15, \sigma = 10$

Table 2: Example of three 'hidden' states S_n and their continuous observation probabilities O_n

The above example shows a continuous hidden Markov model with a one-dimensional Gaussian distribution. The number of parameters, here only 'moisture', translates into the dimensionality of the observation. Unlike the observation types in the discrete HMM, the observations in continuous HMM can consist of multiple dimensions.

Each multi-dimensional observation is defined as follows; where $\mathcal{k}$ indicates the number of dimensions, μ is the mean vector, and Σ is the covariance matrix:

$$O \sim \mathcal{N}_{\mathcal{k}}(\mu, \Sigma)$$

The dimension of temperature measured in Kelvin can expand the observation dimensionality of the example above. The observation for each state would then be defined as a two dimensional or bivariate Gaussian distribution.

The continuous hidden Markov model can even handle Gaussian mixtures as observation values, but this report will not discuss that possibility.

Notation

The short notation of an HMM (λ) is as follows; π contains the initial starting probabilities, A the transition matrix holding the transition probabilities from one state to another and B the emission probabilities responsible for the observation output:

$$\lambda = (\pi, A, B)$$

$$A = \{a_{ij}\} = \{P(x_t = j | x_{t-1} = i)\}, \quad \text{where } i = 1..N; j = 1..N; \sum_j^N a_{ij} = 1$$

$$B = \{b_i(o_t)\} = \{p(o_t | x_t = i)\}$$

Where the output or emission probability density function (pdf) for each *i* is a Gaussian:

$$b_i = N(o_t; \mu_i, \Sigma_i) = \frac{1}{\sqrt{2\pi\Sigma_i}} e^{\frac{-(o_t-\mu_i)^2}{2\Sigma_i}}$$

Types of HMMs

Until now, we have only dealt with HMMs with a full state transition matrix, where transitions can be made from any state to any another state. This is also called an ergodic model. There are also models in which the transition flow is limited.

Another HMM type is a left-to-right model in which a state cannot switch to a lower state. This type of HMM models processes which cannot go back to 'older' states and which will finally end up in an 'end-state'.

A variant of the left-to-right type is the called the cyclic model, it models a process which will go back to the 'begin-state' after it has finished the 'end-state'.

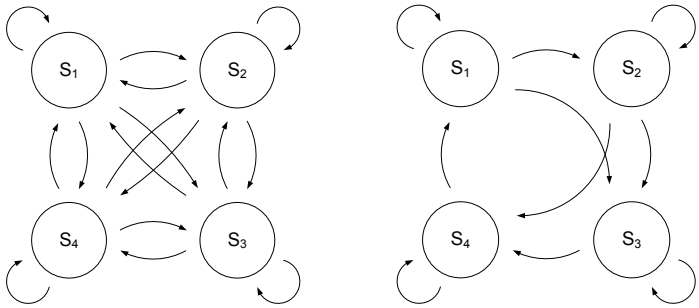


Figure 16: Left: an ergodic HMM model, right: a circular HMM model

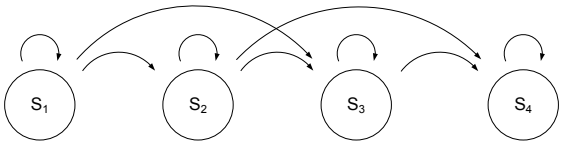


Figure 17: A left-to-right HMM model

2.9 Summery

This chapter provided background information on several different aspects of the prototype. The scope of the project and the theoretical environment the prototype targets is discussed. Intelligent spaces provide a targeted scale range in which the activities will take place. The intelligent office is a great challenge for an intelligent space because most of the activities are recurrent, the activity set is limited, but the type of activities can be quite complex. It is a place where people work and interact with each other.

The third section gave general background information on human action and activity recognition. Activities being the context in which actions take place. The tree major application areas: *surveillance*, *analysis* and *control* are explained.

A general model outline of a human activity recognition method is presented and at each step, different techniques are described for data preprocessing, feature detection and selection, descriptors, training and classification.

Scale space theory and its link with the biological visual perceptive system is explained. By using LoG operators critical points can be found which in turn lead to top points. These top points are stable features that are ideal reference points for structure and motion detection.

By tracking the features and describing their structure / behavior as observations, underlining hidden states and thus patterns can be found, trained and used for classification of unnamed observations. Hidden Markov models are excellent in training and recognizing these temporal patterns.

3. Prototype

3.1 Introduction

The following chapter discusses the model design and implementation. This chapter outline roughly follows the model chain of the prototype. As explained in section 2.4 the general model of human activity recognition is as follows:

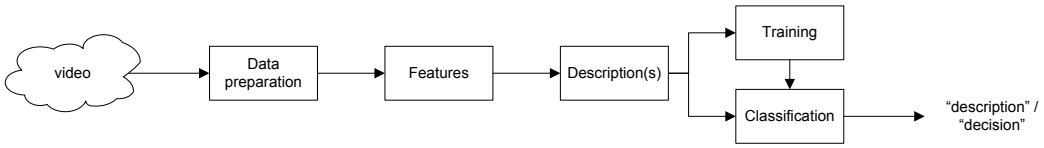


Diagram 3: General flow of a human action/activity recognition model

The model chain of the prototype is inherited from the general model. Below is the prototype's approach, at each step the solution is presented to the problem at hand:

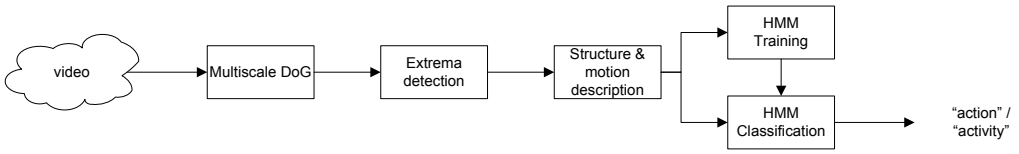


Diagram 4: Model flow of human action/activity recognition of the prototype

In this chapter, each section elaborates on a specific step in the model chain.

3.2 Multi scale Difference of a Gaussian

In order to extract features, or in the case of this model top points, the frames have to be preprocessed first. A multi scale Laplacian of a Gaussian of each frame is needed.

These features are located at the scale-space extrema. Scale-space extrema have the property of being stable and reliable interest points. E.g. different frames containing the same object(s) output corresponding scale-space extrema.

In order to extract scale-space extrema a multi-scale image has to be derived from the frame. This is done by computing the Laplacian of a Gaussian (LoG) of the frame at different scales. Each following scale if received by multiplying the previous scale with the scale factor *k*.

A LoG image can be computed by convolving the frame with a Laplacian kernel (a second order derivate Gaussian kernel), but for optimization purposes a Laplacian is computed by computing a difference of a Gaussian (DoG). A DoG is an approximation of the LoG when the scale factor k is around ~ 1.6 [24]. The DoG is obtained by subtracting two Gaussian blurred images with a different scale.

Given an input frame, where f is the gray color value ranging from 0.0 to 1.0 at pixel location (x, y):

$$f(x, y)$$

This frame is convolved with a normalized 2-dimentional Gaussian kernel at scale σ, the 2-dimensional Gaussian the following function:

$$g(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x^2+y^2)}{2\sigma}}$$

The scale σ determines the width of the Gaussian kernel. In statistics it is called the *standard deviation* and the square of it, σ^2 , *variance*. In the context of a scale space kernel, it is referred to as the *scale*. The scale can only take positive values, $\sigma > 0$.

Convolving the kernel with the frame gives the scale space representation of the frame:

$$L(x, y; \sigma) = g(x, y, \sigma) * f(x, y)$$

The normalization ensures that the average grey level of the image stays the same when convolving at different scale levels. 2D convolution is very computational intensive, luckily the Gaussian kernel can be described as a regular product of two one-dimensional kernels.

The Laplacian is a differential operator given by the divergence of the gradient (or second order derivate) of a function. In two dimensions, this is given by:

$$\Delta L = \frac{\partial^2 L}{\partial x^2} + \frac{\partial^2 L}{\partial y^2} = L_{xx} + L_{yy}$$

As noted earlier it is computational more efficient to compute the Difference of a Gaussian, because it reduces the number of times a frame has to be convolved. The DoG is an approximation of the Laplacian, with *k* as the scale factor:

$$D = L(x, y, k\sigma) - L(x, y, \sigma)$$

The scale factor *k* for a multi-scale image is usually somewhere between 1.5 and 5. For multi-scale images that have been computed with a DoG *to approximate a Laplacian operator*, it is around ~ 1.6 [24]. The DoG approximates receptive fields of ganglion cells in the retina well with a scale factor of ~ 5.

A lower scale factor means a finer scale step, top points can be found at a more precise scale level and connecting toppoints will be more accurate, because of the smaller gap between scale layers. However, because more layers may be needed it may be computational more expensive. The prototype has a scale factor of 1.5. At first 1.6 was chosen. However, with 5 scales and a factor of 1.6 the computation time is very high per frame. Although computational complexity has no priority in the prototype, the practical downside was that it significantly slowed down development and testing the prototype. Decreasing the scale factor to 1.5, significantly reduced computation time and in some cases even increased the recognition rate for some cases slightly (see 7.3), while keeping the value around 1.6.

The scale value of the prototype starts at 2. Scale factor 2 greatly reduces pixel noise without erasing too much detail. Lower values do not filter out noise enough, and higher values tent smooth out relevant finer details.

The intelligent space limits the scale range in which the activities can be observed. Inside a space, the distance between the person and the device observing his or her activities can only vary within the dimensions of the space, limiting the scale range of the observations as opposed to open spaces.

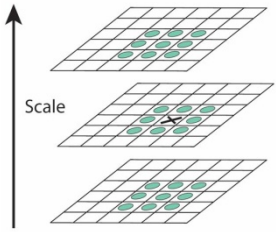
The computation scale range exceeds the observation scale range, because top points can only be found at annihilations of critical paths. Therefore, higher scale levels can be needed to distinguish top points from critical points.

The prototype uses five scales; it is a compromise between the number of scales and the performance. More than five scales significantly slow down calculations. At the fifth scale, the scale value is 10,125, which is more than 5 times the initial starting scale of 2, meaning that the same feature can be recognized at a difference of 5 times its size, which is enough for indoor observations. After the DoG is computed for several scales, a multi scale image is produced for a single frame, ready for the next step: feature detection.

3.3 Feature detection & selection

The second step of the prototype extracts local relevant points from each frame, used as features. The features used by the prototype are stable top points and critical points in the multi-scale image.

A common approach to search for top points, also called scale space extrema, is by detecting if these points are local extrema with respect to both space and scale. This is done by comparing each pixel value with the 26 nearest neighbors in a 3 * 3 * 3 scale space volume. Many candidate key points are detected, some of which are unstable. These points are then filtered.



Because of the discrete steps in scale and pixel size, the locations of the candidate points are not that precise. By interpolation of nearby data, an accurate position can be computed.

The prototype takes another approach. Local space extrema, not scale space extrema, are found in the first scale level by comparing eight neighbors in a 3 * 3 space area. This off course produces many candidate points, more than the approach described above, because the scale dimension is not used as filter condition. As a candidate point is located, a description of the point's local structure is made and the strength of that description is calculated. If the strength is above a certain threshold, the point is saved.

How the descriptor describes the candidate point's environment and how the descriptors strength is calculated, is part of the next section covering the descriptor.

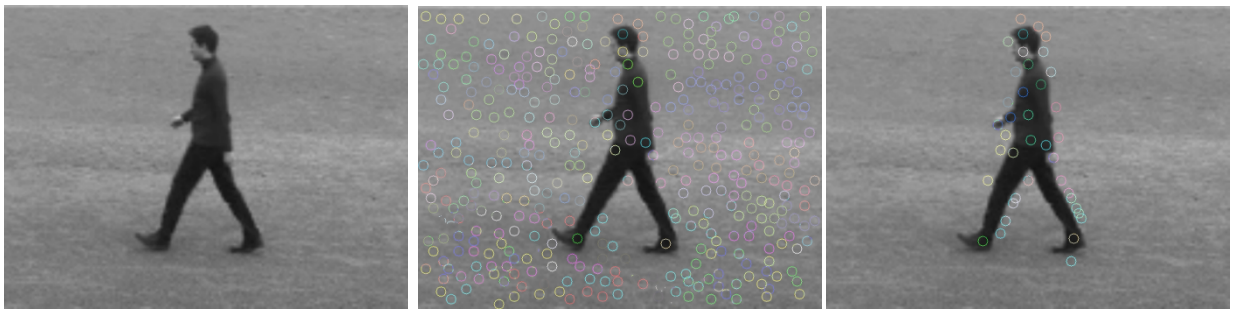


Figure 18: Left: A random frame from video, middle: frame with all candidate key points, right: frame with filtered key points.

Candidate points at the first scale level are now localized, described, filtered and can be labeled as features. Now, features at higher scales need to be located as well. Instead of finding local extrema by comparing each pixel to their eight neighbors at higher scales a novel approach is taken. A hill climber algorithm is used to find critical points in higher-level consecutive scale layer. For each feature in the previous scale layer, the hill

climber starts climbing from the feature's location towards a new feature's location in the current scale layer. The hill climber climbs in the case the key point is a maximum local extrema, the hill climber descends in the case of a minimum local extrema.

The hill climber algorithm is ideal for local maximum or local minimum search in this case, because of the behavior of critical points and their paths through scales. See section 2.6.

A hill climber is an iterative algorithm, usually used for optimizations. There are different variations of the algorithm and for this case, the *steepest ascent hill climbing* variation is chosen, because it is simple and efficient. The hill climber starts at given location. At that location it compares its pixel value and the values of its eight neighboring pixels and it steps toward the pixel location with the highest value, or lowest in the case of a descending hill climber. At the new location it will again search for and step toward the highest (or lowest) neighboring value and will keep doing so until it reaches a point where the location is the local maximum (or minimum). That point is assumed to be the top (or lowest) peak of the region.

There your two major advantages to this approach:

1. It is far less computational intensive to find new key points. Instead of comparing every pixel to its neighbors, the search is targeted small areas.
2. If a key point is found by hill climbing, a relation between the 'base' key point and the 'top' key point is easily made. That way a whole tree structure emerges.

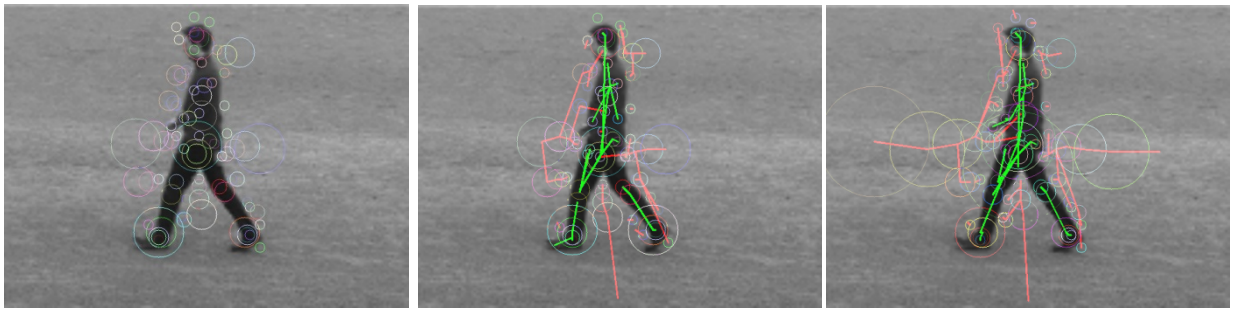


Figure 19: Left: key points; size of radius indicates scale level, middle: connecting base and top key points create tree structures, right: tree structure with two extra scales. Green lines connect minimum local extrema. Red lines connect maximum local extrema.

3.4 Descriptor

The prototype has two kinds of descriptors: A local feature descriptor to describe a feature's structure and a global descriptor that combines a summary of a frame's structure and a summary of a frame's motion.

The local structural descriptor describes the local structure at a feature's location. The description of the features helps with: 1) filtering out unstable candidate points and 2) making the features traceable over consecutive frames.

The global descriptor is needed for the next step: training and classification. The HMM needs a sequence of observations as input. Each observation has a fixed dimension and because the number of features constantly changes in each frame, a global description is needed to describe all local features' individual structure and movement. With order words, an observation is formulated as a summary of all features' structure and all features' movement.

Local structure descriptor

In previous steps, features have been localized in space and scale. These features need to be described in a manner that they can be recognized in a following frame within certain margins of transformation. These transformations include rotation, scale and affine transformations.

The relation between the scale of the feature and the size of the neighborhood that is used for the description is linear. This is necessary in order to make the description scale invariant. Assume the case a feature is found on the shoe of a person in scale-space, the scale being σ_1 the location in space is not relevant. Assume that in another frame the shoe has moved halfway towards the camera. The same feature will -in theory- be found at the same location of the shoe, but at twice a higher scale ($2 * \sigma_1$) . In order to cover the same structural area that results in the same description of the local structure, the description area also has to be twice as big.

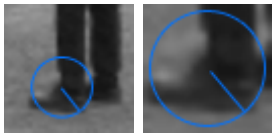


Figure 20: Same feature found at scales σ_1 and σ_2 .

So in order to make a description scale invariant, the described area should be relative to the scale location of the feature. A circular area is chosen to make the feature rotation invariant.

$$r_{descriptor} = \sigma * r_{factor}$$

The relation between the size of the neighboring region is clear, but the radius factor is not. The factor description area cannot be too small: 1) not enough structure to describe and 2) a small amount of pixel data makes the description unstable due to the effects of noise. On the other hand, the description size cannot be too big, because surrounding (changing) structures can have undesirable effects on the description making it unstable as well.

Features are detected in the first place because the local structure reacted to the LoG kernel that passed by. Although the range of a LoG function is infinite, only a small range is responsible for highlighting the feature and that is from -4σ to 4σ. Outside that range, the function approaches zero and has little to no affect on the convolution. So for the r_{factor} the value of 4 is chosen.

$$r_{descriptor} = \sigma * 4$$

Within the radius, the magnitude of each pixel is computed:

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2}$$

The orientation of the pixels is computed as well:

$$\theta(x,y) = \tan^{-1}\left(\frac{L(x,y+1) - L(x,y-1)}{L(x+1,y) - L(x-1,y)}\right)$$

An orientation histogram with 36 bins is formed for every feature, each bin covering a range of 10°. If the r_{factor} increases, the description area increases by ${r_{factor}}^2$, so in order to keep the magnitude values scale invariant the division with σ^2 is applied. The calculated magnitude of each pixel is divided by σ^2 , the σ being

the scale location of the feature, and added to appropriated orientation bin. This orientation histogram represents the description of the feature's local structure.

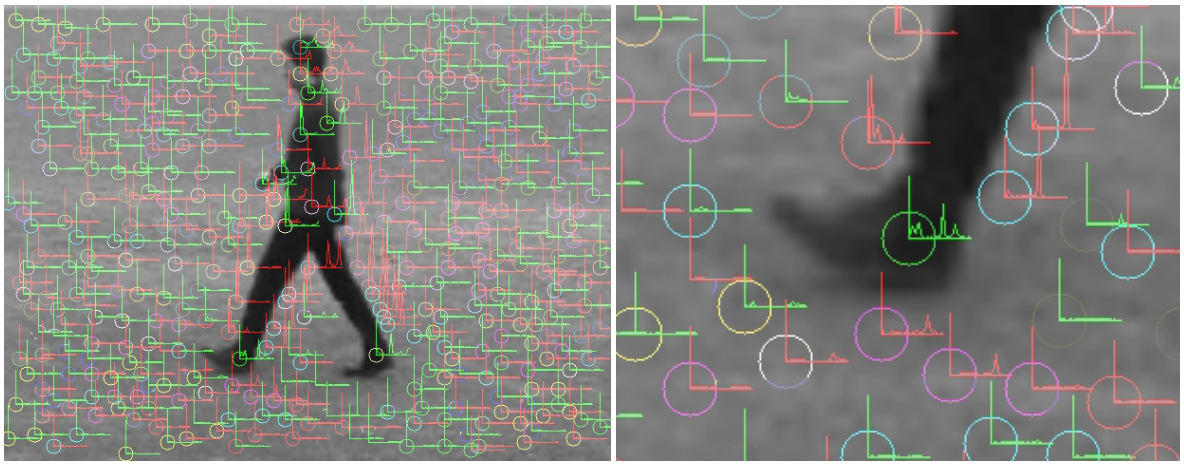


Figure 21: Local descriptors as orientation histograms of the magnitude. Green histograms are bottom-peak-critical-points, red histograms are top-peak-critical-points.

Local feature filter

At this point, the local descriptor is defined as an orientation histogram of the magnitudes. A description is made of every critical point / candidate feature in the first scale level. All these critical points are local extrema and because of noise, this leads too many candidate points as can be seen in the image above.

These candidates need to be filtered. The strength of a descriptor is calculated by summing up al bin values that are above a certain threshold. If the sum is above a 'strength-threshold', the description is strong enough to pass. The bin-by-bin threshold is needed in order to flatten out the effects of noise.

The idea behind this filter is that flat surfaces in an image structure generate no significant magnitude values. Flat surfaces provide no image structure. Thus no relevant structure information and can be filtered out by ruling out surfaces with low magnitude values. Figure 21 to the right shows the difference between histograms near observable structure (the histograms at and near the foot / leg) and the histograms where the image has no significant structure. The histograms are displayed with the x and y-axes. The histograms near observable structure show clear peaks at certain orientations, meaning significant gradients in the local image structure. Single steep peaks often depict edges.

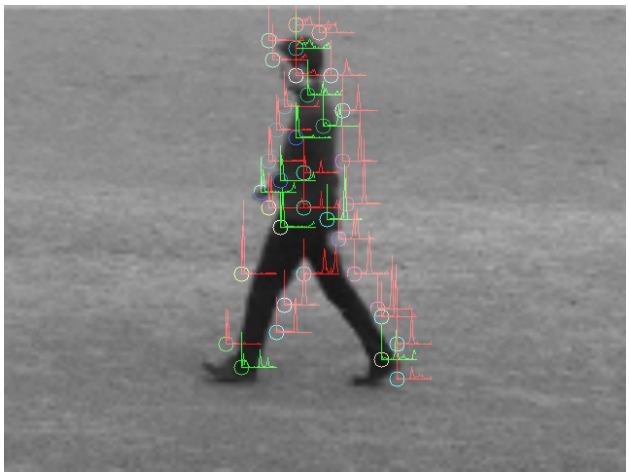


Figure 22: Result after filtering low strength descriptors

Global structure and motion descriptor

The number of local descriptors can vary heavily. The HMM however, takes observations with a fixed number of dimensions. Each dimension type has a fixed place within the observation. For example, if an observation would be expressed as (x, y, z) the next observation cannot be formulated as (y, x, z) but must have the same order.

In order to make the translation from local descriptors per frame to observation per frame a global descriptor is made. This global descriptor will contain a summary of the structural information and a summary of the motion between two frames.

First the structure part which is very simple. An orientation histogram of 12 bins is formed and each bin is filled with the sum of the related bins in all local histograms. Dividing the values by the number of local histograms normalizes the bin values

local descriptor 1	1	2	3	4	5	6	7	8	9	…	34	35	36
local descriptor 2	1	2	3	4	5	6	7	8	9	…	34	35	36
..	1	2	3	4	5	6	7	8	9	…	34	35	36
local descriptor n	1	2	3	4	5	6	7	8	9	…	34	35	36
	Σ			Σ			Σ				Σ		
global descriptor	1			2			3			…	12		

The structure part of the global descriptor is derived from a single frame. The motion part of the global descriptor is derived from the difference between two consecutive frames.

The local descriptors are used, besides for filtering and for the global structure descriptor, to track features from one frame to another. All features of a frame are compared to features of the previous frame. If the locations of two features are within a certain range and the descriptions of those features match, then it is assumed that those features represent the same region. The location of the feature in the previous frame is corrected with the direction and velocity it had at that moment.

Matching descriptions is done as follows: The sum is calculated of the differences of each bin. If this sum is below the threshold of 1.5, it is a match. When a match is made, the direction and distance moved are calculated of the feature.

A 12 bin directional histogram, each bin covering a 30° directional range, forms the motion part of the global descriptor. The distance each feature has moved is added in the appropriate bin. The bin values are also normalized by dividing the values with the number of features found.

The 12 bin structural histogram and the 12 bin motion histogram are concatenated to form a 24 dimensional observation for the HMM.

3.5 Training and Classification

The theory behind the HMM is already extensively treated in section 2.8 and this section will only elaborate on how it is implemented. For the development and testing purposes, this part of the prototype is built as a separate part of the prototype.

The HMM source code from Wataru Kasai is used in the prototype. The use of this source code is governed by a BSD-style license. The source of the code can at the time of writing this report no longer be found and referenced. The source was used for detecting patterns in silhouette motion sequences.

The previous steps generate per video a text file as output containing an observation at each line. These files are used as input in the final step to train and classify activities.

The assumption is made that actions / activities have abstract hidden patterns and that they produce observable signals. Therefore, each activity can be defined as a single HMM. For training and classification, a continuous ergodic 3 state hidden Markov model is used to represent the patterns of each activity.

A continuous HMM was chosen instead of the discrete HMM, because the observations generated by the previous steps contain continuous values. Second, discrete HMM cannot handle multi dimensional observations, which the observations are.

To keep the prototype as generic as possible the ergodic type HMM is chosen. Ergodic transition matrix counts for all transition possibilities and can therefore in theory model left-to-right or circular patterns. In addition, preliminary tests were done on ergodic and circular HMMs (see section 4.3) showing that the ergodic HMM produces slightly better results.

In the same preliminary tests, I found that a 3-state HMM outperformed a 4-state and 5-state HMM using a 24-bin combination histogram. A 4-state HMM outperformed the 3-state HMM using only the 12-bin structure histograms.

Classification

Given the parameters λ of an activity the probability, expressed as the log likelihood, of an observation sequence O can be computed: $P(O|\lambda)$. The probability of the observation sequence is computed for several different activities (different HMMs), and the model which produces the highest log likelihood is chosen as best fit for the observations. With other words, classification is done by selecting an activity model that best matches the observations.

Training

The parameters of each activity first have to be trained. Given an observation sequence the parameters of λ can be computed by maximizing $P(O|\lambda)$. The parameters of λ are optimized to best fit the observations. Traditionally the Baum-Welch method is used to calculate the parameters of λ, the prototype however uses the implementation of Wataru Kasai, which uses k-means clustering for parameter estimation.

3.6 Environment

This section is about the languages, tools, libraries, applications and platform used to build the prototype. With other words the environment.

OpenCV

There are several computer vision libraries available. The most popular computer vision library, also used for this project, is OpenCV. OpenCV is the most mature and it provides a full suite of excellent vision processing tools. It is the most used framework for computer vision, its open source and it has an active development community.

The library is written primary in C with C++ wrappers, the latter making object orientated programming possible. The library also has wrappers for python, which was a big plus in choosing the framework, making it easy to start and play with.

MATLAB was also a good candidate as application with its computer vision library, but it had two major drawbacks: First, I was not familiar with the application and its scripting language and second, the computer vision library lacked many tools OpenCV had. A third minor reason was that the OpenCV library was more low level than the MATLAB library, which gave me more control about the programming.

Language

The programming language primarily chosen for the project was Python. The language was recommended to me, because of its steep learning curve and ease of use. Both factors also contributed to a fast development process.

Although Python lived up to its reputation, when tackling computational expensive calculations, the language seemed rather slow. Furthermore, not all functions of the OpenCV library seemed to have python wrappers. That included functions that were necessary for the prototype. So eventually, a switch was made to C/C++, still quite early in the development phase.

C/C++ did not have the programming comfort of Python, but it made up for speed, low-level control and the availability of the total toolset.

Platform

- Windows 7 Enterprise 32-bit OS
- Intel® Core ™ i5 CPU M540 @ 2.53 GHz 4GB RAM
- Microsoft Visual C++ 2010

3.7 Summery

Step by step, the model behind the prototype is explained in this chapter. It starts with the data preparation step where a multi-scale Laplacian of a Gaussian image is generated from each frame. A multi-scale difference of a Gaussian is used to approximate the LoG.

From the first level in multi-scale LoG local extrema are extracted as candidate features. The features are described by generating an orientation histogram of each pixel's magnitude within a 4σ radius. The descriptions are used for filtering out weak features and tracking features in consecutive frames. From the base scale level, features in higher scales are found by using a hill climber algorithm.

A global descriptor of each frame is represented by a concatenation of a 12-bin orientation histogram summarizing all local magnitude values and by a 12 bin directional histogram of all distances the features have moved over two consecutive frames.

The global description of each frame is used as an observation for training and classification. Every activity is represented by a single HMM. First, they are trained by maximizing the parameters of the models given the observations known to be part of that activity. The trained models are then used for classification by calculating the probability of an observation sequence for each HMM and choosing model with the highest log likelihood.

4. Validation

4.1 Introduction

This chapter discusses how the prototype is tested to validate the model behind the prototype. First, the datasets are discussed. Witch datasets have been chosen and why. Because of the generic goal of the model, two very different datasets have been used to broaden the test cases as much as possible.

The actual test is discussed: the techniques used and the setup. Finally, the results are presented in the last section. Due to the amount of tests only a summary of the results are shown in this chapter. The full list of results can be looked up in the Appendices.

4.2 Datasets

Introduction

In order to validate the model the prototype has to be tested on several datasets. The prototype was tested on two datasets. It was tested on the KTH dataset used in several other researches of human action recognition so the results can be compared. The prototype is also tested on an office dataset specially created for this project. The video's from the office dataset where taken in an office of DDSS. There are several big differences between the two datasets:

1. The sequences in the KTH dataset were taken at different locations but all locations contain homogeneous backgrounds, while all sequences in the office dataset are taken at the same location but contain a complex office background with many details. Note that although the latter is recorded at the same location, the background varies over the sequences because of different illumination conditions and subtle displacements of objects like chairs or other office stationeries.

2. Each sequence in the KTH dataset was shot individually in a set up environment, while the sequences in the office dataset were cut from 7 hours of video taken of the daily activities in an office environment.

3. The activity types in the KTH dataset are relatively simple in the sense that they consist of a single type cyclic action compared to the activity types in the office dataset where the movements can be subtle and the patterns are more complex. (The activity types in the KTH can also be argued to be actions (see Section 2.3). The authors of the dataset defined it as an action dataset.)

KTH dataset

The KTH dataset was originally developed by Schuldt, Laptev and Caputo, Proc. ICPR'04, Cambridge, UK. The KTH dataset database contains six types of human activities as illustrated further below with some screenshots performed several times by 25 subjects in four different scenarios.

Activities:

1. walking
2. jogging
3. running
4. boxing
5. hand waving and

6. hand clapping

Scenarios:

- outdoors
- outdoors with scale variation (zooming in and out while filming or moving from / towards the camera)
- outdoors with different clothes
- indoors

Of each of the different actions, the dataset contains 100 videos and have an average length of four seconds. All videos are down sampled to a resolution of 160 * 120 pixels. Although most of the videos are taken with a static camera, sometimes the shots are filmed by hand and therefore slights camera movements can be recognized.

Office dataset

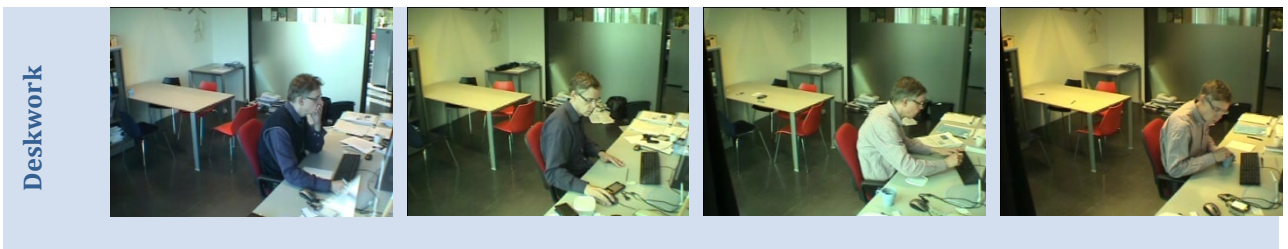
The office dataset is produced by recording daily office activities. Because the activities in the office were not orchestrated in any way like in the KTH dataset. Participants did not get instructions, in fact, the participants forgot most of the time they were being taped. The activities recorded are far more natural and thus allowed a more real life dataset. The recordings are shot over 7 hours on different days and on different hours of the day. This adds to difference background illumination, clothes and people.

The videos in the office dataset are colored, but that does not matter. The prototype grayscales the frames as it reads the video. The videos are significantly larger than the KTH dataset; they have a resolution of 352 * 288 pixels

After the recordings, the video's were examined and different activities were manually distinguished. The 7 hours of video were then cut in samples of 4 seconds each and categorized according to the found activities. The following activities were used for the dataset, each activity containing at least 100 samples:

1. Deskwork
2. Meeting (with 2 or 3 people)
3. (Desk) discussion
4. Movement (random movement around the office)
5. Empty office room

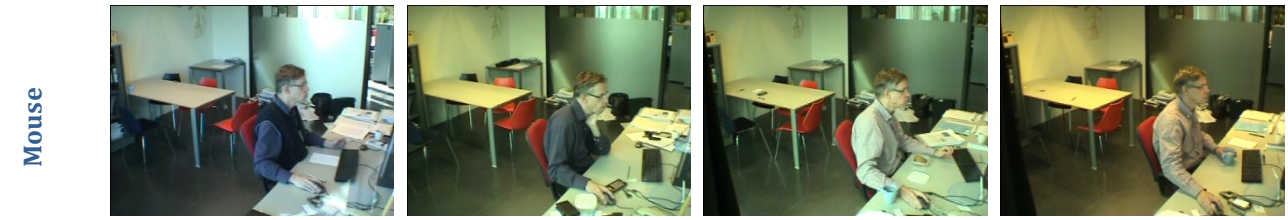
Al samples of each activity are listed in random order. When testing the prototype, the first 50 are used for training, the second batch of 50 video's are used for testing / recognition.

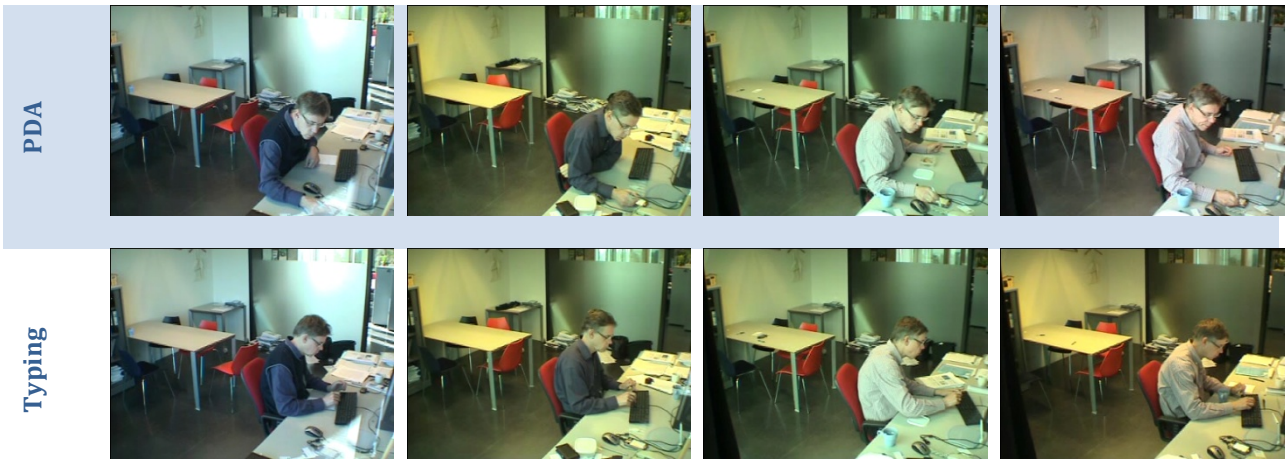


Desk dataset

Because there was a lot of material of desk activities, the desk activities are sub categorized in:

1. Mouse activity
2. PDA activity
3. Typing





Combined office dataset

The desk dataset and office data set are combined into a third and fourth set, the desk activities in the office data set are replaced by the desk activities into two different ways. One in which the pda activities are not part of the set and a second in which movement is not part of the set. The reason for leaving out those activities is so that the set consists of six activities, similar to the amount used in the KTH data set.

1. Mouse activity
2. *PDA activity*
3. Typing
4. Meeting (with 2 or 3 people)
5. (Desk) discussion
6. *Movement (random movement around the office)*
7. Empty office room

4.2 Tests

The prototype is divided in two parts. The first part processes the video and saves the observation sequences to files. The second part consists of the training and classification of the data. In that way it is easier to tune and tweak the HMM and not have to process the video data each time. The parameters that were tuned, were the type of HMM model (ergodic, left-to-right and cyclic) and the number of states.

The first part of the prototype generated three types of observation sequences per action. Observations represented by:

1. structure descriptors (12 dimensions)
2. motion descriptors (12 dimensions)
3. combination of structure and motion descriptors (24 dimensions)

The results are visualized in confusion matrices. A confusion matrix contains the information of the actual classifications and the predicted classifications.

The first tests are done on the KTH dataset. Because the KTH set contains 100 videos for each action. Each action is trained by the first 50 videos and the second 50 videos are used for classification.

In the KTH dataset there is a batch (s2) of videos that have scale invariance. While the actions are performed, the camera zooms in and out or the person varies the distance between him and the camera. Tests have also been done without the scale invariance batch. Still 50 videos are taken to train the models, but only 25 could be used to classify the actions.

Each activity in the office data set contains a minimum of 100 videos as well. For each activity, a file was generated of 100 videos in a random order. In addition, as with the KTH data set, the first 50 videos were taken for training and the second 50 for classification.

In the office set the 'desk' videos have many sub activities, like 'typing', using the mouse and using the pda. Of each sub-activity, tests are done as well and finally the desk activities are tested as part of the general office activities.

4.3 Results

I found in the preliminary tests that a 3-state HMM performed better than the 4-state and 5-state HMM. Therefore, a 3-state HMM was taken as the standard for all further tests. The tests were performed on the global office data set; see section 7.4 and 7.5 for the full results and the graphs below for the summery.

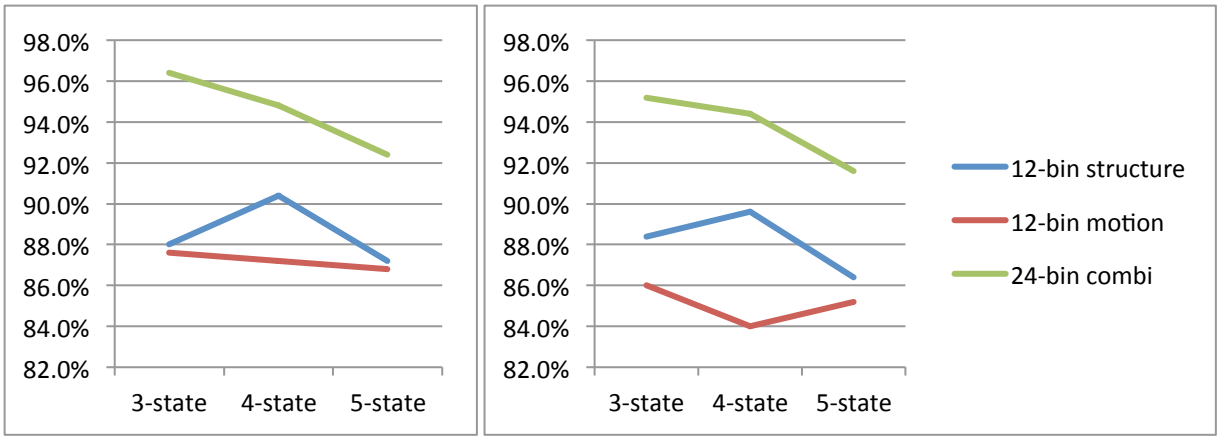


Table 3: Class-relative accuracy of different typed and N-state HMMs, tests performed on the global office dataset. Left: Ergodic HMM, Right: circular HMM

The ergodic typed HMM seems to perform better than the circular type HMM. The ergodic type HMM was taken for further tests because of its performance, but also because of its generic nature.

Below the results of the classifications of the different datasets using a 3-state ergodic HMM. In the appendices, the full results of the tests can be seen.

Dataset	Actions	motion	structure	combined
KTH (with zoom variance)	6	64,3%	49,0%	62,0%
KTH	6	72,7%	46,0%	69,3%
Office global	5	87,6%	88,0%	96,4%
Office subtle actions	3	74,7%	97,3%	97,3%
Office combined 1	6	82,7%	92,3%	97,0%
Office combined 2	6	79,7%	98,0%	98,7%

Table 4: Class-relative accuracy of the 3-state ergodic HMM on different datasets.

KTH vs. Office datasets

The first thing that becomes clear from the results is that the prototype scores significantly better at the office data set than the KTH data set. This is surprising because the activities in the office data set are a lot more complex than the actions in the KTH dataset. The actions in the KTH dataset have recurrent/cyclic patterns which should be easier for an HMM to train and classify. Besides that, the actions in the KTH dataset show a lot more movement, they are far more expressive than the activities in the office set. Some of the activities in the office dataset hardly show movement at all. A reason why the prototype performed better on the office data set can have two reasons:

1. The quality of the videos between the two datasets is very big. The videos of the KTH dataset have a resolution of 160 * 120, while the videos of the office dataset have a resolution of 352 * 288. That is 19200 vs. 101376 pixels a frame. This could in theory lead to far more features and thus more stable observations. In addition, the videos of the KTH dataset seem to have a lot of noise as well, which could significantly lower performance.

2. A single activity might not vary as much in execution over the whole dataset in the office dataset as the actions in the KTH dataset. This could lead, regardless of how complex the activity is, to a more stable HMM of that activity. This leads to better classification results.

Motion vs. Structure descriptors

The motion-based descriptors seem to work better than the structure based descriptors in the KTH dataset and this is the other way around in the office data sets. This is not surprising.

The actions in KTH dataset have, as stated before, very expressive movement, but the videos themselves have a lot of noise. This is why the structural descriptors have trouble calculating stable descriptions.

The activities in the office datasets on the other hand have subtle to no movements. The video's themselves are of a high quality and therefore provide a lot of stable structural information.

Combined descriptors

Combining the motion and structure descriptors has lead to an overall better recognition. This is more applicable for the results in the office dataset, than the KTH dataset. It seems that in the latter case the structure descriptors are the blame.

KTH dataset

The result matrices of the KTH dataset bring clues to where the prototype had trouble. The prototype had trouble with actions that look similar like walking, jogging and running, or with the actions: boxing, hand waving and hand clapping. Especially the first group was mixed up a lot using the structure descriptor. This is not surprising, because a person walking, jogging or running produces the same kind of structural information. The motion descriptor could handle this better, because it contained the notion of speed, which is the big difference between those actions.

Taking out the zoom invariance greatly improved the performance of the motion descriptor. The videos of the camera zooming in and out apparently had a lot effect on the motion descriptor, which is not surprising. Which was surprising is that the structure descriptor performed worse.

Office dataset

The prototype performed well on the office datasets. The motion descriptor only had trouble with the distinction between deskwork and the desk discussion. Although the setup for both activities is similar, the desk discussion should trigger more movement than the deskwork.

The structure descriptor had trouble between the meeting and the movement around the office. This is surprising because the two settings should produce very different image structures. A reason could be that the movement around the office has such a variety of patterns, that the HMM of that activity is less stable and can therefore lead to false classifications.

The prototype performed above expectations at the subtle desk activities (typing, pda, and mouse). The activities are very subtle and only have slight differences between them. Although I have to remark that this activity set only contained three types, so performance is automatically higher than in previous tests. The motion descriptor was the least stable and that is because the motions themselves where very subtle so hard to detect.

What also is surprising, is that the prototype performed better on the combination activity sets despite having to choose between more activities. A reason could be that the activities are sharper categorized, so better HMMs could be trained and better classifications could be made.

Overall, the results were satisfactory, although I expected the prototype would score higher on the KTH set.

5. Conclusion

5.1 Summary

This report presents background information of human activity recognition in combination with intelligent spaces. With the research done a prototype has been developed and put to test with two different datasets.

A low-level approach to the problem at hand is taken. Because neither preliminary assumptions about humans, their structure and their movement nor preliminary assumptions about the prototype's environment are build into the model, the method is as generic and flexible as possible. This is done by using abstract local features that are described in a scale and rotation invariant manner by motion and structure descriptors.

5.2 Conclusions

The objective was to develop a biological inspired generic prototype for robust human activity recognition for in the use in intelligent spaces. Although the results of the prototype using the KTM were below my expectations, the results on the office dataset were above expectations. Although some results were surprising, they were explainable. Moreover, because the office dataset is a far better representation for the use cases in intelligent spaces, I can say that the objective has been met. The prototype successfully proved to be able to cope with daily office activities.

The difference in performance between the KTH dataset and the office dataset are explained by the difference in video quality. Despite the activities in the office dataset being more complex, the videos of office dataset have far less noise and have a larger frame resolution, resulting in more features, which are also more stable. A smaller video resolution also means that the pixel size is relatively larger. Because no interpolation is used to find sub pixel accurate locations of the features, the feature's location tend to jump between pixels.

Object recognition in the prototype is limited to structure descriptions and so by definition does not play a role in the prototype. This made the problem far less complex. In addition, as the results show, object recognition is not necessary for activity recognition at this complexity level.

By combining the motion and structure descriptors, the recognition results improved significantly and are therefore a good addition. The way they were combined is relatively simple and so open for further improvements.

The intelligent space plays an important role in the prototype's scope. The setting offers more stable backgrounds; the background noise is far less and because of space's limited dimensions the scale range is more predictable. If an office is taken as setting for an intelligent space, the activity scope is limited and the number of people using the space at any certain time is limited. Even if multiple people are in the same space in most cases they will be part of the same activity. In that sense the prototype can handle multiple people in a same space.

5.3 Recommendations

There are a number of improvements that can be made to the presented model.

First, each consecutive scale value is calculated by multiplying the scale dimension with a scale factor, this eventually results in very large kernels. Convolving these kernels with the original video frame is very computational expensive. Those calculations take up almost all the processing power. Instead of increasing

the Gaussian kernel for each scale step, the image can be down sampled each step to half the size its previous step. The multi-scale image that results from this is called a Gaussian pyramid. The calculations speed drastically improves, making real time processing possible.

The accuracy of feature positions is limited to the discrete positions of the pixels. By interpolation of nearby data, positions that are more accurate can be computed.

The prototype can only recognize a single activity in a given timeframe. It is not able to recognize a start of an activity of the end of it. When testing the prototype, videos with a duration of 5 seconds were fed to it. Those 5 seconds contained a single activity. If a single video has contained a sequence of different activities, the prototype would only match the most likely one. A solution can be by using a timeframe in which the prototype can match an activity. This can also be improved by taking this timeframe over several timescales, making the activities time invariant.

The prototype cannot handle multiple activities in series, let alone activities that occur in parallel. That was the main reason object recognition should be incorporated in the prototype. That way, because the prototype knows who are what he's dealing with, and can focus more on the individual objects themselves, instead of constantly looking at the whole picture. This could help with the recognition of parallel occurring activities. Besides that, knowing the object can add extra context.

The motion descriptor describes the movement of a feature by the absolute path it has taken. What can be interesting is to describe the features movement relative to its parent feature in a higher scale (see figure 19 right). As explained before stable critical points are chosen to be features. Critical points form paths through different scales. These paths end because they end up as a 'detail' in higher layer image structures, which in their turn will also end as top points. Sub structures can change in relation to their parent structures. If the motion of sub structures is described in relation to their parent structures, the motion is invariant to the viewports movement. I played with the idea in the current prototype, but the critical point tree(s) did not seem to be stable enough. I recommend improving the stability of the feature tree.

In the prototype, the training and classification are separate. The prototype has to be trained first before it can classify activities. This gives some extra problems. First, training data needs to be acquired. The prototype is only as good as the training data is. Once it is trained and starts classifying, it cannot improve itself anymore. Ideally, the prototype needs to have the property of *unsupervised learning.* This could fit perfectly in the bigger picture of a plug and play system.

5.4 Last notes

First of all, I want to thank Bauke de Vries for his support, guidance and patients. Especially the last part of the project, which was a bit of a struggle for me. I want to thank Joran Jessurun, because his door was always open for helping me out with C/C++, python, or anything which had to do with programming the prototype. I want to thank Oliver Amft for helping me out with the classification concepts of the prototype, in particular his help on the Hidden Markov models. I also want to thank Sjoerd Buma, who was always willing to help me out in the Computer Lab.

And last but not least, I want to thank family, friends and my girlfriend Angie for their support and confidence in a good ending of the project.

6. Literature

1. Aarts, E. & S. Marzano (2003). *The new everyday: Views on ambient intelligence*, 010 Publishers, Rotterdam.

2. Ali, A., & Aggarwal, J. (2001) Segmentation and recognition of continuous human activity, *In IEEE Workshop on Detection and Recognition of Events in Video*, 2001, pp. 28–35.

3. Aggarwal, J.K. & Cai, Q. (1997) Human Motion Analysis: A Review, *Computer Vision Image Understanding*, 73 (3)

4. Andrieu, C., De Freitas, N., Doucet, A., & Jordan, M. I. (2003) An Introduction to MCMC for machine learning. *Machine learning*, Vol. 50, pp. 5–43

5. Ben-Arie, J., Wang, Z., P. Pandit, P., & Rajaram, S. (2002) Human activity recognition using multidimensional indexing, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 24, No. 8, pp. 1091-1104.

6. Bodor, R., Jackson, B., & Papanikolopoulos, N. (2003) Vision-Based Human Tracking and Activity Recognition, In *Proceedings of 11th Mediterranean Conference on Control and Automation*, June 2003.

7. Bobick, A.F. & Davis, J.W. (2001) The Recognition of Human Movement Using Temporal Templates, In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 23, No. 23, March 2001

8. Brooks, R.A., Coen, M. , Dang, D., DeBonet, J., Kramer, J., Lozano-Perez, T., Mellor, J., Pook, P., Stauffer, C., Stein, L., Torrance, M., Wessler, M. (1997) The Intelligent Room Project, In *Proceedings of the Second International Cognitive Technology Conference* (CT'97), August 1997, Aizu, Japan.

9. Brox, T., Bruhn, A., Papenberg, N. & Weickert, J. (2004) High Accuracy Optical Flow Estimation Based on a Theory for Warping, In: *Computer Vision, Proc. 8th European Conference on Computer Vision*, T. Pajdla and J.Matas (Ed.) vol. 3024 of Lecture Notes in Computer Science, Springer, Prague, Czech Republic, pp. 25–36.

10. Bruhn, A., Brox, T., Didas, S. & Weickert, J. (2006) Highly Accurate Optic Flow Computation with Theoretically Justified Warping, *International Journal of Computer Vision*, 67 (2), 141–158

11. Chang, T., Gong, S. (2001) Tracking Multiple People with a Multi-Camera System, In *Proceedings of IEEE Workshop on Multi-Object Tracking*, with ICCV '01, Vancouver, B.C., Canada, July 2001.

12. Du, W. & Piater, J. (2007) Multi-camera People Tracking by Collaborative Particle Filters and Principal Axis-Based Integration, In*: Asian Conf on Comp. Vision* (ACCV)

13. Flusser, J. & Suk, T. (1993) Pattern recognition by affine moment invariants, *Pattern Recognition*, Vol. 26, pp. 167–174.

14. Grimson, E. (1990) Object Recognition by Computer: The Role of Geometric Constraints*, MIT Press*, Cambridge, MA.

15. Hongeng, S., Nevatia, R., Bremond, F. (2004) Video-based event recognition: activity representation and probabilistic recognition methods, *Computer Vision and Image Understanding*, Elsevier.

16. Hongeng, F.B.S. & Nevatia, R. (2000) Representation and optimal recognition of human activities, In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (CVPR'00).

17. Jansen, B.J., Florack, L.M.J., Duits, R. & Ter Haar Romeny, B.M. (2006) Optic Flow from Multi-scale Dynamic Anchor Point Attributes, In: *Campilho*, A., Kamel, M.S. (eds.), ICIAR 2006. LNCS, vol. 4141, pp. 767–779. Springer, Heidelberg

18. Jhuang, H. Serre, T., Wolf, L. & Poggio, T (2007). A biologically inspired system for action recognition. In *ICCV*, 2007.

19. F.M.W. Kanters, B. Platel, L.M.J. Florack & B.M. ter Haar Romeny. (2003) Content based image retrieval using multiscale top points. In *Proceedings of the 4th international conference on Scale Space Methods in Computer Vision* (Isle of Skye, UK, June 2003), pp. 464–478.

20. Kim, K. & Davis, L. S. (2006) Multi-camera Tracking and Segmentation of Occluded People on Ground Plane Using Search-Guided Particle Filtering In: *ECCV*, pp. 98–109

21. Lindeberg, T. (2008) Scale-Space, In: *Encyclopedia of Computer Science and Engineering* (Benjamin Wah, ed), John Wiley and Sons, Volume IV, pages 2495-2504, Hoboken, New Jersey.

22. Lowe, D. (1987) Three-Dimensional Object Recognition from Single Two-Dimensional Images, In: *Artificial Intelligence*, 31, 3 (March 1987), pp. 355–395.

23. Madabhushi, A. & Aggarwal, J. (1999) A bayesian approach to human activity recognition, In: *Proc. 2nd International Workshop on Visual Surveillance*, 1999, pp. 25–30.

24. Marr D, Hildreth E. (1980) Theory of Edge Detection, In: *Proceedings of the Royal Society of London. Series B*, Biological Sciences, Vol. 207, No. 1167. (Feb. 29, 1980), pp. 187-217.

25. Medioni, G., Cohen, I., Brémond, F., Hongeng, S., & Nevatia, R. (2001) Event detection and analysis from video streams, In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 23, No. 8, pp. 873–889, 2001

26. Meng, H., Pears, N. (2009) Descriptive temporal template features for visual motion recognition, *Pattern Recognition Letters*, Elsevier

27. Mitchell, T., Buchman, B.. DeJong, G., Dietterich, T., Rosenbloom, P. & Waibel, A. (1990) Machine Learning, Annual Review *Computer Science*, 4:417-33

28. Moeslund, T.B., Granum, E. (2001) A Suvey of Computer Based Human Motion Capture, *Computer Vision and Image Understanding* 81, 231–268

29. Moeslund, T.B., Hilton, A. & Krüger, V. (2006) A survey of advances in vision-based human motion capture and analysis, *Computer Vision and Image Understanding* (CVIU) 104 (2–3) 90–126.

30. Moore, D.J., Essa, I.A. & Hayes, M. (1999) Exploiting human actions and object context for recognition tasks. In: *IEEE International Conference on Computer Vision*, 1999.

31. Niebles, J. C., Wang, H. & Fei-Fei, L. (2006) Unsupervised learning of human action categories using spatial temporal words. In *British Machine Vision Conference*, Edinburgh, 2006.

32. Olivier, N., Horovitz, E., Garg, A. (2002) Layered representations for human activity recognition, In: *IEEE International Conference on Multimodal Interfaces*, 2002.

33. Oliver, N., A Garg & E Horvitz (2004) Layered representations for learning and inferring office activity from multiple sensory, *Computer Vision and Image Understanding*, Elsevier.

34. Oliver, N. & Horvitz, E. (2004) S-SEER: Selective perception in a multimodal office activity recognition system, *Multimodal Interaction and Related Machine Learning Algorithms*, pp. 122-135.

35. Osmani, V., Balasubramaniam, S., Botich,W. (2007) Human activity recognition in pervasive health-care: Supporting efficient remote collaboration*, Journal of Network and Computer Applications*, 31(4), 628-655.

36. Rabiner, L.R. & Juang, B.H. (1986) An Introduction to Hidden Markov Models. *IEEE ASSP Magazine*, Januari 1986, pp. 4-16.

37. Ribeiro, P. C. & Santos-Victor, J. (2005) Human activity recognition from video: Modeling, feature selection and classification architecture, In: *Proceedings of the International Workshop on Human Activity Recognition and Modeling* 2005, Vol. 1, pp. 61–78.

38. Pope, A. R. (1994) Model-based object recognition: A survey of recent research, In: *Tech. Rep*, TR-94-04, University of British Columbia.

39. Robertson, N., Reid, I. (2006) A general method for human activity recognition in video, *Computer Vision and Image Understanding*, 2006, Vol. 104, No. 2-3, pp. 232-*.

40. Roth, P.M. & Winter, M. (2008) Survey of Appearance-based Methods for Object Recognition, *Technical Report ICG-TR-01/08*, Graz University of Technology, Institute for Computer Graphics and Vision

41. Ryoo, M.S., Aggarwal (2009) Semantic Representation and Recognition of Continued and Recursive Human Activities, *International Journal of Computer Vision* (IJCV), 82(1):1–24, April.

42. Singh, M., Basu, A. & Mandal, M., (2004) Human Activity Recognition based Silhouette Directionality, *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 9, pp. 1280–1292, 2008

43. Sun, X. & Chen, C.W. (2002) Probabilistic motion parameter models for human activity recognition, In*: IEEE International Conference on Pattern Recognition* (ICPR02).

44. Turaga P, Chellappa R, Subrahmanian V, Udrea O (2008) Machine recognition of human activities: A survey. In: *IEEE Transactions on Circuits and Systems for Video Technology*, 28, 18:1473–148

45. Wojek, C., Nickel, K., Stiefelhagen, R. (2006) Activity Recognition and Room-Level Tracking in an Office Environment, In: *IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems* (MFI'06), September 2006, Heidelberg, Germany.

46. Yao, J. & Odobez, J. (2008) Multi-Camera Multi-Person 3D Space Tracking with MCMC in Surveillance Scenarios, In: *Workshop on Multi-camera and Multi-modal Sensor Fusion Algorithms and Applications*, M2SFA2 2008, Marseille, France

47. Zouba, N., Boulay, B. Bremond, F. & Thonnat, M. (2008) Monitoring activities of daiy living (adls) of elderly based on 3d key human postures, In: *The 4th International Cognitive Vision Workshop* (ICVW08), Santorini, Greece.

7. Appendices

7.1. Results KTH dataset

	boxing	Hand clapping	Hand waving	walking	jogging	running
boxing	44%	46%	4%	6%	0%	0%
handclapping	12%	78%	6%	4%	0%	0%
handwaving	10%	28%	60%	2%	0%	0%
walking	0%	0%	0%	90%	10%	0%
jogging	0%	0%	0%	10%	82%	8%
running	0%	0%	0%	8%	60%	32%

Only motion description, class-relative accuracy: **64,3%**

	boxing	Hand clapping	Hand waving	walking	jogging	running
boxing	22%	0%	16%	20%	18%	24%
handclapping	2%	28%	50%	14%	2%	4%
handwaving	4%	6%	68%	14%	2%	6%
walking	0%	0%	0%	66%	6%	28%
jogging	0%	0%	0%	24%	36%	40%
running	0%	0%	0%	10%	16%	74%

Only structure description, class-relative accuracy: **49,0%**

	boxing	Hand clapping	Hand waving	walking	jogging	running
boxing	54%	12%	22%	12%	0%	0%
handclapping	4%	44%	42%	10%	0%	0%
handwaving	4%	6%	78%	12%	0%	0%
walking	0%	0%	0%	72%	20%	8%
jogging	0%	0%	0%	14%	68%	18%
running	0%	0%	0%	6%	38%	56%

Combined motion and structure descriptor, class-relative accuracy: **62,0%**

7.2. Results KTH dataset (no zoom invariants)

	boxing	handclapping	handwaving	walking	jogging	running
boxing	52%	40%	4%	4%	0%	0%
handclapping	4%	88%	8%	0%	0%	0%
handwaving	0%	16%	84%	0%	0%	0%
walking	0%	0%	0%	84%	4%	12%
jogging	0%	0%	0%	4%	68%	28%
running	0%	0%	0%	4%	36%	60%

Only motion description, class-relative accuracy: **72.7%**

	boxing	handclapping	handwaving	walking	jogging	running
boxing	18%	4%	10%	8%	8%	2%
handclapping	2%	16%	26%	4%	2%	0%
handwaving	4%	8%	32%	2%	2%	2%
walking	0%	0%	0%	24%	22%	4%
jogging	0%	0%	0%	14%	24%	12%
running	0%	0%	0%	4%	22%	24%

Only structure description, class-relative accuracy: **46,0%**

	boxing	handclapping	handwaving	walking	jogging	running
boxing	24%	8%	12%	2%	0%	4%
handclapping	4%	30%	14%	0%	2%	0%
handwaving	0%	4%	42%	2%	0%	2%
walking	0%	0%	0%	34%	4%	12%
jogging	0%	0%	0%	0%	32%	18%
running	0%	0%	0%	0%	4%	46%

Combined motion and structure descriptor, class-relative accuracy: **69,3%**

7.3. Comparing scale factor 1.5 & 1.6

Test were done on Global Office Activity Dataset with an Ergodic HMM. The results are shown of the 24-bin combi descriptor. Class-relative accuracy is noted to the bottom right of the table.

k = 1.5 (used for all other tests)

3-State HMM

	desk	meeting	empty	discussion	movement
desk	46	-	-	-	4
meeting	-	48	-	-	2
empty	-	-	50	-	-
discussion	-	-	-	47	3
movement	-	-	-	-	50

96,4%

4-State HMM

	desk	meeting	empty	discussion	movement
desk	42	-	-	2	6
meeting	-	48	-	-	2
empty	-	-	50	-	-
discussion	-	-	-	47	3
movement	-	-	-	-	50

94,8%

5-State HMM

	desk	meeting	empty	discussion	movement
desk	43	-	-	-	7
meeting	-	41	-	-	9
empty	-	-	50	-	-
discussion	-	-	-	47	3
movement	-	-	-	-	50

92,4%

k = 1.6

3-State HMM

	desk	meeting	empty	discussion	movement
desk	45	-	-	-	5
meeting	-	44	-	-	6
empty	-	-	50	-	-
discussion	-	-	-	48	2
movement	-	-	-	-	50

94,8%

4-State HMM

	desk	meeting	empty	discussion	movement
desk	42	-	-	2	6
meeting	-	45	-	-	5
empty	-	-	50	-	-
discussion	-	-	-	48	2
movement	-	-	-	-	50

94,0%

5-State HMM

	desk	meeting	empty	discussion	movement
desk	44	-	-	-	6
meeting	-	40	-	-	10
empty	-	-	50	-	-
discussion	-	-	-	48	2
movement	-	-	-	-	50

92,8%

7.4. Results Global Office Activity Dataset Ergodic HMM

Class-relative accuracy is noted to the bottom right of the table

3-State HMM

12-bin structure

	desk	meeting	empty	discussion	movement
desk	43	-	-	-	7
meeting	-	35	-	-	15
empty	-	-	50	-	-
discussion	2	-	-	43	5
movement	1	-	-	-	49

88,0%

12-bin motion

	desk	meeting	empty	discussion	movement
desk	42	-	2	5	1
meeting	2	44	3	-	1
empty	-	3	47	-	-
discussion	5	-	-	45	-
movement	3	3	1	2	41

87,6%

24-bin combi

	desk	meeting	empty	discussion	movement
desk	46	-	-	-	4
meeting	-	48	-	-	2
empty	-	-	50	-	-
discussion	-	-	-	47	3
movement	-	-	-	-	50

96,4%

4-State HMM

12-bin structure

	desk	meeting	empty	discussion	movement
desk	42	-	-	-	8
meeting	-	38	-	-	12
empty	-	-	50	-	-
discussion	-	-	-	47	3
movement	-	1	-	-	49

90,4%

12-bin motion

	desk	meeting	empty	discussion	movement
desk	41	1	1	7	-
meeting	1	45	3	-	1
empty	-	3	47	-	-
discussion	7	-	-	43	-
movement	2	2	2	2	42

87,2%

24-bin combi

	desk	meeting	empty	discussion	movement
desk	42	-	-	2	6
meeting	-	48	-	-	2
empty	-	-	50	-	-
discussion	-	-	-	47	3
movement	-	-	-	-	50

94,8%

5-State HMM

12-bin structure

	desk	meeting	empty	discussion	movement
desk	38	-	-	-	12
meeting	-	33	-	-	17
empty	-	-	50	-	-
discussion	-	-	-	47	3
movement	-	-	-	-	50

87,2%

12-bin motion

	desk	meeting	empty	discussion	movement
desk	43	-	2	5	-
meeting	1	45	4	-	-
empty	-	4	46	-	-
discussion	8	-	-	42	-
movement	2	2	2	3	41

86,8%

24-bin combi

	desk	meeting	empty	discussion	movement
desk	43	-	-	-	7
meeting	-	41	-	-	9
empty	-	-	50	-	-
discussion	-	-	-	47	3
movement	-	-	-	-	50

92,4%

7.5. Results Global Office Activity Dataset Circular HMM

Class-relative accuracy is noted to the bottom right of the table

3-State HMM

12-bin structure

	desk	meeting	empty	discussion	movement
desk	42	-	-	-	8
meeting	-	35	-	-	15
empty	-	-	50	-	-
discussion	-	-	-	45	5
movement	1	-	-	-	49

88,4%

12-bin motion

	desk	meeting	empty	discussion	movement
desk	39	1	1	8	1
meeting	1	42	6	-	1
empty	-	2	48	-	-
discussion	5	-	-	45	-
movement	2	3	1	3	41

86,0%

24-bin combi

	desk	meeting	empty	discussion	movement
desk	45	-	-	-	5
meeting	-	46	-	-	4
empty	-	-	50	-	-
discussion	-	-	-	47	3
movement	-	-	-	-	50

95,2%

4-State HMM

	desk	meeting	empty	discussion	movement
desk	41	-	-	-	9
meeting	-	38	-	-	12
empty	-	-	50	-	-
discussion	-	-	-	46	4
movement	-	1	-	-	49

89,6%

	desk	meeting	empty	discussion	movement
desk	30	1	1	18	-
meeting	1	44	4	-	1
empty	-	3	47	-	-
discussion	2	-	-	48	-
movement	2	2	2	3	41

84,0%

	desk	meeting	empty	discussion	movement
desk	42	-	-	-	8
meeting	-	47	-	-	3
empty	-	-	50	-	-
discussion	-	-	-	47	3
movement	-	-	-	-	50

94,4%

5-State HMM

	desk	meeting	empty	discussion	movement
desk	37	-	-	-	13
meeting	-	33	-	-	17
empty	-	-	50	-	-
discussion	-	-	-	46	4
movement	-	-	-	-	50

86,4%

	desk	meeting	empty	discussion	movement
desk	41	-	2	7	-
meeting	1	42	6	-	1
empty	-	5	45	-	-
discussion	7	-	-	43	-
movement	2	2	2	2	42

85,2%

	desk	meeting	empty	discussion	movement
desk	41	-	-	-	9
meeting	-	41	-	-	9
empty	-	-	50	-	-
discussion	-	-	-	47	3
movement	-	-	-	-	50

91,6%

7.6. Results Subtle Office Actions Dataset

	typing	mouse	pda
typing	80%	18%	2%
mouse	10%	72%	18%
pda	0%	28%	72%

Only motion description, class-relative accuracy: **74,7%**

	typing	mouse	pda
typing	100%	0%	0%
mouse	2%	98%	0%
pda	4%	2%	94%

Only structure description, class-relative accuracy: **97,3%**

	typing	mouse	pda
typing	100%	0%	0%
mouse	4%	96%	0%
pda	4%	0%	96%

Combined motion and structure descriptor, class-relative accuracy: **97,3%**

7.7. Results Combined Global and Subtle Office Actions Dataset

	typing	mouse	meeting	empty	discussion	movement
typing	74%	18%	0%	0%	8%	0%
mouse	10%	84%	0%	0%	6%	0%
meeting	2%	4%	86%	6%	0%	2%
empty	0%	0%	10%	90%	0%	0%
discussion	4%	18%	0%	0%	78%	0%
movement	2%	0%	6%	2%	6%	84%

Only motion description, class-relative accuracy: **82,7%**

	typing	mouse	meeting	empty	discussion	movement
typing	100%	0%	0%	0%	0%	0%
mouse	0%	98%	0%	0%	0%	2%
meeting	0%	0%	70%	0%	0%	30%
empty	0%	0%	0%	100%	0%	0%
discussion	0%	0%	0%	0%	86%	14%
movement	0%	0%	0%	0%	0%	100%

Only structure description, class-relative accuracy: **92,3%**

	typing	mouse	meeting	empty	discussion	movement
typing	100%	0%	0%	0%	0%	0%
mouse	2%	94%	0%	0%	2%	2%
meeting	0%	0%	90%	0%	0%	10%
empty	0%	0%	0%	100%	0%	0%
discussion	0%	0%	0%	0%	98%	2%
movement	0%	0%	0%	0%	0%	100%

Combined motion and structure descriptor, class-relative accuracy: **97,0%**

7.8. Results Combined Global and Subtle Office Actions Dataset 2

	typing	mouse	pda	meeting	empty	discussion
typing	72%	18%	0%	0%	0%	10%
mouse	10%	72%	14%	0%	0%	4%
pda	0%	20%	64%	0%	2%	14%
meeting	2%	4%	2%	86%	6%	0%
empty	0%	0%	0%	6%	94%	0%
discussion	0%	8%	2%	0%	0%	90%

Only motion description, class-relative accuracy: **79,7%**

	typing	mouse	pda	meeting	empty	discussion
typing	100%	0%	0%	0%	0%	0%
mouse	0%	98%	0%	0%	0%	2%
pda	4%	2%	94%	0%	0%	0%
meeting	0%	0%	0%	100%	0%	0%
empty	0%	0%	0%	0%	100%	0%
discussion	4%	0%	0%	0%	0%	96%

Only structure description, class-relative accuracy: **98,0%**

	typing	mouse	pda	meeting	empty	discussion
typing	100%	0%	0%	0%	0%	0%
mouse	2%	96%	0%	0%	0%	2%
pda	4%	0%	96%	0%	0%	0%
meeting	0%	0%	0%	100%	0%	0%
empty	0%	0%	0%	0%	100%	0%
discussion	0%	0%	0%	0%	0%	100%

Combined motion and structure descriptor, class-relative accuracy: **98,7%**