Examensarbete

Human Motion Tracking Using 3D Camera

Examensarbete utfört i Reglerteknik
vid Tekniska högskolan i Linköping
av

Daniel Karlsson

LiTH-ISY-EX--10/4292--SE
Linköping 2010
Human Motion Tracking Using 3D Camera

Examensarbete utfört i Reglerteknik
vid Tekniska högskolan i Linköping
av

Daniel Karlsson

LiTH-ISY-EX--10/4292--SE

Handledare: Christina Grönwall
              FOI
Fredrik Lindsten
              ISY, Linköpings universitet
Joakim Rydell
              FOI

Examinator: Fredrik Gustafsson
              ISY, Linköpings universitet

Linköping, 10 March, 2010
The interest in video surveillance has increased in recent years. Cameras are now installed in e.g. stores, arenas and prisons. The video data is analyzed to detect abnormal or undesirable events such as thefts, fights and escapes. At the Informatics Unit at the division of Information Systems, FOI in Linköping, algorithms are developed for automatic detection and tracking of humans in video data. This thesis deals with the target tracking problem when a 3D camera is used. A 3D camera creates images whose pixels represent the ranges to the scene. In recent years, new camera systems have emerged where the range images are delivered at up to video rate (30 Hz). One goal of the thesis is to determine how range data affects the frequency with which the measurement update part of the tracking algorithm must be performed. Performance of the 2D tracker and the 3D tracker are evaluated with both simulated data and measured data from a 3D camera. It is concluded that the errors in the estimated image coordinates are independent of whether range data is available or not. The small angle and the relatively large distance to the target explains the good performance of the 2D tracker. The 3D tracker however shows superior tracking ability (much smaller tracking error) if the comparison is made in the world coordinates.
Abstract

The interest in video surveillance has increased in recent years. Cameras are now installed in e.g. stores, arenas and prisons. The video data is analyzed to detect abnormal or undesirable events such as thefts, fights and escapes. At the Informatics Unit at the division of Information Systems, FOI in Linköping, algorithms are developed for automatic detection and tracking of humans in video data. This thesis deals with the target tracking problem when a 3D camera is used. A 3D camera creates images whose pixels represent the ranges to the scene. In recent years, new camera systems have emerged where the range images are delivered at up to video rate (30 Hz). One goal of the thesis is to determine how range data affects the frequency with which the measurement update part of the tracking algorithm must be performed. Performance of the 2D tracker and the 3D tracker are evaluated with both simulated data and measured data from a 3D camera. It is concluded that the errors in the estimated image coordinates are independent of whether range data is available or not. The small angle and the relatively large distance to the target explains the good performance of the 2D tracker. The 3D tracker however shows superior tracking ability (much smaller tracking error) if the comparison is made in the world coordinates.

Sammanfattning

Acknowledgments

This thesis concludes my studies for a Master of Science in Applied Physics and Electrical Engineering at Linköping University. I would therefore here like to express my gratitude to everyone who has accompanied me on this journey.

The thesis work was conducted at and funded by the Swedish Defence Research Agency (FOI) in Linköping. I would like to thank my supervisor Christina Grönwall for providing an interesting and challenging topic for my thesis. She has guided me a lot in the progress and pushed me to the finish line. Also, I would like to thank my supervisor Joakim Rydell for many discussions, ideas and inputs to the project. Further, I would like to thank my supervisor Fredrik Lindsten for engagement and lucidity, helping me to organize the work and report. I also want to thank my examiner Fredrik Gustafsson for his feedback.

Thanks also to Håkan Larsson, Gustav Tolt and Thomas Chevalier who contributed a lot to the understanding of the camera and its output data. Furthermore, I would like to thank Fredrik Hemström for quick development of an application for reading out data from one of the cameras, and Katarina Åberg for proof reading this report. I would also like to thank my opponent Robert Stribeck for his feedback and a good opposition. Special thanks to Alexander Lindmaa for listening to me rehearsing the oral presentation.

Finally, I would like to thank my family and friends for their support and encouragement. It has helped me a lot and I appreciate it very much!

Linköping, February 2010

Daniel Karlsson
# Contents

1 Introduction ................................................. 1  
  1.1 Problem formulation ...................................... 2  
  1.2 Objectives .............................................. 2  
  1.3 Limitations .............................................. 3  
  1.4 Related work ............................................ 3  
  1.5 Outline of the report .................................... 7  

2 Sensor system ................................................. 9  
  2.1 Range measurement methods ............................. 9  
  2.2 FLASH sensor ............................................ 10  

3 Camera model ................................................ 13  
  3.1 Pinhole camera ........................................... 13  
  3.2 Path of the light ......................................... 15  
  3.3 Field of view ............................................. 15  
  3.4 Validity of the pinhole camera model .................. 15  

4 Modeling human motion ....................................... 17  
  4.1 Overview ................................................. 17  
  4.2 Choice of coordinate system ............................ 18  
  4.3 Linear models ........................................... 18  
    4.3.1 White noise acceleration model ................. 18  
    4.3.2 Piecewise constant white acceleration model .... 19  
  4.4 Choice of process model ................................. 19  

5 Head detection ................................................ 21  
  5.1 Foreground computation .................................. 21  
  5.2 Head detector ............................................ 22  

6 Tracking algorithms ......................................... 27  
  6.1 Kalman filter ............................................ 27  
  6.2 Extended Kalman filter .................................. 28  
  6.3 Converted Measurement Kalman filter with Debiasing ... 29  
  6.4 Choice of tracking algorithm ............................ 30
7 Simulations
7.1 Overview .................................................. 33
7.2 What to test ................................................. 33
7.3 Measures ..................................................... 34
7.4 Results ....................................................... 35
  7.4.1 Scenario 1: Straight line, parallel to the image plane ........ 36
  7.4.2 Scenario 2: Straight line, not parallel with image plane,
                 range from 3m to 170m .................................. 38
  7.4.3 Scenario 3: Two adjacent circles ........................ 42
  7.4.4 Scenario 4: Two adjacent rhombuses ...................... 45

8 Tracking in real video sequences ............................... 51
  8.1 Measures .................................................. 51
  8.2 Results .................................................... 52
  8.2.1 Sequence 1: Simple motion near camera ................. 52
  8.2.2 Sequence 2: Occlusion by truck and noise .............. 58
  8.2.3 Sequence 3: Random walk at a range of 60 m ............ 65

9 Conclusions ..................................................... 69

10 Discussion and future work ..................................... 73
  10.1 Discussion ............................................... 73
      10.1.1 The statement ....................................... 73
      10.1.2 Video frame rate ................................... 73
      10.1.3 Chain of observation processing ...................... 74
      10.1.4 Target selection ..................................... 75
      10.1.5 Constant velocity ................................... 76
      10.1.6 Comparing performance ............................... 76
      10.1.7 Sampling frequency .................................. 77
      10.1.8 Studied scenarios .................................... 77
      10.1.9 Process modeling ..................................... 77
      10.1.10 Ground-truth for collected data ..................... 78
      10.1.11 Segmentation in 3D ladar data ....................... 78
      10.1.12 Singletarget vs. multitarget tracking .............. 78
      10.1.13 Multisensor tracking ................................ 79
  10.2 Future work ............................................... 79

Bibliography ....................................................... 81
Glossary


3D three-dimensional. 2–6, 9, 10, 13, 17, 19, 21, 22, 30, 31, 33–39, 42, 43, 45, 46, 52, 54, 60, 61, 65, 66, 69–71, 73–80

ASC Advanced Scientific Concepts. 10

BLUE Best Linear Unbiased Estimator. 28, 79

CMKF-D Converted Measurement Kalman filter with Debiasing. 29–31

CV constant velocity. 17, 36, 45, 52

EKF Extended Kalman filter. 28–31, 74, 79

EM Expectation-Maximization. 4

EMEKF Error model based Modified Extended Kalman filter. 30

FOI Swedish Defence Research Agency. 2

FOV Field of View. 15, 16, 52, 69, 71, 75

InSAR interferometric synthetic aperture radar. 9

KF Kalman Filter. 27–30, 34, 79

LADAR laser detection and ranging. 10

LIDAR light detection and ranging. 10

MM multiple model. 18

MMSE Minimum Mean Square Error. 28
PDF Probability Density Function. 28, 35, 75, 79

PPI plan position indicator. 1

RIM range imaging sensor. 9

RMSE Root Mean Square Error. 34–39, 42, 43, 45, 46, 51, 69, 70, 76, 77

SAR synthetic aperture radar. 9

SfS Shapes from Silhouettes. 3

TOF time-of-flight. 3–5, 9–11, 80
Chapter 1

Introduction

Ever since the first radar systems became available, there has been a need for tracking targets. The radar systems registered echoes from sent-out radiowaves and displayed them as dots on a screen, the plan position indicator (PPI) (as seen in Figure 1.1). The dots could represent friends, enemies or false echoes. In order to try to determine their origin, the dots were tracked by manually drawing lines from one time to another, connecting dots together. By doing so, moving objects could be spotted and their next move be predicted. However, determining whether a dot at one moment represented the same target as another dot that was shown earlier or another target or just a false echo, was not and is not an easy task. Many strategies and algorithms to solve the task have been developed since then. They have also been more or less automated by the use of computers.

![Figure 1.1. A radar screen, also known as PPI.](http://www.vectorsite.net/ttradar_2_01.png) Public-domain image from Goebel’s *Introduction to radar technology* [25].

Over time, the algorithms have also come to be used in video sequences captured by image sensors. Today, video cameras are common in for example many stores, in arenas and in prisons (even though there are no official statistics on how
many cameras there are in use [2]. They are used for people surveillance in order to detect thefts, fights or escapes. Analysis of the sequences is however often done manually which is time-consuming. If it could be done effectively by a computer, both manhours and money could be saved. Normally, images are captured in the visual spectrum but now there are also so called range imaging video sensors or three-dimensional (3D) video cameras on the market that can produce range coded images. Each pixel then contains the range to the scene. This type of camera was introduced and launched some years ago and now there is a rapid development going on in this technology. The images are highly interesting for tracking purposes since the position of a target can be measured in all three spatial dimensions.

In analogy with the radar screen, when using a 3D camera, the dots in the plane are replaced by dots in space - just like the real points of the target. The hypothesis in this thesis is that the 3D measurements will facilitate tracking compared to when only two-dimensional (2D) measurements are available.

### 1.1 Problem formulation

At the Informatics unit at the division of Informatics at Swedish Defence Research Agency (FOI), there is a project called “3D imaging lasers for dynamic applications”. This project aims at expanding the knowledge about 3D cameras and how they can be used in different applications. During a workshop\(^1\) on tracking, attended by Grönwall\(^2\), a statement was made, telling that measurements not need to be performed as often when using a 3D camera as when using a conventional 2D camera. This statement has been considered important to be further investigated and motivates the need for this thesis work. In this thesis, the validity of the statement will be evaluated in a human target tracking application. If the statement is found to be true, recommendations will be given on how to use the time that is saved.

### 1.2 Objectives

The objectives of this thesis are as follows.

- Determine a model for the human target dynamics.
- Implement an algorithm (referred to as 3D tracker) for tracking with a 3D camera.
- Test the performance of the algorithm on simulated data.
- Evaluate the performance on real data collected from a 3D camera.
- Compare the performance of the 3D tracker to that of a 2D tracker.

---

\(^{1}\)ATR/ATI (Automated target recognition/identification) Workshop at FGAN FOM in Ettingen, Germany held by NATO SET-077 in Oct. 27-28, 2003

\(^{2}\)Christina Grönwall, FOI
1.3 Limitations

Images from a 3D camera have been collected prior to the start of this thesis work. Further, comparison of the algorithms will be made in image coordinates only since both the 2D camera and the 3D camera can get measurements in these coordinates. Another limitation is that only the problem of single-target tracking will be addressed in this thesis.

1.4 Related work

There are many articles treating the subject of tracking humans in images but not many of them treat tracking with range information. Exceptions to this include:

- **Tracking Without Background Model for TOF Cameras** [10] by Bianchi et al., 2009. In this article, a new approach to foreground segmentation is presented and evaluated on a sequence to detect humans. The method is based on region growing. It is shown to be applicable even when a non-stationary camera is used.

- **People Detection and Tracking with TOF Sensor** [83] by Tanner et al., 2008. The article describes a simple approach for tracking people in a room. Each image was searched for persons and then the detections were kept on record. Based on where the detections were made and the geometry of the room, conclusions were drawn on whether the detection represented a person that had been detected before or if it was a new person. The authors abandoned the Kalman filter since the errors were not Gaussian and the algorithm was time-consuming.

- **TOF Imaging in Smart Room Environments Towards Improved People Tracking** [29] by Guðmundsson et al., 2008. This article describes a framework for improved segmentation when tracking in a smart room, using one time-of-flight (TOF) camera and six standard cameras. The new framework is based on foreground segmentation and the Shapes from Silhouettes (SfS) method. In this way, the geometry of the room is reconstructed by studying how the objects are seen from different views. The authors conclude that the reconstruction is much better than when the TOF camera is not used. The segmentation is also improved but is dependent on whether shadow suppression mode on the camera is used or not.

- **Classification and Localization of Vehicle Occupants Using 3D Range Images** [19] by Devarakota, 2008. This PhD thesis presents a framework for detection of occupants in the seats of a car using a 3D camera. An occupant classification algorithm is introduced and discussed. Then also detection and tracking of the heads of the occupants are performed. The framework could then be used for determining if and when the air bag should release in case of an accident.
• **Head-Pose Tracking with a Time-of-Flight Camera** [57] by Meers and Ward, 2008. In this article, the authors describe their approach to head tracking and identification of persons. They detect the nose tip of the face in 3D space and use this information to construct a 3D model of the face. Via spherical intersections of the face, a person can be identified from a database of 3D models.

• **Cluster Tracking with Time-of-Flight Cameras** [34] by Hansen et al., 2008. The authors examine a method of projecting people in range images to the ground, resulting in a cluster of points. A circle is then fitted to the cluster and tracked. In this way, occlusions more often can be avoided. It is concluded that accurate detections are achieved in 98% of the frames.

• **Range Imaging Technology: New Developments and Applications for People Identification and Tracking** [40] by Kahlmann et al., 2007. A 3D camera and the condensation algorithm were used for tracking of a person and estimation of its 3D trajectory. The authors used a state vector with the position, velocity and acceleration in image coordinates and also the dimensions of the bounding box. From the work it was concluded that the tracking performance was promising but needed further testing.

• **Real-Time 3D Head Tracking Based on Time-of-Flight Depth Sensor** [66] by Parvizi and Wu, 2007. Segmentation is here performed directly in the range image via thresholding of its histogram. Curve analysis and ellipse fitting then gives the head detections. It is suggested that background modeling is unnecessary when using 3D TOF cameras. Another advantage is that the system better can handle occlusions and multiple objects overlapping.

• **Video Surveillance Using a Time-of-Light Camera** [SIC!] [80] by Silvestre et al., 2007. Silvestre has implemented a video surveillance system based on a TOF sensor. After background subtraction, blobs are extracted and tracked using the Expectation-Maximization (EM) algorithm. The system is tested on different sequences and the results are commented. However, no final conclusion was drawn.

• **3D Head Tracking Based on Recognition and Interpolation Using a Time-of-Flight Depth Sensor** [26] by Göktürk and Tomasi, 2004. A tracking algorithm is proposed which consists of two stages; one training stage and one testing stage. In the training stage, long video sequences of humans are captured upon which depth signatures are automatically calculated. The signatures are compared to manual calculations to train the algorithm. In the testing stage, the signature is calculated for the current frame and matched (via correlation) to the training set.

• **Real-Time Head Tracking and 3D Pose Estimation from Range Data** [54] by Malassiotis and Strintzis, 2004. A head tracking system is developed where the range image is thresholded based on the histogram and the head is tracked using the condensation algorithm. The 3D pose of the head is estimated via an appearance-based approach.
Other contributions to tracking with 3D cameras include:

- *Evaluation of a Foreground Segmentation Algorithm for 3D Camera Sensor* [9] by Bianchi et al., 2009. The authors examine a region-growing based method for segmentation of range images from a TOF sensor, avoiding background subtraction or thresholding range data. By using a region-growing-based method, clear boundaries can be obtained between the target and other parts of the scene. The target is first localized in the intensity image, by assuming that it is the most reflective object in the scene, and thereafter tracked in the range image (where the regions are best grown).

- *Using a 3D Time-of-Flight Range Camera for Visual Tracking* [70] by Reiser and Kubacki, 2007. The article discusses tracking of a red ball using a 3D camera put on a robotic arm. A servo system is implemented with the task of positioning the camera so that the target lies along its optical axis. It was concluded that using a 3D camera simplified the task compared to when multiple cameras or more complex algorithms to estimate the positions were used.

- *An Extension to the Condensation Algorithm for Tracking Multiple Objects in Range Images* [47] by Koller-Meier, 2000. The original condensation algorithm is in this PhD thesis extended to also track multiple objects. It is achieved by having only one probability distribution for all objects. The extension is less easy to interpret than the Kalman approach but can instead be more efficient when many objects are to be tracked. The algorithm is tested on simulated images from a range sensor mounted on a car for tracking other cars.

Tracking in conventional 2D images are studied for example in:

- *Understanding Human Interactions with Track and Body Synergies (TBS) Captured from Multiple Views* [64] by Park and Trivedi, 2008. A framework is presented that makes use of images captured from several cameras in order to maintain tracking and recognition of the target. Depending on how similar the current measurement of the target is to the model, different strategies are applied. If it is similar to the model, the specific pose and gesture of the target is identified, otherwise tracking of the position and speed etc. is performed. Interaction between persons are also analyzed by these methods.

- *Estimating Pedestrian Counts in Group* [45] by Kilambi et al., 2008. A general tracking system is presented. It processes data at three levels: image level, blob level and pedestrian level. At the image level, background subtraction is performed to obtain difference images. These images are used at the blob level to identify connected regions, “blobs”, and track them as such. Finally, at the pedestrian level, individuals are tracked using blob association and pedestrian position estimation. A blob is only passed on to this level if it can be tracked for a specified minimum amount of time.
• **Multimodal Technologies for Perception of Humans** [14] by Stiefelhagen et al., 2008. This is a book based on selected articles about video surveillance and human tracking.

• **Dynamic Object Detection, Tracking and Counting in Video Streams for Multimedia Mining** [85] by Vibha et al., 2008. This article describes a complete system for traffic surveillance. Detection is based on frame differencing. Two different methods are compared for counting objects. It was concluded that a proposed background elimination method performed better than a background registration method.

• **Robust Tracking of Human Motion** [15] by Buzan, 2004. This master thesis considers the problem of tracking people and estimate their 3D trajectories. The trajectories are first estimated by a filter algorithm and then clustered by comparing the similarities to earlier trajectory estimates. Similarities are calculated in terms of distances between projections of the trajectories on the coordinate axes. Via clustering, trajectories can be grouped together.

• **Model-based Segmentation and Tracking of Multiple Humans in Complex Situations** [89] by Zhao, 2003. This PhD thesis concerns some problems when tracking several persons. It focuses on the counting problem, tracking problem and also the motion mode and body posture estimation problem. The body parts of a person is here modelled as ellipsoids.

• **Tracking and Modelling of Team Game Interactions** [62] by Needham, 2003. In this PhD thesis, a multiple object tracker for surveillance of team games is described. Feature descriptors based on multiple resolutions are used to describe the shape of different persons. The movements and interactions between players are learnt online in a behaviour model. The thesis also presents different measures for evaluating the performance of a tracking system.

• **Design and Implementation of People Tracking Algorithms for Visual Surveillance Applications** [79] by Siebel, 2003. This PhD thesis describes the “Reading people tracker”, a module in an integrated visual surveillance system. It was developed for many years. The thesis describes the tracker and also the importance of maintainability of the program code. The human model used in the tracker is defined in the image plane.

• **Recognizing and Tracking Human Action** [82] by Sullivan and Carlsson, 2002. In this paper, recognition of human action is performed by calculating point correspondences between the current frame and a frame in a set of key frames. A tennis game is here given as example in which a forehand pose is recognized.

• **Tracking and Modeling People in Video Sequences** [68] by Plankers and Fua, 2001. In this paper, the authors propose a 3D human body model that can be used for tracking. By having restrictions on what motions can be performed, improved performance and robustness against false detections are achieved. The subject is tracked using stereo sensors and fitted to the model.
• **A Novel Method for Tracking and Counting Pedestrians in Real-Time Using a Single Camera** [55] by Masoud and Papanikolopoulos, 2001. Here, a real-time system for detecting, tracking and counting pedestrians using blob models is proposed. A blob may correspond to one or more pedestrians. Depending on the scenario, either blobs or pedestrians are tracked. The classification algorithm used, here assumes that all moving objects are either pedestrians or blobs.

• **Human Tracking in Multiple Cameras** [44] by Khan et al., 2001. A simple approach is here described to synchronize tracking over multiple cameras. It is achieved by comparing the bounding box of detected persons in each camera with the limiting lines of the field of view.

• **Probabilistic Tracking and Reconstruction of 3D Human Motion in Monocular Video Sequences** [78] by Sidenbladh, 2001. In her PhD thesis, Sidenbladh investigates the general problem of human tracking and gives a solution to it without making any assumption on the motion model or appearance of the person.

• **Vision-Based Sensor Fusion for Active Interfaces: H.O.T. - Human Oriented Tracking** [27] by Grange, 2000. This thesis describes a combined tracker and activity recognition system based on fusion of data from different visual sensors. A tracked person is represented by a spherical head and two quadratic hands due to the limited computer capacity.

• **Tracking Human Motion Using Multiple Cameras** [16] by Cai and Aggarwal, 1996. This article deals with tracking in multiple cameras in an indoor environment. It addresses the detection, tracking and matching (of images from different cameras) problems.

General requirements on a human tracking system are discussed in [60].

### 1.5 Outline of the report

In this report, the sensor system is presented shortly in Chapter 2. In Chapter 3, the camera model is introduced. In Chapters 4 and 5, the human motion model and the approach for human detection in the camera images are presented respectively. The applied tracking filters are described in Chapter 6. Theory and results for the simulations are given in Chapter 7, while the results for the collected data are given in Chapter 8. The thesis is summarized in Chapter 9, while discussion of the results and future work is presented in Chapter 10.
Chapter 2

Sensor system

3D camera, depth sensor and range imaging sensor (RIM) are all synonymous for a device that optically captures information about the three spatial coordinates in a scene. In this thesis, the term 3D camera will be used to denote such a device. The increase in computer hardware and development of effective algorithms in recent years have led to 3D camera system solutions where intensity and range images simultaneously are captured at video data rate. This chapter describes different range measurement techniques upon which such a system can be based, and also the specific 3D camera that was used in this thesis work.

2.1 Range measurement methods

Measurements of the range using optics can be obtained according to several different methods. All of them can be divided into three categories: triangulation, interferometry and time-of-flight (TOF). In a triangulation method, two image sensors or an image sensor combined with a light source are used. When their orientation and the distance between them are known, the range to any object in the scene that is visible from both points can be calculated via geometry. For example stereo vision systems utilizes triangulation.

Interferometry methods determine the range by superposition and correlation of two coherent light waves; one that is redirected to the sensor from the light source via a beam splitter and one that comes from the reflections of the target. The range information is then obtained from the phase information of the resulting correlogram by using phase shift algorithms. This technique can for example be used in synthetic aperture radar (SAR) systems when reproducing 3D terrain data [36]. It is then called interferometric synthetic aperture radar (InSAR).

The TOF methods determine the range by measuring the time it takes for a light wave to reach the target and reflect back to the detector. Assuming the detector to be located at the same spot as the light, the range to target, \( d \), can be calculated from the relation \( d = c \cdot t/2 \), where \( c \) is the speed of light in the medium (which is known) and \( t \) is the traveling time for the light wave (back and forth).
10 Sensor system

The camera used in this thesis is a TOF camera. Hence, the triangulation and interferometry methods will not be dealt with any further in this thesis. For a more thorough presentation of different range measurement concepts, see for example [59].

2.2 FLASH sensor

The FLASH sensor (see Figure 2.1) from Advanced Scientific Concepts (ASC)\(^1\) is currently one of the most advanced 3D imaging cameras. It can measure very large ranges and store information from different depths in the scene. A data sheet is provided in Table 2.1. The system consists of a laser beam emitting part, a beam receiving part and a processing unit that controls the interaction between those parts. As it measures the time of a travelling wave produced by a laser and detects it using techniques from the radar\(^2\) field, this combination is sometimes called laser radar. However, this term is somewhat misleading since radar only concerns electromagnetic waves in a certain spectrum (from about 3 MHz to over 60 GHz [53]). The more correct terms are light detection and ranging (LIDAR) and laser detection and ranging (LADAR) [38].

![Advanced Scientific Concepts’ FLASH sensor.](image)

The processor controls when and for how long the beam should be sent and how the receiver should behave. For each beam that is sent, the receiver captures its return with the help of waveform samplers (one for each receiver element). The waveform sampler stores the 20 latest arrived intensities of the returning wave in so called bins. The sampling period depends on the speed of the processing unit. With a processor speed of 430 MHz, the sampling time is 2.33 ns. Assuming the

---

\(^1\)http://www.advancedscientificconcepts.com/

\(^2\)Originally short for Radio Detection and Ranging [53]
medium to be air, this corresponds to a spatial difference of 35 cm between two consecutive bins.

The TOF is measured from the moment the pulse is sent using a high-accuracy timer. The criterion for when to stop measuring depends on in what mode the camera operates. Two modes are available; sular mode and hit mode (also called stop mode).

When operating in hit mode, each unit cell (circuitry that controls a pixel) independently searches for the range. It does so by monitoring the return signal amplitude and compare it to a threshold. The threshold is the same for all pixels and is chosen so that important reflections can be detected. If the threshold is too low, the range will be calculated based on for example returns from disturbances in the air. With a too high threshold, only the most reflecting objects will be detected. When the threshold is met, the start of the returning wave is considered to be found and the time to this first sample of the wave is stored. The corresponding bin is cleared to indicate where the threshold was met. After that, the waveform sampler captures another 19 returns and then it stops.

In sular mode, no amplitude threshold is used. Instead, a maximum range is set. The waveform sampler runs until the corresponding time has elapsed and then the current bin is cleared to indicate where the maximum range was reached.

The hit mode is most useful when searching for a target at unknown range in an open scene, while, on the other hand, the sular mode is most suitable when the target is known to be in a certain range interval. It can for example be used when searching for a target hiding behind a window. Given a good range interval to search for behind the window, the sular mode will find the target and provide a good range estimate. If instead the hit mode was used, the range would be calculated to the reflecting glass of the window, thus giving no information about the range to the real target. For more information about the read-out circuitry, consider [48].

After having registrered the intensities and the times, the ranges can be extracted by using a peak search algorithm. That is because the top value of the returned wave is considered to come from the object in the studied area. If there are more than one object in the area, the most reflecting one is used for the range calculation in the pixel. An example of a range image captured with the FLASH sensor is given in Figure 2.2. It shows a person walking at a range of approximately 13 m.
Table 2.1. Data specification for the sensor system. Values taken from [48] and ASC’s homepage. Lengths that were originally given in feet and inches, have here been converted to the metric system.

<table>
<thead>
<tr>
<th>Quantity</th>
<th>Symbol</th>
<th>Value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensor size</td>
<td></td>
<td>128 × 128</td>
<td>pixels × pixels</td>
</tr>
<tr>
<td>Pixel size/pitch</td>
<td>$K_u, K_v$</td>
<td>100</td>
<td>µm</td>
</tr>
<tr>
<td>Focal length</td>
<td>$f$</td>
<td>85</td>
<td>mm</td>
</tr>
<tr>
<td>Laser wavelength</td>
<td>$\lambda$</td>
<td>1574</td>
<td>nm</td>
</tr>
<tr>
<td>Processor speed</td>
<td>$f_{CPU}$</td>
<td>430</td>
<td>MHz</td>
</tr>
<tr>
<td>Maximum frame rate</td>
<td>$f_{s,max}$</td>
<td>30</td>
<td>Hz</td>
</tr>
<tr>
<td>Minimum range</td>
<td>$r_{min}$</td>
<td>3.05</td>
<td>m</td>
</tr>
<tr>
<td>Maximum range</td>
<td>$r_{max}$</td>
<td>1.52</td>
<td>km</td>
</tr>
<tr>
<td>Range precision</td>
<td>$r_{pres}$</td>
<td>±7.62</td>
<td>cm</td>
</tr>
</tbody>
</table>

Figure 2.2. Range image from the FLASH sensor. Values are presented in meters.
Chapter 3

Camera model

In order to track a target in world coordinates, a camera model is needed that can relate them to the sensor coordinates. This chapter introduces the 3D pinhole camera model and discusses its assumption of no lens.

3.1 Pinhole camera

A pinhole camera is the most primitive type of camera. Ideally, it consists of a black box with an infinitely small opening on one of the sides. Light from the surroundings enters through the opening (called aperture) and is projected on the opposite side. No lens is used and hence the angles are preserved. The situation is depicted in Figure 3.1.


A model based on this scenario is called the *pinhole camera model*. The geometry of the scenario is illustrated in Figure 3.2. The point $Q$ in the world is mapped...
to the point $P$ in the image plane $I$ via the optical center (the aperture) $O$. The distance from the optical center to the image plane is called the \textit{focal length}.

In order to describe the positions of the points in the world, a right-handed Cartesian coordinate system ($\hat{X}, \hat{Y}, \hat{Z}$) is introduced. It has its origin in the optical center, with the $\hat{Z}$-axis aligned with camera direction and the $\hat{Y}$-axis orthogonal to the $\hat{Z}$-axis, pointing towards the sky.

Another coordinate system, $(\hat{u}, \hat{v}, \hat{r})$, forms the basis for the sensor coordinates. Since a stationary camera is used, the image plane is chosen to be parallel to the $XY$-plane. However, the $\hat{r}$-axis varies with the pixel location and is parallel with a ray passing both the pixel and the optical center. Hence, it is only orthogonal to the image plane in the midpoint of the sensor.

![Figure 3.2. Geometry of the pinhole camera.]

The mapping from world coordinates to sensor coordinates are given by

$$u = f \frac{X}{K_u Z} + u_0$$  \hspace{1cm} (3.1a)

$$v = f \frac{Y}{K_v Z} + v_0$$  \hspace{1cm} (3.1b)

$$r = \left(1 + \frac{f}{Z}\right)\sqrt{X^2 + Y^2 + Z^2},$$  \hspace{1cm} (3.1c)

where $K_u$ and $K_v$ are the conversion factors from pixels to meters, $u_0$ and $v_0$ are the pixel coordinates for the center of the image and $f$ is the focal length [39, 61]. The factor $\left(1 + \frac{f}{Z}\right)$ in the expression for $r$ describes the fact that the light travels both through the air and a short angular-dependent distance in the camera.
3.2 Path of the light

With an external lightsource, as is the case for the sensor system described in Section 2.2, the travelled light distance is not exactly given by $c \cdot t/2$. Instead, it varies with the pixel location due to the angle difference between the emitted ray and the reflected (and captured) ray [52, Figure 5.5]. However, in this thesis, it is assumed that the light emitter is located at the optical center of the sensor system, so that the angle difference is zero and no compensation is needed. This is a very good approximation when the range is large.

3.3 Field of view

Field of View (FOV) refers to the angle under which the camera observes the world. It can be defined for the horizontal, vertical or diagonal view of the image plane. With the notation from Figure 3.2, the horizontal FOV $\alpha_h$ can be calculated according to:

$$\alpha_h = 2 \arctan \left( \frac{K_u u_0}{f} \right)$$  \hspace{1cm} (3.2)

while the diagonal FOV can be calculated as:

$$\alpha_d = 2 \arctan \left( \frac{\sqrt{(K_u u_0)^2 + (K_v v_0)^2}}{f} \right).$$  \hspace{1cm} (3.3)

Hence, FOV depends on both the sensor size and the focal length. The camera can be classified into different categories based on the measure [24, page 207]:

- Narrow, $10^\circ - 40^\circ$
- Normal, $45^\circ$
- Wide, $50^\circ - 80^\circ$

For the FLASH sensor, the sensor size and the focal length was given in Table 2.1. Based on that data, the horizontal (or vertical) FOV can be determined to be $\alpha_h = 8.6^\circ$ and the diagonal angle of view $\alpha_d = 12.2^\circ$. Hence, the camera has narrow FOV.

3.4 Validity of the pinhole camera model

Real pinhole cameras are not very useful. Because of the small aperture, only a very small amount of light can enter the camera house and this means that it takes time to produce a bright image. If the aperture is made bigger, more light can enter but the image will become fuzzier and the relatively short time that is saved cannot motivate the loss in image quality.

Instead, a lens is normally placed in the aperture. The lens is an optical device that focuses light from different parts of the world into an area in the image plane.
Each point in the image is therefore exposed to more light, the image will be clear and image capture will be done faster. If a lens is used, the assumption for the pinhole camera model becomes invalid since the rays are bent differently for different pixels. For narrow and normal FOV however, the model is still applicable and accurate enough.

Calibration of the camera has been studied in [48] but results were not available for the work of this thesis. One approach to calibration is described in [88]. Publications on range camera calibration include for example [50] and [42].
Chapter 4

Modeling human motion

There are many different approaches to tracking of humans. In some applications, a full 3D model of the body is developed while in other applications, a single point is used to model the body. Since the focus of the thesis is on comparing 2D and 3D trajectory estimation, a single point is considered sufficient. In this chapter, the constant velocity model and two noise discretization methods are presented.

4.1 Overview

Modeling human motion is far from trivial for several reasons. One of them is that the human body consists of several parts leading to an ambiguity in which one to track. The parts all have different motion patterns that will also generate different trajectories in the world coordinates. Some of the trajectories are therefore harder to describe than other. Depending on the application, different strategies for modeling can be used. In [72], the body is modelled by 14 cylinders and all parts are tracked. The easiest parts to describe are the head and upper body since their positions vary the least around the body’s center of mass. For implementation reasons, only the head position is considered in this thesis. That is, a person is assumed to be fully described by the coordinates of his/her head.

Even though the head is considered to be one of the more easy parts to track, its movements are still hard to predict. The person can choose to move the head in any direction and it can be done intentionally or unintentionally. The movement can for example be initiated by the will of going somewhere or it can be a sudden reaction to something that happens in the environment (see for example [1] for human interaction with other humans and objects). Describing all possible events are practically impossible and hence there is need for simplifications.

A common assumption is that the person moves in the same direction with the same speed all the time. Small speed variations are accounted for by assuming an acceleration that can be described as zero-mean white noise. This is called a constant velocity (CV) model. The approximation will probably be sufficient most of the time, but there will be problem for example if the person starts to move
faster or change direction. (See [73] for a system that recognizes different modes such as walking, running, biking etc.)

If such maneuvers are initiated, the model needs to be changed accordingly. There are at least two ways in how to do this. One is to change the parameters of the current model dynamically to better describe the new behaviour (see i.e. [31] and [63]). Another way is to use a multiple model (MM) algorithm where several models (one for each mode) are run in parallel and then use the currently best model. See [46] for advice on choosing between a single model and an MM approach. The problem of deciding when to change the model occurs in both approaches and there are different suggestions on how to choose a decent change detection strategy (see e.g. [22, 12]).

4.2 Choice of coordinate system

The constant velocity model for human motion is best described in the world fixed Cartesian coordinates since many of the movements consist of walking along a straight line in the world. Another reason for tracking in world coordinates is that it facilitates integration with other sensors. For example, in a surveillance system, people may be tracked from different locations and angles. By using the same states, measurements from different sensors may easily be considered when estimating the motion.

4.3 Linear models

A linear model is a model where the next state is a linear function of the current state. It can be written on the form

\[ x_{k+1} = F_k x_k + N_k v_k \]  \hspace{1cm} (4.1)

where \( x_{k+1} \) is the next state, \( F_k \) is the state transition matrix, \( x_k \) is the current state, \( N_k \) is the noise gain and \( v_k \) is the process noise.

Different linear models exist as discussed above. Here, the linear models are restricted to constant velocity models. For such models, the state vector in one dimension is given by \( x_k = [X, \dot{X}]^T \), where \([\cdot]^T\) denotes matrix transpose, and the state transition matrix is given by \( F_k = \begin{bmatrix} 1 & T_s \\ 0 & 1 \end{bmatrix} \), where \( T_s \) is the sampling time. In [4], two interpretations of the noise are given: White noise acceleration model and Piecewise constant white acceleration model. They are described below.

4.3.1 White noise acceleration model

This model assumes that there is an underlying continuous model that describes the process. The continuous time acceleration is considered to be zero-mean white noise. After discretization, the motion in one dimension is described by Equation (4.1) with \( N_k = [1, 1]^T \). The covariance of the process noise in one dimension is then:
4.4 Choice of process model

\[ Q = E\{v_k v_k^T\} = \begin{bmatrix} \frac{1}{3} T_s^3 & \frac{1}{2} T_s^2 \\ \frac{1}{2} T_s^2 & T_s \end{bmatrix} \tilde{q} \]  \hspace{1cm} (4.2)\]

where \( \tilde{q} \) is the process noise intensity for the continuous time model. \( \tilde{q} \) can be chosen by utilizing that \( Q_{22} = \sqrt{\tilde{q} T_s} \) approximately describes the velocity change during one sample period \( T_s \).

### 4.3.2 Piecewise constant white acceleration model

This model is based on the assumption that the additivive noise \( v_k \) that enters the discrete system is a constant acceleration during the \( k \)-th sampling period. In this case, the noise gain is \( N_k = \begin{bmatrix} \frac{1}{2} T_s^2 \\ T_s \end{bmatrix} \) for one dimension. The covariance of the process noise in one dimension is then:

\[ Q = E\{N_k v_k v_k^T N_k^T\} = N_k \sigma_v^2 N_k^T = \begin{bmatrix} \frac{1}{4} T_s^4 & \frac{1}{2} T_s^3 \\ \frac{1}{2} T_s^3 & T_s^2 \end{bmatrix} \sigma_v^2 \]  \hspace{1cm} (4.3)\]

where \( \sigma_v^2 \) is the process noise variance. \( \sigma_v^2 \) should be of the order of the maximum acceleration of the process.

### 4.4 Choice of process model

When only the position is measured, as is the situation with a 3D camera, the dynamics of the process is best captured if two states are used [11, page 195]. This means that the state vector should consist of the position and velocity of the tracked object. The 3D state vector and corresponding state transition matrix are given by Equation (4.4) and Equation (4.5).

\[ x_k = [X_k, \dot{X}_k, Y_k, \dot{Y}_k, Z_k, \dot{Z}_k]^T \]  \hspace{1cm} (4.4)\]

\[ F_k = \begin{bmatrix} 1 & T_s & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & T_s & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & T_s \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \]  \hspace{1cm} (4.5)\]

The covariance matrix for the process noise has the following structure:

\[ Q = \begin{bmatrix} p_{XX} & p_{X\dot{X}} & p_{XY} & p_{X\dot{Y}} & p_{XZ} & p_{X\dot{Z}} \\ p_{\dot{X}X} & p_{\dot{X}\dot{X}} & p_{\dot{X}Y} & p_{\dot{X}\dot{Y}} & p_{\dot{X}Z} & p_{\dot{X}\dot{Z}} \\ p_{YY} & p_{YY} & p_{YY} & p_{YY} & p_{YZ} & p_{YZ} \\ p_{\dot{Y}X} & p_{\dot{Y}X} & p_{\dot{Y}Y} & p_{\dot{Y}Y} & p_{\dot{Y}Z} & p_{\dot{Y}Z} \\ p_{ZZ} & p_{ZZ} & p_{ZZ} & p_{ZZ} & p_{ZZ} & p_{ZZ} \\ p_{\dot{Z}X} & p_{\dot{Z}X} & p_{\dot{Z}Y} & p_{\dot{Z}Y} & p_{\dot{Z}Z} & p_{\dot{Z}Z} \end{bmatrix} \]  \hspace{1cm} (4.6)\]

where \( p_{ij} \) denotes the covariance of the variables \( i \) and \( j \).
The elements of this matrix were chosen differently depending on the sequence being studied. The values are given in connection with the presentation of each scenario.
This chapter describes the principle of the head detection process. In order to detect a human head in an image, some preprocessing is needed. It might be possible to directly identify different features but it is a rather tedious task. Since there is a sequence that will be used for tracking, it is a good idea to first detect moving parts (the foreground) in the images and thereafter detect human heads in those parts. These steps will be further described below.

5.1 Foreground computation

The foreground mask is here obtained by a process called background subtraction, which is a common method when working with a stationary camera. In that case, the geometry of the background remains the same which facilitates segmentation of an image into background and foreground. Principally, the foreground is then the difference between the current image and the background. However, there are some problems hidden in the description above.

A general background subtraction algorithm consists of several steps (see [74] or [7] for an overview) of which two are the most important ones; background modeling ([23, 87, 77, 69]) and foreground detection ([20, 67, 84]). Many different background subtraction algorithms exists (see e.g.[56]), of which some are recursive while others are non-recursive. A recursive algorithm has one background model (summarizing all earlier information about the background) that is updated for each frame, in contrast to the non-recursive case where the background is estimated from a set of frames. In this thesis, a recursive algorithm with the background represented by a Gaussian distribution for each pixel, is used. That is, each pixel holds its mean value and its background variance. Such a model is recommended by [6]. The algorithm used in this thesis was suggested in [86]. It was reimplemented in [51] but modified to also deliver a range uncertainty by utilizing the 3D info. It is presented in Algorithm 1. The threshold value in this algorithm should be chosen so that shifting illumination does not count as foreground. It can be calculated as the variance over frames with only background or be chosen as a constant.

When the background has been estimated, a foreground probability image is
Algorithm 1 Background modeling

1. Initialization:
   Calculate a mean value background and variation from some frames with only background, and use it as the initial state of the background model.

2. Update:
   3. for all frames do
      4. for all pixels in the frame do
      5. Estimate the deviation from the background model.
      6. if deviation < threshold then
         7. Update the background towards the current pixel value.
      8. else
         9. Do not update the background.
      10. end if
      11. end for
      12. end for

calculated based on the current frame. It is then thresholded to create a binary image for the head detection.

For a 3D camera, Algorithm 1 might be based on either the reflected intensity or the calculated range. Both methods have been tested in this thesis work and they perform differently depending on the intensity of the emitted laser pulse. For a low laser beam intensity, the intensity method works best while for a high laser beam intensity, the range method works best. Examples that illustrate the two methods are provided in Figure 5.1. An early approach to combine the two methods were presented in [58].

By using a laser, the illumination on the surface can be controlled and hence the algorithm is less sensitive to for example outdoor conditions (sun visibility influences on the result). In [8], it was concluded that background segmentation is insensitive to different lighting conditions when using active illumination.

Foreground detection with non-steady background has been studied in [71, 35, 76]. As mentioned earlier, it was also demonstrated in [66] that background modelling is unnecessary when segmenting in range images. However, in this thesis, the range images were affected by more noise than in their study so that segmentation by thresholding the range image was assumed to be less effective.

5.2 Head detector

The head detection algorithm, Algorithm 2, tries to find the top of the head as well as an estimate of the range to it. Head candidates are found (in step 2) by comparing the image with a mask. For each area in the image, the midpoint is given a score depending on how well the mask fits this area. This procedure is implemented via convolution. Convolution of the binary input image $I$ with the kernel $K$ produces the resulting image $R$ according to Equation (5.1) [18].
5.2 Head detector

(a) Input to the segmentation algorithm. Image with range values given in meters.
(b) Input to the segmentation algorithm. Image with intensity values.

(c) Background model after range segmentation. Range values given in meters.
(d) Background model after intensity segmentation.

(e) Foreground probability after range segmentation.
(f) Foreground probability after intensity segmentation.

Figure 5.1. Segmentation based on range (left) and on intensity (right).

\[ R(u, v) = (I \ast K)(u, v) = \sum_{\alpha=\infty}^{\infty} \sum_{\beta=\infty}^{\infty} I(\alpha, \beta) \cdot K(u - \alpha, v - \beta) \quad (5.1) \]
The kernel must be chosen as the $180^\circ$ rotation of the mask to compensate for the rotation in the expression for the convolution. Based on this, the kernel is chosen as:

$$K = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & -1 \\ -1 & -1 & -1 \\ -1 & -1 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

The ones corresponds to pixels in the input image that must contain foreground while the negative ones corresponds to pixels that must not contain foreground. A zero in the kernel means that it does not matter if the pixel is foreground or not. This means that in areas that matches this kernel, the midpoint pixel in the resulting image gets the value 7. By inspection of the kernel, it can be seen that such a high value can only be obtained in the top of connected regions in the image, such as heads. The template can be seen in Figure 5.2. When a head candidate has been found, its range is estimated by taking the median over a foreground area in the vicinity of the candidate. Then, based on this range, a bounding box is estimated around the human by making the assumption that (s)he is of average length. Within the new bounding box, the fillrate is calculated and the presence of a human can be confirmed or rejected. Finally, an average range value $r$ and range variation $\sigma_r$ are calculated from the set of foreground pixels in the bounding box.

![Figure 5.2](image.png)

*Figure 5.2.* Template for the head detector. The white region corresponds to pixels that must be foreground. The black region are pixels that must not be foreground. In the gray regions, it does not matter if it is foreground or background.
Algorithm 2 Head detection

Require: Binary foreground image
1. Remove areas in the image that are too small to possibly represent a human.
2. Locate head candidates by searching in the top of all connected areas.
3. for all head candidates do
   4. Calculate an estimate of the range to the candidate.
   5. Estimate a human surrounding bounding box.
   6. if large part of the bounding box deviates from background then
      7. Consider a human to be found.
      8. Reject other head candidates within the bounding box.
   9. end if
10. end for
11. return remaining head coordinates and variation in range
Chapter 6

Tracking algorithms

The measured head position is in several ways affected by noise and other disturbances. With a tracking algorithm and some assumptions about the noise, a better estimate of the position can be obtained. If no measurement is available, a prediction of the position can be provided. An overview of state estimation in target tracking applications is given in [17]. Many different filters exist and this chapter describes the ones that have been tested in this thesis.

6.1 Kalman filter

The Kalman Filter (KF) is a linear filter that has its name after its inventor Rudolf Emil Kalman. Kalman published his discovery in 1960 [41]. Linear filters are filters where the estimate (the new state) is a linear function of the observable variables (i.e. previous states and measurements). For the KF, the measurement $y_k$ is assumed to be the sum of a linear combination of the current state $x_k$ and an additive measurement white noise term $w_k$. The coefficients in the linear combination are given by the measurement matrix $H_k$ and the full term $H_k x_k$ describes what measurement is expected if the states are correct. The measurement noise models the errors in the measurements as random variables. It has the covariance matrix $R_k$. Equation (6.1) summarizes the requirements on the model of the system for the KF. The state transition equation was described in Section 4.3. KF assumes that also $v_k$ is white noise. In the KF model assumption, it can also contain a linear combination $G_k$ of the control signal $u_k$.

$$
\begin{align*}
  x_{k+1} &= F_k x_k + G_k u_k + N_k v_k \\
  y_k &= H_k x_k + w_k \\
  \text{Cov}(v_k) &= Q_k \\
  \text{Cov}(w_k) &= R_k
\end{align*}
$$

(6.1)

The KF algorithm can be separated into two parts; the time update part and the measurement update part. It is outlined in Algorithm 3. The measurement update uses the Joseph form covariance update, [4], as it preserves non-negativity of the covariance matrix. Under the premises given above, the KF is proven to be
the best linear estimator [4]. If also the Probability Density Function (PDF) of the noises and the initial state error are Gaussian functions, then the KF is also the optimal Minimum Mean Square Error (MMSE) state estimator [4].

Algorithm 3 Kalman filter [4]

1. Time update:
   \[ \hat{x}_{k+1|k} = F_k \hat{x}_{k|k} + G_k u_k \]
   \[ P_{k+1|k} = F_k P_{k|k} F_k^T + N_k Q_k N_k^T \]

2. Measurement update:
   \[ \hat{x}_{k+1|k+1} = \hat{x}_{k+1|k} + K_{k+1} \epsilon_{k+1} \]
   \[ P_{k+1|k+1} = (I - K_{k+1} H_{k+1}) P_{k+1|k} (I - K_{k+1} H_{k+1})^T + K_{k+1} R_{k+1} K_{k+1}^T \]

where \( \epsilon_{k+1} \) is the measurement residual, or innovation, defined as
\[ \epsilon_{k+1} = y_{k+1} - H_{k+1} \hat{x}_{k+1|k} \]

and \( K_{k+1} \) is the Kalman gain, defined as
\[ K_{k+1} = P_{k+1|k} H_{k+1} (H_{k+1} P_{k+1|k} H_{k+1}^T + R_{k+1})^{-1} \].

With measurements given in sensor coordinates and state expressed in Cartesian coordinates, the function relating the measurement to the state will be nonlinear and hence the system cannot be modelled by Equation (6.1). Two different modifications of the KF that aims to solve this problem are presented in the following sections.

6.2 Extended Kalman filter

The Extended Kalman filter (EKF) provides a way to track in Cartesian coordinates with sensor measurements by allowing the measurement function to be nonlinear. In contrast to the KF, the EKF is not an optimal filter. It is only an approximation of the optimal nonlinear filter (which requires the full PDF to be estimated). However, in [75], the EKF was compared to the Best Linear Unbiased Estimator (BLUE) [90] filter and shown to perform up to only 4% worse in simulations. The state and measurement predictions for the EKF are based on Taylor series expansions of the nonlinear state propagation and measurement functions respectively. The EKF assumes that the system can be described by the model in Equation (6.2):

\[
\begin{align*}
x_{k+1} &= f(k, x_k, u_k, v_k) \\
y_k &= h(k, x_k, w_k) \\
\text{Cov}(v_k) &= Q_k \\
\text{Cov}(w_k) &= R_k
\end{align*}
\]  

(6.2)

with the same notation as for the KF, except that the state transition matrix and measurement transition matrix have been replaced with nonlinear functions
of state, noise and control signals. The EKF is outlined in Algorithm 4. As can be seen, it reduces to the standard KF if \( f(\cdot) \) and \( h(\cdot) \) are linear functions.

The measurement function that returns the sensor coordinate representation of the Cartesian state coordinates are given by Equation (3.1). The algorithm also requires the evaluation of two Jacobians. Since the process model is linear (and \( F_k \) trivially given by Equation (4.5)), it is only the Jacobian for the measurement function that needs to be derived. If the index is temporarily skipped, the Jacobian with respect to the state variables \([X, \dot{X}, Y, \dot{Y}, Z, \dot{Z}]^T\) can be expressed as:

\[
H = \begin{bmatrix}
\frac{f}{K_uZ} & 0 & 0 & 0 & -\frac{Xf}{K_uZ^2} & 0 \\
0 & 0 & \frac{f}{K_vZ} & 0 & -\frac{Yf}{K_vZ^2} & 0 \\
(1 + \frac{f}{Z}) \frac{X}{R} & 0 & (1 + \frac{f}{Z}) \frac{Y}{R} & 0 & (1 + \frac{f}{Z}) \frac{Z}{R} & -\frac{f}{Z^2} R & 0
\end{bmatrix}
\]

(6.3)

where \( R \equiv R(X, Y, Z) \equiv \sqrt{X^2 + Y^2 + Z^2} \) is the radial distance from the aperture to the point in the world.

Algorithm 4 Extended Kalman filter [4]

1. Time update:
\[
\hat{x}_{k+1|k} = f(k, \hat{x}_{k|k}, u_{k}, 0) \\
\hat{P}_{k+1|k} = F_k \hat{P}_{k|k} F_k^T + N_k Q_k N_k^T
\]

where \( F_k \) is the Jacobian of \( f \) with respect to \( x \) evaluated at the current time instant: \( F_k = \frac{\partial f(k)}{\partial x} \bigg|_{x=\hat{x}_{k|k}} \)

2. Measurement update:
\[
\hat{x}_{k+1|k+1} = \hat{x}_{k+1|k} + K_{k+1} \epsilon_{k+1} \\
\hat{P}_{k+1|k+1} = (I - K_{k+1} H_{k+1}) \hat{P}_{k+1|k} (I - K_{k+1} H_{k+1})^T + K_{k+1} R_{k+1} K_{k+1}^T
\]

where \( \epsilon_{k+1} \) is the measurement residual, or innovation, defined as \( \epsilon_{k+1} = y_{k+1} - h(k + 1, \hat{x}_{k+1|k}) \),

\( K_{k+1} \) is the Kalman gain, defined as
\[
K_{k+1} = \hat{P}_{k+1|k} H_{k+1}^T (H_{k+1} \hat{P}_{k+1|k} H_{k+1}^T + R_{k+1})^{-1},
\]

and \( H_k \) is the Jacobian of \( h \) with respect to \( x \) evaluated at the current time instant: \( H_k = \frac{\partial h(k + 1)}{\partial x} \bigg|_{x=\hat{x}_{k+1|k}} \).

6.3 Converted Measurement Kalman filter with Debiasing

The Converted Measurement Kalman filter with Debiasing (CMKF-D) is suggested as a better candidate than the EKF in the sense that it guarantees coordinate conversion consistency [5]. This means that the expressions for the mean and
covariance of the measurements are in accordance with the real statistics. Even
the EKF is consistent but only when the errors are small (i.e. the measurement is
close to the predicted value). The errors may grow if the object changes direction
or accelerate much.

In CMKF-D, the measurements are first converted to Cartesian coordinates
and then filtered with the standard KF algorithm (see Algorithm 3). If the orig-
inal measurements are given in the spherical coordinates radius \( r_m(k) \), elevation
\( \phi_m(k) \) and azimuth \( \theta_m(k) \), the transformation to Cartesian coordinates is given
by Equation (6.4).

\[
\eta [y_k] \equiv \begin{bmatrix} r_m(k)\cos\phi_m(k)\cos\theta_m(k) \\
 r_m(k)\cos\phi_m(k)\sin\theta_m(k) \\
 r_m(k)\sin\phi_m \end{bmatrix} = \begin{bmatrix} X \\
 Y \\
 Z \end{bmatrix}
\] (6.4)

When transforming the measurements, the noise will also be transformed and
its distribution will be corrupted. For small angles, these errors may be negligible,
but for larger angles, they will affect the coordinates notably. These conversion
errors can be minimized by debiasing and the converted measurements are then
given by \( y^c_k = \eta [y_k] - \mu_k \), where \( \mu_k \) is the average true converted measurement bias.
The exact expression for this bias, as well as for the corresponding measurement
covariance matrix, is given in [81]. The CMKF-D assumes that the model takes
the following form:

\[
x_{k+1} = F_kx_k + G_ku_k + N_kv_k \\
y^c_k = H_kx_k + w_k \\
Cov(v_k) = Q_k \\
Cov(w_k) = R_k
\]

with the same notation as in Section 6.1 and with \( y^c_k \) calculated as above. In this
case, the measurement matrix becomes particularly simple:

\[
H_k = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\
 0 & 0 & 1 & 0 & 0 \\
 0 & 0 & 0 & 0 & 1 \\
 0 & 0 & 0 & 1 & 0 \end{bmatrix}.
\] (6.5)

6.4 Choice of tracking algorithm

CMKF-D was recommended as the better choice according to [5] as it provides
consistent coordinate conversion for all geometries. Simulations and results were
provided for the case of polar coordinates, showing the good performance. How-
ever, a similar study (of EKF vs. CMKF-D) for the case of spherical coordinates
has, to the author’s knowledge, not yet been made. Admittedly, a comparison of
CMKF-D and variants of EKF was performed for spherical coordinates in [43],
but several separate 2D measurements updates were used to obtain the state es-
timates instead of using the full measurement vector at once. The study showed
that one variant of EKF, called Error model based Modified Extended Kalman
filter (EMEKF), performed better than the other. A similar approach but based
on the 3D sensor coordinates \( u, v \) and \( r \), was presented in [65].
Expressions for transformation of measurements in CMKF-D for spherical measurements were given in [81]. To the best of the author’s knowledge, there is yet no similar expressions published for tracking with 3D sensor coordinates $u$, $v$ and $r$. Hence, the choice fell on the EKF for this thesis. This choice also had the advantage of enabling using the same structure of the tracking algorithm for both 2D and 3D experiments. The CMKF-D method is not applicable for tracking with a 2D camera since the conversion to Cartesian coordinate cannot be performed when the range is unknown. In this thesis, the 2D EKF tracker was realized by removing the last row in the measurement matrix.
Chapter 7

Simulations

In this chapter, predefined trajectories of a moving target are used to evaluate the 2D tracker and the 3D tracker. By having the true trajectory as a reference, it is possible to determine how good the trackers are at estimating it when measurement noise is present. Different performance measures and a specification of what will be tested are also presented in this chapter.

7.1 Overview

Before the chosen filter can be accepted in a real application, it has to be tested on many different datasets. By running the tracking algorithm on sequences from the sensor described in Section 2.2, it is possible to observe how the tracker filters the measured positions. However, this does not imply that the agreement with the true positions are good. What are then the true positions? Since the camera sensor is the subject of evaluation, an independent source must be considered. One option is to measure the position with another sensor and use this as the true value. This requires that the other sensor has a known high accuracy and that it is capable of delivering values at the same speed as the system that is being evaluated. As there were no such sensor available at the time for the data collection, another approach had to be taken, namely simulations. Via simulations, a true trajectory is precomputed and projected on the image plane via a camera model. This model is preferably as accurate as possible. After that, noise is added to emulate the distortions that occur in a real sensor system (camera and head detector). The process may be afflicted with noise in many different combinations and each such combination (true trajectory and noise) is called a realization.

7.2 What to test

The purpose of simulating is to evaluate the ability of the 2D and 3D trackers to estimate the true trajectory of the target. As the human target can undergo many different maneuvers, a selection of representative scenarios have been chosen. Four
different scenarios (to be presented in Section 7.4) will be simulated. For each of
the four scenarios, the capability to predict the trajectory will be investigated
through two studies.

In the first study, data will be provided at all time instants, but two different
levels of measurement noise will be used. In one case, a small noise is modelled
and in the other case, a large noise is modelled. The specific standard deviations
of the noises are presented in Table 7.1. They describe how much errors that are
introduced to the target position measurement by using a sensor system (camera
and lens, range detection, segmentation and head detection). This study aims to
investigate whether the 3D tracker is better than the 2D tracker at filtering the
data or not.

Table 7.1. Setup of measurement noise covariance matrix and periodicity of measure-
ments

<table>
<thead>
<tr>
<th>Quantity</th>
<th>Standard deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\sigma_{v,u,small} )</td>
<td>0.5 px</td>
</tr>
<tr>
<td>(\sigma_{v,v,small} )</td>
<td>0.5 px</td>
</tr>
<tr>
<td>(\sigma_{v,r,small} )</td>
<td>0.03 m</td>
</tr>
<tr>
<td>(\sigma_{v,u,large} )</td>
<td>2 px</td>
</tr>
<tr>
<td>(\sigma_{v,v,large} )</td>
<td>2 px</td>
</tr>
<tr>
<td>(\sigma_{v,r,large} )</td>
<td>0.3 m</td>
</tr>
</tbody>
</table>

Periodicity of measurements \{1, 5, 10, 20\}

In the second study, the performance of the filter will be evaluated when the
frequency of data delivery is changed. With a low frequency of data delivery, the
predictions will become more uncertain than if the frequency is high. This may
for example affect the behaviour of the tracker during target accelerations. For
this study, the smaller measurement variance values according to Table 7.1 will be
used.

7.3 Measures

A variety of measures are available for comparing the filters. If the KF is applicable
to the system, then the state covariance matrix can be used to calculate a measure
for the estimation error. By taking the square root of the trace of the covariance
matrix, a scalar measure is given; the square root of the mean value of the norm
of the estimation error [30]:

\[
\sqrt{tr(P[k])} = \sqrt{E\left\{ \|x^0_k - \hat{x}_k\|^2 \right\}} \tag{7.1}
\]

where \(\hat{x}_k\) is the estimated value and \(x^0_k\) is the true value, both at time instant \(k\).
Equation (7.1) could be interpreted as the standard deviation of the estimation
error. When many realizations of the same process are known, that measure can
be approximated by the Root Mean Square Error (RMSE). It is defined as
7.4 Results

\[\text{RMSE}[k] = \sqrt{\frac{1}{M} \sum_{j=1}^{M} \| x_0^k - \hat{x}_k^{(j)} \|_2^2} \quad (7.2)\]

Equation (7.2) is preferably used in simulations. If only one value for the full sequence is desired, instead the RMSE definition according to Equation (7.3) can be used.

\[\text{RMSE} = \sqrt{\frac{1}{k_{tot}} \sum_{k=1}^{k_{tot}} \left( \frac{1}{M} \sum_{j=1}^{M} \| x_0^k - \hat{x}_k^{(j)} \|_2 \right)^2} = \sqrt{\frac{1}{k_{tot}} \sum_{k=1}^{k_{tot}} (\text{RMSE}[k])^2} \quad (7.3)\]

where \(M\) is the total number of realizations, \(k_{tot}\) is the length of each realization and \(\hat{x}_k^{(j)}\) is the estimation of the true value based on realization \(j\).

All the presented measures assume that the distribution of the state can be fully described by the mean and the variance. Otherwise it might be necessary to calculate the full PDF.

In this thesis, the measures will be used coordinate-wise. This enables studies of each coordinate independently even though it does not fully comply with Equation (7.2) (where the norm is taken over all coordinates at once). The coordinate-wise RMSE measures are indexed with the coordinate, i.e. \(\text{RMSE}_X\) denotes the RMSE measure for coordinate \(X\).

The equation also implies that the norm must be taken over the state variables. Here, however, both the true state and the true measurements are given so that any of them can be evaluated with the measure. The comparison of the 2D tracker and 3D tracker can be performed in either world coordinates or in sensor coordinates. With the world coordinates \((X, Y, Z)\) defined as in Section 3.1, the \(X\) and \(Y\) values will depend on the range, \(r\), to the target. If there is an error in the range estimate, all three coordinates will be afflicted with errors. If instead the sensor coordinates \((u, v, r)\) are used, only the range estimate is afflicted. Then the ability of the 2D tracker to track in image coordinates still can be shown. This was chosen as the main method in this thesis work. The RMSE values for the two algorithms can be compared via a hypothesis test, see [3].

7.4 Results

A selection of interesting scenarios has been chosen to test the 3D camera against the 2D camera. These are presented below.

The first scenario consists of pure translation along a straight line that is parallel to and far away from the image plane. This is the configuration that probably will best suit the conventional camera since the motion can be fully observed. However, it cannot be observed that there is no motion in range.

The second scenario also consists of pure translation, but here the range will change. The target will start close to the camera in the left part of the scene and
end up further away from the camera in the right part of the scene. This may introduce problems for the conventional 2D tracker as it may treat the vanishing target as a decrease of absolute speed. Without range or target dimension information, it cannot determine whether the target approaches or recedes from the camera. It can estimate a lower speed in the image coordinate, but it cannot know if it is due to changed target direction or a lower speed in horizontal direction only.

The third scenario consists of a target that moves through a ground trajectory shaped like the number ‘8’. The camera is fixed at a location near the lower part of the ‘8’, looking at it from a small angle. In this case, the 2D camera may have even bigger problems as the range and direction of the target vary several times.

The fourth scenario consists of a target, moving at ground level in a trajectory shaped like two adjacent rhombuses. This will test the trackers’ performance when sudden direction changes occur.

In all scenarios, the $Y$-coordinate of the head was modelled as a constant, as it is common that the head does not move very much up and down. It was chosen to be $Y = 1$ m in all scenarios except scenario two, where it instead was chosen as $Y = 0.2$ m. For the first two scenarios, the process was modelled in accordance with the piecewise constant noise acceleration model (see Section 4.3.2) as the CV assumption was very well fulfilled. For the more complicated scenarios three and four, diagonal process noise covariance matrices were chosen due to the geometry of the trajectories, see Section 7.4.3. The matrices were chosen so that the RMSE measure was minimized for each sequence. This was done so that the comparison between the 2D tracker and 3D tracker should not depend on bad process modelling. Hence the results were expected to be somewhat optimistic.

### 7.4.1 Scenario 1: Straight line, parallel to the image plane

In this simulation, a target with constant velocity moves at a range of 1000 m from the camera. The range was chosen so that the motion of the target could be maintained for many samples, without disappearing out of camera view. The true accelerations (constants) were used to calculate the process noise covariance matrix for the sequence according to the piecewise constant white acceleration model (see Section 4.3.2). As the true accelerations were zero in this scenario, $\sigma_v^2$ was chosen very small, i.e. 0.0001 m/s$^2$. The position was initialized to the true position and the corresponding variances in the process noise matrix set to small values. The velocity was initialized to zero and the corresponding variances set to values ten times higher than the position variances to express the larger uncertainty in these values (because there is no velocity sensor).

In Figure 7.1, one realization of the true and estimated range trajectory are plotted for the 2D and 3D case respectively. Here, the larger measurement noise setup from Table 7.1 is used. One can see that even in this simple scenario, the 2D tracker will have problems estimating the range correctly, while it is hard to see any errors at all for the 3D tracker. This is further confirmed when comparing the scalar RMSE measure for a sequence, see Table 7.2. The RMSE values are calculated for 100 realizations of the sequence, using both small and large mea-
Figure 7.1. Scenario 1: True and estimated distances when tracking with large measurement noise (see Table 7.1).

In this scenario, where the process noise is modelled very small, the sensor RMSE values will mainly reflect the capabilities of the tracker to filter out the signals from measurement noise. Thus, if the RMSE values are smaller than the corresponding standard deviations, then the tracker helps to improve the knowledge of the true positions.

Table 7.2. Scenario 1: RMSEs for data with two different measurement noise levels for the 2D and 3D case. RMSE calculations based on 100 realizations.

<table>
<thead>
<tr>
<th>Measure</th>
<th>2D, small R</th>
<th>3D, small R</th>
<th>2D, large R</th>
<th>3D, large R</th>
</tr>
</thead>
<tbody>
<tr>
<td>$RMSE_X$</td>
<td>0.292 m</td>
<td>0.275 m</td>
<td>1.082 m</td>
<td>1.065 m</td>
</tr>
<tr>
<td>$RMSE_Y$</td>
<td>0.047 m</td>
<td>0.043 m</td>
<td>0.132 m</td>
<td>0.139 m</td>
</tr>
<tr>
<td>$RMSE_Z$</td>
<td>3.431 m</td>
<td>0.017 m</td>
<td>3.261 m</td>
<td>0.072 m</td>
</tr>
<tr>
<td>$RMSE_u$</td>
<td>0.231 px</td>
<td>0.235 px</td>
<td>0.905 px</td>
<td>0.909 px</td>
</tr>
<tr>
<td>$RMSE_v$</td>
<td>0.040 px</td>
<td>0.037 px</td>
<td>0.113 px</td>
<td>0.118 px</td>
</tr>
<tr>
<td>$RMSE_r$</td>
<td>3.433 m</td>
<td>0.005 m</td>
<td>3.268 m</td>
<td>0.034 m</td>
</tr>
</tbody>
</table>

Inspecting the value of $RMSE_r$ reveals that it is almost 100 times higher for the 2D tracker than for the 3D tracker, when the large measurement noise is simulated. If small measurement noise is used, $RMSE_r$ is more than 500 times higher for the 2D tracker than for the 3D tracker. This demonstrates that the 3D tracker really works as expected. However, concerning the image coordinates,
there is no significant difference in errors when comparing the 2D tracker and the 3D tracker. One explanation to this is that the range is so large and the velocity so small so that an error of about 10 m in range does not affect the estimated image coordinates to any greater extent.

The tested scenario describes an almost ideal situation (no maneuvers and good observation data) and is mainly included to demonstrate the (non-existing) range estimation capability of the 2D tracker.

For reference, the RMSE values for the image coordinates when there are only detections in every 30th sample are presented in Table 7.3. As the followed trajectory is nothing but a straight line, both trackers performs well even when many consecutive measurements are missing.

**Table 7.3.** Scenario 1: RMSEs for data with detections in every 30th sample. RMSE calculations based on 100 realizations.

<table>
<thead>
<tr>
<th>Measure</th>
<th>2D, small $R$</th>
<th>3D, small $R$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$RMSE_{u}$</td>
<td>1.314 px</td>
<td>1.307 px</td>
</tr>
<tr>
<td>$RMSE_{v}$</td>
<td>0.143 px</td>
<td>0.145 px</td>
</tr>
<tr>
<td>$RMSE_{r}$</td>
<td>3.135 m</td>
<td>0.034 m</td>
</tr>
</tbody>
</table>

**7.4.2 Scenario 2: Straight line, not parallel with image plane, range from 3m to 170m**

In this test case, a similar scenario is considered. The target starts at a range of 3 m from the camera and moves along a straight line to a range of 170 m from the camera at a constant speed of 1.67 m/s ($\approx$6km/h), see Figures 7.2(a) and 7.2(b).

![Trajectory in XY plane](image)

![Trajectory in XZ plane](image)

(a) $XY$ plane. The target starts in the left end of the line. (b) $XZ$ plane. The target starts in the bottom left end of the line.

**Figure 7.2.** Predefined trajectories for sequence 2.

The same process noise covariance matrix was used as in last section. In Figure 7.3, the true and estimated trajectory plots in sensor coordinates for the case of
large measurement noise are plotted. It is hard to see how good/bad the filters are in those plots, hence it is better to study the RMSE plots.

In Figure 7.4, $RMSE_u[k]$ is plotted and in Figure 7.5, $RMSE_v[k]$ is plotted for one realization with the larger measurement noise configuration. It can be seen that the errors decrease as more measurements enter the filter. An error of less than 0.5 pixels in horizontal direction is reached within 83 frames for the 3D camera while the same error level is reached after 285 frames for the 2D camera. The large values in the beginning of the sequence occurs due to the large acceleration. The target motion model is initiated with zero speed and a very low acceleration, while the real speed is non-zero constant from the very first frame, so that it takes some time to match the detections.

Table 7.4. Scenario 2: RMSE for data with two different measurement noise levels for the 2D and 3D case. RMSE calculations based on 100 realizations.

<table>
<thead>
<tr>
<th>Measure</th>
<th>2D, small $R$</th>
<th>3D, small $R$</th>
<th>2D, large $R$</th>
<th>3D, large $R$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$RMSE_u$</td>
<td>0.329 px</td>
<td>0.192 px</td>
<td>1.035 px</td>
<td>0.295 px</td>
</tr>
<tr>
<td>$RMSE_v$</td>
<td>0.196 px</td>
<td>0.127 px</td>
<td>0.617 px</td>
<td>0.386 px</td>
</tr>
<tr>
<td>$RMSE_r$</td>
<td>1.942 m</td>
<td>0.073 m</td>
<td>2.891 m</td>
<td>0.185 m</td>
</tr>
</tbody>
</table>

Table 7.4 shows the RMSE value for the full sequences, based on 100 Monte Carlo simulations. The RMSE indicates an improvement in both image coordinates when the range information becomes available. Except for the $RMSE_u$ for large $R$, the values for the 3D tracker are approximately 60 % of the values for the 2D tracker. For $RMSE_u$ for large $R$, the values for the 3D tracker are only 30 % of the values for the 2D tracker.

Table 7.5 shows the results from simulation with the small measurement noise and different data delivery rate. When the sampling frequency is lowered with a factor five, the $RMSE_u$ and $RMSE_v$ values for the 3D tracker are in average 75 % of the values for the 2D tracker. If the factor is changed to ten, the percentage is raised to 85 % and if it is changed to twenty, the percentage is raised to 111 %. Hence, there is no use to track with a 3D camera if so few measurements are available. Furthermore, both trackers perform poor in this case.

Table 7.5. Scenario 2: RMSEs for data with small measurement noise levels and different amount of data available. RMSE calculations based on 100 realizations.

<table>
<thead>
<tr>
<th>Measure</th>
<th>2D, 5th</th>
<th>3D, 5th</th>
<th>2D, 10th</th>
<th>3D, 10th</th>
<th>2D, 20th</th>
<th>3D, 20th</th>
</tr>
</thead>
<tbody>
<tr>
<td>$RMSE_u$</td>
<td>1.083 px</td>
<td>0.788 px</td>
<td>3.317 px</td>
<td>2.792 px</td>
<td>8.418 px</td>
<td>9.314 px</td>
</tr>
<tr>
<td>$RMSE_v$</td>
<td>0.585 px</td>
<td>0.439 px</td>
<td>1.778 px</td>
<td>1.504 px</td>
<td>4.526 px</td>
<td>5.002 px</td>
</tr>
<tr>
<td>$RMSE_r$</td>
<td>1.195 m</td>
<td>0.093 m</td>
<td>11.303 m</td>
<td>0.138 m</td>
<td>48.433 m</td>
<td>0.313 m</td>
</tr>
</tbody>
</table>
Simulations

Figure 7.3. Scenario 2: True and estimated sensor coordinates when large noise is applied to the measurements.
7.4 Results

Figure 7.4. Scenario 2: RMSE for image coordinate $u$ using large measurement noise.

Figure 7.5. Scenario 2: RMSE for image coordinate $v$ using large measurement noise.
7.4.3 Scenario 3: Two adjacent circles

When tracking a target that maneuvers like an eight on the ground (see Figure 7.6), the white acceleration noise assumption is no longer applicable. Since one component of the acceleration vector always points toward the center of one of the circles, the autocorrelation assumption is violated. To compensate for this, a much larger value for the velocity variances must be chosen. If the piecewise constant noise model is used, then also the position variances will be affected via the nondiagonal terms. This behavior is not desirable since it will make the position estimates less certain. To avoid this, the piecewise constant white acceleration model was skipped and the process noise matrix $Q$ chosen to be diagonal. Since the head position is known to not change much around the height of the person in the sequence, the noises corresponding to $Y$ and $\dot{Y}$ were chosen much smaller than for the other states. Based on the covariance matrix structure in Equation (4.6), the values that best described the process noise in this sequence was chosen as in Equation (7.4),

$$ Q = \frac{1}{1000} \begin{bmatrix} 0.01 & 0 & 0 & 0 & 0 & 0 \\ 0 & 50 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.0010 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.00001 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.01 & 0 \\ 0 & 0 & 0 & 0 & 0 & 50 \end{bmatrix} $$

(7.4)

The resulting RMSE values are shown in Table 7.6. It is seen that $RMSE_u$ does not depend on whether a 2D tracker or a 3D tracker is used. The values are mainly dependent on the noise level. More changes are seen in the values of $RMSE_v$. The 3D tracker filters the image coordinate $v$ about twice as good as the 2D tracker does. However, both the trackers render $RMSE_v$ values smaller
than one pixel, which is very good, so the halving of the $RMSE_u$ value might not motivate the need for a 3D tracker.

Figure 7.7 shows the sequences $RMSE_u[k]$ and $RMSE_v[k]$ when small measurement noise is used. It is seen that $RMSE_u[k]$ sequences are very similar to each other but that $RMSE_v[k]$ in the 2D case (dashed line) varies more between samples than it does in the 3D case. The dotted line represents a 2D tracker where the process noise for the $Y$ coordinate is chosen in the same way as the other coordinates’ process noises. Using such a configuration, it is seen that the variations decreases and that the mean of the sequence increases. With the larger process noise, the 3D tracker estimates the values similar to the 2D tracker (not illustrated here) and no difference can be seen. However, for the remainder of this section, the process noise is chosen according to Equation (7.4).

Table 7.6. Scenario 3: RMSEs for data with two different measurement noise levels for the 2D and 3D case. RMSE calculations based on 100 realizations.

<table>
<thead>
<tr>
<th>Measure</th>
<th>2D, small $R$</th>
<th>3D, small $R$</th>
<th>2D, large $R$</th>
<th>3D, large $R$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$RMSE_u$</td>
<td>0.451 px</td>
<td>0.450 px</td>
<td>1.622 px</td>
<td>1.610 px</td>
</tr>
<tr>
<td>$RMSE_v$</td>
<td>0.236 px</td>
<td>0.093 px</td>
<td>0.658 px</td>
<td>0.277 px</td>
</tr>
<tr>
<td>$RMSE_r$</td>
<td>5.734 m</td>
<td>0.025 m</td>
<td>4.219 m</td>
<td>0.188 m</td>
</tr>
</tbody>
</table>

The RMSE values from simulations with the smaller measurement noise and different data delivery rate are shown in Table 7.7. Table 7.8 shows the RMSE values in the 3D case in percentage of the corresponding RMSE values in the 2D case. It is seen that $RMSE_u$ even here is independent of the type of tracker being used, but that the value increases when fewer measurements are available. $RMSE_v$ is about twice as good in the 3D case compared to the 2D case (as was seen in Figure 7.7(b)), but the gain with using a 3D tracker slowly decreases when measurements are given more seldom. However, the advantage of using the 3D tracker is notable even when many measurements are missing.

Table 7.7. Scenario 3: RMSEs for data with small measurement noise levels and different amount of data available. RMSE calculations based on 100 realizations.

<table>
<thead>
<tr>
<th>Measure</th>
<th>2D, 5th</th>
<th>3D, 5th</th>
<th>2D, 10th</th>
<th>3D, 10th</th>
<th>2D, 20th</th>
<th>3D, 20th</th>
</tr>
</thead>
<tbody>
<tr>
<td>$RMSE_u$</td>
<td>1.915 px</td>
<td>1.914 px</td>
<td>7.025 px</td>
<td>7.186 px</td>
<td>28.943 px</td>
<td>29.794 px</td>
</tr>
<tr>
<td>$RMSE_v$</td>
<td>0.435 px</td>
<td>0.176 px</td>
<td>0.600 px</td>
<td>0.250 px</td>
<td>0.848 px</td>
<td>0.472 px</td>
</tr>
<tr>
<td>$RMSE_r$</td>
<td>4.075 m</td>
<td>0.106 m</td>
<td>4.919 m</td>
<td>0.377 m</td>
<td>4.781 m</td>
<td>1.462 m</td>
</tr>
</tbody>
</table>
Figure 7.7. Scenario 3: Calculations based on 100 realizations with small measurement noise.

Table 7.8. Scenario 3: RMSEs for the 3D tracker as percentage of the RMSEs for the 2D tracker when small measurement noise is simulated. The values are directly calculated from Table 7.6 and Table 7.7. Values less than 100% indicate that the 3D tracker performs better.

<table>
<thead>
<tr>
<th>Measure</th>
<th>every</th>
<th>5th</th>
<th>10th</th>
<th>20th</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{RMSE}_u )</td>
<td>99.7%</td>
<td>100.0%</td>
<td>102.3%</td>
<td>102.9%</td>
</tr>
<tr>
<td>( \text{RMSE}_v )</td>
<td>39.3%</td>
<td>40.5%</td>
<td>41.7%</td>
<td>55.6%</td>
</tr>
<tr>
<td>( \text{RMSE}_r )</td>
<td>0.4%</td>
<td>2.6%</td>
<td>7.7%</td>
<td>30.6%</td>
</tr>
</tbody>
</table>
7.4 Results

7.4.4 Scenario 4: Two adjacent rhombuses

The scenario with a constantly turning target in previous chapter is rarely seen in practise. Therefore, a slightly more realistic scenario was set up. It consists of straight lines and 90° turns. Figure 7.8 shows the trajectory from above. Even this scenario will test how sensitive the trackers are to a violation of the CV assumption.

Figure 7.8. Scenario 4: The predefined trajectory in the $XZ$ plane. The target starts in the center and travels down left along the line.

The process noise covariance matrix is given in Equation (7.5). It is almost the same as the matrix in Section 7.4.3 but here, larger changes in velocity are accepted. Different values for the matrix was tested and the model evaluated on the sequence before it was determined that the chosen values generated the smallest RMSE values. The RMSE values for the different measurement noise levels are presented in Table 7.9 while the values for the study with smaller measurement noise and different data delivery rate are presented in Table 7.10.

$$Q = \frac{1}{1000} \begin{bmatrix} 0.01 & 0 & 0 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.0010 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.00001 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.01 & 0 \\ 0 & 0 & 0 & 0 & 0 & 100 \end{bmatrix} \quad (7.5)$$

Even for this scenario, the $u$ values are almost the same between the 2D case and the 3D case, while the $v$ value is approximately twice as good in 3D. Another interesting thing to observe is that the $RMSE_r$ values seems to decrease when less data become available in the 2D case. In order to see if there is such a trend in the data, the RMSE errors for an interval of different data delivery rates were studied and plotted in Figure 7.9. Since the trajectory (Figure 7.8) consists of 200 samples, it is not interesting to get measurements more seldom than what
Table 7.9. Scenario 4: RMSEs for data with two different measurement noise levels for the 2D and 3D case. RMSE calculations based on 100 realizations.

<table>
<thead>
<tr>
<th>Measure</th>
<th>2D, small $\hat{R}$</th>
<th>3D, small $\hat{R}$</th>
<th>2D, large $\hat{R}$</th>
<th>3D, large $\hat{R}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$RMSE_u$</td>
<td>0.432 px</td>
<td>0.431 px</td>
<td>1.467 px</td>
<td>1.469 px</td>
</tr>
<tr>
<td>$RMSE_v$</td>
<td>0.208 px</td>
<td>0.096 px</td>
<td>0.600 px</td>
<td>0.295 px</td>
</tr>
<tr>
<td>$RMSE_r$</td>
<td>3.341 m</td>
<td>0.026 m</td>
<td>4.566 m</td>
<td>0.186 m</td>
</tr>
</tbody>
</table>

Table 7.10. Scenario 4: RMSEs for data with small measurement noise levels and different amount of data available. RMSE calculations based on 100 realizations.

<table>
<thead>
<tr>
<th>Measure</th>
<th>2D, 5th</th>
<th>3D, 5th</th>
<th>2D, 10th</th>
<th>3D, 10th</th>
<th>2D, 20th</th>
<th>3D, 20th</th>
</tr>
</thead>
<tbody>
<tr>
<td>$RMSE_u$</td>
<td>2.155 px</td>
<td>2.184 px</td>
<td>7.365 px</td>
<td>7.137 px</td>
<td>26.538 px</td>
<td>25.352 px</td>
</tr>
<tr>
<td>$RMSE_v$</td>
<td>0.409 px</td>
<td>0.185 px</td>
<td>0.545 px</td>
<td>0.273 px</td>
<td>0.783 px</td>
<td>0.429 px</td>
</tr>
<tr>
<td>$RMSE_r$</td>
<td>8.981 m</td>
<td>0.111 m</td>
<td>7.223 m</td>
<td>0.422 m</td>
<td>5.313 m</td>
<td>1.053 m</td>
</tr>
</tbody>
</table>

corresponds to the short straight lines, that is in each 25th sample. If there were measurements, say at only each 50th sample, information could only be retrieved at the top, middle and bottom of the trajectory and this would mostly yield a better range value on average than when data is available also on the very left and right sides of the trajectory.

From the figures, the observation from above can be confirmed; there is actually a decrease in $RMSE_r$ if only the factors 5, 10 and 20 are studied. However this is more a coincidence than a linear relationship between sampling time and RMSE values. The values fluctuate seemingly randomly in the graph. This is explained by the non-observability in range. For the image coordinates, more measurements results in smaller RMSE values, but the world coordinates depend more or less on the range so they show some random behaviour.

The 2D tracker can actually make use of the lower uncertainty in $Y$ to better estimate $Z$. Intuitively, this is seen by inspecting Equation (3.1b). $Y$ and $v$ are much less uncertain than $Z$ (and also less uncertain than $X$), hence the filter prioritizes this information when it calculates $Z$. By letting the true $Y$ value of the head coordinate be $Y = 0$ instead of $Y = 1$, information about $Z$ can no longer be obtained from the equation and then the tracker can be expected to perform worse. The RMSE values for this case are shown in Table 7.11. It indicates that there is no longer a difference in RMSE for the image coordinates between the 2D and 3D case. The $RMSE_v$ values are smaller and the $RMSE_r$ values are larger for the 2D tracker compared to the former values (see Table 7.9). Table 7.12 shows results from simulation with the smaller measurement noise and different data delivery rate. Even here, the RMSE seems to show no big difference between the 2D and 3D case. Figure 7.10 shows the RMSE measures plotted for different measurement data delivery rates. This shows clearly that the $Y$ and $Z$ coordinates now are decoupled from each other. The behaviour of the 3D tracker can be seen in Figure 7.11.
7.4 Results

(a) $RMSE_u$

(b) $RMSE_X$

(c) $RMSE_v$

(d) $RMSE_Y$

(e) $RMSE_r$

(f) $RMSE_Z$

Figure 7.9. Scenario 4: RMSE as a function of measurement delivery frequency, using no range information. The $Y$ positions of the camera and target differ by one meter.

Table 7.11. Scenario 4: RMSEs for data with two different measurement noise levels for the 2D and 3D case. Target has the same $Y$ position as the camera. RMSE calculations based on 100 realizations.

<table>
<thead>
<tr>
<th>Measure</th>
<th>2D, small $R$</th>
<th>3D, small $R$</th>
<th>2D, large $R$</th>
<th>3D, large $R$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$RMSE_u$</td>
<td>0.431 px</td>
<td>0.431 px</td>
<td>1.474 px</td>
<td>1.469 px</td>
</tr>
<tr>
<td>$RMSE_v$</td>
<td>0.092 px</td>
<td>0.096 px</td>
<td>0.286 px</td>
<td>0.290 px</td>
</tr>
<tr>
<td>$RMSE_r$</td>
<td>14.929 m</td>
<td>0.026 m</td>
<td>9.356 m</td>
<td>0.186 m</td>
</tr>
</tbody>
</table>
Table 7.12. Scenario 4: RMSEs for data with small measurement noise levels and different amount of data available. Camera and target are at the same \( Y \) position.

<table>
<thead>
<tr>
<th>Measure</th>
<th>2D, 5th</th>
<th>3D, 5th</th>
<th>2D, 10th</th>
<th>3D, 10th</th>
<th>2D, 20th</th>
<th>3D, 20th</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{RMSE}_u )</td>
<td>2.341 px</td>
<td>2.184 px</td>
<td>9.582 px</td>
<td>7.137 px</td>
<td>26.713 px</td>
<td>25.351 px</td>
</tr>
<tr>
<td>( \text{RMSE}_v )</td>
<td>0.189 px</td>
<td>0.181 px</td>
<td>0.240 px</td>
<td>0.241 px</td>
<td>0.293 px</td>
<td>0.311 px</td>
</tr>
<tr>
<td>( \text{RMSE}_r )</td>
<td>9.662 m</td>
<td>0.111 m</td>
<td>5.600 m</td>
<td>0.422 m</td>
<td>9.570 m</td>
<td>1.054 m</td>
</tr>
</tbody>
</table>

Figure 7.10. Scenario 4: RMSE as a function of measurement delivery frequency, using no range information. Camera and target are at the same \( Y \) position.
Figure 7.11. Scenario 4: RMSE as a function of measurement delivery frequency, using range information. Camera and target are at the same Y position.
Chapter 8

Tracking in real video sequences

Tests were carried out on video sequences collected by the FLASH sensor. Comparisons were made to show similarities and differences between tracking with and without range information available. In this chapter, the sequences and the results are presented and commented.

8.1 Measures

When tracking with a real camera, there is no longer a known truth. This makes it impossible to use the RMSE measure in the form that was used in the simulation part (see Section 7.3). Different methods for making a pseudo ground truth are discussed in [21]. Making pseudo ground truth is time-consuming and in the case of having a range image, the target range value is hard to choose - it cannot be chosen based only on visual inspection.

An alternative idea is to calculate the variance error around the Monte Carlo mean in each sample of the sequence (see e.g. [30]). If the Monte Carlo mean is close to the real mean, this will yield a rather good approximation of the RMSE measure. The problem with this is that it is not easy to reproduce data under the same circumstances. Only one realization of each process is available here so this approach cannot be used in this case.

When only one realization is available, the bootstrap approach may be fruitful [30]. After having applied the filter algorithm to the sequence, the noise statistics can be calculated in reverse. Then the filter can be run again but with values for the noise randomly drawn from the calculated noise statistics. In this way, several different artificial sequences can be obtained, that in turn can be used to calculate a Monte Carlo mean and an estimate of RMSE based on this mean. There are however many difficulties involved in this algorithm such as how to decide the noise statistics (what is noise and what is signal?) and whether it is justified or not to use noise from different times (a close target may have different noise behaviour.
than one far away). For such reasons, the bootstrap approach was abandoned and instead the trackers are compared to each other and to the detections. Differences in updated estimates between the two trackers are calculated and compared.

### 8.2 Results

Information about the data sets can be found in [49]. Using the CV model in Section 4.3.1 with the values $\tilde{q}_u = \tilde{q}_v = 0.04$ s and $\tilde{q}_w = 0.001$ s for the process noise intensities of the coordinates results in the following process covariance matrix, which is used in all sequences:

$$Q = \frac{1}{100000} \begin{bmatrix} 0.5328 & 8.0000 & 0 & 0 & 0 & 0 \\ 8.0000 & 160.0000 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.0003 & 0.0050 & 0 & 0 \\ 0 & 0 & 0.0050 & 0.1000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5328 & 8.0000 \\ 0 & 0 & 0 & 0 & 8.0000 & 160.0000 \end{bmatrix} \quad (8.1)$$

#### 8.2.1 Sequence 1: Simple motion near camera

In this sequence, the target moves several times between the left and right parts of the scene at a range of 12.7 m away from the camera (see Figure 8.1). Due to the almost constant range over time and a relative large distance to the wall in the background, segmentation based on range is used. The background variance is initiated to 0.2 m because the resulting foreground is good input to the head detector. The target moves with approximately constant speed in the horizontal direction except during the turns. The second and third turn are made outside the FOV which results in lack of measurements between samples 70 and 94 and between samples 132 and 146 (as seen in Figure 8.2(d)). The head detector delivers values for the image coordinate $u$ in the range $[11, 118]$ because it cannot verify the presence of a head closer to the edge than approximately 10 pixels. The standard deviations for the measurements $u$ and $v$ were both set to one pixel. Figure 8.2 shows that both the image coordinates are filtered in similar ways for the 2D and 3D tracker. Considering the range images, the 3D tracker performs well in the sense that it filters the measurements and also predicts a reasonable range when measurements are missing. The 2D tracker also shows a good performance in range but only for the first 25 frames. After that, the filtered range starts to increase with each frame and end up at about 13.4 m.

An interesting detail to notice is that the range measurement in sample 95 seems to be filtered in a strange way (see Figure 8.3 for a closer view) by the 2D tracker. After the update part of the filter algorithm, the updated range estimate is larger than the prediction, even though the measured range is smaller than the prediction. See Example 8.1. This behaviour is not restricted to occur only in 2D tracking but can also be seen for the 3D tracker in the next sequence (see Section 8.2.2). The prediction would have been a better choice in both cases. This
remarkable behaviour arises as the tracking is performed in Cartesian coordinates but the comparison is made in the sensor coordinates. The Cartesian update is good but that is hidden when transformed to sensor coordinates due to dependency between coordinates. These kinds of errors are often very small in the studied applications and fastly corrected for by the subsequent measurements.

--- Example 8.1: Filtered value worse than prediction? ---

The situation seen in Figure 8.3 shows the “estimated” range, which is the norm of the three position values in the state vector. In frame 91-94, no measurements are available and then only the prediction part of the tracker algorithm can be executed. This is seen in that the slope of the line is constant here. In frame 95, a measurement is provided to the filter and the states are updated. The “predicted” range (i.e. the norm of the predicted position vector) is about 13.3 m while the measurement is about 12.65 m. Yet, the “filtered” range ends up at 13.36 m, which is further away than the prediction. How did this happen?

Consider that the measured range is smaller than the “predicted” range. In order to come closer to the measurement (which a linear filter always strives for), the filter tries to make each component (X, Y and Z) of the vector smaller. If the measurements in the other directions (u and v) coincide with their respective “prediction”, then all the position states must be scaled equally. Doing so, the problem that was illustrated above cannot occur. Hence, consider instead the case that also the measurement of u is smaller than its prediction. Reviewing Equation (3.1a), it is seen that a smaller u value is obtained by making the quotient X/Z smaller. The case with a smaller v measurement than v prediction is analogous. The quotients can be made smaller in many ways (including both decreasing and
increasing the values of $X$, $Y$ and $Z$). Combining this choice with the choice above (on how to decrease the norm of the position vector) gives an idea of the complexity of the problem. Which property is more important? Is it to decrease the norm of the position vector or is it to lower the quotients? For the tracker, this is not a binary decision, but in this example, the tracker seems to have prioritized the $u$ and $v$ coordinates, on the expense of raising the $Z$ value. The tracker reaches its solution by using the information in the covariance matrices. They decide how much the state variables can change. The short answer to the question of how the situation could arise is that there are at least two criterions that need to be met. In this example there was a conflict in how to choose the new $Z$ value. One criterion wanted it to increase while the other wanted it to decrease.

Figure 8.2. Detected and estimated sensor coordinates for the sequence Simple motion near camera.

Figure 8.4 shows the uncertainties in the states. This indicates normal behaviour of a filter since the prediction errors increase when measurements are not given and decrease when there are.

When comparing the 2D tracker with the 3D tracker for different sampling intervals, as in Figure 8.5, it turns out that the estimates mainly coincide most of
Figure 8.3. Study of the estimated range value in frame 95 in sequence *Simple motion near camera*. The updated value of the 2D tracker is larger than the prediction despite the measurement being smaller than the prediction.

Figure 8.4. The estimation error development for the 3D tracker in sequence *Simple motion near camera*. Square root of the mean value of the norm of the estimation error as function of the frame number. Deviations are given in meters.
the time. The similarities are most pronounced for a high sampling frequency while the trackers differ more if the sampling frequency is low. Differences between the two trackers often occur after the target has performed a turn and measurements again become available after that maneuver.

Such a situation can be seen in frame 95. The target has been out of the picture for some frames and there made a U-turn. In frame 95 it reappears in the scene and is detected again. When using the highest sampling frequency, the difference between the trackers is 2 pixels in image coordinate \( u \) and 0.15 pixels in
image coordinate \( v \) (see also Table 8.1). If the frequency is lowered by a factor 20, the same differences becomes 3.67 pixels in \( u \) and 0.05 pixels in \( v \). However, these values does not reflect the real situation. The information about the U-turn is not considered until frame 100 if the lower frequency is used, so the comparison is preferably made relative this frame. At frame 100, the difference is 11.40 pixels in \( u \) and 0.18 pixels in \( v \), which means that the difference is almost six times higher for \( u \) in this case. The difference in \( v \) is almost unchanged but can be explained by the smaller spread in the values supplied by the head detector. The estimated world coordinates are presented in Figure 8.6.

![Figure 8.6. Estimated world coordinates for the sequence Simple motion near camera.](image)

**Figure 8.6.** Estimated world coordinates for the sequence *Simple motion near camera.*

**Table 8.1.** Differences in target image coordinates in frame 95 and frame 100 for different sampling times in sequence *Simple motion near camera.*

<table>
<thead>
<tr>
<th>Frame and sampling time factor</th>
<th>( \text{diff}_u )</th>
<th>( \text{diff}_v )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frame 95, ( d = 1 )</td>
<td>2.00 px</td>
<td>0.15 px</td>
</tr>
<tr>
<td>Frame 95, ( d = 20 )</td>
<td>3.67 px</td>
<td>0.05 px</td>
</tr>
<tr>
<td>Frame 100, ( d = 20 )</td>
<td>11.40 px</td>
<td>0.18 px</td>
</tr>
</tbody>
</table>
8.2.2 Sequence 2: Occlusion by truck and noise

The scene which can be seen in Figure 8.7 consists of three buildings and one truck. In frame 43, a person appears from behind the rightmost building. He walks to the left and then gets covered by the truck in frame 57. The person is then covered by the truck until frame 89 where he again becomes visible. At frame 141 the person walks behind the leftmost building and no further detections can be made.

![Frame no. 46](image)

**Figure 8.7.** Intensity image from the sequence *Occlusion by truck and noise*. Intensity-based segmentation proved to be difficult for the sequence.

In order to get decent detections in this noisy data, the range was limited to between 96 m and 103 m before entering the segmentation and detection algorithms, see Figure 8.8. Many of the faulty detections from the atmosphere and the buildings are then avoided, which would otherwise render mainly false head detections. The interval was also chosen so that the target was inside it all the time. The initial background standard deviation, based on the range, was chosen as the constant $\sigma_{\text{initial}} = 0.01$ m.

Figure 8.9 shows the detected and filtered sensor coordinates and also frames with detections. It can be concluded that a maneuver is performed during the frames 58-89, when the target is occluded by the truck. Before the occlusion, the target is slowly approaching the camera, but when it reappears after the occlusion, it is detected at a range 3 m further away. Figure 8.10 shows the standard deviation of the range values. It is less than 0.7 m at all frames except sample 54 where it is 3.4 m. This means that it is quite likely that there is a maneuver being performed during the occlusion and the big range difference is less likely explained by bad
Figure 8.8. Range image with ranges given in meters. Tracking in scene with target occluded by truck and noise. The measurement is marked by a small red square. The estimated output is marked by a big red square. (Electronic version only: Click for movie. Xvid Codec is required, see http://www.xvid.org/.)

range measurements.

In Figure 8.11, the image coordinates are plotted for different sampling times of the two trackers. For the \( u \)-direction, the constant velocity assumption is very well satisfied. As soon as the trackers have received some measurements, they are able to predict the rest of the sequence quite accurately. However, the first detections come at different times when different sampling times are used and that will also affect the prediction accuracy at this instant.

Differences can be seen in Figure 8.12. The biggest differences are during the prediction-only frames, but those are not very interesting to investigate. They depend only on how the tracker estimated the states during the latest update part. The update parts are more interesting to investigate and one such frame is studied here. At frame 90, which is where the target reappears after having been occluded by the lorry for 33 frames, the difference between the two trackers is 0.02 pixels in \( u \) and 0.12 pixels in \( v \), when all detections are available. If detections are only available in every tenth frame, then the difference is 0.01 pixels in \( u \) and 0.08 pixels in \( v \). The differences are therefore smaller in this case. One explanation to this is that the uncertainty in the states is larger when the filter has been fed with fewer measurements, hence it trusts new measurements more. As the situation is the same for the two filters, both trust more in the measurements and then the difference between the filter outputs is smaller.


When looking at the $v$ coordinate however, there is also another explanation. Here, the initial position almost coincides with the first measurement for the lower measurement delivery rate. Hence, the state covariance can be very small. The difference would also have been even smaller if the estimates had not been updated in different directions. As can be seen, the 3D tracker found that it was better to increase the estimate than to decrease it (this was discussed in Section 8.2.1).

The case with measurements given only in every 20th frame are not studied here. Frame number 100 is the first frame where a measurement is given and hence it is only the initial estimate and covariance that affect the update, which is not very interesting to examine. The difference in $u$ between the trackers when measurements are accepted only in every fifth frame is the same as when all measurements are given; 0.02 pixels. For $v$, the difference is 0.05 pixels in this case.

Another problem that is worth to be mentioned, valid for both trackers, is the growing dominance of the detection in frame 140. When a high sampling frequency is used, this causes a jump of approximately 0.1 pixels in the filtered estimates. By using the lowest sampling frequency instead, this jump rises to 0.7 pixels, even though the difference in detected values are the same. That is, a lower sampling frequency let outliers have more influence on the estimates, since the covariances are larger.
Figure 8.10. Standard deviations of the range measurements (from the detector) in the sequence *Occlusion by truck and noise*.

All the studied differences in this section are very small and nothing can be said about whether the 2D tracker or the 3D tracker performs better. The estimated true world coordinates are presented in Figure 8.13.
Figure 8.11. Different sampling times for the sequence *Occlusion by truck and noise.*
(a) Difference in estimated $u$, all measurements available  
(b) Difference in estimated $v$, all measurements available  
(c) Difference in estimated $u$, every 5th measurement available  
(d) Difference in estimated $v$, every 5th measurement available  
(e) Difference in estimated $u$, every 10th measurement available  
(f) Difference in estimated $v$, every 10th measurement available  

Figure 8.12. The difference between the estimated outputs from the 2D tracker and the 3D tracker for the sequence *Occlusion by truck and noise.*
Figure 8.13. Estimated world coordinates for the sequence *Occlusion by truck and noise.*
8.2.3 Sequence 3: Random walk at a range of 60 m

In this sequence, sular mode is used. The maximum detectable range is close to the wall in the background and so there is often missing range information about this wall (see Figure 8.14(b)). Missing range data is normally identified by a large value but if it is changed to a constant maximum value close to the maximum detectable range of the camera, the background calculation becomes easier. That is, the variances in the background will be relatively small and it will be possible to detect a target passing by. However, for this sequence, segmentation based on intensity data (see Figure 8.14(a)) turned out to generate more detections. The range image is then only used to calculate the mean and variance of the range for the person in the detection part. For this sequence, the background model was initiated with the mean and variance calculated from a set of only background images.

![Intensity image](a) Intensity image  ![Range image](b) Range image

Figure 8.14. Intensity and range images from the random walk sequence.

As can be seen in the figure, lots of noise is present under the person in picture. The values covers the whole range interval 58 m - 65 m. This noise is caused by the sular mode. Since the sensor system does not expect any returns at ranges closer than 58 m, it does not stop the counter in those regions. When no clear intensity peaks can be detected in the area, the peak search algorithm will output mostly nonsense range data. Hence, if the target moves out of the sular range interval, its range will be lost by the tracker. An improved algorithm that could account for this is however out of the scope for this thesis.

The target moves in a more random pattern than in the previous datasets. It also moves at yet another range, about 60 m away. The estimated world coordinates are presented in Figure 8.17. The detections and converted filter outputs is seen in Figure 8.15. Both the trackers seems to track the image position equally well in this case too. However, the range is tracked much worse for the 2D case in this sequence. As soon as the target changes direction, the 2D tracker predicts a shorter range to the target, while the 3D tracker maintain a good range prediction. A possible explanation to why the 2D tracker predicts a shorter range is that the
measured $v$ simultaneously is smaller in the studied frame, and from what have been seen earlier, the (more certain) $Y$ state is used as information in the range estimation process. Since $v$ probably is more certain than $u$ (because $v$ does not vary very much around its mean), the filter trusts more in $v$.

The detections and estimated values for different values of the sampling time are shown in Figure 8.16. For $u$, the two trackers exhibit almost the same behaviour. The biggest difference can be seen for $d = 20$ in frame 160 where both a difference in position and in speed is present. Again, these differences occur after a U-turn has been performed and enough details (measurements) about the turn are missing.

Another thing to note is that the 2D tracker generally appears to adapt faster to the measurements in $v$ than the 3D tracker does. This might be explained by the absence of constraints on the range estimates. When a detection (here: of $v$) is far from the predicted value, the 2D tracker might decrease the estimate of the range value so that the $v$ value better fits the detection. More specific, it is the states $Z$ and $\dot{Z}$ that are less restricted and can easier be changed to achieve better accordance between the detected and estimated values. For example, if the $u$ prediction is small compared to the measurement, the $Z$ state can be raised to
8.2 Results

(a) Detected and estimated $u$, $d=5$

(b) Detected and estimated $v$, $d=5$

(c) Detected and estimated $u$, $d=10$

(d) Detected and estimated $v$, $d=10$

(e) Detected and estimated $u$, $d=20$

(f) Detected and estimated $v$, $d=20$

Figure 8.16. Different sampling time for the random walk sequence.

increase the $u$ value (see Figure 3.2). This can also be described as overfitting the available data. Obviously this relationship does not necessarily lead to good performance in both $u$ and $v$ due to both their dependence on $Z$, but it may do so occasionally.
Figure 8.17. Estimated world coordinates for the sequence *Random walk at a range of 60 m.*
Chapter 9

Conclusions

In this thesis work, four typical tracking scenarios were simulated to test how well a 3D tracker and a 2D tracker could estimate the true trajectories of a human target. After that, three sequences captured with the FLASH sensor were used to investigate the trackers’ performance under real conditions.

The first simulated scenario consisted of motion along a straight line at a large distance, parallel to the image plane. No large difference could be seen between the two trackers when the filtered image coordinates, \( u \) (horizontal) and \( v \) (vertical), were compared. Changes in measurement noise or data delivery rate revealed no difference between the trackers. In some cases, the 2D tracker performed slightly better than the 3D tracker but none of them then performed bad since the RMSE values were low for both the trackers. It was suggested that the large range and small FOV helped the 2D tracker to perform as good as the 3D tracker. Furthermore, the superior capability of the 3D tracker to accurately track the range to target was demonstrated.

In the next scenario, the target moved along a straight line not parallel to the image plane. Here, the 3D tracker proved to be the better choice, showing \( RMSE_u \) and \( RMSE_v \) values that were 60% of the values for the 2D tracker with the highest sampling frequency. The 3D tracker was also more than three times faster than the 2D tracker in reaching below an RMSE value of less than 0.5 pixels in the \( \hat{u} \) direction. However, the 3D tracker was only advantageous if the tracker was fed with enough amount of measurements. Otherwise the gain with using a 3D tracker was smaller, but in that case none of the trackers performed well so a comparison was not relevant.

Both the eight-like trajectories showed no difference in RMSE for image coordinate \( u \) between the two trackers, but the RMSE for image coordinate \( v \) was twice as good in the 3D case. This might have been a consequence of modeling small process noise covariance in the \( Y \) position, enabling the tracker to use this to estimate \( Z \). When process noise values were chosen equally for each dimension, the difference between the 2D tracker and the 3D tracker could no longer be seen. As was the case with the former scenarios, even here the advantage of the 3D tracker decreased when the sampling frequency was lowered.
A study was performed to investigate the behaviour of the $RMSE_r$ values for the 2D tracker. It was shown that they were a non-monotonous mapping from the number of skipped measurements, most likely due to non-observability in distance. The placement of the camera in relation to the target was also shown to have a big influence on the $RMSE_r$ values for the 2D tracker. With a small expected variance of the $v$ coordinate and a difference in $Y$ position of camera and target, there was a strong correlation between the $Y$ and $Z$ states. It was shown that this had a bad influence on the $RMSE_v$ value when the 2D tracker was used, but no influence when the 3D tracker was used.

When the collected data sets from the FLASH sensor was examined, it was found that the measurements were affected by many different factors. The sequence *Occlusion by truck and noise* suffered from lots of noise. The noise might have originated from disturbances in the air so that the receiver got erroneous beam answers. Another explanation was that it came from electrical disturbances in the camera system. In the sequence (captured in hit mode), it was also noted that intensity data was sometimes missing or erroneous for some pixels. This was due to a bad implementation of the peak search algorithm (not part of this thesis).

It was also discovered that tracking using the camera in sular mode might be devastating if the peak search algorithm is not carefully implemented. In sular mode, only beam reflections corresponding to a certain range interval will be recorded. Hence, areas in the image that represent objects that are closer, should not obtain a range estimate. Even so, this was the case. Since the algorithm was based on the assumption that there must be exactly one peak, it always found one. This peak was random since none of the searched values in the wave represented a reflection from an area illuminated by the laser. Then, also the range estimate got a random value.

This means that no target can be closer to the camera than the specified interval for sular mode. If it is, it will introduce errors in a 3D tracking system. Even the video segmentation will perform worse as the threshold between foreground and background must be larger. In this scenario, the problem was minimized by limiting the range image to the minimum and maximum range to target, leading to smaller variations and better segmentation.

The comparison of the 2D tracker and the 3D tracker could not be made in terms of RMSE due to the absence of a ground truth. Instead they were compared to each other to see when there were differences. Comparing them in sensor coordinates also made it possible to make a comparison to the measurements. It was shown that differences in estimated output between the two trackers were most visible when the target reappeared in the scene after having been out of the camera view for some frames; 2 pixels difference in filtered $u$ value and 0.15 pixels difference in filtered $v$ value. If instead the twenty times lower sampling frequency was used, the differences were 11.4 pixels in $u$ and 0.18 pixels in $v$. During the frames out of the scene, the target performed a U-turn in the $\hat{u}$ direction which might explain the larger difference in $u$. A smaller spread in the values supplied by the head detector explained why the difference in $v$ had not grown more when the sampling frequency was lowered.

In the sequence where the target was occluded by a truck for some frames,
testing a lower sampling frequency instead led to more similar performance of the two trackers after the update part when the target reappeared. Two explanations were given to this. The first was that the new measurement was considered to be more trustworthy by both trackers when the sampling frequency was lowered (less measurements means larger uncertainty), so they both updated the states much toward the same value and hence the difference got small. The other explanation concerned mainly the \( v \) coordinate. It was based on the observation that the first detection in \( v \) was very close to the initial value, so the state covariance matrix got very small. Then in the studied frame, yet a detection at the same \( v \) value arrived which both trackers then considered to be trustworthy.

Finally, for the random walk sequence, no obvious benefits could be seen when using a 3D tracker. It was suggested that the 2D tracker might use the \( Z \) state (which describes the target position along the camera direction) as a tuning parameter when estimating the other states. This way the 2D tracker could sometimes respond and adapt quicker to image coordinate measurements.

For all video sequences, and also in some simulations, it was noted that the trackers sometimes generated a worse updated value than the predicted value. This peculiarity occurred due to tracking in Cartesian coordinates while getting the measurements and making the comparisons in sensor coordinates. The influence of such events occurred only temporarily and was only seen in at most three consecutive frames. Moreover, these deviations were most often very small.

To summarize, using a 3D camera for tracking human motion has no obvious advantages in relation to the conventional 2D camera when the comparison is made in image coordinates. In cases where the 3D tracker actually performed better, the performance of both the trackers were very good so using a 3D tracker could not be motivated in those cases. Two explanations to the good performance of the 2D tracker was given. Firstly, large range, small FOV and low target velocity facilitates the task. Secondly, it was assumed that the 2D tracker modified the non-observable states \( Z \) and \( \dot{Z} \) to make the other states more consistent with the measurements. Hence, the good performance was reached at the expense of bad range tracking. With the 3D tracker, the error is small in all coordinates so that it is especially useful in precision applications, where the world position of the target is critical.

Based on this study, it is stated that the 3D tracking algorithm cannot be run with a lower frequency than the 2D tracking algorithm and at the same time also show similar or better performance for target tracking in the image coordinates. However, the 3D tracker is the better alternative if the comparison should be made in world coordinates.
Chapter 10

Discussion and future work

This chapter discusses the possibilities with using 3D imaging ladar for head tracking and also the different assumptions that were made in this thesis. The chapter concludes with suggestions for future work.

10.1 Discussion

In this thesis, it has been investigated whether or not it is possible to decrease the sampling frequency for a tracking algorithm based on 3D measurements without losing performance compared to a tracking algorithm based on 2D measurements. It was further stated that this is not the case; if a 3D tracker runs with a lower sampling frequency, it cannot be expected to perform better than the 2D tracker with unchanged sampling frequency. A number of assumptions and simplifications have been made that affects the outcome of the study. They are discussed below.

10.1.1 The statement

The conference statement (see Section 1.1) was made in conjunction with a discussion on how to best determine the time of detonation before a missile collides with its target. That will probably imply the following differences to the application studied in this thesis:

- A missile moves with a velocity of several thousands kilometer per hour while a human moves at about 6 km/h.
- The human target is much more unpredictable than a missile.
- The location of the sensor is in one case stationary on the ground and in the other case attached to the moving missile.

10.1.2 Video frame rate

Normally, video rate is defined as 30 Hz, but in this thesis, the highest studied frame rate was 10 Hz. This was also the rate at which the frames were delivered
by the FLASH sensor. If a higher maximum frame rate would have been studied in the simulations, there might had been a chance that the 3D tracker showed better performance compared to the 2D tracker.

10.1.3 Chain of observation processing

The information provided to the filters in this thesis are obtained after the steps in the following list have been applied.

- **Measuring the time.** The pixel circuitry has a counter that counts how many times the bins have rotated. The value of this counter is then multiplied with the bin distance (the distance between two consecutive bins), which is about 35 cm. This distance depends on the medium and the processor speed. A more precise distance can be obtained if a higher processor speed is used.

- **Peak search algorithm to find the most probable range in each pixel.** The peak search algorithm interpolates the sampled data and suggests a range peak that most probably inherits from the target. Depending on what algorithm is used, a more or less accurate range will be delivered. Most likely, this step will give a range measure that deviates much less than 70 cm (twice the bin distance) from the correct range.

- **Image segmentation to find foreground regions.** With the video segmentation algorithm (background modeling and foreground detection) used in this thesis, noise can lead to faulty classification of pixels as background or foreground. This was very common in sequences where the noise had a large variance.

- **Detection of the head.** If the foreground area is distorted, then the calculated image position of the head may be affected by a bias. The distance is calculated over all foreground pixels in a bounding box, hence a faulty foreground pixel in this box will contribute to an error. If, furthermore, the camera is located above the person, then the range is shorter to the head than to the feet. Averaging all detected values of the body then gives a larger range value to the head than the true range to the head.

- **Tracker update.** The tracker can use the information in many ways. In the EKF, the sensor coordinates are used to update the states via an approximation of the camera model. If instead the measurements are converted via an inverted camera model before entering the filter, some debiasing needs to be done to compensate for the errors that are also transformed. If the debiasing is performed correctly then this is the better choice of tracking algorithm [5].

In this thesis work, the measurement noise covariance matrix elements associated with the range are only calculated from the foreground in the bounding box (step 4 above), as a variance over an area. Those values are probably too
small since there are more error sources listed above. The image segmentation is probably the second biggest error source since the noise can differ by many meters in some situations. Such a value might be avoided with a better segmentation algorithm. The problem with having too small measurement noise covariance values is that the filter trust the measurements more than what is realistic.

The pinhole camera model is accurate for narrow to normal FOV only [37]. In this thesis, a FOV of 8.6° has been used which is considered to be narrow so that the model is applicable here. For wider angles, more detailed models might be considered, such as the one used in [13, page 14].

An improved calibration of the camera has been suggested in [48], but it has not yet been implemented.

The camera model assumes that the illuminator (the laser source) and the detector is located at the same spot. In a realistic application this is never the case but the errors that this gives rise to is here considered to be very small (the distance between the center of the detector and each of the laser sources are about 5 cm). However, if the illuminator is located far away (several meters or more) from the detector, then this needs to be modelled accordingly.

A problem with using active illumination might arise if several 3D cameras are used for the same scene. Then the detector thresholds might stop too early and/or the peak search algorithm can calculate the wrong range. This problem can however be eliminated by coding the signals differently.

Other factors that affect the measurements badly are electrical disturbances in the camera and atmospheric disturbances such as aerosols and fog. In the simulations, Gaussian noise was added to the true measurements in order to model the distortions that occur in the real sensor system as discussed above. The chosen small and large measurement noises are considered to cover most situations so this has not affected the results of the study to any greater extent. The measurement noise matrix was chosen with standard deviations of the image coordinates measurement of one pixel. This is a choice that was made based on experience from other experiments with cameras. In a final implementation, these variations have to be estimated for the particular sensor and detection system. The PDF of the noise of the real measurements in a sensor system is probably also centered around some mean value but that has not been examined in further detail here.

10.1.4 Target selection

There was also a question of what part of the human to track. Here, the position of the head was selected as there was already an algorithm available to detect the head but one might as well consider to detect other parts such as the torso. The advantage with choosing the head is that it is almost always visible and it also has many features that helps to identify the person. A downside is that the head is quite small and can be harder to detect at larger ranges. From this point of view, detection of the torso can be more robust.
10.1.5 Constant velocity

The constant velocity assumption about the motion of the human target is an approximation that is sufficient most of the time, but there will be problem for example if the person starts to accelerate for a longer period of time, change direction or make another maneuver such as jump or crawl. In military and security/safety applications this constant velocity model will limit the applicability of the system; a model that can treat more advanced target motions is needed. Many different methods exists that can achieve this. One of them is to change the parameters of the current model dynamically to better describe the new behaviour. Another way is to use a multiple model algorithm in which several models (one for each mode/maneuver) are run in parallel and the currently best model is selected at each time instant.

10.1.6 Comparing performance

Tracking in image coordinates had the advantage that the 2D and 3D trackers both could measure the same quantities that were studied and compared. This obviously makes the comparison more fair but also the difference between the two trackers smaller and maybe misleading. If it is more important to have a good estimate of the true trajectory than to track in the image plane, then the comparison should be carried out in the world coordinates.

Tracking a correct range is most crucial when many systems cooperate, since the image coordinates that are calculated are only valid for the point in the world where the sensor is located. If another device needs the estimated values, they have to be represented in a common coordinate system. One example when this is less of a problem is when a missile will be launched towards a target and the sensor is located close to the barrel. Then it is mainly just to set up the barrel in the camera direction. Assuming that the velocity is high, it is most important to know the direction to the target. For long-range targets, the range is more crucial as a more exact parabola needs to be calculated for the missile to reach its destination. Correct range tracking is also very useful when determining whether a target is approaching or receding from the sensor. A fast approaching target can characterize a threat and some defensive actions can be taken in advance with this knowledge.

In surveillance applications however, it is most important to recognize and track suspicious persons. The exact world position is not always crucial as long as no other system will make use of the location. A future application might be to send the coordinates to a guard that will use them in a positional system to find the suspect. With the 2D camera this is not possible without making some assumptions. What could be done is to arrange the camera so that it looks in a very steep angle towards the ground. Assuming a height of about 170 cm (typical value for a human) and knowing the orientation of the ground plane, an estimate of the range can be calculated. From this, the world coordinates can be calculated approximatively.

For comparing the algorithms, a modification of the RMSE measure was used. Normally, it is defined for the state vector, but here it was used for the sensor
representation of the state vector. Furthermore, the norm was taken over one coordinate only instead of over all coordinates. By doing so, errors in each coordinate could be studied separately.

An informal measure of the variance of the RMSE measure between different realizations was used but not described in this document. It showed that the “variance” was quite small in all cases so the RMSE measures were reliable.

A downside with using one value for the whole sequence is that transient errors are hidden. They can only be seen by plotting the actual signals or $RMSE[k]$.

### 10.1.7 Sampling frequency

The sampling frequencies were chosen sufficiently low so that any differences between the 2D tracker and the 3D tracker should be revealed. On the other hand, the highest sampling frequency that was studied was 10 Hz, which is a bit lower than the standard definition of video rate which is 30 Hz. There might be a frequency higher than 10 Hz where the 2D tracker can no longer show the same good performance as the 3D tracker.

### 10.1.8 Studied scenarios

The scenarios that were simulated were chosen with the intention to show both similarities and differences between using a 3D tracker and using a 2D tracker. The first scenario was arranged to suit both the trackers very well, while the second scenario aimed at illustrating how the 2D tracker treated an observed decrease in speed in image coordinates differently than the 3D tracker. The last two scenarios illustrated how a manuevering target was tracked. In one of them, the manuevers occurred during some samples only and in the other, manuevers dominated the sequence.

### 10.1.9 Process modeling

In the modeling of the human target, it was assumed that the variation in height of the person was much smaller than the variation in the other directions. When there was a difference in height between the camera and the target, the 2D camera could use that difference to better estimate the range. If the difference was eliminated, there was no longer any information for the range estimation and at the same time the estimate of the height got better. Hence, the position of the camera relative to the target will affect the performance of the 2D tracking system. In a final implementation for a real-time application, the process noise must be modelled larger to cover different human behaviours. When a larger process noise was modelled, it was seen that the difference in performance was even smaller between the 2D tracker and the 3D tracker when comparing the image coordinates.

The initial position was chosen in compliance with the recommendations in [4]. The corresponding covariance matrix was chosen experimentally but with larger values for the unknown velocities than for the positions.
10.1.10 Ground-truth for collected data

For the collected data sets, the 3D tracker was compared to the 2D tracker but there was no relation to the true values. That study aimed at investigate in what situations the estimated values can be expected to deviate. It would have been interesting to also examine the absolute performance of the trackers but that would have required a ground truth which was not available here. Making a pseudo-ground truth would have been an alternative but due to the difficulties of making a good one, comparing the performance measures may not have contributed with much more knowledge than what was given by the chosen strategy.

10.1.11 Segmentation in 3D ladar data

In this thesis, the same segmentation strategy was used for the 2D tracker as for the 3D tracker. This means that if the target was detected using range segmentation for the 3D tracker, then range segmentation was used also for the 2D tracker. In this way, the trackers received exactly the same measurements of the image coordinates.

For the sequence Occlusion by truck and noise, the range image was limited to the maximum and minimum values of the range to the target that was to be tracked, in order to reduce the effect of the noise in the measurements. Such a simplification is probably not motivated in a real-time tracking system since the target can move at any range. A possible extension to the real-time tracking system is to have different gates based on predictions of the real trajectory. However, in that case, the assumption of a steady background becomes invalid and another video segmentation approach has to be taken.

It was also noticed that using the sular mode in tracking applications may lead to faulty detections and maybe even loss of track, due to a bad implementation of the peak search algorithm. If the person moves outside the range boundaries that are associated with the sular mode, no reliable range detections can be made. When using the sular mode, one must make certain that the target will remain within the specified range.

10.1.12 Singletarget vs. multitarget tracking

As was seen, the 3D camera made no big improvement of the tracking performance. Tracking with a 2D camera in a cluttered environment works almost as well as with a 3D camera - at least when it is known that at most one target is present in the sequence. As soon as another target enters the scene, the situation becomes more complex. Each detection must now somehow be associated with at most one of the targets. Different approaches can be taken to achieve this. It is possible to assign each detection to the nearest target but it might be risky when the targets are close to each other. If one target moves and change direction when it is hidden behind the other target, there are no guarantees that the correct association is made when the target again becomes visible. The association problem might be facilitated by trying to identify some features of the detected heads and then compare and match them with the correct target.
The 3D camera offers means to solve both the problems more efficiently than the 2D camera does. With all three spatial coordinates, the trajectories can more easily be predicted. For the recognition part, a 3D head model can be built from the range image and, textured with the intensity data, be compared with the target models.

10.1.13 Multisensor tracking

Association cannot only be made between a measurement and a target. When tracking with multiple sensors, there is also a problem of coordinating the measurements. The sensors most certain have different noise characteristics and maybe also different representations of the environment and the target. It is not trivial how to find out if a detection in the first sensor also corresponds to a detection in the second sensor. This is the problem that multisensor tracking addresses. With the range information of a 3D camera, occlusions can be resolved much easier and point correspondencies between different views be calculated more robust and reliable.

10.2 Future work

Based on the discoveries during this thesis work, a number of problem areas have been identified and are suggested as future extensions:

- Higher sampling frequencies than 10 Hz should be studied as the common definition of video rate is 30 Hz and the hardware will also allow higher speeds in the future.

- A study could be done to investigate the measurement noise. As seen in the previous section, the observations are processed in several steps. It would be of interest to examine how much each of them contribute to the uncertainty in the values.

- When having both range and intensity images available, a 2D PDF could be a better choice for describing the background. By using both information, a more robust video segmentation algorithm could be developed.

- The EKF is by no means optimal, other than when it reduces to the linear KF. A better alternative might be to use the BLUE estimator for nonlinear measurements (see [90]). However, the expressions for tracking in $u$, $v$ and $r$ have not yet been published and need to be derived.

- A multiple model algorithm could improve the performance when maneuvers occur. It is recommended for human motion tracking as different maneuvers are very common in this application.

- The filtered world coordinates should be more thoroughly investigated. If those are found to be good, there will be many possible applications in where the 3D tracker can gain ground.
• In the thesis, a small process noise covariance was chosen for the \( \hat{Y} \) direction since the heads were assumed to remain at an almost constant height throughout the sequences. However, in reality, this is a very restrictive assumption. Therefore, a setup with a larger process noise covariance should be studied for different scenarios. The scenarios could include humans performing actions such as jumping, climbing or walking in stairs.

• The conversion script (including the peak search algorithm) should be improved so that no range estimate can be reported in pixels where the true range is shorter than the lower limit on the range in sular mode.

• Multitarget tracking [5] is one of the areas that probably would gain the most when replacing a 2D camera with a 3D camera. Different problems could be studied, such as targets entering or exiting the scene, targets occluding each other, recognizing reentering targets and targets passing each other’s paths.

• Multisensor tracking [5, 33] is another area in where the 3D camera is proposed to show good performance. An example of combining a TOF camera with stereo-vision technology is given in [28]. It was proposed that the combination rendered higher spatial resolution and higher quality dense stereo disparity than when the techniques were used separately. This has also been tested in [32]. Such a combination might also make an improvement in tracking applications.
Bibliography


