Probabilistic framework for multi-target tracking using multi-camera: applied to fall detection

Victoria Rudakova

Masteroppgave
Master i Teknologi - Medieteknikk
30 ECTS
Avdeling for informatikk og medieteknikk
Høgskolen i Gjøvik, 2010

Avdeling for
informatikk og medieteknikk
Høgskolen i Gjøvik
Postboks 191
2802 Gjøvik

Department of Computer Science
and Media Technology
Gjøvik University College
Box 191
N-2802 Gjøvik
Norway

Probabilistic framework for multi-target tracking using multi-camera: applied to fall detection

Victoria Rudakova

2010/07/01

Abstract

The developments in health care lead to longer life expectancy in developed countries. The growth of the number of seniors put new challenges to health care services, caregivers and family members. One of the major challenges is how to prevent elderly people from falling down. Especially, if the person lives alone, falling can be quite dangerous for his / her health. Falling down can cause serious consequences like broken bones or lost consciousness for elderly people. Therefore, it is of key importance to be able to provide help as soon as possible, or, even try to prevent the incident. Combining video-based surveillance and computer vision techniques could bring us towards intelligent systems that could help to monitor the elderly and raise an alarm in case of unusual and dangerous events.

The main question is how to automate fall detection using only video information (like camera network) with application in public health and health services (i.e., elderly houses, hospitals). Generally, when talking about video surveillance systems, there are some elements that would remain the same in every system irrespective of the application. These are: tracking, data association in context of camera network and post-processing based on extracted information (like activity recognition). So, this thesis addresses each of these questions and tries to find a generalized framework that could integrate all the methods in one system.

One of the main criterion of the project was to avoid camera calibration - that is, correspondence between views should be based on some confidence or probability function between camera views. Therefore, it was decided to adopt a Bayesian framework for multi-target multi-camera tracking [1] that fulfills our requirements. When considering a single view from one camera, the system allows to represent each target as a set of samples (or particles) that have probabilistic measures (weights) proportional to the likelihood of the target being at that position. For multi-camera system, one needs to associate corresponding targets among camera views. Gale-Shapley algorithm for finding stable matching between two classes (in our case - cameras) was modified for this purpose. For fall detection part, silhouette- and velocity-based set of features were extracted and passed through Support Vector Machine (SVM) to separate fall events from other activities.

A set of test videos were recorded using two cameras to test stability of the tracking framework and the reliability of fall detection algorithm. The initial results are encouraging. Further integration of additional features may improve the performance of the system.

Keywords: surveillance, multiple camera tracking, multiple person tracking; recursive Bayesian estimation, Sequential Monte-Carlo Methods, particle filtering; camera network, data association, stable marriage problem; fall detection, activity recognition, support vector machine.

Preface

This Master thesis was done at Gjøvik University College (Norway), Department of Computer Science and Media Technology, in the period from January 2010 to July 2010, under supervision of Dr. Tech. Faouzi Alaya Cheikh.

I would like to thank Dr. Faouzi Alaya Cheikh from all my heart for everything he did for me, for his ideas, support and encouragement. It was a reward for me just to talk with him - whether we discussed my thesis, or a life. His sensible and priceless advices, his guidance and attitude changed me in a better way and, of course, taught me many things that I will never forget.

I thank my CIMET and HIG friends warmly for all the great time we spent together and for all the sweet memories. Special thanks to my best friend Talha Ahmed, for his belief in me and constant support.

Victoria Rudakova, 2010/07/01

Contents

Abstract . . . . **iii**
Preface . . . . **v**
Contents . . . . **vii**
1 Introduction **1**
1.1 Motivation 1
1.2 Problem Statement 1
1.3 Research questions 2
1.4 Methodology 3
1.5 Thesis outline 4
2 Literature Review **5**
2.1 Fall detection systems 5
2.2 Existing fall detection video-based systems 6
2.3 Target tracking 7
2.3.1 Object representation 7
2.3.2 Feature selection for tracking 9
2.3.3 Tracking algorithms 9
2.4 Multi-camera systems and data fusion 12
3 Proposed Framework **15**
3.1 System overview 15
3.2 Multi-target tracking 15
3.2.1 Bayesian sequential estimation 15
3.2.2 Bayesian framework for multiple target tracking 18
3.2.3 Sequential Monte-Carlo implementataion 20
3.2.4 Modeling of densities 26
3.3 Multi-camera data fusion 30
3.3.1 Problem statement 30
3.3.2 Stable marriage problem 32
3.4 Target initialization 35
3.5 Fall detection 35
3.5.1 Feature extraction 35
3.5.2 Classification 36
3.6 Implementation issues 37
4 Experimental Results **39**
4.1 Experimental setup 39
4.2 Analysis 39
5 Conclusions and Future Directions **43**
Bibliography **47**

1 Introduction

1.1 Motivation

Nowadays we witness an increase senior numbers and population aging in many counties. According to [2] in 2006, 37 million people over the age of 65 in US made up 12% of the total population. Over the 20^{th} century, the older population grew from 3 to 37 million; and as to oldest-old population (those age 85 and over) made it from $100,000$ in 1900 to 5.3 million in 2006. The same situation is happening now in the western countries - Europe and Norway as well. The phenomenon affects many aspects of our society, challenging policymakers, families, business, health-care providers to meet the needs of aging individuals. And one of the main questions we consider here is how to provide safety to elderly allowing them to live longer independently.

It was stated that each year, an estimated one third of older adults fall; and the probability to fall down increases with advancing age [3]. Statistics shows that in 2005, a total of $15,802$ persons aged ≥ 65 years died as a result of injuries from falls in US. However, this does not include the number of elderly who fell and were not injured or who experienced minor or moderate injuries.

Now we clearly can see the problem: fall represents a great danger for elderly people, especially if the person can not get up afterwards by him- or herself and has to spend hours lying down, in pain or unconsciousness. Due to increasing number of elderly, it is not possible to take care for them personally, therefore, there must be other way to detect a fall and even try to prevent it.

Thanks to today's technologies, several solutions already exist that help to address the problem. Small electronic devices called wearable buttons and wearable sensors are supposed to be worn by a person (e.g., on the hip) can rise an alarm in case of fall. However, both of the mentioned devices have disadvantages. Wearable button can be useless in case if person loses consciousness and does not press the button. As to wearable device, the person can forget to wear it, or simply feel uncomfortable using them.

To overcome this drawback, another solution must be considered; something that does not include the direct interaction with the person. Commonly used video-based approach could be of greater use, since it is contact-less and is already used for many different security applications.

1.2 Problem Statement

Considering all the said above, this Master thesis addresses the question of how to build a robust video-based system, the main purpose of which is monitoring and analysis of people's actions. To be more specific, the system must be supplied with multi-target multi-camera tracking algorithm, which will feed data to for the next high-level activity analysis. Therefore, we define the requirements for our system:

- track multiple targets and maintain their identities over multiple cameras

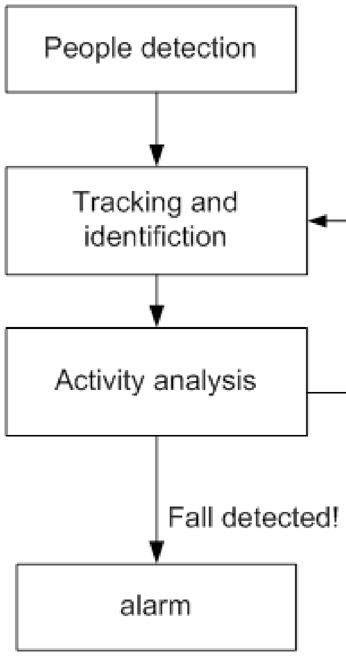


Figure 1: General block-diagram of the system.

- cope with background clutter, illumination changes, shadows, etc.
- handle mutual occlusions that can occur from other targets and also from furniture, in the room

The general workflow of the desired system is shown on the figure 1. Each of the shown blocks will be expanded and explained later in section 3.2.

1.3 Research questions

The objective of this thesis is to develop a system that can detect falling down events of people based on video analysis using multiple cameras. Figure 2 represents typical situation, where we may have people occluding each other and there is also a chance that somebody can fall.

When working with multiple cameras, it's necessary to think how the data fusion will be performed. For this project it was decided to exclude calibration procedure, since it has its own disadvantages like it is not so easy to conduct if cameras are located quite far from each other; and also it needs re-calibration in case if camera locations were suddenly changed.
The research questions we proposed for consideration:

1. Questions related to multiple target tracking:
    - how to perform target detection (initialization)?
    - how to provide target tracking over period of time and over multiple cameras?
    - how to solve problems of mutual occlusions when dealing with multi-target tracking? how

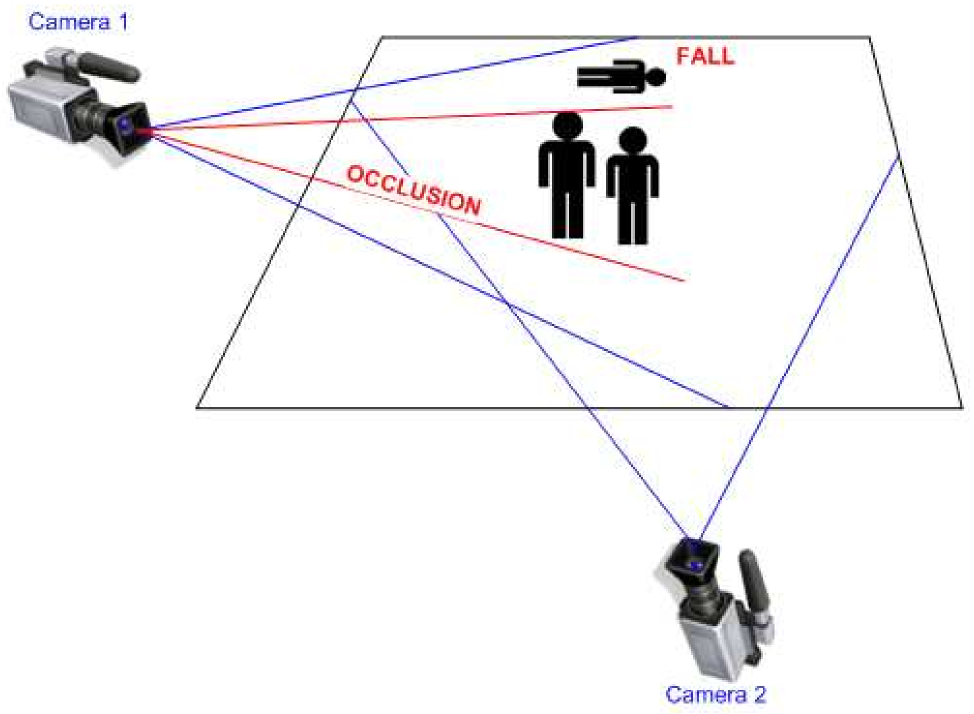


Figure 2: A typical scenario of multi-target tracking with two camera system.

to take into consideration the impact of environment (i.e., furniture) that also creates occlusions?

2. Questions related to data fusion among cameras:
   - how to avoid camera calibration or any other initial configuration?
   - how to perform association between the data from different cameras without calibration information?
3. Question related to fall detection:
   - how to distinguish falling down from other everyday people activities?

1.4 Methodology

The next methodology was proposed to find answers on the raised research questions:

1. Literature review: a lot of work can be found in the field. Literature survey was conducted in order to find an appropriate theoretical strategy.
2. Design of the algorithm: when thinking about this project globally, it can be divided into three main parts: people tracking, data fusion and fall recognition. Each of the tasks was treated in steps. First, a general probabilistic framework was adopted for the tracking; second, we tried to find a way how to improve system performance by using information from other cameras;

and third, fall detection was embedded.

3. Implementation: it was decided to use Visual C++.
4. Setup the experiment: several videos with two and more interacting people were recorded by using two USB cameras.
5. Analysis and results: must include how good tracking is performed, how often the track can be lost and also false and positive alarms of fall detections.

1.5 Thesis outline

The remainder of the thesis is organized as follows. The next chapter contains a review of the literature on modern tracking systems, also previous work on object tracking and data association when dealing with multi-camera system. Then an overview of existing particle filter based systems and algorithms is presented. Chapter 3 gives details about the chosen method and implementation issues. In chapter 4 experimental setup is described, as well as system output is provided. Finally, chapter 5 concludes the thesis results with future extension included as well.

2 Literature Review

2.1 Fall detection systems

This section provides a brief overview of fall detection systems and the advantages of the video-based ones.

Nowadays many contributions are made to address the problem fall detection due to the big demand and social values of such products and technologies. Work made during recent years produced a series of devices helping to lift the quality of elderly's life. Depending on how fall is detected, solutions can be divided into three main approaches based on [4]:

- wearable devices
- ambient devices
- cameras (vision-based) approach

Though, we have different approaches and methods to detect falls, all the existing systems follow one general framework, shown in figure 3. The differences are only in the principles of working of each device or sensor, for example, a setting of specific threshold can be already an indicator of a fall; at the same time, the fall can be induced from more complicated image processing algorithm.

So, there are wearable devices which are supposed to be worn by a person. Such devices help to detect the posture and motion of the body; later classifier helps to identify the specific events that may include falls [5], [6], [7].

In general the goal of such systems is to generate a warning to caregivers or an alarm to monitoring authorities. It means that no human interaction is needed as opposed to manual systems [8]: where the output of the device / system is operated by a human and does not automatically detect a fall. The person who wears it, can simply push a button in case if he / she needs assistance. However, it will not be so effective in case of losing consciousness, for example.

Considering the above fact, we can conclude that, probably, wearable devices are not the best option for fall detection; also because there is a big chance that the senior can simply forget to

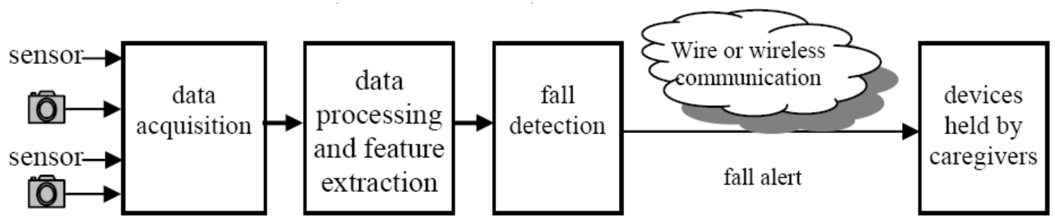

sensor
sensor
data acquisition
data processing and feature extraction
fall detection
Wire or wireless communication
fall alert
devices held by caregivers

Figure 3: General framework of fall detection [4].

wear it or feel uncomfortable. This was the main reason why should look towards the camera-based systems. Furthermore, in most elderly homes / nursing homes there are video surveillance systems that may be used to include additional functionality such as fall detection.

2.2 Existing fall detection video-based systems

In this section a brief overview of video-based fall detection systems will be given.

Among all fall detection algorithms, a first major subdivision is based on clear immediate visual cues, for example, sudden change in dimensions [8]:

- aspect ratio of the bounding box [9], [10]: it is a simple technique to detect a fall. The bounding box is represented by a rectangle which can be drawn around the moving object and which indicates its location on the image; the aspect ratio is the ratio between the two dimensions in x and y directions of the bounding box. When a person falls, the bounding box is supposed to change drastically in both x and y directions, therefore, the aspect ratio will change too [11].
- horizontal and vertical gradient [12]: during the fall, a horizontal gradient will be less than the horizontal one
- fall angle [13]: defined as an angle between the ground and the person from where it is certain that the person will fall
- centroid of the person [14]: it changes significantly and rapidly when the person is falling down

Another class of algorithms that needs some sort of learning phase such as Hidden Markov Models (HMM's) based approaches. Usually, they require learning phase over the data that must include sequences with falling person. One of the examples, where HMM is applied, is described in [15]: first wavelet transform of aspect ratio is computed, and then this value serves as an input for HMM-based classification. Later, this information is used to classify the motion of the person into walking or falling. Another example can be seen from [13] where authors suggest to use Layered Hidden Markov Model for motion modeling in conjunction with Fuzzy logic for posture classification.

When talking about fall detection, other different information like surroundings and personal data might be of use. For example, [10] suggests to incorporate an individual knowledge of a person (i.e., current health condition status) into fall detection procedure. [16] Shows how learned models of spatial context are used in conjunction with a tracker. The method enables inactivity outside usual zones of inactivity (e.g. chairs, beds) to be detected. Combined with body pose and motion information, this provides a cue for fall detection. Another example is [17] which presents incorporation of such personal information as height, weight and electronic health history. Based on this knowledge, the detection sensibility can vary - to reduce unnecessary alarms and improve the system's performance in detecting falls.

Some literature suggests broadening the problem to human activity recognition and conside-ring falling down a particular case. In [18] and [19] Fuzzy logic serves for modeling the human activity; and to define a fall, thus one should determine a specific set of rules. The system described in [16] generalizes an 'after fall' to 'unusual inactivity' and is defined in terms of location and posture of an actor on the scene. In [20] the authors describe a machine learning approach to activity recognition where the attributes characterizing the user's behavior and a machine learning algorithm must be selected. The attributes are the locations of body parts and the angles between adjacent body parts. After testing eight machine learning algorithms, it was stated that Support Vector Machines gives the most reliable result. In [21] authors use behavioral maps to represent the probability of moving in a particular direction; deviation from learned behavior helps to detect abnormal and atypical motions.

2.3 Target tracking

To narrow down the task, it's better to refer to what was done in the field of object tracking, because the chosen model will determine the fall detection procedure accordingly.

Object tracking is a very active research area in the field of computer vision due to availability of inexpensive cameras, constantly increasing computation power in modern computers and the need for automated video monitoring. All these factors raised a great interest in the topic and numerous algorithms and approaches were proposed.

The main task of the tracker is maintaining a consistent labeling of the tracked targets in different frames of a video. Depending on the application, the tracker can also provide another types of information about the target, like its orientation, shape, speed, etc.

Approaches to the object tracking can differ from each other by the way they consider the following issues:

- object representation
- used image features
- modeling of motion, appearance and shape of the target

Depending on the context and end use, these questions can be treated differently. A large number of tracking methods exist which attempt to define a model for the according scenario [22].

2.3.1 Object representation

An object can be anything that we are interested in recognizing and / or tracking: people, vehicles, animals, etc. These objects can be represented by different feature descriptors, for example, shape and appearance. Here, the main shape descriptors will be considered:

- Primitive geometric shapes. Object shape can be represented by a rectangle or ellipse. Though, this kind of representation is more suitable for rigid targets, it is commonly used in the literature for non-rigid objects as well. For example, [23] suggests using rectangle object representation, [24] uses an elliptic one. Example of rectangle shape descriptor is shown on the figure 4.
- Object silhouette and contour. A contour representation defines the boundary of an object and the region inside the contour is called the silhouette of the object. This descriptor is more suitable for non-rigid objects. i.e., [25] uses curve representation for describing a contour. An

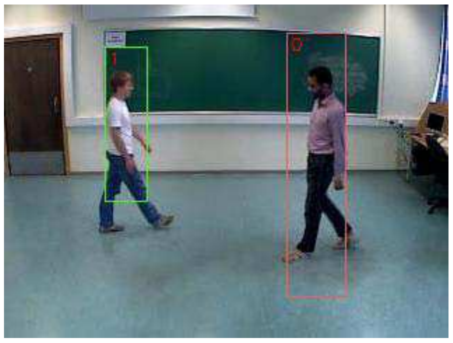


Figure 4: Rectangle can simply describe a human body location [27]

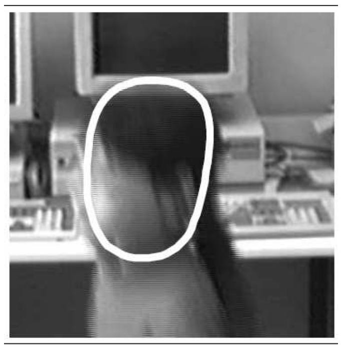


Figure 5: A curve serves for representation of a human head [25]

example is represented on figure 5.

- Articulated shape models. It helps to decompose an articulated objects into parts that are held together, i.e., when we deal with human body, we have many articulated parts: arms, legs, torso, head. The relationships between these parts are described by kinematic motion models like joint angle. [26] proposes to use three elliptic cylinders for one person representation - first cylinder plays role of head, second - torso and third one stands for legs (see figure 6)

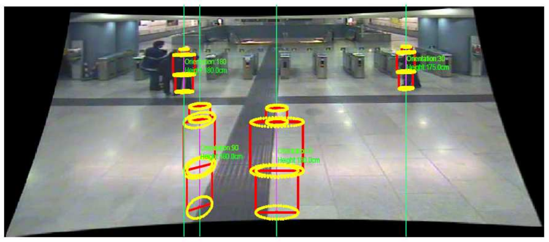


Figure 6: Human body is represented by elliptical cylinders [26]

2.3.2 Feature selection for tracking

Different features can be used for this task, but it's obvious that their selection plays a critical role for reliable tracking. The most desirable property of a visual feature is its uniqueness so that the objects can be easily distinguished from each other in the feature space. Feature selection is closely related to the object representation. For example, color is used as a feature for histogram-based appearance representations, while for contour-based representation, object edges are of use [22]. Most of the tracking algorithms do a combination of several features - they are fused so that in case if one fails, another one will provide desired output. The very commonly used visual features are:

- Color. A variety of color spaces has been used for color description of an object. On one side, an image processing gives us RGB (red-green-blue) representation; however, it is not perceptually uniform (difference between the colors in RGB does not correspond to color differences perceived by humans). On the other side, color spaces such as $L^*a^*b^*$ and $L^*u^*v^*$ are perceptually uniform and HSV (hue-saturation-value) is also approaching uniformity; but at the same time they are quite sensitive to the noise. In the literature, HSV color space can be met more frequently, i.e., in [28] color information is represented as GMM in $H-S$ space. In [29] the authors offer to use HSV color space for model representation. But still, there is no standard or obvious answer to which color space is better to use.
- Edges. Numerous edge detectors exist to identify strong changes in image intensities. One important property of edges is that it is less sensitive to the illumination changes than color descriptors. Edge features are usually used with algorithms that track boundary of the objects [22].
- Optical flow. It is defines by a set of vectors that show displacements of the each pixel in a region. Using of optical flow as a motion-based feature is common technique among segmentation and tracking applications. As an example, AVITRACK project described in [30], the main purpose of which is activities recognition around a parked aircraft in an airport area. The Kanade-Lucas feature tracking algorithm [31] is used for tracking objects - this combines a local feature selection criterion with feature-based matching in adjacent frames.
- Texture. It is a property of a material which measures regularity and smoothness. As the edge descriptors, it is not so sensitive to illumination changes as color.

The choice of features is usually done manually and depends on the application. Generally, color is the most widely used feature for tracking and it is represented by calculation of color histogram. Although, as it was said before, different kinds of features can be combined to provide more reliable tracking.

2.3.3 Tracking algorithms

The goal of the target tracker is finding the trajectory of an object over the time by specifying its position in every frame of the video. At the same time it also can provide an information about the complete region in the image that is occupied by the target. When we want to detect a target and establish correspondence between the target's instances across the frames, these two actions can be done either separately or jointly. In the former class of approaches, possible

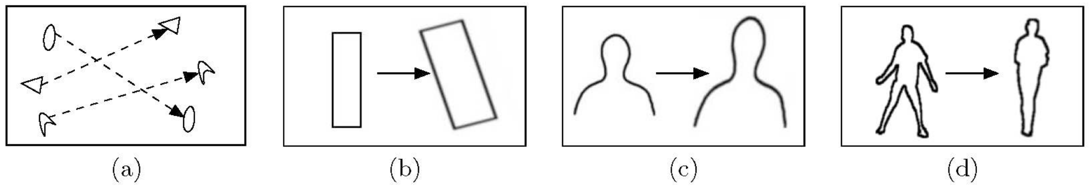


Figure 7: Tracking approaches: (a) multipoint correspondence, (b) parametric transformation of a rectangular patch, (c, d) examples of countour evolutions [22]

target regions in each frame are obtained by an target detection algorithm, and then tracker sets correspondences between targets across the frames. The latter assumes that the target region and its correspondences are jointly estimated by iteratively updating object location and region information obtained from previous frames. A brief classification of some tracking categories is given below.

Kernel tracking

Kernel can be both object shape and appearance, it can be a rectangular template or an elliptical one with its associated color histogram. Object tracking is performed by computing the motion of the kernel in the consecutive frames. The motion can be described by a parametric transformation like translation, rotation and affine transformations or optical flow computed based on two consecutive frames as illustrated in figure 7, (b).

Example of using Kernel tracking can be found in [32] - it is achieved by applying the meanshift algorithm to an image where pixel values represent likelihood of belonging to the tracked person. In [27] the tracking algorithm is performed by augmenting CAMSHIFT algorithm with motion information so that it allows tracking people in the case of occlusions.

Silhouette tracking

When it gets difficult to track complex shapes like hands, head by just using simple geometric shapes, silhouette-based methods provide much more accurate shape descriptors. The main goal of the methods is finding target region in each frame based on a target model which was generated using the previous frames. The model itself can have different forms - color histogram, edge histogram or object contour. When the model is given, the silhouettes can be tracked by either shape matching or contour evolution - see schematic examples on the represented figure 7, (c) and (d).

An example, using of silhouette-based tracking can be found in [18].

Point tracking

Here, the targets are represented by points and the method allows to associate these points across the frames based on the previous target's state which can include position and motion. The object correspondence example is shown on the figure 7 - (a). Furthermore, the point correspondence methods can be divided into two broad categories: deterministic methods and statistical methods.

The former uses qualitative motion heuristic to constrain the correspondence problem [22]. As to the latter, they take into account both object measurements and uncertainties to establish correspondence. [33] describes a non-linear / non-Gaussian tracking problem and how Bayesian approach is used for approximating the solution: different strategies are considered, including Kalman and Particle filtering.

Since the main focus of the project will be based on the statistical methods, deeper overview of the related work will be presented here. Considering the problem of the tracking, we should keep in mind that most of the models encountered in visual tracking are nonlinear, non-Gaussian, multi-modal or any combination of these [34]. The object motion can easily undergo random perturbations and especially when the object moves in maneuvering way. Therefore, our main focus is given to the overview of Monte-Carlo methods (also known as particle filter tracking) and its use for object tracking.

Particle filtering, that is also known as the Condensation algorithm [25] or bootstrap filter [35], or Sequential Monte-Carlo algorithm [24] as proven to be a strong and reliable method for estimating the states of nonlinear and non-Gaussian systems [33]. Particle filtering techniques are also commonly used because of their quality to be able to fuse different sensor data, to take into account different uncertainties, to withstand with data association problem for not only multiple targets, but also for multi sensors. Their main idea is to keep track based on sample-based representation of probability density functions [34]. All the samples make a contribution to the state evaluation, and the state evolution is described by a system state dynamics. The relationship between measurements and states are given by observational dynamics.

A modeling of those two probabilities plays critical role in system performance. Below a brief overview is given how they can be modeled:

1. Observation probability.
   Reliability of the particle filter is very dependent on the chosen method for obtaining object measurements. Some of the papers rely on a single cue that can be chosen accordingly to the application context, for example, [29] integrates the color distribution into particle filtering; and later this distribution can be updated to account for the illumination or target's localization changes. However, using only color cue can fail, if, for example, the region is cluttered; so using of multiple cues can help to avoid this situation. In [34] authors present a consistent histogram-based framework for analysis of color, edge and texture cues. [36] uses edge-based likelihood function as the only cue, but at the same time second-order dynamical model is applied for better prediction step. Color histogram cue combined with a shape metric was proposed in [37], called Spatially Weighted Oriented Hausdorff Measure.

2. State dynamics.
   It was found that better importance density function may make particles more efficient [38]. Conventional particle filtering offers quite simple model - random sampling [25]. To provide better performance, [39] includes motion-based proposal; while [40] introduces boosted particle filter by constructing the proposal distribution using an information from the dynamic models of each target and the detection hypotheses generated by Adaboost. In [41] the authors propose to use unscented particle filter that integrates current observation to generate sophisticated proposal distribution.

Various techniques had been proposed to extend particle filters to multi-object tracking (MOT).

The most common approach that can be found in the literature is by exploiting a join state space representation (joint probability data association - JPDA), the main idea of which is linking all the objects' states to form a large meta data [24]. The joint data association problem is inferred from characterization of possible associations between objects and observations [42]. The obvious disadvantage of such approach is tremendous computational cost due to the high dimensionality of the joint space state.

More recent works tried to relax this method by applying different models and approximations. For example, [24] uses magnetic-inertia potential model to handle the 'error merge' and 'labeling' problems. Specifically, they propose to model the interactions by a 'gravitation' and 'magnetic' repulsion schemes, that is representing the cumulative effect of virtual physical forces that objects undergo while interacting with others. Another work [43] proposes to use a 'Data Driven Markov Chain Monte-Carlo' approach to sample the solution space efficiently. The sampling is driven by an informed proposal scheme controlled by a joint probability model combining motion and appearance. In [44] the data association posterior is represented in an information form where the 'proximity' of objects and tracks are expressed by numerical links. Updating these links requires linear time, compared to exponential time required for computing the exact posterior probabilities. In [45] a full polynomial randomized approximation scheme for JPDA is proposed that by presenting a single-scan version of Markov chain Monte Carlo data association. The proposed approach in [46] integrates Markov Random Field (MRF) motion prior to the model interactions and substitutes traditional importance sampling step by Markov Chain Monte-Carlo (MCMC) sampling step that allows to get rid from exponential complexity in the number of tracked targets.

2.4 Multi-camera systems and data fusion

When considering the tracking of a number of people in indoor environment, multiple-camera system will be of greater help. In this case multiple-camera tracking (MCT) not only serves for maintaining consistency of identity labels in the different camera views, but also allows to expand the Field of View (FOV). Therefore, the main baseline will consist of:

1. matching objects in consecutive frames in single camera (tracking itself), and
2. matching objects across fields of view of the multiple cameras so that to maintain identities when a moving object moves from one field of view of one camera to another (correspondence between views)

Multi-camera systems are used for interpreting the dynamics of objects moving in wide areas or for observing objects from different viewpoints to achieve 3D interpretation [47]. For wide-area monitoring. Cameras with non-overlapping or partially overlapping fields of view are used; cameras with overlapping fields of view are used to disambiguate occlusions. In both cases, determination of the relative location and orientation of the cameras is often essential for effective use of a multi-camera system.

For setting correspondence between camera views, methods can be generally divided into three categories [48]:

1. geometry-based

2. color-based
3. hybrid approaches

The first class can include both calibrated and uncalibrated methods. For calibrated approaches most of the literature uses a popular method called homography. Where a matrix that describes a transformation between any two images of the same planar surface in a space. The most commonly-used strategy for MCT [49]:

- detect people in each camera
- calculate the correspondence between cameras (homography)
- provide a feedback from ground-plane tracker to each camera tracker

In [32] authors use homography for restoring 3D world coordinates of each target combined with CAMSHIFT tracking algorithm for each single camera. Another example is work done in [28] where homography is used only when occlusion occurs to check the location of targets in world coordinates and particle filter is used for tracking of each target individually.

An example of uncalibrated approach can be seen from [50] where computation of so-called 'Edges of FOV' (the lines delimiting the FOV of each camera) is performed, and, thus, defining the overlapped regions. It is done through a learning procedure where a single target moves from one view to another, and an automatic procedure computes those edges that are later applied to keep consistent labels on the objects when they pass from one camera to adjacent.

Regarding color-based approaches, the matching is essentially based on the color information of the tracks as in [51], where appearance features for matching between views are calculated by reducing color texture information and, therefore, getting centroids of homogeneously colored body segments.

Hybrid approaches mix information from calibration and geometry with visual appearance. Like [52] uses Sequential Belief Propagation as a tool for information fusion and particle filter is used for tracking.

3 Proposed Framework

3.1 System overview

The block diagram of the proposed framework is shown on figure 8. The process it describes can be split into four main steps: Initialization, Sequential Monte-Carlo tracking, data association and fusion, and fall detection. Each of these steps will be explained further in the following sections.

3.2 Multi-target tracking

The chosen models and algorithms are described in this section. Since Bayesian filtering was chosen as a basic skeleton of the tracking system, its theory will be first reviewed, also its application in tracking. Thr main notations will be presented and a Bayesian framework for multi-target multi-camera tracking will be proposed.

3.2.1 Bayesian sequential estimation

When we think about tracking process, a target movements can be considered as a state transition process, where one state is represented by a vector that can include [53]:

- coordinates of the target
- its scale
- velocity
- shape or any other parameters that are able to depict characteristics of the target.

Usually, the *state variables* are denoted as $x_t \in \mathbb{R}^{n_x}$, $t \in \mathbb{N}$ defines the current time step and n_x is a dimension of the state variable. For our application the main task is to estimate the current state of the target through given output data (named observations). That is why state variables are also called 'hidden' variables, because they are not directly available, but must be induced from the observable variables.

The *observations* of the hidden variables are usually denoted as $z_t \in \mathbb{R}^{n_z}$, where n_z is the corresponding dimension for observed variable. For most of applications it's not necessary that $x_t \in \mathbb{R}^{n_x}$ and $z_t \in \mathbb{R}^{n_z}$ are in the same space, therefore, $n_x \neq n_z$ usually.

The conventional tool that helps to represent a relationship between the two variables is graphical model [54] as is shown on the figure 9, where hidden state is denoted by a rectangle and observable one - by a circle. The direct link from a target's state to its observation explains that the latter only depends on the former; and the edge characterizes the *local observation likelihood* and is denoted as $p(z_t|x_t)$.

The Hidden Markov Model (HMM) serves quite well for describing the sequential data (in our case - tracking data). It makes links not only between states and observations, but also between consecutive time steps (consecutive frames); this can be represented by a node structure,

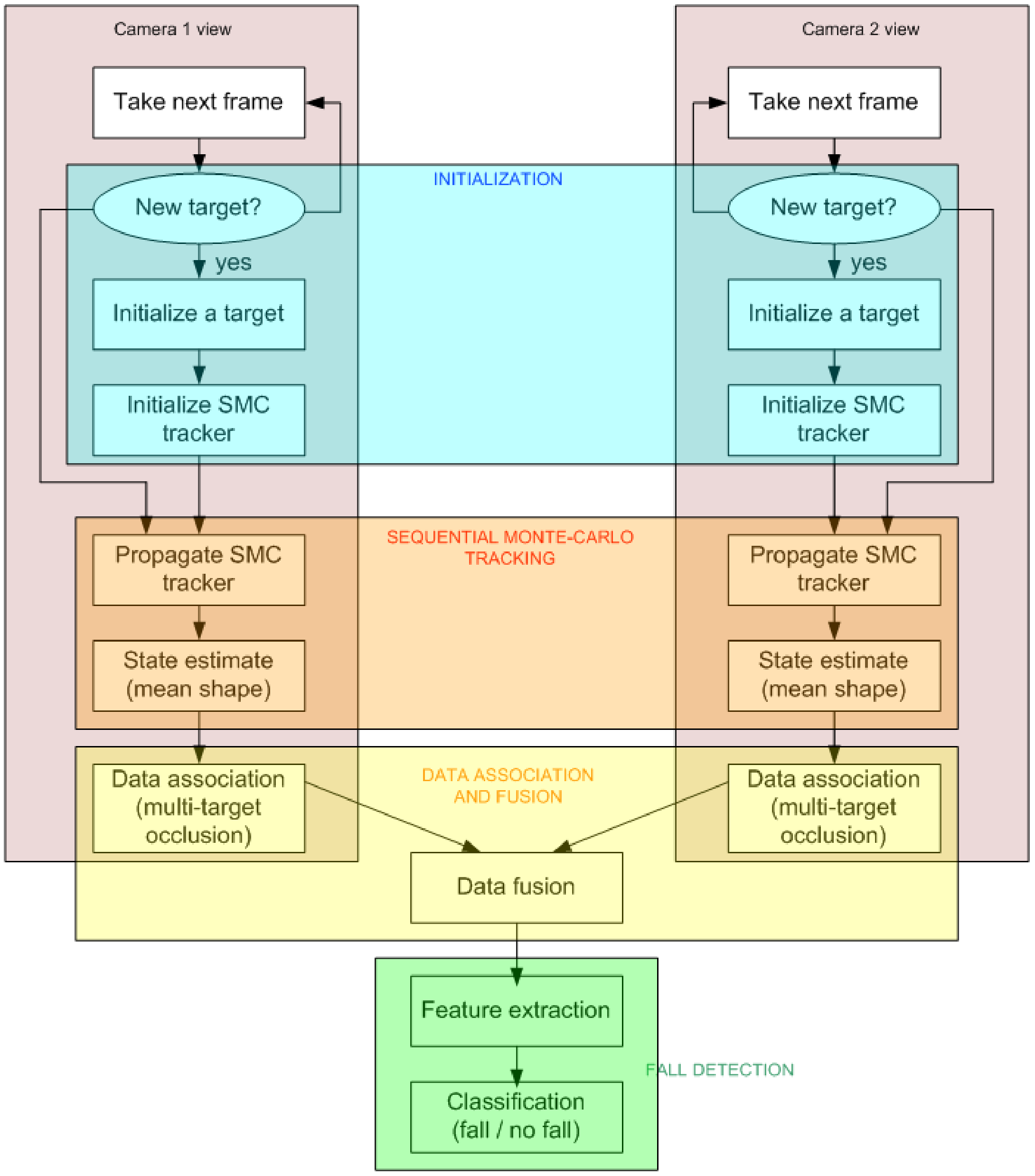


Figure 8: System block-diagram (SMC stands for Sequential Monte-Carlo).

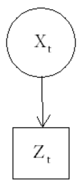


Figure 9: Graphical model representation of the relationship between an observable variable z_t and hidden one x_t.

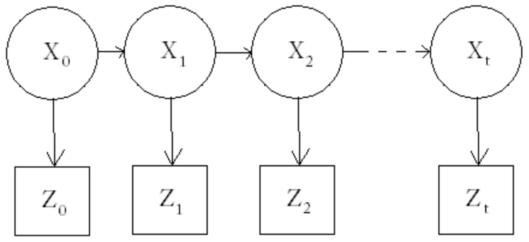


Figure 10: Graphical model representation of the HMM.

example of which is shown on the figure 10. The subscript 0 stands for defining the initial state of the system, the direct link between consecutive states represents a *state dynamics*, or transition probability which is described by $M \times M$ matrix for M possible values and denoted as $p(x_t|x_{t-1})$.

The Bayesian sequential estimation helps to estimate a *posterior distribution* noted as $p(x_{0:t}|z_{1:t})$ or its marginal $p(x_t|z_{1:t})$, where $x_{0:t}$ includes a set of all states up to time t and $z_{1:t}$ - set of all the observations up to time t accordingly. The evolution of the state sequences $\{x_t, t \in \mathbb{N}\}$ of a target is given by equation 3.1; and the observation model is given by equation 3.2:

$$x_t = f_t(x_{t-1}, v_{t-1}), \tag{3.1}$$

$$z_t = h_t(x_t, u_t), \tag{3.2}$$

where $f_t : \mathbb{R}^{n_x} \times \mathbb{R}^{n_v} \longrightarrow \mathbb{R}^{n_x}$ and $h_t : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \longrightarrow \mathbb{R}^{n_z}$ can be both linear or nonlinear, and v_t and u_t are sequences of i.d.d. process noise and measurement noise respectively; at the same time n_v and n_u are their dimensions. So, the main goal is to find a filtered estimate of x_t based on the set of all available measurements $z_{1:t} = \{z_i, i = 1, ..., t\}$ up to time t.

In Bayesian context, the tracking problem can be considered as recursive calculation of some belief degree in the state x_t at time step t, given the data observation $z_{1:t}$. That is, we need to construct a pdf $p(x_t|z_{1:t})$. It is assumed that initial state

$$p(x_0|z_0) \equiv p(x_0) \tag{3.3}$$

of the system (also called prior) is given. Then, the posterior distribution $p(x_t|z_{1:t})$ will be calculated by following next two steps given in equations 3.4 and 3.5 (we can apply the recursion only based on the Markov assumptions [1]):

1. prediction involves using the system model defined in 3.1 to get the prior pdf of the state at time t:
$$p(x_t|z_{1:t-1}) = \int p(x_t|x_{t-1})p(x_{t-1}|z_{1:t-1})dx_{t-1}, \tag{3.4}$$
where the recursion is initialized by the prior distribution in equation 3.3 and dynamic model described in equation 3.1;

2. update is performed after measurement z_t becomes available; later it will be used to update the prior through Bayes' rule:
$$p(x_t|z_{1:t}) = \frac{p(z_t|x_t)p(x_t|z_{1:t-1})}{p(z_t|z_{1:t-1})} = \frac{p(z_t|x_t)p(x_t|z_{1:t-1})}{\int p(z_t|x_t)p(x_t|z_{1:t-1})dx_t}, \tag{3.5}$$
where the denominator is normalization constant and it depends on the likelihood function $p(z_t|x_t)$ - as described by equation 3.2.

In this case, the optimal Bayesian solution will be formed from the two mentioned relations in equations 3.4 and 3.5, however the presented recursive propagation is just a conceptual solution and can not be applied in practice. Because of this, several approximate solutions were proposed, to help obtain optimal Bayesian solution. The section 3.2.3 will give appropriate explanations of the sequential Monte-Carlo implementation.

3.2.2 Bayesian framework for multiple target tracking

The idea to build such a framework was adopted [1]. Here the main concept will be presented and later in section 3.2.4 each of the system elements will be expanded and discusses.

For multiple targets, one tracker per target was used in each camera. The main notations used in the algorithm are as follows:

$x_t^{A,i}$ - one state in camera A at the time t of the ith target;
$z_t^{A,i}$ - image observation of the state $x_t^{A,i}$;
$x_{0:t}^{A,i}$ - set of all states up to time t with $x_0^{A,i}$ - initialization prior;
$z_{1:t}^{A,i}$ - set of all observations up to time t.

Similarly, notations for the corresponding targets in a camera B will be denoted:

$x_t^{B,i}$ - 'counterpart' of $x_t^{A,i}$ in the camera B;
z_t^{A,J_t^i} - to denote all the neighboring observations of $z_t^{A,i}$ which are in the interaction with $z_t^{A,i}$ at time t;
$J_t^i = \{j_{l_1}, j_{l_2}, ...\}$ - is set of indexes of the targets whose observations are interacting with $z_t^{A,i}$ for the current moment t;
$z_{1:t}^{A,J_{1:t}^i}$ - collection of neighboring observation sets up to time t.

Graphical modeling

As it was said before, we can use graphical model to represent and analyze complex dynamic systems. Such a model of two consecutive frames for multi-targets and for two cameras is shown

[1]First order Markov chain is defined by independency assumptions: $x_t \perp z_{0:t-1}|x_{t-1}$ and $z_t \perp z_{0:t-1}|x_t$, which mean that the current state is independent from the past observations given the previous state; and the current observation depends only on the current hidden state.

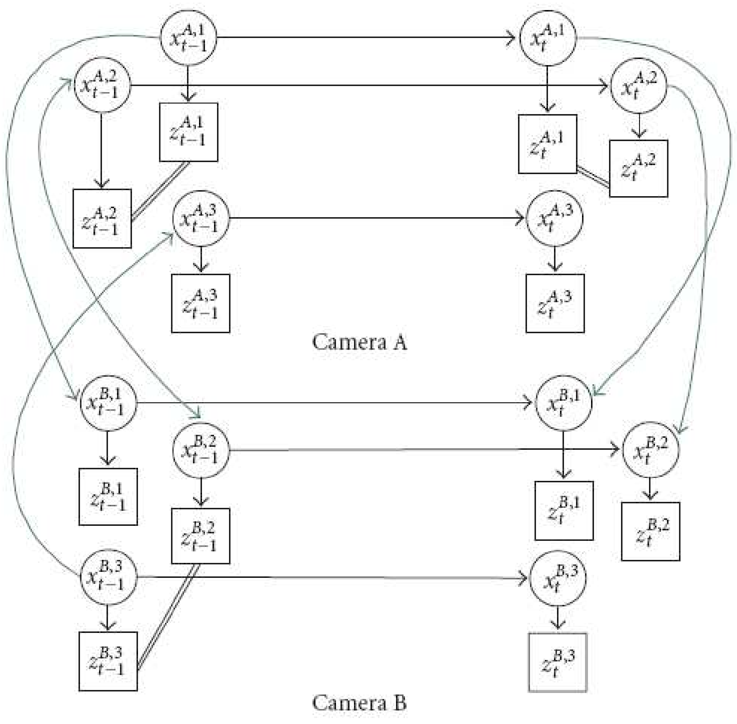


Figure 11: The dynamic graphical model for multi-person multi-camera tracking [1].

on the figure 11. Similarly to figure 9, circled notes represent hidden layer and rectangles stand for observable layer; link between consecutive states of the same target in one camera is state dynamics $p(x_t|x_{t-1})$; direct link between targets' states and their observations denote local observation likelihood $p(z_t|x_t)$; the indirect (double line) link in each camera between neighboring observations means that they are interacting at the current moment of time (determined by spatial locations of the observations); the direct curve link between the corresponding states in different camera views represents a camera collaboration. According to [1], this collaboration is activated between any possible collection of cameras only for targets which need help to improve their tracking robustness - when, for example, they are occluding each other or are in close proximity in a camera view. The direction of the link shows how this help is organized - which target's data is used for improving other targets' data. The scheme is supposed to be activated only when an occlusion issues occur, therefore, a lot of computational costs would be saved.

Bayesian tracking for one target

This subsection will present a general Bayesian conditional density propagation framework for each of the targets in one camera. The main goal is to provide a generic statistical framework to model interactions among targets and, possibly, cameras. For one target, one tracker will be used; also we want to take into account observations coming from both the target and its neighborhood for the current camera view, and also data gotten from other camera views - that is the desirable posterior $p(x_{0:t}^{A,i}|z_{1:t}^{A,i}, z_{1:t}^{A,J_{1:t}^{i}}, z_{1:t}^{B,i})$ for one tracker and for one camera. Based on equations 3.4, 3.5 and Markov properties rules derived in [1], a recursive conditional density updating rule for multi-person multi-camera environment [1]:

$$\begin{aligned} p(x_{0:t}^{A,i}|z_{1:t}^{A,i}, z_{1:t}^{A,J_{1:t}^{i}}, z_{1:t}^{B,i}) = k_t \; & p(z_t^{A,i}|x_t^{A,i}) \; p(x_t^{A,i}|x_{0:t-1}^{A,i}) \\ & \times p(z_t^{A,J_t^{i}}|x_t^{A,i}, z_t^{A,i}) \; p(z_t^{B,i}|x_t^{A,i}) \\ & \times p(x_{0:t-1}^{A,i}|z_{1:t-1}^{A,i}, z_{1:t-1}^{A,J_{1:t-1}^{i}}, z_{1:t-1}^{B,i}), \end{aligned} \tag{3.6}$$

where

$$k_t = \frac{1}{p(z_t^{A,i}, z_t^{A,J_t^{i}}, z_t^{B,i}|p(z_{1:t-1}^{A,i}, z_{1:t-1}^{A,J_{1:t-1}^{i}}, z_{1:t-1}^{B,i})} \tag{3.7}$$

is a normalization constant. In equation 3.6 we have four main probabilities:

- $p(z_t^{A,i}|x_t^{A,i})$ is local observation likelihood for target i in a camera A
- $p(x_t^{A,i}|x_{0:t-1}^{A,i})$ stands for state dynamics
- $p(z_t^{A,J_t^{i}}|x_t^{A,i}, z_t^{A,i})$ defines a target interaction function for one camera view
- $p(z_t^{B,i}|x_t^{A,i})$ is for defining a collaboration function between the same target states in different camera views

Of course, it is of key importance to know how to model each of the probabilities. Though, we follow the same framework as in [1], probability models will be different for our project. Section 3.2.4 presents the local likelihood model; section 3.2.4 gives more details about state dynamics probability; target interaction function is discussed in section 3.2.4; and since we can not model the camera collaboration in the same way as in [1] (discussed in section 3.2.4), data fusion procedure is suggested instead in section 3.3.

3.2.3 Sequential Monte-Carlo implementataion

Sequential Monte-Carlo (SMC) methods are a general class of techniques which provides weighted samples from a sequence of distributions using sampling and re-sampling mechanisms [55]. Among different applications of SMC are: terrain navigation, mobile robot localization, neural networks, Bayesian networks, jump diffusions, stochastic volatility, maneuvering target tracking, etc [56]. In this project we employ a particle set to represent the posterior of the Bayesian formulation. The following subsections contain the basic theory concerning SMC. For the sake of simplicity it is provided for single target without considering neighboring targets or camera views information.

Monte-Carlo simulation

The main idea of Monte-Carlo sampling method is to represent the posterior distribution in a form of a set of random samples with appropriate weights. It means that our posterior described by equation 3.5 will be approximated as in [33]:

$$p(x_t|z_{0:t}) \approx \frac{1}{N_s} \sum_{n=1}^{N_s} \delta(x_t - x_t^{(n)}), \tag{3.8}$$

where N_s is the number of samples, $\{x_t^{(n)}, \omega_t^{(n)}\}_{n=1}^{N_s}$ is a particle set drawn from the posterior distribution and their corresponding weights. Initially, the weights of all particles are equal to $\frac{1}{N_s}$ to provide normalized weights. $\delta(\cdot)$ stands for Dirac delta function

$$\delta(x) = \begin{cases} 1 & \text{if } x = 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.9}$$

Therefore, any integral can be approximated by a discreet form using Monte-Carlo simulation:

$$\mathbb{E}(f(x_t)) = \int f(x_t)p(x_t|z_{0:t})dx_t \approx \frac{1}{N_s}\sum_{n=1}^{N_s} f(x_t^{(n)}). \tag{3.10}$$

From this formulation we can see that when the number of samples approaches infinity, the approximation will be approaching the true distribution. Similarly, any query estimation that is sampled and weighted will be close to the optimal Bayesian sequential estimate.

Sequential importance sampling

Usually, it is not possible to sample directly from the proposal distribution, so the particles are sampled from so-called Importance density $q(x_t|z_{1:t})$ instead. In this case, equation 3.8 will be transformed into

$$p(x_t|z_{0:t}) \approx \sum_{n=1}^{N_s} \omega_t^{(n)}\delta(x_t - x_t^{(n)}), \tag{3.11}$$

where

$$\omega_t^{(n)} \propto \frac{p(x_t^{(n)}|z_{0:t})}{q(x_t^{(n)}|z_{0:t})}, \tag{3.12}$$

and

$$\sum_{n=1}^{N_s} \omega_t^{(n)} = 1. \tag{3.13}$$

This technique is called "Importance Sampling" and corresponding weights are "Importance Weights". From equation 3.12 it can be seen that if the proposal function is the same as posterior, then weights will look the same way as in equation 3.8.

Using importance sampling technique, equations 3.4 and 3.5 for optimal Bayesian solution can be re-written in the discreet form:

1. predict:

$$p(x_t|z_{0:t-1}) \approx \sum_{n=1}^{N_s} p(x_t|x_{t-1}^{(n)})\omega_{t-1}^{(n)}, \tag{3.14}$$

2. update:

$$p(x_t|z_{0:t}) \approx \sum_{n=1}^{N_s} \omega_t^{(n)}\delta(x_t - x_t^{(n)}), \tag{3.15}$$

where

$$\omega_t^{(n)} \propto \frac{p(z_t|x_t^{(n)})p(x_t^{(n)}|x_{t-1}^{(n)})}{q(x_t^{(n)}|x_{t-1}^{(n)}, z_{0:t})}\omega_{t-1}^{(n)}. \tag{3.16}$$

$[\{x_t^{(n)}, \omega_t^{(n)}\}_{n=1}^{N_s}] = \text{SIS}[\{x_{t-1}^{(n)}, \omega_{t-1}^{(n)}\}_{n=1}^{N_s}, z_t]$
• FOR $n = 1 : N_s$ - Draw $x_t^{(n)} \sim q(x_t \| x_{t-1}^{(n)}, z_t)$ - Assign the particle a weight, $\omega_t^{(n)}$, according to equation 3.16 • END FOR

Table 1: SIS particle filter algorithm [33].

Normalization constant $\int p(z_t|x_t)p(x_t|z_{0:t-1}dx_t)$ was not shown here just for convenience. The evolution of the states' estimation happens by sequentially updating the weights for each supporting particle; in the literature this method has a conventional name - Sequential Importance Sampling (SIS). A pseudo-code description of this algorithm is provided in table 1.

SIS for the Bayesian framework

Based on [1], an obtained Bayesian formulation can now be written using SIS method for the Bayesian framework (which was presented in section 3.2.2). Based on the derived sequential iteration in equation 3.6 for the framework, SIS approximations in equations 3.14, 3.15 and 3.16; and having the representation posterior as a particle set

$$p(x_{0:t}^{A,i}|z_{1:t}^{A,i}, z_{1:t}^{A,J_{1:t}^{i}}, z_{1:t}^{B,i}) \sim \{x_{0:t}^{A,i,(n)}, \omega_t^{A,i,(n)}\}_{n=1}^{N_s}, \tag{3.17}$$

where $\{x_{0:t}^{A,i,(n)}, n = 1, \cdots, N_s\}$ are samples with the associated weights $\{\omega_t^{A,i,(n)}, n = 1, \cdots N_s\}$, and if the particles $x_{0:t}^{A,i,(n)}$ are sampled from the importance density

$$q(x_t^{A,i}|x_{0:t-1}^{A,i,(n)}, z_{1:t}^{A,i}, z_{1:t}^{A,J_{1:t}^{i}}, z_{1:t}^{B,i}) = p(x_t^{A,i}|x_{0:t}^{A,i,(n)}), \tag{3.18}$$

then corresponding weights are defined by

$$\omega_t^{i,(n)} \propto \omega_{t-1}^{i,(n)}\ p(z_t^{A,i}|x_t^{A,i,(n)})\ p(z_t^{A,J_t^i}|x_t^{A,i,(n)}, z_t^{A,i})\ p(z_t^{B,i}|x_t^{A,i,(n)}). \tag{3.19}$$

Re-sampling

Degeneracy phenomenon is one of the common problems with SIS, which means that after several interactions most of the particles will have very low weights due to the very far locations from the target observation, and, therefore, most of the computational effort is wasted on those particles with negligible weights. It was proven that it happens because the variance of the importance weights increase over time [33], so it is impossible to avoid the degeneracy phenomenon without treating it. Therefore, the main idea is to update the particles whose contribution to the approximation to $p(x_t|z_{1:t})$ is almost zero and concentrate an attention on the more promising particles . The equation 3.20 helps to define the threshold of generacy [33]:

$$N_{eff} = \frac{N_s}{1 + Var(\omega_t^{*(n)})}, \tag{3.20}$$

where $\omega_t^{*(n)} = \frac{p(x_t^{(n)}|z_{1:t})}{q(x_t^{(n)}|x_{t-1}^{(n)},z_t)}$ stands for the 'true' weight. It can not be calculated exactly, so another estimate is obtained $\widehat{N_{eff}}$ instead of N_{eff}:

$$\widehat{N_{eff}} = \frac{1}{\sum\limits_{n=1}^{N_s}(\omega_t^{(n)})^2}, \tag{3.21}$$

where $\omega_t^{(n)}$ is the normalized weight from equation 3.16. N_{eff} is always smaller or equal to N_s, and small value of N_{eff} indicates a severe degeneracy. Of course, it's always better try to avoid degeneracy phenomenon, and, theoretically, it can be achieved by increasing the number of samples (works like brute force approach), but it can not be realized in practice due to high computational costs. Therefore, other tricks can be applied: for example, using a re-sampling scheme.

The basic idea of the re-sampling is generation of a new set $\{\tilde{x}_t^{(n)}, N_s^{-1}\}_{n=1}^{N_s}$ by re-sampling with replacement N_s times from an approximate discreet representation of $p(x_t|z_{0:t})$, that is given by equation 3.11, so that $Pr(\tilde{x}_t^{(n)} = x_t^{(m)}) = \omega_t^{(m)}$. Since the resulting sample will be an i.d.d. sample from the density defined by equation 3.11, it means that weights must be reset to $\omega_t^{(n)} = \frac{1}{N_s}$ (we denote it N_s^{-1}). As a result we will have more replicates of particles that had higher weights before, and most of the particles with lower weights will be eliminated.

The whole process of the Bayesian estimation (or particle filtering) including re-sampling is shown on figure 12. The SIR process starts at $t-1$ with a set of equally weighted particles $\{\tilde{x}_{t-1}^{(n)}, N^{-1}\}_{n=1}^{N}$, which represent the distribution $p(x_{t-1}|z_{0:t-2})$. Importance weights are updated by using measurements obtained at time $t-1$ to represent $p(x_{t-1}|z_{0:t-1})$ with $\{\tilde{x}_{t-1}^{(n)}, \tilde{\omega}_{t-1}^{(n)}\}_{n=1}^{N}$. Re-sampling is performed to make it an equally weighted particle set $\{x_{t-1}^{(n)}, N^{-1}\}_{n=1}^{N}$: particles with lower weights are re-sampled around the particles with higher weights. Finally, particles are propagated to $\{\tilde{x}_t^{(n)}, N^{-1}\}_{n=1}^{N}$ at time t by the proposal distribution to represent $p(x_t|y_{0:t-1})$ and iterations repeat forward [41].

There are several types of re-sampling scheme implementations exist in the literature, and for this project we decided to apply a systematic one [57]: it takes $O(N_s)$ time and minimizes MC variation [33]; although, this parameter can easily be changed inside the program, to test other techniques. A pseudo-code of systematic re-sampling is provided in table 2, where $\mathbb{U}[a,b]$ is the Uniform distribution on the interval $[a,b]$. There, for each re-sampled particle $\tilde{x}_t^{(m)}$, the information about the index of its its parent, denoted by $n_{(m)}$, is also stored. Then, a pseudo-code of generic particle filtering is described by table 3.

State estimate

Until this point, there was no explicit state estimation maintained - whatever we have, it is just a cloud of particles that represents the posterior probability distribution of optimal Bayesian solution. At every time step, it's necessary to know a current estimate since it will give us all the information about target location, scale, shape, etc. The mean state ('mean' particle, 'mean' shape or weighted sum of particles) is commonly used for this purpose

$$\varepsilon[X_t] = \sum_{n=1}^{N_s} \omega_t^{(n)} x_t^{(n)} \tag{3.22}$$

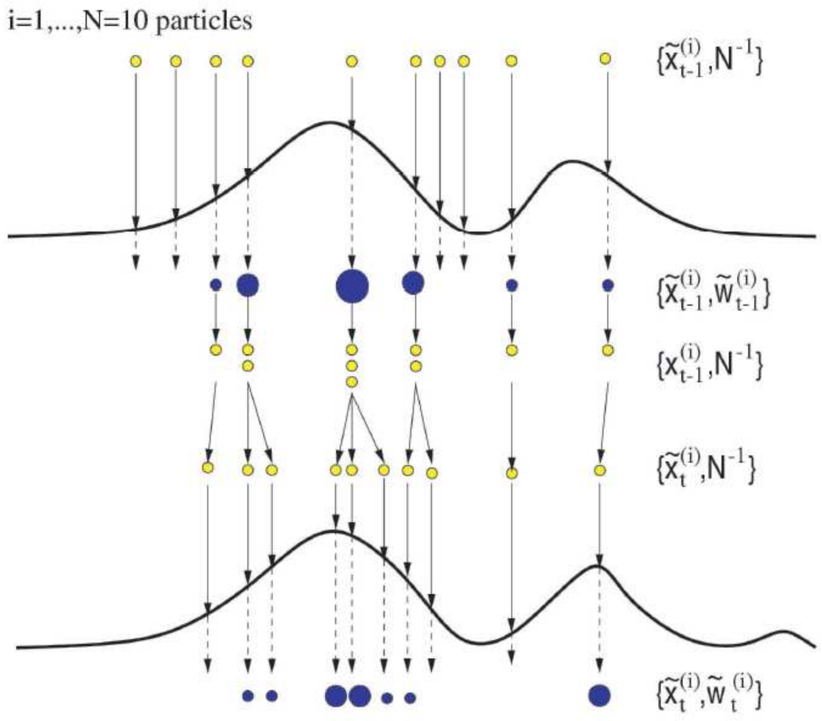

i=1,...,N=10 particles
$\{\tilde{x}_{t-1}^{(i)}, N^{-1}\}$
$\{\tilde{x}_{t-1}^{(i)}, \tilde{w}_{t-1}^{(i)}\}$
$\{x_{t-1}^{(i)}, N^{-1}\}$
$\{\tilde{x}_{t}^{(i)}, N^{-1}\}$
$\{\tilde{x}_{t}^{(i)}, \tilde{w}_{t}^{(i)}\}$

Figure 12: Graphical illustration of particle filtering including re-sampling step

```
$[\{\tilde{x}_t^{(m)}, \omega_t^{(m)}, n_m\}_{n=1}^{N_s}] = \text{RESAMPLE}[\{x_t^{(n)}, \omega_t^{(n)}\}_{n=1}^{N_s}]$

• Initialize the CDF: $c_1 = 0$
• FOR $n = 2 : N_s$
    - Construct CDF: $c_{(n)} = c_{(n-1)} + \omega_t^{(n)}$
• END FOR
• Start at the bottom of CDF: $n = 1$
• Draw a starting point: $u_1 \sim \mathbb{U}[0, N_s^{-1}]$
• FOR $m = 1 : N_s$
    - Move along the CDF: $u_{(m)} = u_1 + N_s^{-1}(m-1)$
    - WHILE $u_{(m)} > c_{(n)}$
          $n = n + 1$
    • END WHILE
    • Assign sample: $\tilde{x}_t^{(m)} = x_t^{(n)}$
    • Assign weight: $\omega_t^{(m)} = N_s^{-1}$
    • Assign parent: $i^{(m)} = (n)$
• END FOR
```

Table 2: Resampling algorithm [33], CDF stands for cumulative density function.

```
$[\{x_t^{(n)}, \omega_t^{(n)}\}_{n=1}^{N_s}] = \mathrm{PF}[\{x_{t-1}^{(n)}, \omega_{t-1}^{(n)}\}_{n=1}^{N_s}, z_t]$

• FOR $n = 1 : N_s$
    - Draw $x_t^{(n)} \sim q(x_t | x_{t-1}^{(n)}, z_t)$
    - Assign the particle a weight, $\omega_t^{(n)}$, according to formula 3.16
• END FOR
• Calculate total weight $k = \mathrm{SUM}[\{\omega_t^{(i)}\}_{n=1}^{N_s}]$
• FOR $n = 1 : N_s$
    - Normalize: $\omega_t^{(n)} = k^{-1}\omega_t^{(n)}$
• END FOR
• Calculate $\widehat{N_{eff}}$ using equation 3.21
• IF $\widehat{N_{eff}} < N_{thresh}$
    • Re-sample using algorithm table 2: $[\{x_t^{(n)}, \omega_t^{(n)}, -\}_{n=1}^{N_s}] = \mathrm{RESAMPLE}[\{x_t^{(n)}, \omega_t^{(n)}\}_{n=1}^{N_s}]$
• END IF
```

Table 3: Generic particle filter algorithm [33].

Figure 13 shows an illustration of sample-set representation and its mean state estimation.

3.2.4 Modeling of densities

Modeling the densities in equations 3.19 and 3.18 is not an easy task, but at the same time plays critical role in the system performance. Since we do not use the same models as the reference [1], the following subsections will cover the construction of each of the densities.

Target representation

First, it is necessary to define what the state vector will be. As it was shown before, a target can have different representation - from 3D object models [26] to dynamic contour model [25]. Though, the targets have non-rigid nature in the project (moving people), it was decided to use simple 5D parametric ellipse model in order to decrease computational costs, and at the same time, it can be sufficient to represent the tracking results. Figure 14 illustrates state space representation for our system.
Therefore, one state $x_t^{A,i}$ is defined by

$$x_t^{A,i} = (cx_t^{A,i}, cy_t^{A,y}, a_t^{A,i}, b_t^{A,i}, \rho_t^{A,i}), \tag{3.23}$$

where i is target index, t - current time, (cx, xy) - coordinates of the center of ellipse, (a, b) - major and minor axises, and ρ - orientation angle.

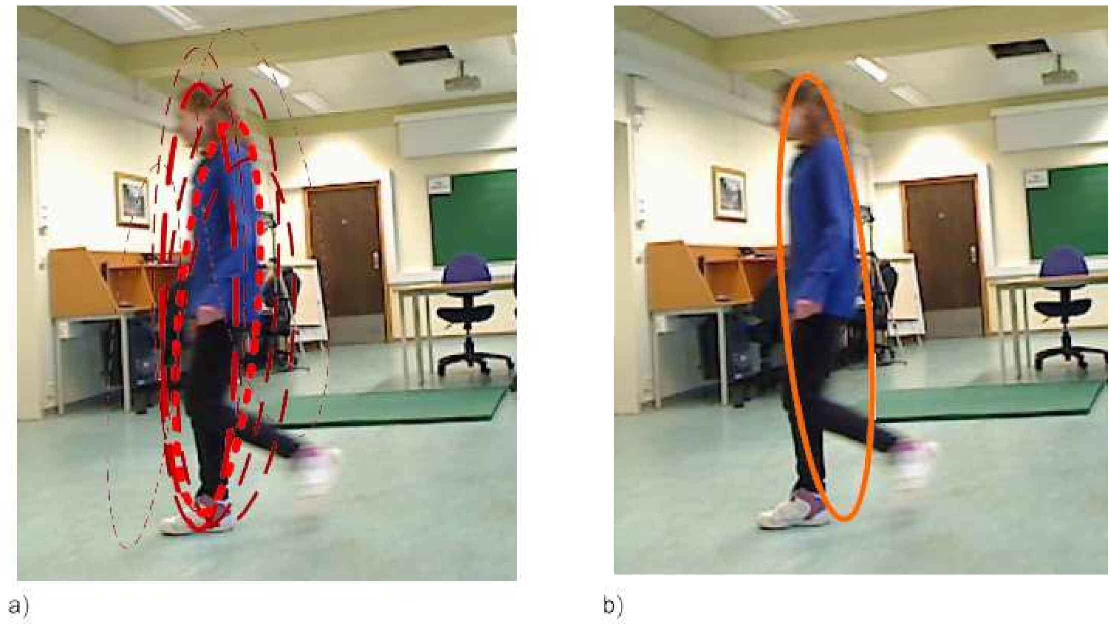


Figure 13: (a) illustrates sample-set representation: each particle is represented by an ellipse $x_t^{(n)}$ with varying position, scale and orientation; the thickness and brightness correspond to the weight $\omega_t^{(n)}$ (larger weights correspond to thicker and brighter lines). (b) shows the final state estimate, that is, mean of the sample set.

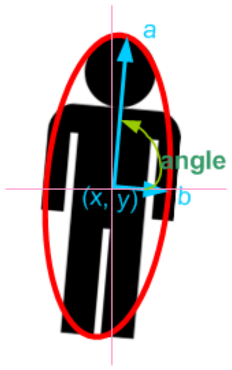


Figure 14: Ellipse representation for describing a human body.

Local observation model

As a local likelihood $p(z_t^{A,i}|x_t^{A,i,(n)})$ it was decided to use single cue - color histogram like [29], but without updating procedure (for simplicity); since the test video sequences are quite short and even if color distribution changes over time due to illumination, it is not very critical at this time. Using different cues (i.e., combined with PCA-appearance based model [58]), of course, could give more robust results, but it was not employed for the current moment leaving it for future extensions.

State dynamics model

Since for local observation likelihood we used single color histogram model, the prediction step must be improved by the use of an importance density function. Usually, for conventional particle filter, simple transition model is used, therefore, many particles can be wasted in low-likelihood region. For the project, it was decided to include motion information and use it to predict the next step.

Motion-based proposal [39] adds next two benefits to the system performance:

- efficient allocation of the samples,
- adaptivity to sudden changes in motion directions.

As a motion estimation algorithm we chose Lucas-Kanade (LK) optical flow algorithm [59], with implementation available in OpenCV [60]. It allows to calculate coordinates of the feature points on the current video frame given their coordinates in the previous frame. Later, the mean of optical flow values will be used for better prediction of the center (cx_{t+1}, cy_{t+1}).
If we consider the 5D ellipse representation, a motion vector will be defined by:

$$\Delta V_t^{A,i} = (cx_t^{A,i} - cx_{t-1}^{A,i}, cy_t^{A,i} - cy_{t-1}^{A,i}, 0, 0, 0). \tag{3.24}$$

If we model the computational errors by a zero-mean white Gaussian noise, then sampling scheme will look like:

$$x_t^{A,i} = x_{t-1}^{A,i} + \Delta V_t^{A,i} + \Omega_t^{A,i}, \tag{3.25}$$

where $\Omega_t^{A,i} \sim N(0, \Sigma_G)$, represents the process noise with

$$\Sigma_G = \begin{vmatrix} \sigma_x^2 & 0 & 0 & 0 & 0 \\ 0 & \sigma_y^2 & 0 & 0 & 0 \\ 0 & 0 & \sigma_a^2 & 0 & 0 \\ 0 & 0 & 0 & \sigma_b^2 & 0 \\ 0 & 0 & 0 & 0 & \sigma_\rho^2 \end{vmatrix}, \tag{3.26}$$

where σ_x^2, σ_y^2, σ_a^2, σ_b^2 and σ_ρ^2 are the variances of the motion diffusion in the x-y directions, scaling (a, b) and rotation ρ respectively. In this case the motion proposal in equation 3.18 can be described by

$$q(x_t|x_{t-1}, z_t) = \sum_{n=1}^{N} N_{x_t}(x_{t-1}^{(n)} + \Delta V_t, \Sigma_G), \tag{3.27}$$

function $N_x(\mu, \Sigma)$ is represented by a Gaussian distribution:

$$N_x(\mu, \Sigma) \equiv \frac{1}{2\pi|\Sigma|} \exp(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)). \tag{3.28}$$

The tests showed that motion-based approach gives more useful states' predictions rather than random-based model, and, therefore, better results for tracking single target are achieved. For initialization, simple random-based proposal is applied.

Interaction model

When targets start to interact with each other (i.e., occlusion), we cannot rely on the motion based proposal anymore (at least, not for the targets that are not located on the front). As it was mentioned before, some techniques were already proposed to deal with multi target interactions in particle filtering: MCMC-based method [46] and its modifications [43], JPDA filter [42], etc.

Because in this project we adopted Bayesian framework approach, firstly it was decided to model interactions the same way as in [1], which assumes that interaction modeling will be part of the proposed framework and will be performed by re-weighting procedure [24]. The authors of [24] propose a method of building target interaction probability $p(z_t^{A,J_t^i}|x_t^{A,i}, z_t^{A,i})$ in equation 3.6 by representation of interaction as a magnetic-inertia potential model. Based on the particles' re-weighting, [24] shows how it's possible to deal with occlusions by using velocity and targets' location information. This method was implemented for our system too, however, it did not give as good results as expected, probably due to the reasons of targets' non-rigidity or even the use of little prior information.

Therefore, it was decided to avoid re-weighting procedure and set the mentioned interaction probability to uniformly distributed; though, it forces to use another technique, outside from the Bayesian framework, that could help to deal with occlusion problems in a better way. Other re-weighting models can be tested and added later, if it will be necessary, so they will be part of the Bayesian framework.

The magnetic-repulsion inertia re-weighting was tested against the behavior of the simple conventional particle filter, which uses random based proposal (instead of motion based) that is based on the inertia information (previous motion direction). And since it gave better results, it was decided to use random based prediction:

$$x_t^{A,i} = G \cdot x_{t-1}^{A,i} + \Omega_t,$$

where G defines the deterministic system model and Ω_t is a random vector drawn from the noise distribution of the system [29] (process noise).

As it was said before, using random-based model sometimes does not give good prediction results, even if inertia information is used; therefore, it will be better if particle set will be more spread during target interactions, so that particles would cover more space and capture the target after occlusion is diminished. Increasing the process noise Ω_t serves well for this. The block-diagram of interacting objects is shown in figure 15.

The interaction itself is defined when the distance between the centers of the targets observations is less than predefined threshold. Although, theoretically, the interaction must be defined when targets' observation have intersect, therefore, each time we must check whether the ellipses have intersection or not - this gives good performance, but, probably, is a little bit complex. So, for the current project it was decided to use simple thresholding procedure:

$$thre = \alpha(a_t^k + a_t^l),$$

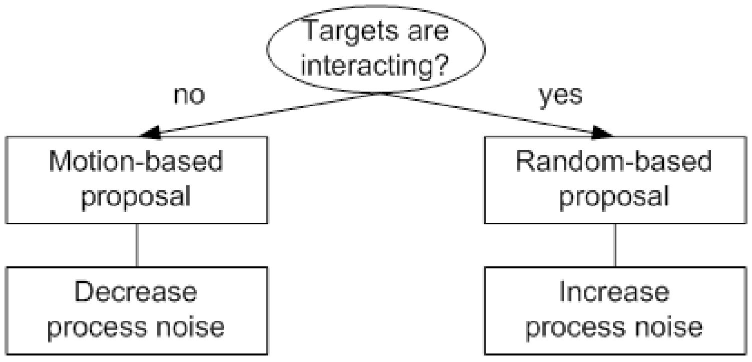


Figure 15: Target interaction scheme: left - no interaction, $x_t = x_{t-1} + \Delta V_t + \Omega_t^-$; on the right - when interact, $x_t = G \cdot x_{t-1} + \Omega_t^+$

where α is a constant, a_t^k and a_t^l are minor axises of the two considered ellipse k and ellipse l.

Camera collaboration model

Since one of the main requirements of our system is excluding any calibration etc. information, we can not apply the technique of camera collaboration likelihood model described in [1], because it is based on the calculation of epipolar lines which can not be performed without some prior knowledge about the relative camera locations. It means that we will also discard collaboration likelihood distribution $p(z_t^{B,i}|x_t^{A,i})$ from equation 3.6 and set it to uniform distribution. Again, like for interaction model, we get rid of the re-weighting procedure and replace it by another method of data fusion between views. The description of this data fusion procedure is given in the section 3.3. Although, it's again quite straightforward to embed any proposed camera collaboration model into equation 3.6.

3.3 Multi-camera data fusion

The situation when we deal with two cameras is illustrated in figure 16. The main goal there is how to correctly associate corresponding targets (assign the same identity to each them across the frames) and how to maintain this association in time, especially, when occlusions occur. As it was mentioned before, most of the literature suggests using homography, and, therefore, calibration is required. Although, they give reliable results, we cannot adopt the same strategy. Therefore, we need to think about another, probabilistic-based method that helps to fuse the data.

3.3.1 Problem statement

First, it's necessary to give a problem description - what are the input (what we have) and the output (what need to know). So, we have two cameras and certain number of targets with their current states (ellipse parameters) in each camera view. The data fusion procedure must assign the same identities to each of the targets across the camera views (if they are not set correctly) based only on the appearance information.

Let's assume that illumination does not change very drastically from one view to another (that

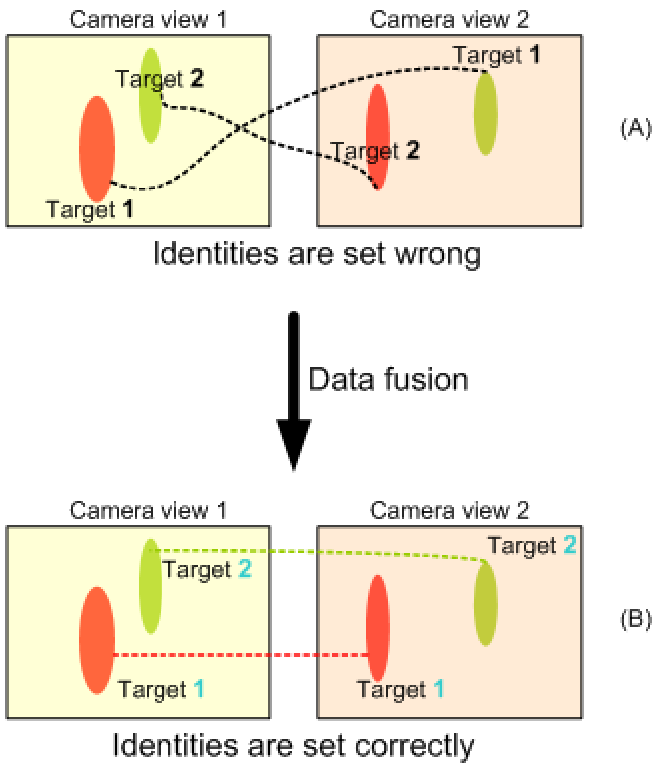


Figure 16: The figure reflects the problem: (A) shows that identities are not set correclty (i.e., red target has different identites in two views), but after association, (B) illustrates that identities are maintained, so we know that we monitor the same target in different views.

is, cameras will not have too broad points of view), and also people appearances remain almost the same on the both views (i.e., color of the T-shirt will not have different colors on the back and on the front of the person). Another assumption that we need to have is that the number of targets is the same in each view.
The solution should give an answer to the following question:

1. how to make the association between the targets within the given two cameras based on appearance information?
2. how to achieve robust labeling of the different targets across the different views?

3.3.2 Stable marriage problem

Stable marriage problem (SMP) is the problem of how to find a stable matching [61]. A matching is defined as a mapping from some elements (in our case - targets in first camera view) to other elements (targets in a second view). And it is called stable whenever there is **no** element A of the first matched set (first camera view) that prefers an element B (that it is not matched with) of the second matched set (second camera view); and at the same time B also prefers A over the one B is matched with. The matching algorithm also bears the name Gale-Shapley algorithm.
According to [62], the SMP is stated as:

> Given n men and n women, where each person has ranked all members of the opposite sex with a unique number between 1 and n in order of preference. Marry the men and women off such that there are no two people of opposite sex who would both rather have each other than their current partners. If there are no such people, all the marriages are 'stable'.

And the proposed solution [62]:

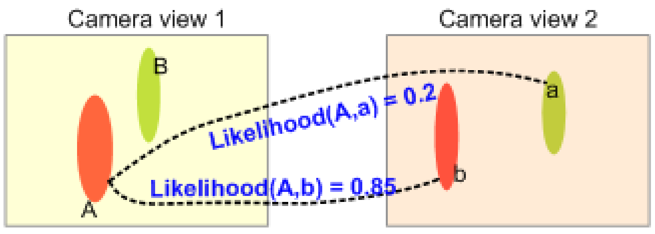
> The Gale-Shapley algorithm (GSA) involves a number of 'rounds' (or 'iterations') where each unengaged man 'proposes' to the most-preferred woman to whom he has not yet proposed. Each woman then considers all her suitors and tells the one she most prefers 'Maybe' and all the rest of them 'No'. She is then provisionally 'engaged'. In each subsequent round, each unengaged man proposes to one woman to whom he has not yet proposed (the woman may or may not already be engaged), and the women once again reply with one 'maybe' and reject the rest. This may mean that already-engaged women can 'trade up', and already-engaged men can be 'jilted'.

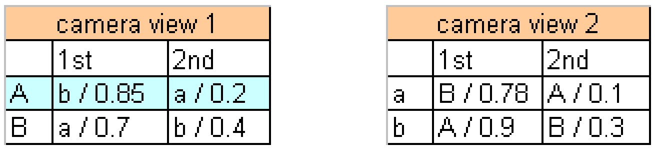
The GSA generates an optimal solution [61]:

- everyone gets married
- the marriages are stable

It was decided to use the same technique for setting correspondences between the targets' identities. In the context of data fusion the two sets are represented by two camera views and targets will play a role of the elements. Figure 17 shows how ranked list for each target is organized.
Therefore, presence of likelihoods for each target adds some modifications into the statements of the original SMP:

- preference list for each target contains information about the likelihoods (or weights); and actually based on these weights, the preference list is sorted.





camera view 1		
	1st	2nd
A	b / 0.85	a / 0.2
B	a / 0.7	b / 0.4

camera view 2		
	1st	2nd
a	B / 0.78	A / 0.1
b	A / 0.9	B / 0.3

Figure 17: Example of how the preference profiles for the targets might be built. Illustration on the top: for each target in the first camera view, based on the likelihood function, probability is calculated for each of the targets from second camera view. Tables on the bottom: the obtained probibilities are sorted to form a preference list for each of the target in each camera view - from the most preferrable target to the least (descending order of likelihoods)

- the preference list must be built beforehand (that is, the procedure must be performed each time before the running GSA).

The block-digram describing the algorithm and how it's intergated into the system is presented on figure 18. Once the pairings are set based on the GSA, correct identities are assigned to the targets that belong to the class 'woman' based on the pairing links.

Gale-Shapley properties

One of the GSA properties that must be considered are proposers-optimality and acceptors-pessimality [63] (chapter 1). Just assume that there is a matching between proposer A and acceptor B and it is stable. A proposer-optimal pairing means that every proposer is matched with its highest ranked acceptor. In the same way, an acceptor-optimal pairing is the one where each acceptor is paired with its lowest ranked proposer. For our system, it will mean that stable matching output can be different - depending on how are two classes will be split (which camera view will be defined as proposers and which one - acceptors). Although, if the assumptions stated earlier are hold (appearance of the targets is quite different), then the property will not affect much the output result.

It is worth mentioning that there is no need to apply the GSA for every frame - once it was initialized, identity information is actually a part of the state vector of a particle system, therefore, it will be propagated correctly until occlusion occurs. The algorithm must be applied again after occlusion (when targets do not interact anymore) - to check consistency. And if labels were suddenly switched during the interaction, GSA will be able to manage it.

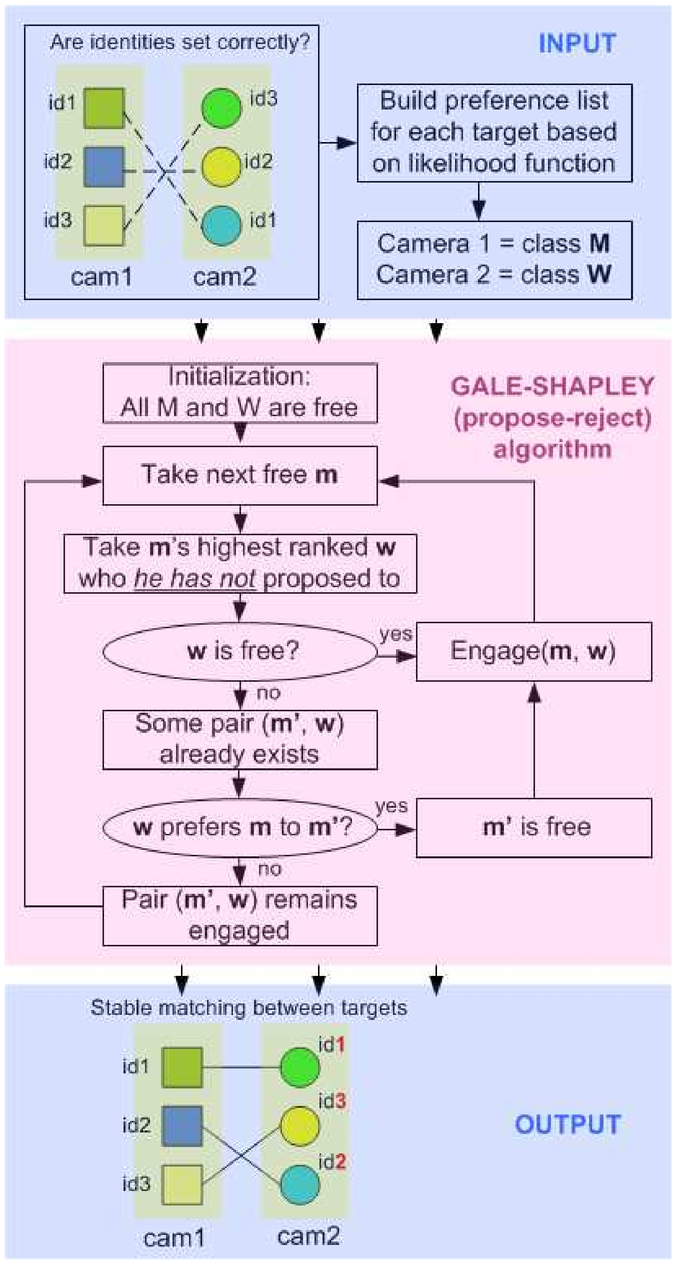

INPUT
Are identities set correctly?
id1
id2
id3
id3
id2
id1
cam1
cam2
Build preference list for each target based on likelihood function
Camera 1 = class **M**
Camera 2 = class **W**
GALE-SHAPLEY (propose-reject) algorithm
Initialization: All M and W are free
Take next free **m**
Take **m**'s highest ranked **w** who *he has not* proposed to
w is free?
yes
Engage(**m**, **w**)
no
Some pair (**m'**, **w**) already exists
w prefers **m** to **m'**?
yes
m' is free
no
Pair (**m'**, **w**) remains engaged
Stable matching between targets
id1
id2
id3
id1
id3
id2
cam1
cam2
OUTPUT

Figure 18: The block-diagram shows how GSA is used for data fusion. The blue block on the top: illustrates the problem when three targets (different in color within one camera view) have not got the correct identities; for each of the targets, list of preferences can be built based on the color likelihood; for GSA algorithm it's necessary to split the targets into two classes - proposors ('men') and acceptors ('women'), what is performed based on camera views' criterion (rectangles denote proposors and cirlces - acceptors). Pink block on the middle: represents the original GSA. Blue block on the bottom: output of the GSA, identities were set based on the pairing information.

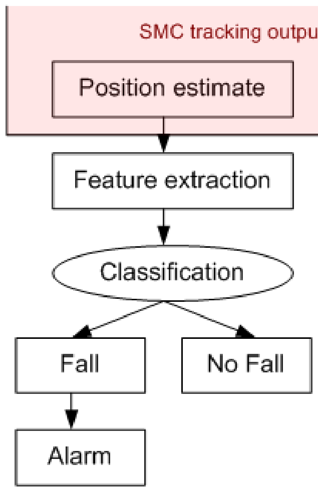


Figure 19: General block-diagram for fall detection part.

3.4 Target initialization

It was decided to use a previously done work at HIG [27] which uses CAMSHIFT-based algorithm for tracking. When the target just appears on the scene, it outputs the initial ellipse parameters, which are saved and used for initialization of the system.

3.5 Fall detection

The block-diagram for this part is displayed on figure 19 which takes previously estimated state of the target as input and performs further analysis.

3.5.1 Feature extraction

The aim of the feature vector is to give a good description of the moving objects position. These features will help to distinguish different activities - in our case, we want to be able to separate ’fall’ activity from ’no fall’ activity (walking, sitting down, lying down, running, etc.). All the features can be divided into two main groups: silhouette-based and motion-based features.
Silhouette-based features are extracted from still images:

- Aspect ratio - defined by ratio between the height and width of the bounding box.
- Height of the center mass - represented by the distance between the center mass of a person and the floor lever (bottom edge of the minimum bounding box).
- Height of the bounding box.
- Orientation is actually one of the our 5D ellipse parameters and it gives the overall direction.

More advanced methods may include calculation of edge histogram, i.e., when the person is lying, most of the edges will have horizontal orientation.

Since the human body was represented by ellipse when the tracking was performed, it’s straightforward to extract bounding box and center parameters for frame features extraction.

Motion-based features may also be used, these will be calculated:

- Direction of motion is defined by the direction of movements of the center of mass, can be calculated based on the current and previous positions on two consecutive frames.
- Speed value is a rate of motion, it measures how fast position changes.
- Motion history image (MHI) helps to detect when a large motion of a person is happening (when fall occurs, motion will be large). MHI is represented by an image where intensity of each pixel shows how recent the was motion in an image region [64].

Another class of features that can also be of use is audio-based features. Usually, a person falling produces high amplitude sounds as opposite to those when the person is sitting down or bending. At this point, previously done work at HIG [65] can be used, where audio analysis is based on wavelet domain signal decomposition. The main reason why the author used wavelet decomposition is that it can easily reveal aperiodic characteristics that are intrinsic to the falling down event. In [65] two main features were used to discriminate a fall event from different background sounds (music, speech, etc.):

- Zero-cross rate (ZCR) which shows how often the dominant frequency in the signal crosses zero level. Higher dominant frequency will produce higher crossing rate. For the falling down event ZCR values will be decreased.
- Variance - shows how spread the distribution is. When fall occurs, variance will increase rapidly.

To make use of both ZCR and variance, [65] suggests to use their ratio for optimization, since they are related (during the fall variance values increase and ZCR goes down).

3.5.2 Classification

The classification algorithm must be able to distinguish falls from other everyday activities. It takes as an input a feature vector and outputs an alarm if fall was recognized. For the project, it was decided that Support Vector Machine (SVM) can serve well for this. In [20] author show that SVM outperforms other seven classification algorithms.

SVM is related to the techniques that require supervised learning, therefore, some rules defining a fall must be known beforehand. If given a set of different human actions (walking, running, falling down), it's necessary to mark each as belonging to one of two categories ('fall' and 'no fall'). Later an SVM training algorithm will build a model that is able to predict - whether new sample belongs to the first or second class.

Thinking about it intuitively, if each sample is represented by a point in space, then SVM needs to build an optimal separating hyperplane between categories, what can be done by focusing on the training samples that are close to the edge of separation. Those training samples are called support vectors. There is no need to take into consideration other training samples, therefore, high classification accuracy can be achieved with small training data sets. For one-class classification (also called novelty detection or OCSVM), we can just define the one class that will represent the characteristics of all normal data (normal human activities, i.e., walking, sitting down, running, etc.). Samples that will not fit the same description very well will be considered

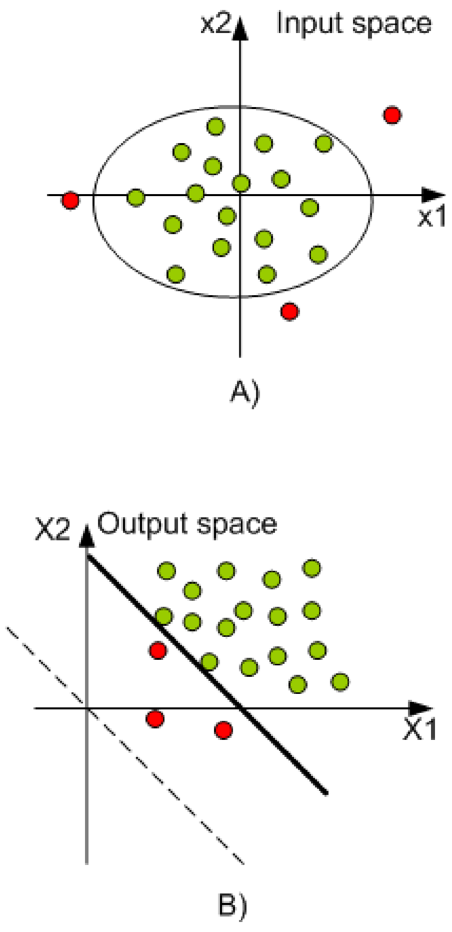


Figure 20: Support vector machine illustration. One-class category that stands for normal data is represented by green circles and outliers are red circles. A) represents data mapping by OCSVM from input space to B) output space.

as abnormal data or outliers [66] as illustrated on figure 20. OCSVM constructs a hyperplane to separate the normal data such that the margin between the hyperplane and outliers is maximized [67] and [68]. Small sphere and large margin (SSLM) method was introduced recently by [69], where the normal data is surrounded by optimal hypersphere with a margin which gives the maximum distance from outliers to the optimal hypersphere.

3.6 Implementation issues

It was decided to use Visual C++ as a main programming language for implementation. Also, other libraries were used: OpenCV (for image and video processing) [60], Sequential Monte-Carlo in C++ (SMCTC) [55] (particle filter functions), GNU Scientific Library (random values generation and error processing).

4 Experimental Results

4.1 Experimental setup

Two USB Logitec web cameras were used. All the video sequences with people were recorded from these cameras. For this purpose to make synchronized recordings, software developed by Nils Fjeldsø at HIG was used for two cameras. To avoid manual initialization of the targets, code of the previous work at HIG [27] was used, which is based on the CAMSHIFT algorithm.

Several experimental camera set-ups were organized, where people number, their activities (hand-shaking, simulation of falling down, walking and occluding each other) and also illumination (daylight, artificial room light) varied. Original videos were recorded with the frame size of 640×480. For our tests it was decided to decrease the frame size to 320×240 to lower computational costs needed for processing one frame. For all the sequences, we use 50 particles for each target. Different color and index is attached to an ellipse for labeling the target.

4.2 Analysis

It is still an open question how we can evaluate performance of multi-target tracking [70]. Some metrics exist for single-target tracking and for target detection systems [71]. Providing an exact performance evaluation for multi-target tracking approaches is a challenging problem due to varying number of targets and the frequent multi-target occlusions [70]. Plus, if we use multi-cameras, the target identity correspondence across different cameras increases the difficulty of the problem [1]. If the main goal of the tracking is the correctness of target location and its label, then we may use the *false position rate* FR_p and the *false label rate* FR_l described by equations 4.1 and 4.2 for quantitative tracking performance [72] - the test can be run for each cameras separately and then for all the cameras.

$$\mathsf{FR}_p = \frac{\text{the number of position failures}}{\text{the total number of targets in one / all camera(-s)}}, \tag{4.1}$$

$$\mathsf{FR}_l = \frac{\text{the number of lables failures}}{\text{the total number of targets in one / all camera(-s)}}, \tag{4.2}$$

where a *position failure* happens when the tracker loses its target in a camera, and *label failure* takes place when the tracker gives the wrong label to a target. Table 4 represents the statistical performance for video sequences for two and three targets on the scene. As it can be seen, our system is able to maintain tracks and identities for two targets. Video sequences with three targets were more challenging due to more complex interactions that occurred between targets.

One of the main reasons for target loss is color similarities between the target appearance and background. Figure 21 demonstrates the problem, where the target is lost after interaction occurs. An example of tracking for three targets is represented on figure 22.

	2 targets	3 targets
FR_p	8.3%	48.5%
FR_l	0.0%	16.5%
# of particles	50	50

Table 4: Quantative performance.

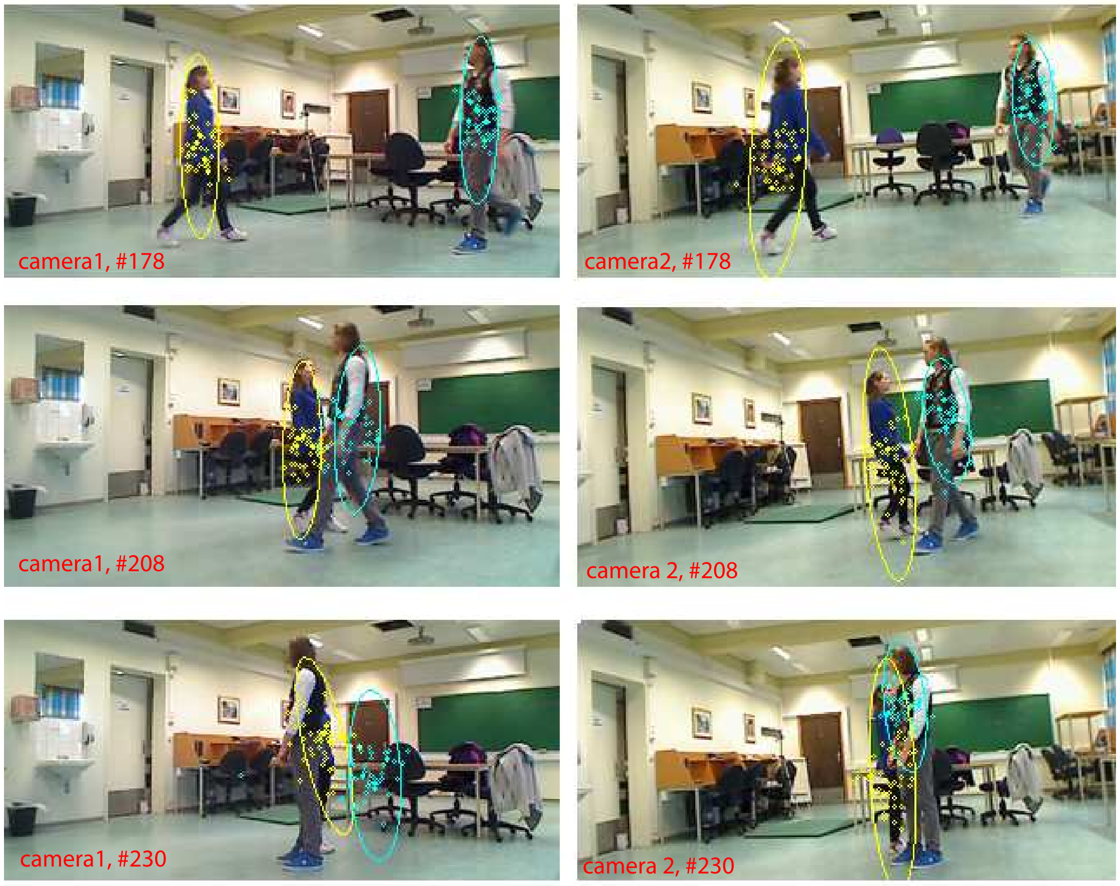


Figure 21: Video sequences obtained from two camera views. The blue tracker loses track of target in first camera view due to interaction and close similarity between the target appearance and background (it seems that color of the door is closer to the color distribution model of the blue tracker).

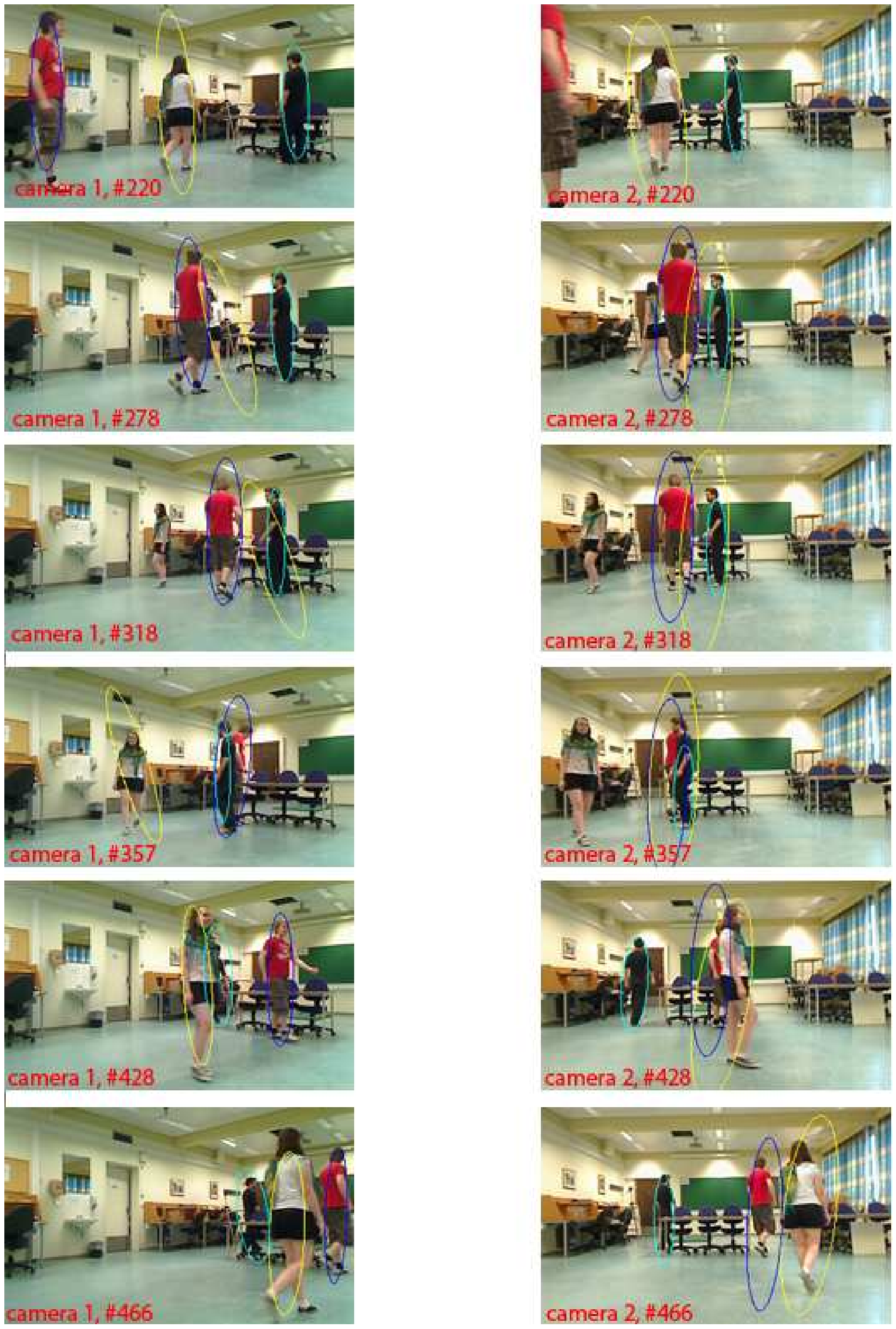


Figure 22: Example of our system behaivor for three targets on the scene. The yellow track loses its target in frames 278-357 and finds it back just because the target appears at the place where tracker failed. Another problem is ellipse over-expanding (visible on frames 428-466 of the second camera view), the reason of which is simplicity of modeling of observation likelihood function and target dynamics.

5 Conclusions and Future Directions

A probabilistic framework was proposed and implemented for multi-camera multi-target tracking. The advantage of this framework is that it does not require any camera calibration or view correspondence between the different cameras. It adopts a fully probabilistic approach for target detection, tracking and identification.

To solve multi-camera data fusion, simple technique 'stable marriage' (probability-based) was proposed. It does not check the correctness of coordinates of the tracked objects in the image coordinate system like most of the systems do, but serves for maintaining correct identities of the targets among the different camera views and gives more reliable results in fall detection procedure.

The experiments demonstrate quite accurate tracking results for two targets on the scene under the different visual challenges mentioned in section 1.2. Tests with video sequences with three or more people were not as successful. The tracker loses its target when an occlusion occurred. This is due to the similarity between target appearance and background (in terms of colors in the sequences we have).

Based on the study of the different parts of the system, we can conclude that the proposed system can be improved by modifying the method of local likelihood calculation. More advanced color descriptors could also be of more use since they allow to take into account not only color distribution, but also spatial distribution of the colors [51] for example. Also using multiple cues will give more reliable results, i.e., edge histogram, shape descriptors [34], SIFT descriptor [73], etc.

The non-rigid nature of the targets and their rigid representation (ellipse) also adds a need for improvements. Figure 23 describes the problem, where it is shown that ellipses also capture undesired background regions that we do not want to consider for target descriptors calculation. One possible solution is to use different weights for the pixels inside an ellipse when calculating a descriptor [29]. An intuitive weighting scheme would be to give larger weights to the central pixels and smaller for the edge-located ones.

Another important point to consider is how to include prediction of a, b and ρ the ellipse parameters for state dynamics model. Until now it was possible to predict only the center coordinates (cx, cy) of the ellipse based on the LK optical flow calculation (section 3.2.4). At the same time parameters a, b and ρ propagate according to random-based prediction which is not very effective sometimes since it can lead to over-expanding or over-squeezing of the ellipse bounds which is not desirable. It is not possible to apply the same method for a, b and ρ as for cx and cy prediction, since optical flow calculation does not give shape description (or shape bounds), but only a direction of motion. The possible solution lies in using a method that helps to get shape parameters or shape outline - for example, three-frame-differencing can be used [74] to get a contour which will help to get circumscribed ellipse and, therefore, its scale and rotation parameters.

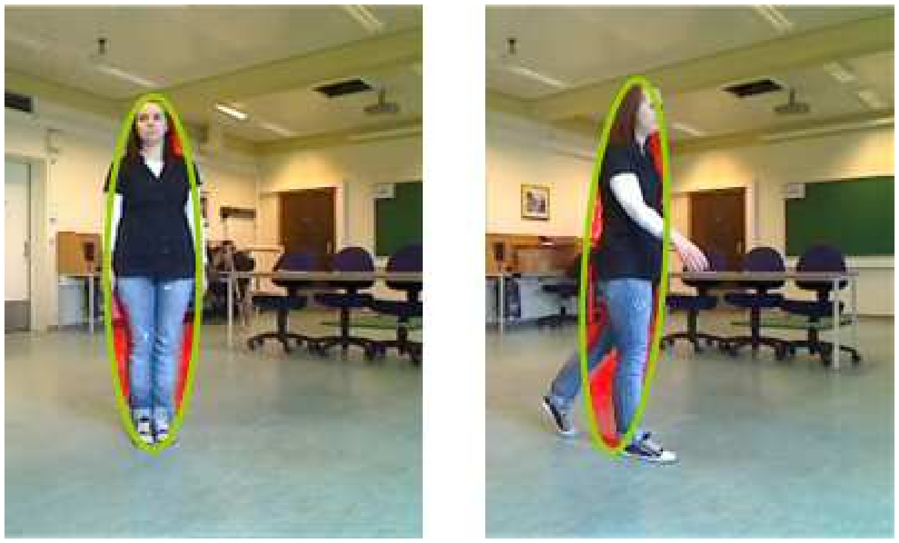


Figure 23: Ellipse representation of a human body is just a rough approximation. Red highlighted regions inside the ellipse bear background information and should not be considered when calculating target descriptor (i.e., color histogram). These regions become larger when walking person is observed from the side, thus, probability to lose the target is increased, especially when the target is occluded.

As to target interaction function, it also can be improved. Relying only on the random walk model for interacting targets does not work well for more than two targets. Furthermore, better results can be achieved if interaction model will be part of Bayesian framework as in [1]. Magnetic-repulsion inertia model [24] was actually implemented and tested as a part of our Bayesian framework, but did not provide as good performance as expected. One possible reason of it may lie in the aim of our application. The primary focus of our project is not only tracking, but also fall detection, and this fact must be taken into account when building state dynamics model. We must keep in mind that evolution of person movements can also have vertical direction (as shown on figure 24). This means that process noise for a, b and ρ ellipse parameters must be increased to provide larger variances for scale and orientation. In [24] the application is general and it is not assumed that target can rapidly change orientation or scale, therefore, only coordinates of the ellipse center cx and cy may change rapidly based on the direction of target's movement. Therefore, when applying magnetic-potential inertia model [24] for our system, large variance of orientation and scale can influence the re-weighting procedure and, thus, it does not perform so well. More tests with lower variances and different constant values could improve the performance.

The definition of targets' interaction is another issue that could be considered. For our project simple distance threshold was used. If this threshold is too large (i.e., sum of major semi-axises), interaction function starts working even when there is no actual interaction between targets which may lead to slower performance and worse state estimation. Too small threshold (i.e., sum of minor semi-axises) cannot describe interaction when one target is much closer to camera than the other, therefore, such interaction is better defined in terms of sum of major semi-axises

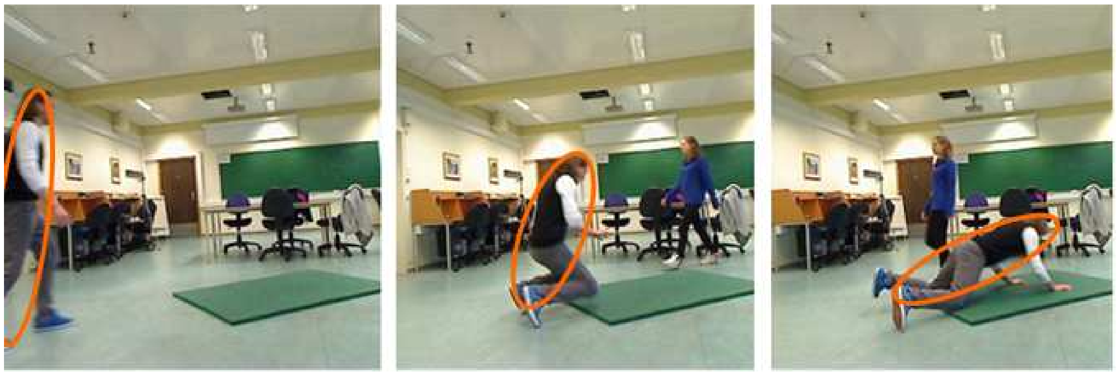


Figure 24: Fall evolution requires an ellipse parameters to be able to change rapidly so that it could follow the target while it changes vertical posture to horizontal.

of targets. In this case the tracker will most probably lose the target in the back since the prediction will be made based on the motion information from frontal target. Thus, it would be more efficient to replace the distance threshold by a test which allows to check whether ellipses intersect or not. If intersection occurs, it means that targets are interacting.

Though, data fusion procedure was implemented, it was tested only for simulated data. The procedure is quite simple and based only on similarities of color histograms. Again, like for local observation likelihood, more advanced cues for comparison could be added here as well. One question that might be also considered is whether it's possible to update targets' coordinates based on the obtained motion vector direction - i.e., whether it was changed during occlusion or not (same direction or opposite direction).

Regarding code optimization, the computational speed could be improved a lot by an optimized function for calculating color histogram (or any other descriptor) for elliptic region. For the project, OpenCV standard function was used that also includes prior ellipse mask extraction. Both operations make likelihood calculation a very expensive step.

A fall detection step was not implemented for the project, but only described theoretically. Code from previous work at HIG [65] which extracts features and performs classification based on KNN-neighborhood could be used for this purpose (but with OCSVM classifier instead of KNN).

Bibliography

[1] Qu, W., Schonfeld, D., & Mohamed, M. Distributed Bayesian multiple-target tracking in crowded environments using multiple collaborative cameras. *EURASIP Journal on Applied Signal Processing*, Issue 1, 21–21, 2007.

[2] Carbonell, J., Cohen, S., Nardone, T., & Hogan, H. 2008 older american update: Key indicator of wellness. Technical report, Federal Interagency Forum on Aging-Related Statistics, 2008. `http://www.aoa.gov/agingstatsdotnet/Main_Site/Data/2008_Documents/OA_2008.pdf`.

[3] Stevens, J., Mack, K., Paulozzi, L., MD, & Ballesteros, M. Self-reported falls and fall-related injuries among persons aged over 65 years. Technical report, Div of Unintentional Injury Prevention, National Center for Injury Prevention and Control, CDC, 2006.

[4] Yu, X. Approaches and principles of fall detection for elderly and patient. *Proceedings of the 10th IEEE Intl. Conf. on e-Health Networking, Applications and Service (HEALTHCOM) 2008*, 978-1-4244-2280-7, 42–47, 2008.

[5] Bourke, A., O'Brien, J., & Lyons, G. Evaluation of a threshold-based tri-axial accelerometer fall detection algorithm. *Gait & Posture*, 26, 194–199, 2007.

[6] Chen, J., Kwong, K., Chang, D., Luk, J., & Bajcsy, R. Wearable sensors for reliable fall detection. *IEEE Proceedings on Engineering in Medicine and Biology (EMB) 2005*, 4, 3551–4, 2005.

[7] Lee, Y., Kim, J., Son, M., & Lee, M. Implementation of accelerometer sensor module and fall detection monitoring system based on wireless sensor network. In *29th Annual International Conference of the IEEE. Engineering in Medicine and Biology Society (EMBS), 2007.*, Lyon, France. August 23-26 2007.

[8] Willems, J., Debard, G., Bonroy, B., Vanrumste, B., & Goedemé, T. How to detect human fall in video? an overview. In *Positioning and Context-Awareness International Conference (POCA) 2009*, Antwerp, Belgium. May 2009.

[9] Young-Sook, L. & HoonJae, L. Multiple object tracking for fall detection in real-time surveillance system. In *Proceedings of the 11th international conference on Advanced Communication Technology (ICACT) 2009*, 2308–2312, Phoenix Park, Republic of Korea. IEEE Press. February 15-18 2009.

[10] Huang, B., TIAN, G., & LI, X. A method for fast fall detection. In *Proceedings of the 7th World Congress on Intelligent Control and Automation (WCICA) 2008*, Chongqing, China. June 25-27 2008.

[11] Vishwakarma, V., Mandal, C., & Sural, S. Automatic detection of human fall in video. *Lecture Notes in Computer Science*, 4815/2007, 616–623, 2007.

[12] Auvinet, E., Revéret, L., St-Arnaud, A., Rousseau, J., & Meunier, J. Fall detection using multiple cameras. In *30th IEEE Engineering in Medicine and Biology Conference (EMBC) 2008*, Vancouver, Canada. August 20-24 2008.

[13] Thome, N., Miguet, S., & Ambellouis, S. A real-time, multiview fall detection system: A LHMM-based approach. *IEEE Transactions on Circuits and Systems for Video Technology 2008*, 18(11), 1522–1532, November 2008.

[14] Fu, Z., Delbruck, T., Lichtsteiner, P., & Culurciello, E. An address-event fall detector for assisted living applications. *IEEE Transactions on Biomedical Circuits and Systems 2008*, 2(2), 88–96, 2008.

[15] Töreyin, B. U., Dedeoğlu, Y., & Çetin, A. E. HMM based falling person detection using both audio and video. In *IEEE International Workshop on Human-Computer Interaction (HCI) 2005*, 211–220, Beijing, China. Springer-Verlag GmbH. October 2005.

[16] Nait-Charif, H. & McKenna, S. J. Activity summarisation and fall detection in a supportive home environment. *In proceedings of the International Conference on Pattern Recognition (ICPR) 2004*, 4, 323–326, August 2004. Cambridge, UK.

[17] Huang, B., Tian, G., & Wu, H. A method for fast fall detection based on intelligent space. In *IEEE International Conference on Automation and Logistics (ICAL) 2008*, Qingdao, China. September 1-3 2008.

[18] Anderson, D., Luke, R. H., Keller, J. M., & Skubic, M. Extension of a soft-computing framework for activity analysis from linguistic summarizations of video. In *IEEE International Conference on Fuzzy Systems (FUZZ) 2008*. June 1-6 2008.

[19] Derek, A., Robert, H. L., James, M. K., Marjorie, S., Marilyn, J. R., & Myra, A. Modeling human activity from voxel person using Fuzzy logic. *IEEE Transaction On Fuzzy Systems 2009*, 17(1), 39–49, 2009.

[20] Luštrek, M. & Kaluža, B. Fall detection and activity recognition with machine learning. *Informatica*, 33, 205–212, 2009.

[21] Berclaz, J., Fleuret, F., & Fua, P. Multi-camera tracking and atypical motion detection with behavioral maps. In *Proceedings of the European Conference on Computer Vision (ECCV) 2008*, Marseille, France. October 12-18 2008.

[22] Alper, Y., Omar, J., & Mubarak, S. Object tracking: A survey. *ACM Computing Surveys*, 38(4), 13, 2006.

[23] Francois, F., Jerome, B., Richard, L., & Pascal, F. Multicamera people tracking with a probabilistic occupancy map. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2), 267–282, 2008.

[24] Wei, Q., Dan, S., & Magdi, M. Real-time interactively distributed multi-object tracking using a magnetic-inertia potential model. In *Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV) 2005, Volume 1*, 535–540, Washington, DC, USA. IEEE Computer Society. 2005.

[25] Blake, A. & Isard, M. The Condensation algorithm - conditional density propagation and applications to visual tracking. In *Advances in Neural Information Processing Systems (NIPS) 1996*, 36–1, Denver, CO, USA. The MIT Press. December 2-5 1996.

[26] Jian, Y. & Jean-Marc, O. Multi-camera 3d person tracking with particle filter in a surveillance environment. In *16th European Signal processing Conference (EUSIPCO) 2008*, Lausanne, Switzeralnd. August 25-29 2008.

[27] Guraya, F. F. E., Bayle, P.-Y., & Cheikh, F. A. People tracking via a modified CAMSHIFT algorithm. 2009.

[28] Ting-Hsun, C. & Shaogang, G. Tracking multiple people with a multi-camera system. In *Proceedings of the IEEE Workshop on Multi-Object Tracking (WOMOT) 2001*, 19, Washington, DC, USA. IEEE Computer Society. 2001.

[29] Nummiaro, K., Koller-Meier, E., Gool, L. V., & Gaal, L. V. Object tracking with an adaptive color-based particle filter. `http://www.koller-meier.ch/esther/dagm2002.pdf`. 2002.

[30] Thirde, D., Borg, M., Ferryman, J., Aguilera, J., Kampel, M., & Fernandez, G. Multi-camera tracking for visual surveillance applications. In *Computer Vision Winter Workshop (CVWW) 2006*, Telč, Czech Republic. February 6-8 2006.

[31] Lucas, B. D. & Kanade, T. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI) 1981*, 674–679, Vancouver, British Columbia, Canada. August 1981.

[32] Collins, R. T., Amidi, O., & Kanade, T. An active camera system for acquiring multi-view video. In *Proc. International Conference on Image Processing (ICIP) 2002*, 517–520, Rochester, New York, USA. June 24-28 2002.

[33] Maskell, S. & Gordon, N. A tutorial on particle filters for on-line nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing 2001*, 50, 174–188, 2001.

[34] Paul, B., Lyudmila, M., David, B., & Nishan, C. Sequential Monte Carlo tracking by fusing multiple cues in video sequences. *Image Vision Comput.*, 25(8), 1217–1227, 2007.

[35] Gordon, N. A hybrid bootstrap filter for target tracking in clutter. *IEEE Transactions on Aerospace and Electronic Systems 1997*, 33, 353–358, 1997.

[36] Poon, E. & Fleet, D. J. Hybrid Monte Carlo filtering: Edge-based people tracking. In *IEEE Workshop on Motion and Video Computing (WMVC) 2002*, Orlando, Florida, USA. December 2 2002.

[37] Xiong, T. & Debrunner, C. Monte carlo visual tracking using color histograms and a spatially weighted oriented hausdorff measure. In *10th International Conference on Computer Analysis of Images and Patterns (CAIP) 2003*, 190–197, Netherlands. August 25-27 2003.

[38] Doucet, A. On sequential simulation-based methods for Bayesian filtering. Technical report, Cambridge University, Department of Engineering, Cambridge, UK, 1998.

[39] Bouaynaya, N., Qu, W., & Schonfeld, D. An online motion-based particle filter for head tracking applications. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP) 2005*, 225–229, Philadelphia, PA, USA. March 18-23 2005.

[40] Okuma, K., Taleghani, A., Freitas, N. D., Freitas, N. D., Little, J. J., & Lowe, D. G. A boosted particle filter: Multitarget detection and tracking. In *The 8th European Conference on Computer Vision (ECCV) 2004*, 28–39, Prague. May 11-14 2004.

[41] Rui, Y. & Chen, Y. Better proposal distributions: Object tracking using unscented particle filter. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR) 2001*, II:786–793, Kauai Marriott, Hawaii. December 9-14 2001.

[42] Frank, O., Nieto, J., Guivant, J., & Scheding, S. Multiple target tracking using sequential Monte Carlo methods and statistical data association. In *Proceedings of the IEEE / RSJ International Conference on Intelligent Robots and Systems*, Las Vegas, Nevada. October 27-31 2003.

[43] Qian, Y. & Gerard, M. Multiple-target tracking by spatiotemporal Monte Carlo Markov chain data association. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12), 2196–2210, 2009.

[44] Schumitsch, B., Thrun, S., Bradski, G., & Olukotun, K. The information-form data association filter. In *Proceedings of Conference on Neural Information Processing Systems (NIPS) 2005*, Vancouver, B.C., Canada. MIT Press. December 2005.

[45] Oh, S., Russell, S., & Sastry, S. Markov chain Monte Carlo data association for general multiple-target tracking problems. `http://www.eecs.berkeley.edu/~sho/papers/cdc04_mcmcda.pdf`. 2004.

[46] Khan, Z., Balch, T., & Dellaert, F. An MCMC-based particle filter for tracking multiple interacting targets. In *Proceedings of the 8th European Conference on Computer Vision (ECCV) 2004*, 279–290, Prague. May 11-14 2003.

[47] Aghajan, H. & Cavallaro, A. *Multi-Camera Networks: Principles and Applications*. Number ISBN: 978–0–12–374633–7. Elsevier. 2009.

[48] Cucchiara, R., Prati, A., & Vezzani, R. A multi-camera vision system for fall detection and alarm generation. In *Expert Systems, Volume 24 Issue 5*, 334–345. 2007.

[49] Du, W. & Piater, J. Multi-camera people tracking by collaborative particle filters and principal axis-based integration. *8th Asian Conference on Computer Vision (ACCV) 2007*, LNCS 4843, 365–374, 2007.

[50] Khan, S. & Shah, M. Consistent labeling of tracked objects in multiple cameras with overlapping fields of view. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25, 1355–1360, 2003.

[51] Monari, E., Maerker, J., & Kroschel, K. A robust and efficient approach for human tracking in multi-camera systems. *IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS) 2009*, 0, 134–139, 2009.

[52] Du, W. & Piater, J. Data fusion by belief propagation for multi-camera tracking. In *The 9th Asian Conference on Computer Vision (ACCV) 2009*, XiAn, China. September 23-27 2009.

[53] Cai, Y. Robust visual tracking for multiple targets. Master's thesis, The University of British Columbia, 2005.

[54] Whittaker, J. *Graphical Models in Applied Multivariate Statistics*. Wiley Publishing. 2009.

[55] Johansen, A. M. SMCTC: Sequential Monte Carlo in C++. *Journal of Statistical Software*, 30(6), 1–41, 4 2009.

[56] Doucet, A. & de Freitas, N. *Sequential Monte Carlo methods in practice*. New York, Springer. 2001.

[57] Kitagawa, G. Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *Journal of Computational and Graphical Statistics*, 5, No. 1, 1–25, 1996.

[58] Moghaddam, B. & Pentland, A. Probabilistic visual learning for object representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence 1997*, 19(7), 696–710, 1997.

[59] Bouguet, J. Y. Pyramidal implementation of the Lucas Kanade feature tracker: Description of the algorithm. Intel Corporation Microprocessor Research Labs, `http://robots.stanford.edu/cs223b04/algo_tracking.pdf`. 2002.

[60] Bradski, G. The OpenCV library. `http://opencv.willowgarage.com/documentation/`. 2000.

[61] Gale, D. & Shapley, L. S. College admissions and the stability of marriage. *American Mathematical Monthly*, 69, 9–14, 1962.

[62] Wikipedia. Stable marriage problem. `http://en.wikipedia.org/wiki/Stable_marriage_problem`. 2010.

[63] Kleinberg, J. & Tardos, E. *Algorithm Design*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA. 2005.

[64] Bobick, A. F. & Davis, J. W. The recognition of human movement using temporal templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence 2001*, 23, 257–267, 2001.

[65] Alaliyat, S. Video - based fall detection in elderly's houses. Master's thesis, Gjøvik University College, 2008.

[66] Moya, M., Koch, M., & Hostetler, L. One-class classifier networks for target recognition applications. In *Proceedings of world congress on neural networks 1991*, Seattle, USA. 1991.

[67] Schölkopf, B. & Smola, A. J. *Learning with kernels : support vector machines, regularization, optimization, and beyond*. Adaptive computation and machine learning. MIT Press. 2002.

[68] Tax, D. M. J. & Duin, R. P. W. Support vector data description. *Mach. Learn.*, 54(1), 45–66, 2004.

[69] Wu, M. & Ye, J. A small sphere and large margin approach for novelty detection using training data with outliers. *IEEE Transactions on Pattern Analysis and Machine Intelligence 2009*, 31, 2088–2092, 2009.

[70] Black, J., Ellis, T., & Rosin, P. A novel method for video tracking performance evaluation. In *In Joint IEEE Int. Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance (VS-PETS) 2003*, 125–132, Nice, France. October 11-12 2003.

[71] Chiani, M., Giorgetti, A., Mazzotti, M., Minutolo, R., & Paolini, E. Target detection metrics and tracking for UWB radar sensor networks. In *IEEE International Conference on Ultra-Wideband (ICUWB) 2009*, Vancouver, Canada. September 9-11 2009.

[72] Ellis, T. Performance metrics and methods for tracking in surveillance. In *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS) 2002*, Copenhagen, Denmark. June 2002.

[73] Lowe, D. G. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60, 91–110, 2004.

[74] Zhao, S., Zhao, J., Wang, Y., & Fu, X. Moving object detecting using gradient information, three-frame-differencing and connectivity testing. *AI 2006: Advances in Artificial Intelligence*, 4304, 510–518, 2006.