Dipl.-Ing. Peter Holzer, Bakk.techn.

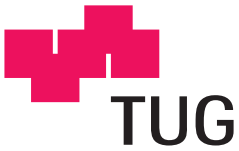
Multibody Structure and Motion from Structure and Motion in conjunction with Space Time Appearance Analysis

Dissertation

zur Erlangungung des akademischen Grades
"Doktor der Technischen Wissenschaften"
(Dr. techn.)

erreicht an der

Technischen Universität Graz



durchgeführt am

Institut für Elektrische Messtechnik und Messsignalverarbeitung

1. Begutachter und Betreuer:	Ao. Univ.-Prof. Dipl.-Ing. Dr.techn. Axel Pinz
2. Begutachter:	Dipl.-Ing. Dr.techn. Univ.-Doz. Bernhard Rinner

April 2012

Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

Peter Holzer

Acknowledgments

This PhD thesis was created from 2008 to 2011 at the Institute of Electrical Measurement and Measurement Signal Processing at the Graz, University of Technology. The financial support during that period was given by the Austrian Science Foundation (FWF project S9103), the Austrian FFG project MobiTrick (825840) and EVis (813399).

First of all, I want to spread out a big thank you to my supervisor Axel Pinz for giving me the opportunity to join the Vision based Measurement Group (VMG) and to make all this possible.

Another big thank you to Bernhard Rinner, who agreed to be the second examiner of this thesis.

Furthermore I want to thank all my colleagues at the VMG as well as the institute for the excellent working atmosphere and working conditions.

Thank you!

Abstract

This thesis addresses the extension of Structure and Motion (SaM) towards Multi-body Structure and Motion (MSaM). Beside the reconstruction of the (unknown) scene and the observer pose estimation, MSaM identifies independent foreground motion in a scene. In particular, the work described in this thesis not just identifies foreground motion, but allows the classification of specific object classes. Furthermore, appearance change information is used to harvest good features to track for the observer pose estimation. By that, MSaM is able to estimate the observer pose with a small number of point features. All the algorithms used to build and extend MSaM are tested in several experiments.

Kurzfassung

Diese Arbeit befasst sich mit der Erweiterung von *"Structure and Motion"* (SaM) hinsichtlich *"Multibody Structure and Motion"* (MSaM). Während SaM die Rekonstruierung einer (unbekannten) Szene und die Schätzung der Pose des Betrachters ermöglicht, bietet MSaM zusätzlich die Möglichkeit, unabhängige Objektbewegungen im Szenen-Vordergrund zu erkennen. Neben dem Erkennen von Objektbewegungen beschreibt diese Arbeit auch, wie Objekte klassifiziert werden können. Weiters wird die visuelle Beschreibung von *"Point features"* genutzt, um *"good features to track"* zu selektieren. Mithilfe dieser *"good features to track"* wird gezeigt, dass auch mit einer kleinen Menge an *"Point features"* die Pose des Betrachters bestimmt werden kann. Alle Algorithmen, die für MSaM entwickelt wurden beziehungseise MSaM erweitern, werden in verschiedenen Experimenten getestet.

Publications

While doing research for this thesis a number of papers were published. All of them are co-authored with my supervisor Axel Pinz and are related to the topic of this thesis. These publications are:

[HLP11] Peter Holzer, Chunming Li, and Axel Pinz. Detecting and tracking people in motion - a hybrid approach combining 3d reconstruction and 2d description. In *6th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory*, 2011.

[HP11] Peter Holzer and Axel Pinz. Mobile surveillance by 3d-outlier analysis. In *ACCV 2010 Workshops, Pt. 1*, 2011.

[HP10] Peter Holzer and Axel Pinz. Using outliers in structure and motion analysis to reconstruct foreground motion. In *Proceedings of the Computer Vision Winter Workshop 2010*, 2010.

Contents

1 Introduction

3D reconstruction of dynamic scenes and tracking of independent foreground motion play an important role in application areas such as video surveillance, robotics, or augmented reality. In mobile surveillance, moving cameras substitute stationary ones, and pose estimation of the observing camera is an essential task for such kind of systems. In cases where uncrewed robots are the only possibility to explore the environment (e.g. rescue operations), one is interested in building a map of the environment as well as the poses of the mobile robot for navigation purposes. Structure and Motion (SaM) or Simultaneous Localization And Mapping (SLAM) are vision-based approaches which address this task. They are able to reconstruct the scene and estimate the observer pose. However, both approaches require stationary scenes, i.e. their results deteriorate in case of foreground motion. SaM/SLAM are a good choice for exploring empty rooms or collapsed environments (both mainly contain a stationary scene structure). However, non-stationary places (e.g. streets with opposing traffic or crowded places) are not the field of application for SaM/SLAM. While SaM/SLAM are able to (i) reconstruct the scene and (ii) estimate the observer pose, Multibody Structure and Motion (MSAM) additionally (iii) estimates

independent foreground motion. I.e. MSaM deals with environments which consist of both, stationary background and foreground motion. Obviously, SaM/SLAM algorithms are a good starting point to build an MSaM algorithm, as the scene reconstruction and observer pose estimation is already done by them.

1.1 Motivation and Goals

We are motivated by extending SaM or SLAM to MSaM systems, i.e. find a way to analyze information to cope with foreground motion. Basically, SaM/SLAM algorithms analyze point features. SaM/SLAM algorithms divide point features into inliers and outliers. Inliers are point features located on the stationary background. Outliers are point features which do not behave stationarily, i.e. a point feature on a moving object is regarded as an outlier by SaM. State-of-the-art SaM/SLAM algorithms use inliers for the scene reconstruction and the observer pose estimation. Outlier information is detected but not processed in state-of-the-art SaM/SLAM algorithms. However, in case of foreground motion, outliers can be used to model this foreground motion. Furthermore, outliers can provide information of independently moving foreground objects. By analyzing this outlier information, we are in the position to extend conventional SaM algorithms to MSaM systems.

Existing SaM/SLAM algorithms use the entire inlier information for scene reconstruction and observer pose estimation. As a minimum of three stable point features is sufficient for the observer pose estimation when using stereo or general cameras, one can think about methods to reduce the amount of inliers for the pose estimation. By that, one can speed up the pose estimation, which is a huge benefit for online SaM/SLAM and MSaM. Additionally one gets rid of requiring more inliers than outliers. Imagine a scenario with many independently moving foreground objects. Such a scene provides fewer stationary background information than outliers in the foreground. SaM/SLAM algorithms cannot distinguish between the stationary background and the foreground motion in such a scenario, i.e. this will deteriorate the result. The observation of the appearance information of point features over time provides information on the stability of point features. I.e. we

can use this information to retrieve a subset of point features located in stationary background.

The goal of this work is to implement an MSaM system, extending an SaM by outlier analysis. The MSaM itself should be modularly expandable, such that the foreground motion detection can be extended to object classification. Beside using all the stationary background information for the observer pose estimation, another goal is to use only a subset of stationary point features, which can be derived by spatial temporal information.

1.2 My Contribution

My work contributes in four ways. First, we use SaM by Schweighofer et al. [SSP08] to identify point features which are located in the stationary background and point features which are not. With the stationary point features - inliers - we reconstruct the 3D scene and estimate the observer pose. We use the non-stationary point features - outliers - to identify and track independent foreground motion in scenes. By that, we are able to extend an SaM implementation towards MSaM. Second, we use our MSaM for object tracking. More precisely, we modularly extend our MSaM to classify moving people. Third, we search for good features to track among the inliers. By that, significantly fewer point features are sufficient for the observer pose estimation. We retrieve the good features to track by analyzing appearance change information of point features. We use the Space-Time Appearance (STA) descriptor introduced by Brkic et al. [BPSK11]. We implement an online implementation of the STA descriptor and evaluate the outcome by several histogram statistics evaluation methods. Fourth, we integrate the good features to track approach in our MSaM. We estimate the observer pose by various good features to track subsets and compare them to the estimated poses retrieved by all inliers. In the next two paragraphs I give a brief overview of the developed framework and its algorithms.

The developed framework is shown in Fig. 1.1. It extends an SaM algorithm [SSP08] to MSaM by outlier analysis. The MSaM can be used for moving object classification. In our case, we are interested in moving people.

Furthermore the framework uses the Space-Time Appearance descriptor [BPSK11] for retrieving good features to track. Together with the Space-Time Appearance information, MSaM collects subsets of stationary background point features. We pass this information back to the SaM to re-estimate the observer pose on a subset of inliers only. Blocks indicate the core modules of the framework. Blocks indicating a chapter number depict developed algorithms. The flow information is illustrated by arrows.

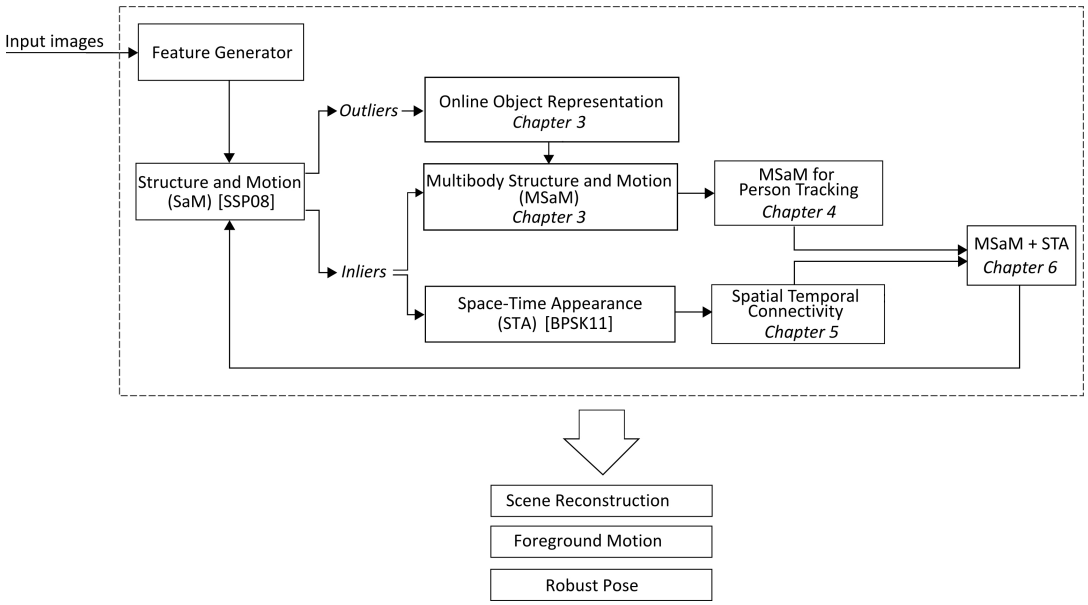


Figure 1.1: Developed framework for extending SaM [SSP08] to MSaM and good features to track detection by appearance change information with the STA descriptor [BPSK11].

Camera(s) provide images as input to a feature generator. The generated features are passed to an online SaM [SSP08] algorithm which computes the 3D scene structure and estimates an initial pose for the observer. As shown, we do not discard outlier information, but generate an online object representation of potentially moving objects. With this object representation we extend the used SaM towards MSaM. We detect, track, and classify moving people by using MSaM together with a human shape descriptor and a 2D tracker in a feedback control system. We create spatial-temporal descriptors out of the set of inliers by using the Space-Time Appearance (STA) descriptor [BPSK11]. By that, we can harvest stable, station-

ary point features. This information is passed back to the SaM algorithm which recomputes the observer's pose by these good features to track.

1.3 Related work

The research field of SaM or SLAM reaches back several decades to the problem of scene reconstruction from multi-viewpoint images. In 1952, Semple and Kneebone published an introduction to projective geometry [SK52]. A milestone for SaM occurred in 1981, when Longuet-Higgins [LH81] presented an approach to reconstruct the scene from two calibrated camera views. By finding common point features in the two projective views, the algorithm - known as the eight-point-algorithm - estimates the scene structure. Several extensions for the linear relationship to three [Har94, SiW95] and four views [FM95] followed, until Triggs [Tri95] showed in 1995, that there are no linear relationships for more than four views. In 1997, Hartley [Har97] presented the normalized eight-point algorithm, which allows to estimate the fundamental matrix from two uncalibrated camera views. For more than four views, different research fields came into existence. Basically one can distinguish between Structure and Motion (SaM) approaches and Simultaneous Localization and Mapping (SLAM).

Both, SaM and SLAM algorithms mainly rely on point features and can simultaneously reconstruct 3D scene information and observer motion. Additionally, the pose estimation approach of Ansar and Daniilidis [AD03] can handle either points or lines. Both SaM (in computer vision) and SLAM (in robotics terminology) are general approaches because they are purely geometry-based, but SLAM requires real-time performance. Due to the active control of the robots and additional sensors, SLAM can use additional information for the structure and motion estimation. Both approaches, SaM and SLAM, do not need prior model information, but their range of applications is limited to stationary scenes only. SaM/SLAM fails or produces erroneous reconstruction results in case of (dominant) independent foreground motion in the scene. A way to deal with noise or foreground motion was introduced by Nistér et al. [NNB04] in 2004. They estimate the relative

pose for a calibrated perspective camera by using the five-point algorithm [Nis04] and using a preemptive RANSAC [Nis03] to deal with noise or not-wanted foreground motion. Followed by iterative refinement, Nistér estimates the observer pose. However, he is discarding any information related to foreground motion or noise. Newcombe and Davison [ND10] introduce the combination of state-of-the-art components to solve real-time monocular dense reconstruction of cluttered natural scenes. By combining SLAM with variational optical flow, accurate depth maps are generated. This approach is computationally expensive, as it needs a Desktop PC with a GPU. SaM/SLAM algorithms can roughly be categorized into continuous tracking approaches [NNB04, DRMS07, SSP08] and keyframe-based approaches [KM07, WKR07, KM08, WKR11]. While the continuous tracking uses all frames for structure and motion estimation, the idea of the keyframe-approach is to create a static map out of frames at certain time intervals. This allows a very fast re-localization, once the map is set up. However, the keyframe-approaches are not applicable to deal with foreground motion, as the map is generated at discrete time steps. Even a continuous map-update will not solve this problem. While continuous tracking introduces an increasing error, the keyframe-approach keeps the error constant due to the a priori generated map. Specific SaM or SLAM applications such as Geiger et al. and Kitt et al. [GRU10, KGL10] are applicable for on-road vehicle motion only. They use stereo image sequences in conjunction with outlier rejection by a specific RANSAC [SFS09] for egomotion estimation.

Multibody Structure and Motion (MSaM) is an extension of SaM/SLAM. The term MSaM was introduced by Fitzgibbon and Zisserman [FZ00] in 2000. It ports the functionality of SaM/SLAM (i.e. scene reconstruction and observer pose estimation) to non-static scenes. I.e., MSaM systems are able to distinguish between stable, static background and dominant foreground motion. They further extend the functionality of SaM/SLAM. The core tasks of MSaM are (i) the detection and tracking of independently moving foreground objects by spatial-temporal trajectories, (ii) the reconstruction of the (unknown) scene structure, (iii) the pose estimation of the moving observer. In MSaM, Schindler et al. [SSW08] distinguish between algebraic methods and non-algebraic methods. Most of the algebraic meth-

ods are based on matrix factorization (e.g. [CK95, CK98, YP06]). An iterative algebraic method was introduced by Li et al. [LKSV07]. In contrast to algebraic methods, non-algebraic methods combine rigid SaM with segmentation. Non-algebraic methods handling multi-view perspective sequences in dynamic scenes are addressed by [FZ00, SSW08, OSG10]. But most existing MSaM methods are computationally expensive and thus not applicable in real-time. Schindler et al. [SSW08] and Ozden et al. [OSG10] for example cluster objects in the image plane, as this decreases the run time compared to full 3D information processing. Online MSaM systems, such as Leibe et al. [LSCG08] and Ess et al. [ELSvG08a] differ from basic SaM because their approaches are not purely geometry-based and require quite elaborated object detection algorithms. Furthermore, they are restricted to the processing of certain classes of objects only, cars and people.

Basing on the concept of point features, one has to think about the quality of the tracked point features. In 1994, Shi and Tomasi [ST94] came up with the idea of analyzing the appearance change of point features over time. They introduced a measure of feature dissimilarity. By that, they analyze the space-time information of a point feature and are able to identify good features to track. The Space-Time Appearance (STA) descriptor of Brkic et al. [BPSK11] introduces a concept similar to Dollár et al. [DRCB05], Laptev and Perez [LP07], and Luo et al. [LKZF10]. It allows the retrieval of appearance change information of point features.

In this thesis we develop an MSaM system which is - compared to the non-factorization and factorization-based MSaM algorithms - applicable in real-time. Our MSaM system processes 3D information and is not restricted to a certain class of objects. Instead, it works for any rigidly moving foreground object. Additionally we follow the idea of Shi and Tomasi, identifying good features to track [ST94]. We introduce evaluation methods of space-time information to identify certain subsets of static background point features for the observer pose estimation. Compared to the full set of background point features, the subsets contain significantly fewer point features but still provide useable results on observer pose estimation.

1.4 Outline

First, chapter 2 gives an overview of the used datasets. Chapter 3 introduces the *Online Rigid Object Representation* together with the *Multibody Structure and Motion* (MSaM). With this approach, we are able to reconstruct independent foreground motion. In chapter 4, the introduced MSaM algorithm is extended by appearance-based detectors and trackers to classify moving objects. In detail, we show how to detect and track moving people. In chapter 5 we are focusing on the Space-Time Appearance (STA) descriptors of point features. We analyze the appearance change of point features over time. By that, we try to find *Good Features To Track* (GFTT) for pose estimation. Chapter 6 fuses MSaM with the Space-Time Appearance (STA) descriptors. Having different subsets of *Good Features To Track* (GFTT), we are able to decrease the required inlier/outlier ratio considerably, i.e. we can reliably estimate the observer pose out of a subset of the background information only. Finally, chapter 7 concludes this thesis, providing a summary and discussing the results of this work.

2 Datasets

Ideally, it should be possible to use any public dataset available, including uncalibrated and monocular videos. However, our MSaM requires calibrated stereo datasets. This is due to the underlying SaM by Schweighofer et al. [SSP08], which does not accept uncalibrated or monocular datasets. By that, we are limited to few available public stereo datasets. But such stereo datasets seem to experience an upswing, especially high resolution datasets in combination with groundtruth data (e.g. GPS information).

In this work, also monocular datasets are used in chapter 5 (among stereo datasets). This is possible, as chapter 5 does not rely on the MSaM introduced in chapters 3 and 4. It shows analysis on space-time appearance changes of point features. However, the performed analysis is used together with MSaM in chapter 6, where solely stereo datasets are used again.

There exist many monocular datasets. Most of them are related to person detection (Caltech pedestrian dataset [Cal], INRIA person dataset [Dal05], KTH action dataset [SLC04], Daimler pedestrian datasets [EG09]). A few stereo datasets for person detection are also available (ETH stereo datasets for multi-person track-

ing [ELSvG08b], Middlebury stereo dataset [SS02]). The ETH datasets contain wide angle images. The sequences consist of solely forward motion, which causes moving objects (people) to disappear rapidly. The Middlebury stereo dataset [SS02] consists of a maximum of seven views per sequence without calibration information. From the public video surveillance datasets (BEHAVE Interactions Test Case Scenarios [BF09], PETS datasets [PET]), most do not provide a moving observer. Another interesting dataset is introduced by Aanæs et al. [ADP12]. It provides 60 scenes with high-resolution images and precise ground truth information for both, camera positioning and the 3D surface. Additionally, the scenes were recorded with different artificially relighting conditions. However, the camera positions are too diverse. It is not possible to use them for continuous camera motion, i.e. it is not possible to use this dataset in chapter 5.

Beside the monocular and stereo datasets recorded by our own, we decided to use the KIT datasets [Gei]. The KIT datasets provide high-resolution stereo video sequences recorded from a moving car in Karlsruhe. Additionally, the datasets provide GPS observer pose information. Having both, GPS information and high-resolution stereo datasets, is the main reason why we chose the KIT datasets.

2.1 VMG_Bike_01

This monocular dataset consists of 76 color images and covers a street sequence. Each image has a resolution of 1280 x 960 pixels. The sequence was recorded by our own with a wide-angle camera mounted on the helmet of the biker. The moving observer - a biker - is overtaken by a car. Figure 2.1 shows frames 1, 24, and 72 of the sequence. For the space-time appearance change analysis of point features, this sequence is interesting in terms of the different point features, located on the static background passing by (either nearby or far from the observer) and on the overtaking car.

Figure 2.1: Frames 1, 24, and 72 of experiment VMG_Bike_01.

2.2 VMG_Bike_02

This monocular dataset consists of 300 color images with a biker as moving observer. Again, the sequence was recorded by our own with a wide-angle camera mounted on the helmet of the biker. Each image has a resolution of 1280 x 960 pixels. The sequence shows the observer - a biker - driving along a winding street. Figure 2.2 shows frames 1, 210, and 281 of the sequence. This sequence provides detailed information on the appearance change due to the length of the sequence. We chose this sequence, because of its changing light conditions as well as the long fence appearing on the right hand in the sequence (see 2.2, frame 210).

Figure 2.2: Frames 1, 210, and 281 of experiment VMG_Bike_02.

2.3 VMG_Lab_01

This stereo dataset consists of 180 black and white images per camera which were recorded by our own. The image dimensions are 752 x 480 pixels. The frames were captured with two μeye 1220C USB cameras with 6.5 mm Cosmicar lenses mounted

on a stereo rig and a baseline of approximately 30 *cm*. We used a constant frame rate of 20 Hz.

The sequence is recorded by a moving observer and shows an indoor scene containing textured background and two moving objects in the foreground, a toy cow and a cup. While the cup is moved from the left to the right by pulling a cord, the toy cow is moved behind the cup from the right to the left. Figure 2.3 shows frames 42, 107, and 135 of the sequence.

We selected this experiment, because it shows the potential of our MSaM in a controlled setup. It consists of multiple foreground motion (cup and toy cow), textured background, and a moving observer.

Figure 2.3: Frames 42, 107, and 135 of experiment VMG_Lab_01.

2.4 VMG_Person_01

This stereo dataset consists of 99 black and white images per camera which were recorded by our own. The image dimensions are 752 x 480 pixels. The frames were captured with two μeye 1220C USB cameras with 6.5 *mm* Cosmicar lenses mounted on a stereo rig and a baseline of approximately 30 *cm*. We used a constant frame rate of 20 Hz.

The sequence consists of a moving observer focusing on a moving person, which moves from the left to the right. Figure 2.4 shows frames 42, 58, 85, and 92 of the sequence.

Due to the lack of stereo people datasets, we decided to record a couple of datasets by our own. This sequence is interesting, as the observer focuses on a sidewards passing person. I.e., in contrast to the ETH datasets [ELSvG08b], sidewards motion of the observer is considered.

Figure 2.4: Frames 42, 58, 85, and 92 of experiment VMG_Person_01.

2.5 VMG_Person_02

This stereo dataset consists of 161 black and white images per camera which where recorded by our own. The image dimensions are 752 x 480 pixels. The frames were captured with two μeye 1220C USB cameras with 6.5 mm Cosmicar lenses mounted on a stereo rig and a baseline of approximately 30 cm. We used a constant frame rate of 20 Hz.

The sequence consists of a moving observer observing a person walking towards the camera. Figure 2.5 shows frames 28, 93, 119, and 148 of the sequence.

This experiment was chosen due to the scale-change of the observed person. In contrast to the ETH dataset [ELSvG08b], the observer is not moving fast forward, i.e. less motion blur occurs and the observed person is long enough in the scene to enable tracking.

Figure 2.5: Frames 28, 93, 119, and 148 of experiment VMG_Person_02.

2.6 VMG_Person_03

This stereo dataset consists of 55 color images per camera. The image dimensions are 752 x 480 pixels. The frames were captured with two μeye 1220C USB cameras with

6.5 *mm* Cosmicar lenses mounted on a stereo rig and a baseline of approximately 30 *cm*. We used a constant frame rate of 20 Hz.

The moving observer focuses on a person far from the observer. The person walks from right to the left, partially occluded by bikes. Figure 2.6 shows frames 1, 26, and 50 of the sequence.

We chose this sequence to demonstrate the detection and tracking capabilities of our MSaM. The observed person is uniformly colored, small scaled, and partially occluded. Still we want to demonstrate, that detection and (partial) tracking is possible by outlier analysis.

Figure 2.6: Frames 1, 26, and 50 of experiment VMG_Person_03.

2.7 KIT_Seq_01

This experiment is a subsequence of the Karlsruhe stereo dataset [Gei] sequence 2009_09_08_drive_10. The complete scene consists of 180 black and white images per camera. Frame 1 of this experiment match with frame 1 of the sequence 2009_09_08_drive_10. The image dimensions are 1344 x 372 pixels. The sequence was recorded by Pointgrey Flea2 firewire cameras and stored as rectified images. The KIT dataset sequences were recorded with a frame rate of 10 Hz. The stereo baseline is approximately 57 cm. The GPS information was collected with an OXTS RT 3000 GPS/IMU system.

The scene contains one moving observer, one car ahead, and one pedestrian. Figure 2.7 shows frames 16, 51, and 117 of the sequence. Beside the high resolution of the images and the available GPS information, we are interested in this sequence

because of it's multiple moving objects (person and car) and the fast moving observer.

Figure 2.7: Frames 16, 51, and 117 of experiment KIT_Seq_01.

2.8 KIT_Seq_02

This experiment is a subsequence of the Karlsruhe dataset [Gei] sequence 2009_09_08_drive_10. The sequence of this experiment consists of 85 black and white images per camera. Frame 1 of this experiment is frame 715 of the sequence 2009_09_08_drive_10. The image dimensions are 1344 x 372 pixels. The sequence was recorded by Pointgrey Flea2 firewire cameras and stored as rectified images. The KIT dataset sequences were recorded with a frame rate of 10 Hz. The stereo baseline is approximately 57 cm. The GPS information was collected with an OXTS RT 3000 GPS/IMU system.

The scene contains one moving observer and a group of three pedestrians. Figure 2.8 illustrates frames 2, 39, and 65 of the sequence. We are interested in this scene, as a group of people is observed. As the people move jointly, we are interested how our MSaM is clustering the objects, i.e. detecting the group vs. the individual people.

Figure 2.8: Frames 2, 39, and 65 of experiment KIT_Seq_02.

2.9 KIT_Seq_03

This experiment is a subsequence of the Karlsruhe dataset [Gei] sequence 2010_03_09_drive_82. This experiment's sequence consists of 98 black and white images per camera. Frame 1 of this experiment is frame 15 of the sequence 2010_03_09_drive_82. The image dimensions are 1344 x 372 pixels. The sequence was recorded by Pointgrey Flea2 firewire cameras and stored as rectified images.

The KIT dataset sequences were recorded with a frame rate of 10 Hz. The stereo baseline is approximately 57 cm. The GPS information was collected with an OXTS RT 3000 GPS/IMU system.

The scene contains one moving observer, one car passing from left to right, and one car passing from right to left. Figure 2.9 illustrates frames 21, 46, and 74 of the sequence. The observer moves fast and at a 90°-turn, motion blur occurs. This scene was chosen to show the capabilities of our MSaM and the impact of the image quality on the MSaM results.

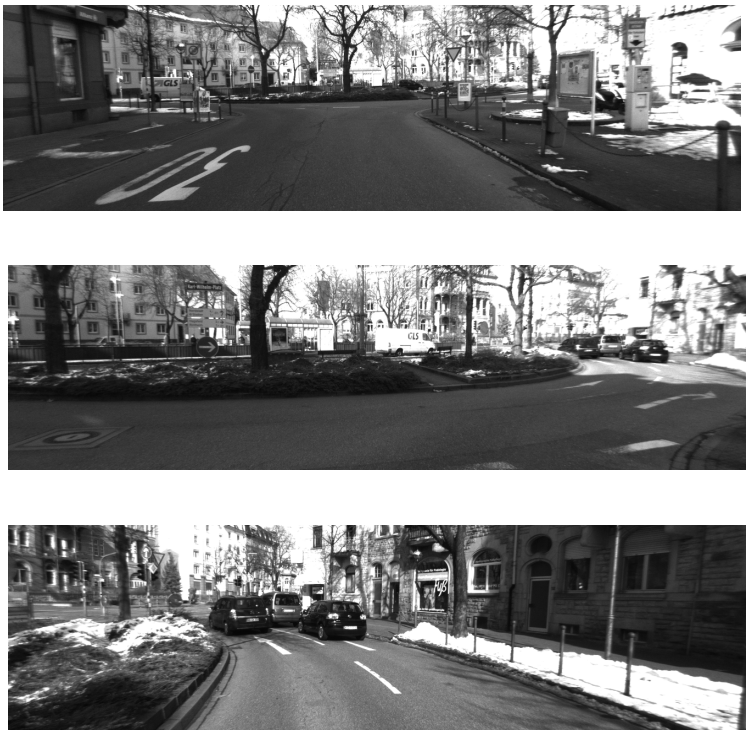


Figure 2.9: Frames 21, 46, and 74 of experiment KIT_Seq_03.

2.10 KIT_Seq_04

Again, this experiment consists of a subsequence of the KIT dataset. The experiment's sequence consists of 230 black and white images per camera. Frame 1 of this experiment is frame 272 of the sequence 2010_03_09_drive_82. The image dimensions are 1344 x 372 pixels. The sequence was recorded by Pointgrey Flea2 firewire cameras and stored as rectified images. The KIT dataset sequences were recorded with a frame rate of 10 Hz. The stereo baseline is approximately 57 cm. The GPS information was collected with an OXTS RT 3000 GPS/IMU system. It shows static, textured background, two moving trams, one biker, and four people. Figure 2.10 shows frames 28, 62, and 115 of the sequence. We chose this sequence, as it bundles properties of multiple datasets described above. Motion blur occurs due to multiple moving trams and people moving individually and in groups.

Figure 2.10: Frames 28, 62 and 115 of experiment KIT_Seq_04.

3

Extending Structure and Motion towards Multibody Structure and Motion*

Multibody Structure and Motion (MSaM) extends Structure and Motion (SaM) or Simultaneous Localization and Mapping (SLAM) algorithms. SaM/SLAM algorithms provide (i) the reconstruction of the (unknown) scene structure and (ii) the pose estimation of the moving camera (observer). MSaM extends this approach by detecting and tracking foreground motion. In general, MSaM describes the problem of simultaneously solving the segmentation of independently moving objects (including the observer) along with the motion estimation for each object. Schindler et al. [SUW06, SSW08] apply MSaM to image sequences containing rigid object motion. A discussion on practical issues of realistic sequences is introduced by Ozden et al. [OSG10].

In this chapter we develop an MSaM algorithm extending an existing SaM algo-

*This chapter builds on a paper that originally appeared in the *Proceedings of the ACCV 2010 Workshops, Pt. 1.* Holzer P. and Pinz A.: *Mobile Surveillance by 3D-Outlier Analysis*, ACCV 2010 Workshops, Part 1, 2011, pages 195 - 204.

rithm that requires point correspondences. We model rigid object foreground motion by SaM outlier analysis. We show that outliers contain information of potentially moving foreground objects. The main contribution of this work is given by the rigid object representation for online tracking in section 3.5.

First, in section 3.1 we specify the required scene settings. Second, in section 3.2, we explain the design and functionality of the algorithm. The three subsections 3.3-3.5 explain the information gathering by the used SaM [SSP08], motion-clustering, and maintenance of the online rigid object representation, the core of the MSaM algorithm. The purely geometry-based approach is extended by descriptors in section 3.6. We highlight some implementation issues in section 3.7. In section 3.8 we present various experiments including a controlled indoor setup, scenes with moving people, and real-life street scenes observed by a moving car. We also show an experiment demonstrating the limits of our MSaM. Finally, we conclude with section 3.10.

3.1 Scene Settings

In order to apply the MSaM algorithm, the scene needs to meet a number of criteria (fig. 3.3):

1. Foreground motion is allowed, but there is a set P of stable point features corresponding to the static scene, i.e. $P = \{\boldsymbol{x}_S|\text{static}\}$.

2. There exits exactly one observer in the scene that can view a subset of P. The observer can be any kind of general camera, e.g. a calibrated stereo-rig.

3. There are $n \geq 0$ objects in the scene. if $n > 0$, any point feature $\boldsymbol{x}_i \notin P$ contributes to an object j

$$\boldsymbol{x}_i \rightarrow O_{j,i} \quad j = 1 \ldots n \tag{3.1}$$

I.e., we try to classify any point feature $\boldsymbol{x}_i$(scene) viewed by the observer either into the set of stationary background features P or to an object O_j (eq. 3.2).

$$\boldsymbol{x}_i \widehat{=} \begin{cases} P & \text{stationary background} \\ O_{j,i} & \text{independently moving foreground} \end{cases} \tag{3.2}$$

3.2 Algorithm Overview

We use 3D-outlier information, gathered by an SaM algorithm, as initial per-frame input of our online algorithm. Meanshift-clustering separates the outliers into sets of moving objects. At each step, the current clustering information has an impact on future clustering, which prevents point features to change randomly between nearby objects, i.e. the clustering procedure is a feedback control system.

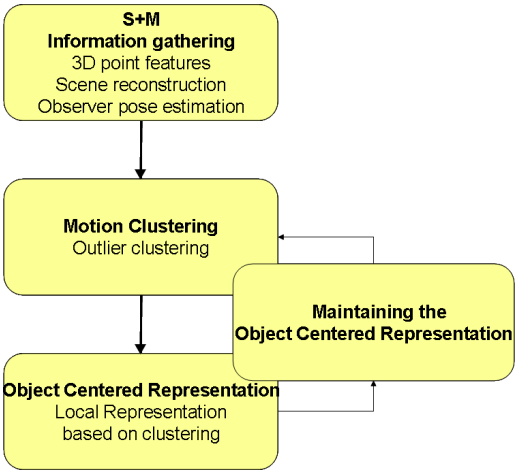


Figure 3.1: Schematic illustration of our MSaM algorithm.

A stable object centered representation is computed per object, which constitutes the core of our algorithm. Fig. 3.1 illustrates the general structure of the algorithm. Based on a stable reference point, the object centered representation allows motion analysis and enables motion prediction based on position, velocity, and acceleration, using a 9-state Kalman Filter. For each foreground object, rotation and translation information is gained over the tracked time. Finally, to solve loop-closing, an update routine for previously lost and re-appeared point features (e.g. after occlusion or

self-rotation) is implemented.

Our algorithm consists of three parts: (i) SaM information gathering briefly discussed in section 3.3, (ii) online clustering of outlier data as introduced in section 3.4, and (iii) an object-centered representation explained in section 3.5.

3.3 Information gathering by SaM

We start by reconstructing the scene and estimating the observer pose using an SaM algorithm that requires point correspondences. Typically, SaM reconstructs the static scene from inliers, i.e. stable and non-moving point features. Any other point features - outliers - are not processed further. Outliers are any point features which do not fit into the pattern of inliers. This may include point features generated by false point correspondences as well as point features located on the independent moving foreground. Conventional SaM algorithms can reconstruct the static scene and the observer pose in case of up to 50% outliers. We extend the functionality of such SaM algorithms by analyzing the outliers as candidates for independent foreground motion.

Our MSaM algorithm can be used with any SaM algorithm that (i) reliably reconstructs 3D-inlier and outlier information and (ii) provides 3D-inlier and outlier information. We build on the continuous tracking SaM algorithm by Schweighofer et al. [SSP08] as it gives us access to both, 3D-inlier and outlier information.

The used SaM [SSP08] is applicable with a general camera model [GN01]. I.e., it is not limited to a specific camera model (e.g. perspective camera), but can be used for any camera model which satisfies the general camera model. In brief, the general camera model assumes that light travels along a line into the camera. Therefore, the sensor position, its intensity, and the direction of the light ray are used as information, replacing the usage of a pixel, which solely contains the sensor position and its intensity. The used SaM implements an optimization algorithm, similar to bundle-adjustment. However, unlike bundle-adjustment, it uses a cost function based on a general camera model.

3.4 Motion clustering

We are focusing on groups of consistently moving outliers as they can represent independently moving foreground objects. Thus, each outlier provides a hypothesis for independent foreground motion.

We start by building 3D-trajectories for each outlier. Once having 3D-coordinates for a minimum of five frames per outlier, the trajectory is passed to a clustering table. One column of the clustering table has the form $[X_t, X_{t-1}, \ldots, X_{t-4}]^T$ where X is the 3D-coordinate and t is the current frame and $t-1$ is the previous frame. Fig. 3.2 illustrates examples of 3D-outlier trajectories gathered by SaM in experiment VMG_Lab_01 (cf. fig. 3.8). For better legibility only the x/z-coordinates of the 3D-trajectories are plotted. The prominent trajectories indicate two moving objects in the scene. In the back, some short trajectories are shown, which are outliers, too. But these outlier trajectories are rather short and isolated from each other.

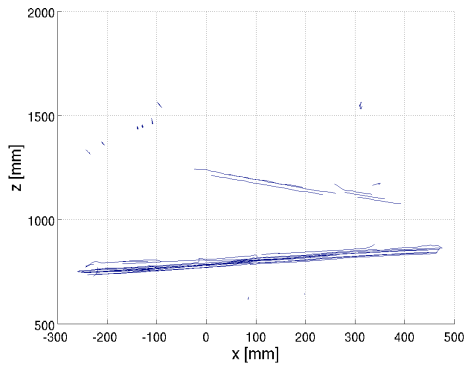


Figure 3.2: x/z-plot of 3D-outlier-trajectories of experiment VMG_Lab_01.

Once the clustering table is set up, it is passed to a Meanshift clustering algorithm [FH75, CM02]. To gain hypotheses for moving objects, we cluster the passed 3D-information online by position and by trajectory behavior. We do not use mean-corrected coordinates, as we want to preserve the trajectories' positions in the scene. As a result, most of the short outlier trajectories in the back of fig. 3.2 are discarded. The longer trajectories are clustered to two separate moving objects.

3.5 Online Rigid Object Representation

In this section, we introduce a stable, purely geometry-based object representation that enables us to model the object behavior online, without prior knowledge of the scene. We obtain the object coordinate system by establishing difference vectors of available neighboring point features per object. This local object coordinate system can move independently with respect to the global scene coordinate system that is attached rigidly to the static background structure. Fig. 3.3 illustrates a global scene coordinate system X_s attached to the static background structure, two local coordinate systems X_k and X_j attached to each independently moving cluster of outliers, and one moving observer.

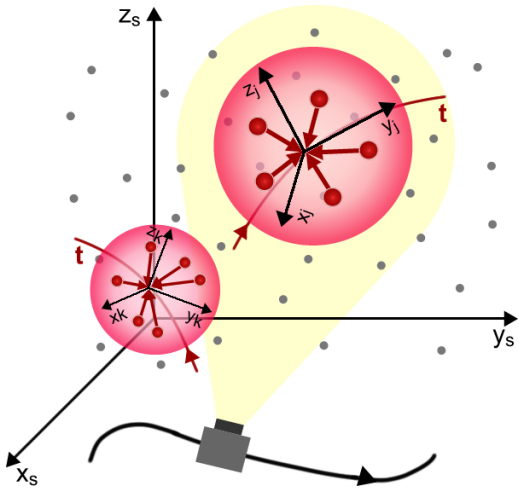


Figure 3.3: Scene representation: one global scene coordinate system, one local coordinate system per object, and one observer in the scene.

The point features per object are determined by the online clustering-procedure described in section 3.4. When a cluster contains enough point features, a reference point is initially computed by the mean values of the global scene coordinates of the cluster's point features. The motion, i.e. rotation and translation of the object, is estimated according to each object's reference point. Thus, in contrast to point cloud matching, we link the object's representation and motion to one single reference point per object.

3.5.1 Initialization of the Object Centered Representation

Once $t \geq 3$ (at least three non-colinear points are needed for pose estimation when using stereo or general cameras [Hor87]) point features are available on an object, the reference point is computed by the mean values of the available point coordinates. Thus, it coincides with a first rough estimation of the object's center of gravity. We define this reference point as the object's coordinate center and store its position in scene coordinates. The coordinates of all point features on the object are stored in object centered coordinates, i.e. the difference vectors $\Delta \boldsymbol{d}_i$ from point feature i to the reference point.

In case of temporary invisibility of an object, a 9-state Kalman Filter (KF) provides prediction for the reference point based on its position, velocity, and acceleration. However, the KF is not able to provide an estimation for the rotation, only the translation can be estimated.

3.5.2 Update

Once the initialization process has been successfully finished, the updating procedure is continuously performed. The update procedure consists of three principal tasks:

- maintaining a common point feature set in two successive frames,
- managing a confidence measure to distinguish between "active" and "inactive" point features, and
- estimating the self-rotation $\Delta \boldsymbol{R}$ between two successive frames.

Using this update procedure, we can compute a stable local object representation per object. The three tasks of the update procedure are explained in detail below.

Common Point Feature Set

In every subsequently processed frame, each point feature on an object provides one hypothesis for the reference point. A hypothesis is a vector retrieved by the difference between a point feature's current scene coordinate and the reference point's scene coordinate computed in the previous frame. We require a feature set containing the same point features in two successive frames for pose estimation. Required point features can disappear over time (e.g. self-occlusion due to object rotation). If all point features of an object disappear, no point feature provides a hypothesis. Then, we can only estimate the motion of the object by the KF. However, if a subset of the object disappears temporarily, we still can compute the reference point. The remaining $r \geq 1$ point features provide valid hypotheses for the reference point. Additionally, at every new frame j, available new point features provide new hypotheses for the object's reference point.

Confidence Measure

To provide a stable reference point, we need a confidence measure that can distinguish between "active" and "inactive" point features. We allow "active" point features to provide hypotheses for the reference point computation, whereas "inactive" point features must not. For each object, we generate the confidence measure by computing the median in $\boldsymbol{x}$, $\boldsymbol{y}$, and $\boldsymbol{z}$ direction of the reference points derived by all hypotheses (i.e. median of 3D-coordinates of visible point features on the object). Then, a certain range around the median values is chosen. In our case, this is 2 times the standard deviation. All point features within this adjusted range in all three directions are set "active". All other point features are set "inactive" and do not contribute to the reference point computation. If all point features are "inactive", i.e. outside the adjusted range, we increase the range stepwise, until a valid hypothesis emerges. The new reference point is then computed as median of all "active" hypotheses. Once the new reference point has been computed, the difference vectors of all point features on the object are updated.

Estimation of the Self-Rotation $\Delta \boldsymbol{R}$

The self-rotation $\Delta \boldsymbol{R}$ between two successive frames has to be estimated for each object. The difference vectors $\Delta \boldsymbol{d}_i$ created in the initialization process do not provide information on rotation. To estimate the rotation $\Delta \boldsymbol{R}$ between the current frame j and the previous frame $j-1$, we proceed as follows:

1. In frame $j-1$, the reference has already been established by $n \geq 3$ neighboring point features. Thus, for each point feature $i \in n$, a difference vector $\Delta \boldsymbol{d}_i$ exists, indicating the position of the point feature w.r.t. the reference point (see fig. 3.4(a)).

2. In frame j, we know the scene coordinates of the same n point features $i_1 \ldots i_n$ found in frame $j-1$. Furthermore, we assume the position of the reference point unchanged. This assumption is required, as we need to build the difference vectors for each point feature, which is not possible with an unknown reference point. In case of an object motion between frames $j-1$ and j, the scene coordinates of the point features will be slightly different (see fig. 3.4(b)). I.e., the difference vectors of each point feature are different in frame $j-1$ and j. We can compute the rotation $\Delta \boldsymbol{R}$ from these 3D point correspondences between frames $j-1$ and j according to Horn [Hor87].

3. The rotation with matrix $\Delta \boldsymbol{R}$ is applied to all point features' difference vectors to obtain the relative position to the reference point in frame j. We update both "active" and "inactive" point features.

4. Now, each point feature of the object provides a hypothesis for the new position of the reference point considering the self-rotation. The mean of all hypotheses per object is used as new reference point.

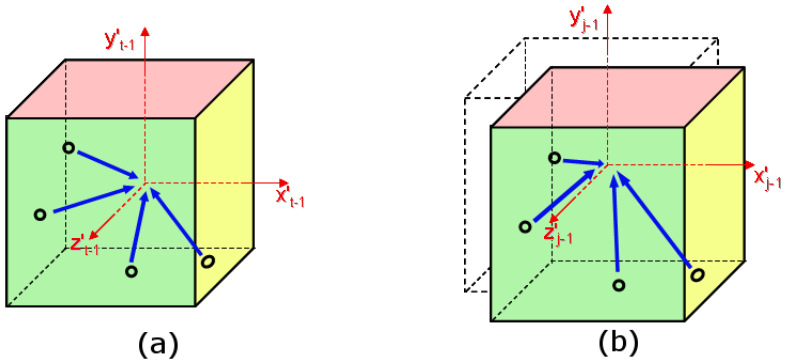


Figure 3.4: Estimation of the self-rotation $\Delta \boldsymbol{R}$. The local object coordinate system and the difference vectors are already established in frame $j-1$ **(a)**. In frame j the global scene coordinates of the point features are different to frame $j-1$, the reference point is assumed unchanged **(b)**. This results in different difference vectors per point feature. Using the algorithm of Horn [Hor87], the rotation between frames $j-1$ and j can be estimated.

3.6 Geometry and Descriptors

To handle loop-closing, we extend this purely geometry-based algorithm by descriptors that are generated for each point feature on an object. Apparently, one could apply descriptors also to the stationary background point features. By that, one would achieve loop-closing for the stationary scene. Within the scope of this work, we apply the descriptors on foreground motion only (descriptive information of stationary point features is introduced in chapter 5, but is not used for loop-closing). Furthermore, we keep track of the visibility of all point features. In case of invisibility, continuous difference vector update can not be performed. Instead, a position estimation routine is used.

We use the Scale-Invariant Feature Transform (SIFT) descriptor [Low04, VF08] which is invariant to rotation and scale. SaM performs continuous tracking, so that a temporarily lost point feature is not recognized on re-appearance. Providing (i) a stable reference point, (ii) a reliable object coordinate system, and (iii) descriptive information by SIFT, re-mapping is possible. Upon re-mapping, descriptor and difference vector of a point feature are updated, as both are similar, but not equal. Fig. 3.5 illustrates this process.

First, the object is detected. The SIFT descriptor is computed for all point

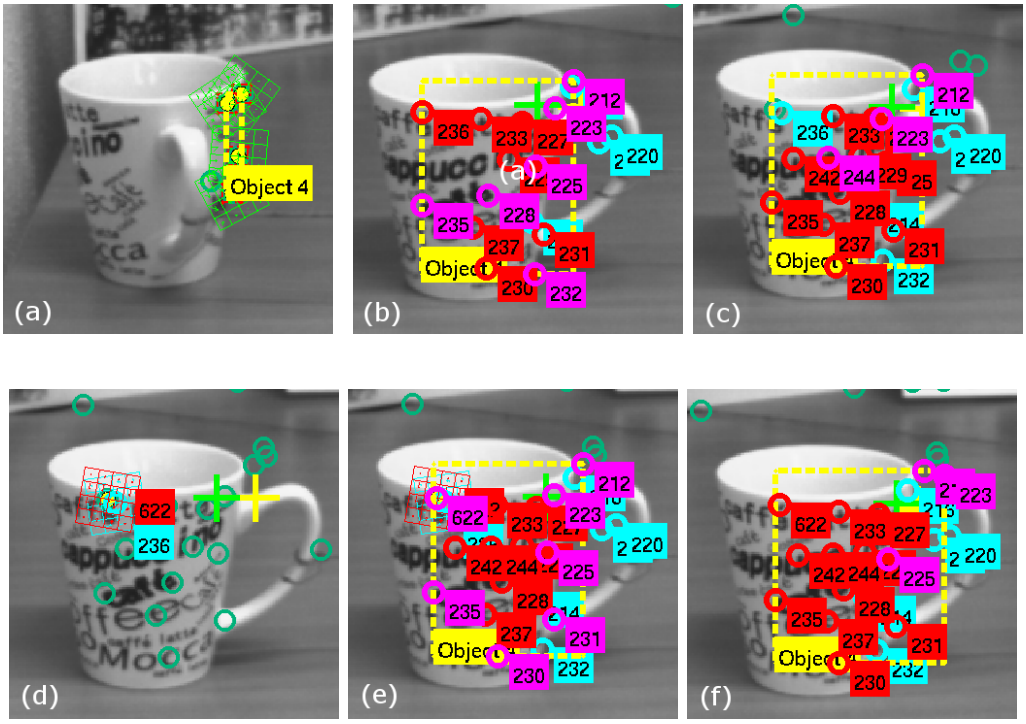


Figure 3.5: Re-detection and re-mapping of a previously lost point feature by SIFT. When the object is detected, the SIFT descriptor is computed for all point features assigned to this object (3.5(a)). Different states of point features: estimated (cyan) due to prior disappearance, actively (red) or inactively (magenta) contributing (3.5(b). The point feature 236 is no longer visible but is still estimated (3.5(c)). Appearance of a new point feature 622 on the object (3.5(d) and 3.5(e)). Point feature 622 is similar to the previous lost point feature 236; point feature 236 is updated to ID 622 and set visible again (3.5(f)).

features assigned to this object (fig. 3.5(a)). During tracking, more point features appear on the object. Point features can contribute either actively (red), inactively (magenta), or they are estimated (cyan) due to prior disappearance (fig. 3.5(b)). In fig. 3.5(c)), point feature 236 is no longer visible, but we can still estimate its position (cyan). Fig. 3.5(e) illustrates the appearance of a new point feature 622 on the object. The new point feature is compared with all previously disappeared and still estimated point features.

The matching is done by VLFeat [VF08], implementing the matching method introduced by Lowe [Low04]. Additionally, we use a threshold on the 3D distance

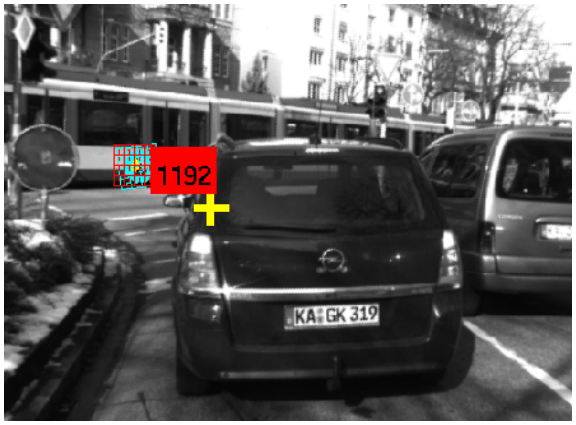


Figure 3.6: Example of re-detection and re-mapping in experiment KIT_Seq_04. A previously lost point feature (yellow circle) is re-detected by descriptor and location comparison. Descriptor of lost point feature (yellow), descriptor of re-detected point feature (red).

between the two matching point features. We use different thresholds between 0.15 and 3 m. The threshold depends on the scene, i.e. the distance of a point feature to the observer or the time of estimating a lost point feature. For closer point features and short term estimation of a lost point feature, a smaller threshold is sufficient. The longer the estimation of a lost point feature, the more the estimation will deteriorate from the actual position, i.e. a larger threshold is required.

In our case, point feature 622 is similar to 236. I.e. both have a similar descriptor and their difference vectors are similar (the 3D distance threshold is 0.15 m). Point feature 236 is now set visible again. Its ID is updated to 622 and the new descriptor (red SIFT in fig. 3.5(d)) replaces the old descriptor (cyan SIFT in fig. 3.5(d)). Now, point feature 622 - former 236 - contributes to the object coordinate system computation, either actively or inactively.

Fig. 3.6 shows the update of a point feature in experiment KIT_Seq_04. A lost point feature (cyan) is re-mapped to a new feature (red) due to similar locations and descriptors.

3.7 Implementation Details

The object centered representation for independently moving objects is implemented as a tree with depth of 2 (see fig. 3.7). The reference point and its global scene coordinates are stored as root node. All the point features per object are stored as children of the reference point. Each child of the root stores its coordinates w.r.t to the reference point (pos), whether it is visible or not (visible), whether it meets the rules of the confidence measure ("active" or "inactive") (active), and its descriptor information (descriptor). In case of a re-mapping by descriptors, the old IDs as well as the old coordinates w.r.t. local object coordinate system are stored (past_ids and past_pos respectively).

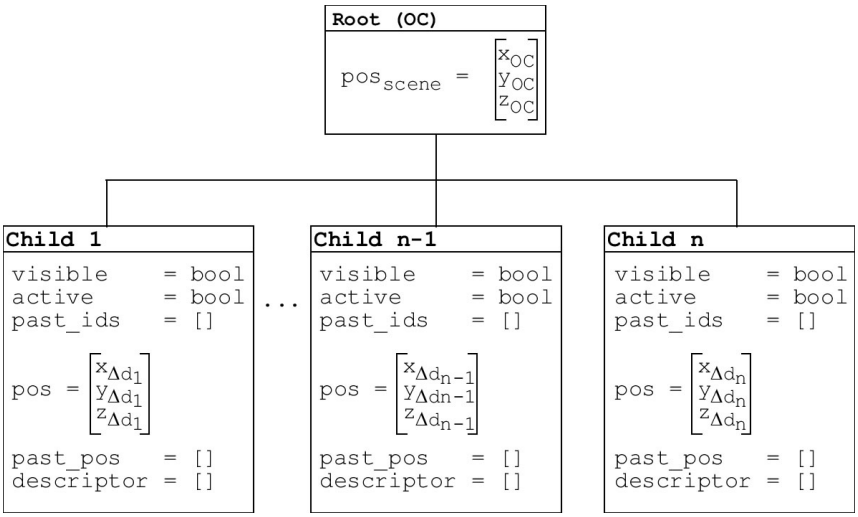


Figure 3.7: Schematic illustration of the object centered representation data structure.

Once the outlier information is clustered into separate clusters, the gained cluster IDs may not match the IDs in the previous frame. Meanshift clustering does not provide the same object order each time it is called. Therefore, our algorithm has to provide a re-order mechanism every time Meanshift clustering has been called. In our implementation we use two parameters to adjust this re-ordering mechanism, (i) a certain minimum common point feature subset between two successive frames and (ii) a threshold indicating after how many frames an object may be invisible. The former parameter ensures that only objects with a minimum common feature

set (compared to all previous objects) are mapped to existing object IDs. If this minimum common feature set is obtained, the new cluster (i.e. object) is mapped to the previous object with the maximum matching point features set. The latter parameter is responsible how long we allow a disappeared object to be mapped to an existing ID. One can think of it as a kind of ID reservation. In our case, the former parameter is set to 0.3, i.e. at least 30 percent of an object's point feature set have to overlap with the point feature set of an object in the previous frame. The latter parameter is set to 1 frame. Listings 3.1 to 3.3 show the pseudo code for the re-mapping routine.

```
1  for id = 1:length(objects)
2     for pid = 1:length(previous_objects)
3       common_subset = ismember(objects(id),previous_objects(pid));
4       weight = sum(common_subset,1)/elemsOn(objects(id));
5       correspondence(id,pid) = weight;
6    end
7    ...
8  end
```

Listing 3.1: Finding object correspondences

```
1  if sum(correspondence(objects(id),:)) > MIN_THRESH & notInFrames(object
       ) < MAX_INVISIBLE_THRESH
2    [val idx] = max(correspondence(objects(id),:));
3    if ~isLocked( previous_objects(idx) )
4      mapping(objects(id),2) = idx: %map previous_object with id idx
5                                    %to current object
6    else
7      mapping(objects(id),2) = -1;  %no mapping assigned, object ID
8    end                             %is locked
9  else
10     mapping(objects(id),2) = -1;    %no mapping assigend
11 end
```

Listing 3.2: Object mapping

First, for a new object in frame j the algorithm determines the common point feature subsets of the new object with all objects in the previous frame $j-1$. We skip any object which is invisible for more than MAX_INVISIBLE_THRESH frames. The amount of common point features is weighted by the amount of point features of the new object. It is stored in a correspondence matrix (listing 3.1). Next, we verify whether a new object has at least MIN_TRESH common point features with any of the objects in frame $j-1$. The new object is mapped to the object with which it shares the maximum common point feature set (see listing 3.2). If there is any new object in frame j, we add it after the matched objects (listing 3.3).

```
1 if length(objects)>length(previous_objects)
2     for k=length(prev_objects)+1:length(objects)
3         mapping(k,1)=k;
4         mapping(k,2)=-1; % -1 means no mapping
5     end
6 end
```

Listing 3.3: Mapping of new objects

3.8 Experiments

In this section, we show five selected experiments. The first experiment, VMG_Lab_01, is an indoor setup we recorded to illustrate the basic proceedings and results. Experiment VMG_Person_01 and experiment VMG_Person_02 are also sequences recorded at our own but consist of outdoor real-world data. More precisely, they contain a moving observer and a moving person. Experiments KIT_Seq_01 and KIT_Seq_04 are subsequences taken from the Karlsruhe (KIT) dataset [Gei] and contain street sequences. Detailed information on the used datasets can be found in chapter 2.

The algorithm has been tested on an Intel Core 2 Quad PC with 2.8 GHz and 1 GB RAM using Matlab 7.6 on a 32 bit version of Ubuntu 9.10. However, Matlab was run on one core only. Online processing depends on the number of objects (i.e. clusters) and point features (descriptor generation) in the scene.

In all experiments we use the MSaM inlier data for estimating the observer pose as well as the stationary scene structure. We expect our MSaM algorithm to identify the moving foreground motion by outlier analysis. At least subparts of each moving object are expected to be detected.

3.8.1 Experiment VMG_Lab_01

This experiment uses the dataset VMG_Lab_01 (see section 2.3) and consists of 180 frames. It shows static, textured background and two moving objects (a toy cow and a coffee cup that slide on a table by pulling them on a string) in the foreground. Fig. 3.8 shows the 3D-output of out algorithm back projected to the left image of the stereo-rig at frames 61, 109, 122, and 145. The yellow dash lined rectangles represent the bounding boxes of each cluster. The point features are shown as colored circles; supporting a hypothesis (red), not supporting a hypothesis (magenta), lost (cyan). The yellow and green cross illustrate the 2D projection of reference point and KF, respectively. Fig. 3.8(a) shows the output at frame 61. In frame 109 (fig. 3.8(b)), the cow is detected the first time. In frame 122 (fig. 3.8(c)), there are not enough point features on the cow, only a KF estimation is possible. The cow is clustered correctly again in frame 145 (fig. 3.8(a)).

In this scene, estimation works well in most parts. This is due to the low noise level in this scene. Fig. 3.9 shows the resulting motion trajectories of the reference points, in 3D (a) and in a 2D x/z plot (b). The computed reference point of the cup over all frames is shown in blue, and of the cow in magenta. Yellow points illustrate the static static structure. The egomotion of the observer is shown as black circles. The KF prediction and its lag are shown as red trajectory.

3.8.2 Experiment VMG_Person_01

This experiment uses the data from the dataset VMG_Person_01 (see section 2.4) and consists of 99 frames and shows static, textured background and the upper part of a walking person. Fig. 3.10 shows the output of our algorithm at frames 48, 60, 87, and 93. Fig. 3.12 presents a 2D plot of the motion trajectory of the

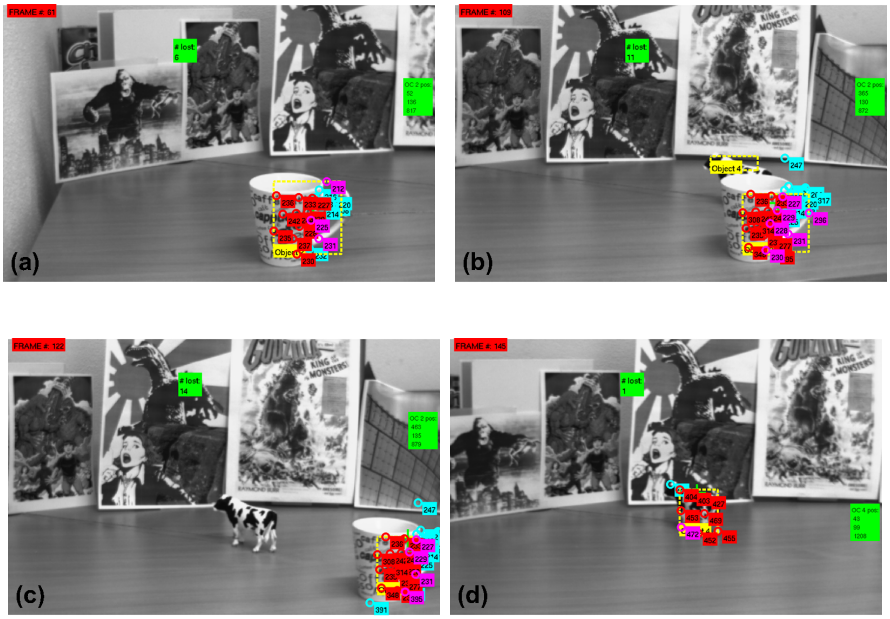
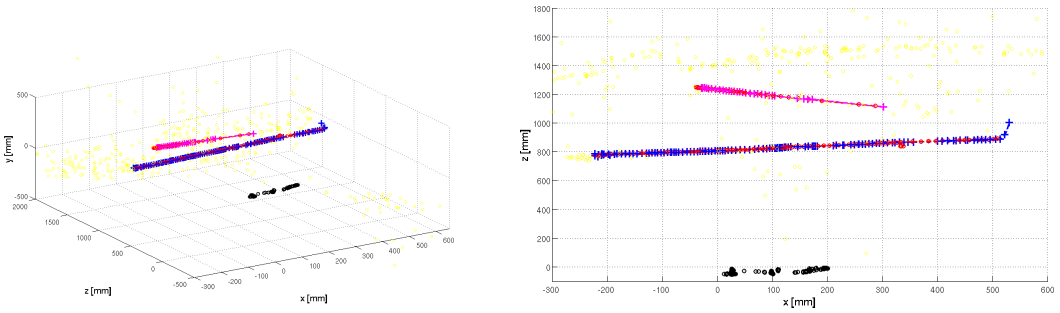


Figure 3.8: Experiment VMG_Lab_01: 3D-output of the described algorithm back-projected to the left image captured by the stereo-rig. Bounding boxes of each cluster (yellow), point features (colored circles), point features which support a hypothesis (red), point features which do not support a hypothesis (magenta), lost point features (cyan), 2D projection of reference point (yellow), and 2D projection of KF (green). (a) Output of the described algorithm at frame 61. (b) Output at frame 109. The toy-cow is visible in both stereo images and is clustered correctly. (c) Output at frame 122. The toy-cow does not provide enough point correspondences, only a KF estimation is possible (d) Output at frame 145. The toy-cow is clustered correctly again.

person (blue). False detections are illustrated as magenta and cyan trajectories, the observer motion is shown as black circles. The motion is relative to the scene coordinate system initialized at the first frame. The false detections occur as at least three point features with wrong stereo correspondences are clustered to an object by the motion clustering procedure (see 3.4). As shown in fig. 3.12, their trajectories are short. I.e., the false detections disappear rapidly as either the SaM eliminates wrong stereo correspondences or the motion clustering procedure stops grouping these point features.



(a) (b)

Figure 3.9: Experiment VMG_Lab_01: Estimated motion trajectories of the reference points, in in 3D (a) and in a 2D x/z plot (b). Computed reference point of cup over all frames (blue), cow (magenta), the estimated observer motion (black), and the static structure (yellow). The KF output smooths the estimation (red). The motion is relative to the scene coordinate system initialized at the first observer view.

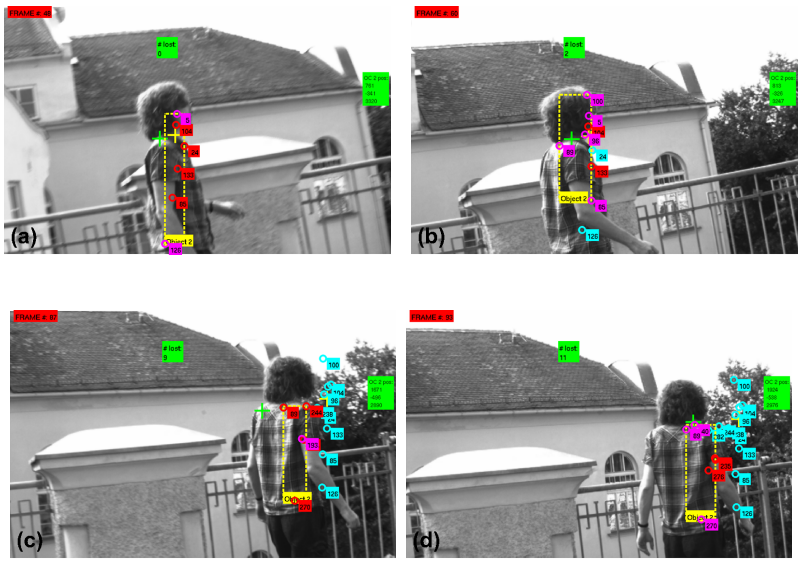


Figure 3.10: Experiment VMG_Person_01: 3D-output back-projected to the left image of the stereo-rig at frames 48 (a), 60 (b), 87 (c), and 93 (d). Bounding boxes of each cluster (yellow), point features (colored circles), point features which support a hypothesis (red), point features which do not support a hypothesis (magenta), lost point features (cyan), 2D projection of reference point (yellow), and 2D projection of KF (green).

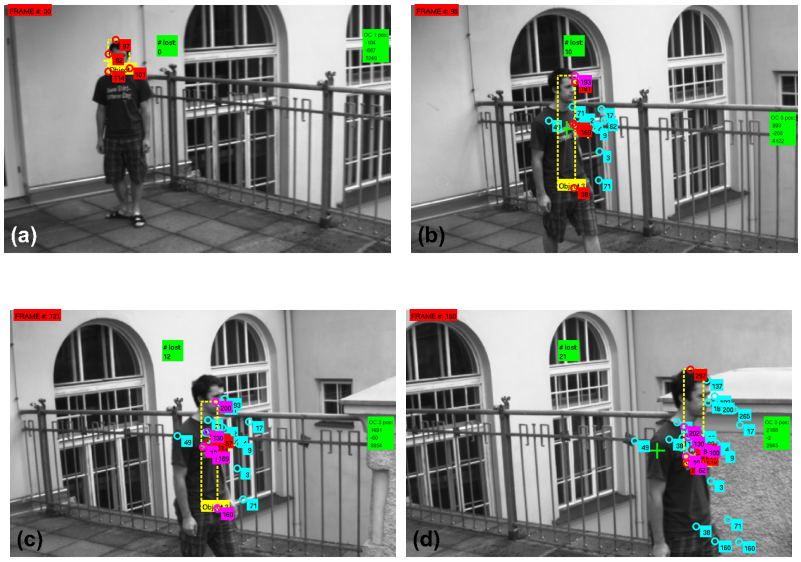


Figure 3.11: Experiment VMG_Person_02: 3D-output back-projected to the left image of the stereo-rig at frames 30 (a), 95 (b), 121 (c), and 150 (d). Bounding boxes of each cluster (yellow), point features (colored circles), point features which support a hypothesis (red), point features which do not support a hypothesis (magenta), lost point features (cyan), 2D projection of reference point (yellow), and 2D projection of KF (green).

3.8.3 Experiment VMG_Person_02

This experiment uses the dataset VMG_Person_02 (see section 2.5) and consists of 161 frames. It shows static, textured background and one moving person. Fig. 3.11 shows the output of our algorithm at frames 30, 95, 121, and 150. Fig. 3.13 presents the motion trajectories of the reference points. One can see, that outliers are not equally distributed over a moving object. Rather there exist sub-regions on an object, where most of the outliers are concentrated. Most of the outliers are located on the head of the moving person (see fig. 3.11(a)), later on the folds and label of the person's T-shirt (see fig. 3.11(c)). Fig. 3.13 illustrates the x/z-plot of the 3D-trajectory of the person (blue) and observer motion (black). Sometimes, the algorithm detects the person twice (magenta). This is caused by detecting two subparts on the person. I.e., the motion clustering procedure (see section 3.4) considers the person as two independently moving objects.

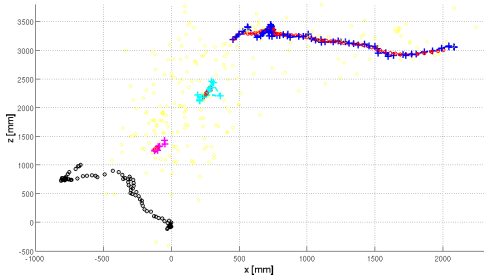


Figure 3.12: Experiment VMG_Person_01: x/z-plot of the 3D-trajectory of the person (blue) and observer motion (black), false detections (cyan and magenta), and static structure (yellow). The KF output smooths the estimation (red).

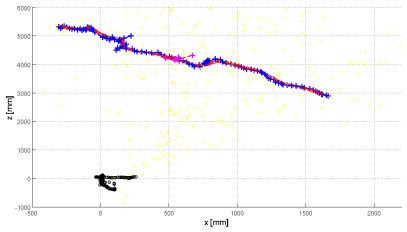


Figure 3.13: Experiment VMG_Person_02: x/z-plot of the 3D-trajectory of the person (blue), the observer motion (black), and the static structure (yellow). The KF output smooths the estimation (red).

3.8.4 Experiment KIT_Seq_01

This experiment uses the dataset KIT_Seq_01 (see section 2.7). It consists of 180 frames. It shows static, textured background, a moving car and one moving person. Fig. 3.14 shows the output of our algorithm at frames 42 and 114. Fig. 3.15 shows the x/z-plot of the 3D-motion trajectories of the reference points. The long black trajectory shows the egomotion of the observer. The moving car is detected independently three times and illustrated as cyan, magenta, and small black trajectory.

The moving person is illustrated as blue trajectory. The scale of moving objects play a major role. The smaller an object appears in the scene, the less outlier information is available. For a long time, the moving person does not provide outlier information. This issue is reflected in the *Points/Obj.* and *Detection in %* values of experiment KIT_Seq_01 in table 3.1. The minimum of 3 outliers required for the moving object's pose computation is hardly achieved, due to the scale issue (person) and large homogenous regions on the object (car).

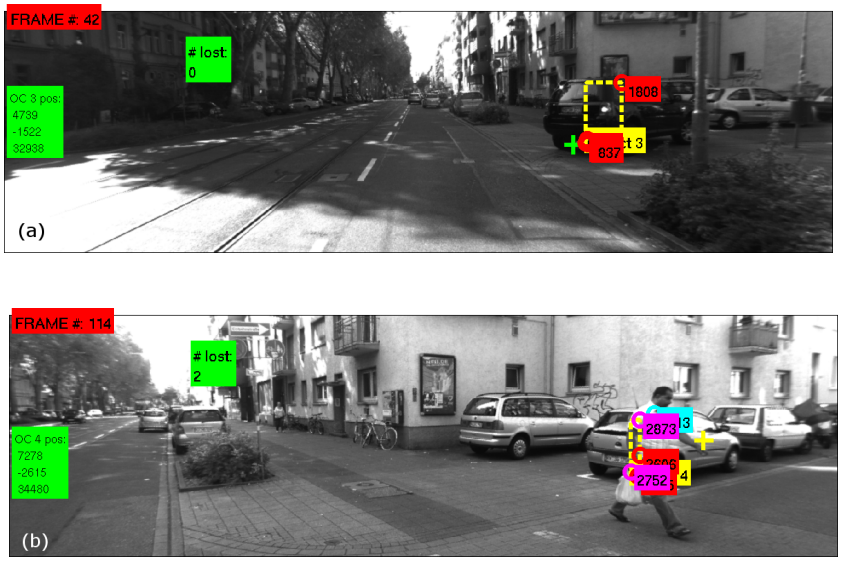


Figure 3.14: Experiment KIT_Seq_01: 3D-output back-projected to the left image of the stereo-rig at frames 42 (a) and 114 (b). Bounding boxes of each cluster (yellow), point features (colored circles), point features which support a hypothesis (red), point features which do not support a hypothesis (magenta), lost point features (cyan), 2D projection of reference point (yellow), and 2D projection of KF (green).

3.8.5 Experiment KIT_Seq_04

This experiment uses the dataset KIT_Seq_04 (see section 2.10). It consists of 230 frames. This experiment shows the limits of our geometry-based approach. In this scene, the observer is not moving. There are seven independently moving objects in the scene, two trams, one biker, and four people. In a set of subsequent frames, two (and later three) people build a group.

In the detection rate results (see table 3.1), we refer to the group as one object, as nearly no individual moving person detection is achieved. In this scene, we consider to have one individual moving person at the beginning which later on builds a group of up to three people, one biker, and one object refers to the two trams. Fig. 3.16 shows the output of our algorithm at frames 30, 64, and 117. In fig. 3.16(a), a single person is identified correctly. In (b) a group of two people is identified as independently moving foreground object. However, outliers on the

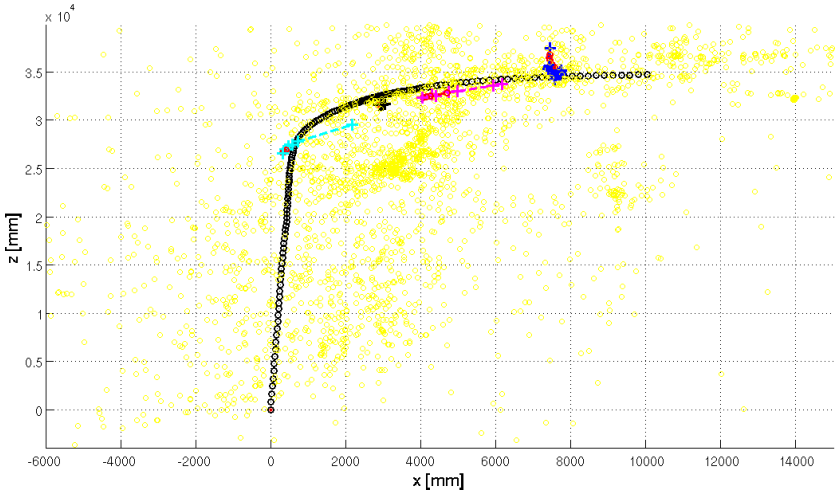


Figure 3.15: Experiment KIT_Seq_01: x/z-plot of the 3D-trajectories of the moving observer (long black trajectory), the car in front (cyan, short black trajectory, magenta), the person (blue), and the static structure (yellow). The KF output smooths the estimation (red).

nearby undergrowth on the left hand side also actively contribute to this object, which results in a false localization of the object. Fig. 3.16(c) illustrates the detection of the tram. Again, outliers on the undergrowth falsify the location of the object. The car ahead is detected wrongly as moving object in fig. 3.16(c).

Fig. 3.17 shows the x/z-plot of the 3D-motion trajectories of the reference points. The position of the observer is at $\boldsymbol{X} = [0, 0, 0]^T$, as the observer is not moving. We are able to detect one single person (see fig. 3.16(a)), illustrated as blue trajectory. Additionally, we identify a group of two people as foreground object (see fig. 3.16(b)). Over time, this group is detected as three different objects, resulting in the magenta, cyan and red trajectories. As mentioned above, the localization results are poor due to falsely clustered outliers contributing to the object. Only the distance of the single person is identified correctly. Something similar happens with the detection of the trams (see fig. 3.16(c)). Our algorithm detects the trams as one object, even though they are moving in opposite directions. This can be explained by the motion blur and the resulting poor texture. All trajectories are illustrated in fig. 3.17.

We want to point out that in experiment KIT_Seq_04 the output varies consid-

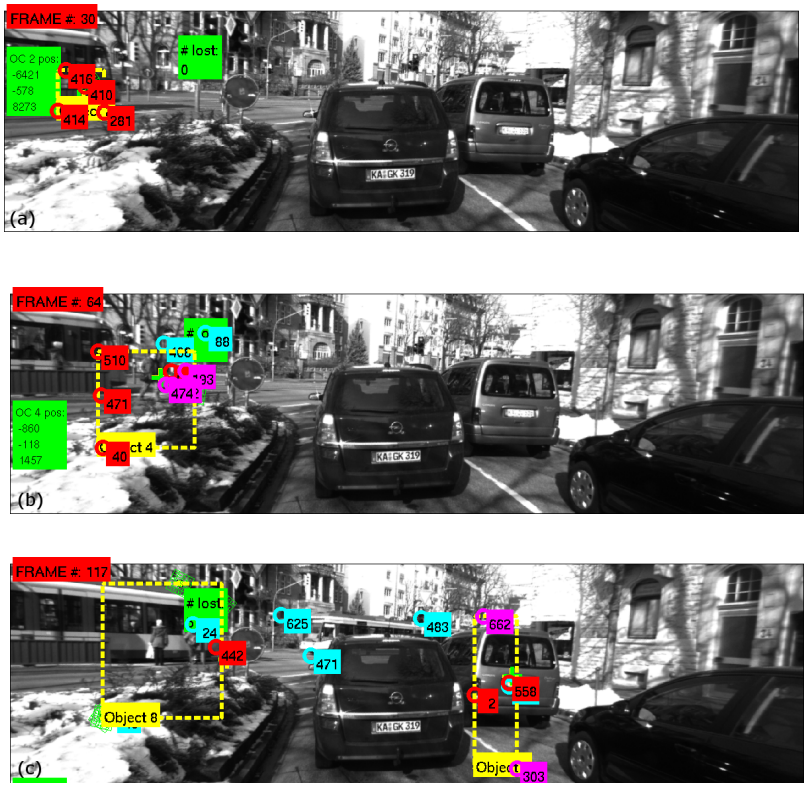


Figure 3.16: Experiment KIT_Seq_04: 3D-output back-projected to the left image of the stereo-rig at frames 30 (a) and 64 (b), and 117 (c). Bounding boxes of each cluster (yellow), point features (colored circles), point features which support a hypothesis (red), point features which do not support a hypothesis (magenta), lost point features (cyan), 2D projection of reference point (yellow), and 2D projection of KF (green).

erably depending on the selected cluster size. Decreasing the cluster size increases the detection rate for the trams. Fig. 3.18 illustrates this issue. With a cluster size of 2.3 m we gain a tram detection rate of 1.4%. With a cluster a size of 1.65 m the detection rate increases to 7.9%. However, this variation in the cluster size deteriorate the detection results on the single person.

Table 3.1 shows a quantitative evaluation of our algorithm. Besides the number of frames per experiment, it contains the total number of point features (inliers and outliers). Furthermore, the number of outliers is listed separately. *Valid outliers* indicates outliers the algorithm has assigned to a moving object. The row *Points/Obj.*

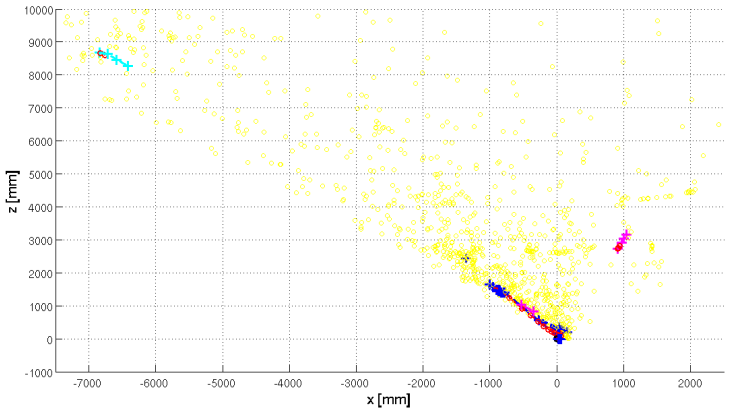


Figure 3.17: Experiment KIT_Seq_04: x/z-plot of the 3D-trajectories of the moving observer (clustered black circles at $\boldsymbol{x} = [0, 0, 0]^T$), the car in front (magenta), the person (short blue trajectory), the group of people (long blue trajectory), the tram (cyan trajectory), and the static structure (yellow). The KF output smooths the estimation (red).

Figure 3.18: Tram detection vs. no tram detection due to a different cluster size in experiment KIT_Seq_04.

lists the average number of point features on the detected object(s). The row *Objects* indicates the number of individual moving objects (neglecting the observer). *Objects detected* gives the number of objects detected in the scene. *Objects detected* may be larger than *Objects* in case of multiple detections of one object. Each experiment was run three times, and the average number of points is given. The experiments were executed multiple times, as the motion clustering procedure (see section 3.4) is not deterministic due to the used meanshift clustering algorithm. I.e., the point features clustered to certain objects may vary slightly.

In experiment VMG_Lab_01, the first number refers to the cup, the second to the cow. In experiment VMG_Person_01, object 2 refers to the moving person, objects 1 and 3 are false positives. In experiment VMG_Person_02 both, object 1 and 2 refer to the person. Object 2 is a second detection at the same time. In experiment KIT_Seq_01, objects 1 to 3 refer to the car, objects 4 and 5 to the person. In experiment KIT_Seq_04, object 1 in *Objects detected* refers to the single walking person. Object 2 and 3 refer to the group of people walking, which was multiply detected at some frames. While object 4 is representing a false detection on the car ahead on the right hand side, object 5 refers to the trams. Finally, the detection rate for moving objects is shown in percent. Again, in experiment VMG_Lab_01, the first number refers to the cup. The detection rate for experiment VMG_Person_01 highlights the detection rate for object 2. The detection rate for the false positives does not exist. For experiment KIT_Seq_01, the detection rate for object 1 refers to the car and is the total detection rate of objects 1 to 3. The same applies for object 2, i.e. it is the total detection rate for objects 3 and 4. At this point it has to be mentioned, that the detection rate does not indicate that the Pascal criterion (50% overlap with the groundtruth, refer to chapter 4) is fulfilled, i.e. only sub-parts of a moving object may be detected. Focusing on the detection rates for experiment KIT_Seq_04, object 1 of refers to the single person detected. Object 2 gives the detection rate of the group of people detected. The biker was not detected in a single frame. Object 4 gives the detection rate of the trams.

As mentioned in section 3.4, we are using Meanshift clustering, i.e. further information processing relies on the output of our clustering routine. Thus, small

Table 3.1: Quantitative evaluation of the the algorithm.

	VMG_ Lab_01	VMG_ Person_01	VMG_ Person_02	KIT_ Seq_01	KIT_ Seq_04
Frames	180	99	161	180	231
Features	820	615	564	4441	1815
Outliers	200	107	128	134	197
Valid Outliers	131	87	108	114	147
Objects	2	1	1	2	7
Obj. detected	2	3	1	5	6
Points/Obj.	12.6/6.2	2.4/6.3/2.9	7.8	4.0/4.3/4.5/ 4.4/4.0	4.3/5.8/4/ 6.0/4.3
Detection in %	91.5/54.5	67.0	79.4	41.5/13.1	9.3/95.2/ 0.0/1.4

variations may occur in the clustered objects which will be handed over to the motion estimation. Fig. 3.19 illustrates the motion estimations of the moving person of experiment VMG_Person_02 with a cluster size of 1.20 m (fig. 3.19(a)) and 1.24 m (fig 3.19(b)). The person is tracked continuously. However, in fig. 3.19(b) the person is detected twice independently, which results in the minor variation of the motion estimation.

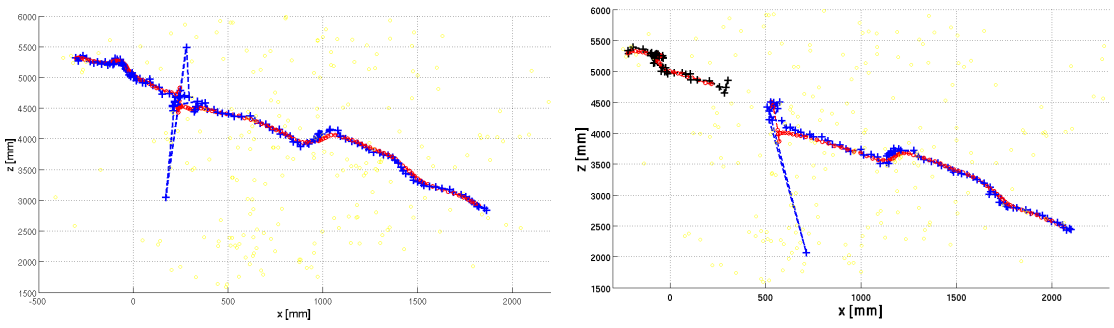


Figure 3.19: Different motion estimation due to variations in the Meanshift clustering process.

The presented purely geometry-based model is very efficient at approximately 2 and 4 frames/second for 2 and 1 objects, respectively. The extension with descriptive

information by the SIFT descriptor [Low04, VF08] does not slow down the execution time considerably. We achieve good estimation results for both, observer motion and independent foreground motion. Solely in experiment KIT_Seq_04 MSaM results deteriorate. This is on the one hand caused by the moving people. Sometimes they behave as independently moving objects, sometimes the behave the same in groups. Due to their distance, they are pretty small. I.e., MSaM is not able to always identify them as foreground objects. On the other hand, due to the frame rate of 10 Hz and the resulting motion blur, the two trams moving in the opposite direction are identified as one moving foreground object.

In contrast to Ozden et al. [OSG10] who require 1 minute/frame, our approach is close to real-time (2-4 frames/second). However, their approach provides higher accuracy in object events like splitting or merging due to the high amount of hypotheses they are maintaining for each frame. But we have to point out, that our object clustering routine is applied on the 3D information, whereas Ozden et al. [OSG10] use 2D information.

3.9 Discussion

As shown in experiment KIT_Seq_01 (see section 3.8.4), sometimes multiple detections of a single object over time occur. This behavior can be traced back to two reasons:

1. the threshold indicating after how many frames an object may be invisible (see section 3.7) is set to one frame. I.e., if the car is not detectable for more than one frame, the object ID is locked and in case of a re-detection a new object ID is assigned to the re-detected object.

2. too many point features disappeared from one to the other frame, i.e. the minimum overlap of common point features is below the threshold specified (see section 3.7).

In order to prevent multiple detections, one can either decrease the threshold for the common point feature set or increase the threshold which indicates how long

a disappeared object is re-detectable. For better comparison of the detection and tracking results, we decided to keep these two parameters constant for all experiments.

3.10 Conclusion

We have presented a purely geometry-based MSaM algorithm which bases on an SaM framework [SSP08]. The developed algorithm models rigid independent foreground motion by outlier analysis. Hence, SaM algorithms can be extended to MSaM by providing both, inlier and outlier information. While the inlier information is used for scene reconstruction and observer pose estimation, the outlier information is related to independent foreground motion and noise. We pointed out that any SaM algorithm is suited for MSaM extension, except keyframe-based approaches. The main contribution of the presented MSaM algorithm is the local object representation gained by a feedback control mechanism, involving Meanshift clustering and rotation estimation. A stable reference point per object and the positions of point features on the object w.r.t. the reference point provide strong information on the object pose and its motion behavior. To stabilize the reference point update, we have introduced a confidence measure. Based on this confidence measure, the algorithm decides whether a point feature is reliable enough to contribute to the reference point computation. The selection of the cluster size contributes to the outliers on the object and thus has influence on the motion estimation. Mainly, the introduced confidence measure is capable to minimize this influence. Sometimes - as shown in an experiment - the impact of the cluster size is still large, i.e. the detection of a moving foreground object depends on the selected Meanshift cluster size. It is shown by the experiments that - once we gather four or more outliers per cluster - the algorithm is able to model the detected foreground motion. Our MSaM algorithm requires more than 50% stable and reliable background information, due to the underlying SaM algorithm [SSP08]. An approach to minimize this set of stable point features will be introduced in chapter 6.

4

MSaM for Person Tracking*

In the previous chapter we described a Multibody Structure and Motion (MSaM) algorithm which is able to detect foreground motion by outlier analysis. We pointed out the main tasks of MSaM, which are the (i) detection and tracking of moving objects, (ii) observer pose estimation in a global scene, and (iii) scene reconstruction. The major benefit is certainly that all available information is in 3D, i.e. we gain information on depths and object sizes.

In this chapter, we extend this MSaM algorithm to classify the detected foreground motion. We show, that we can do that by connecting state-of-the-art components to our MSaM. In particular, we are interested to identify moving people in our scenes. Having an MSaM system, which allows to identify moving foreground objects, the idea of object classification comes to mind. However, active tracking of people as well as other objects is challenging. It requires an object classification routine attached to the core MSaM tasks. I.e. for person detection, a person detection algorithm has to be added to MSaM.

*This chapter builds on a paper that originally appeared in *Proceedings of the 6th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications.* Holzer P., Li C. and Pinz A.: *Detecting and Tracking People in Motion - A Hybrid Approach Combining 3D Reconstruction and 2D Description, March 2011, pages 561-568*

Person detection methods can be classified into probabilistic-based and non-probabilistic algorithms. Probabilistic-based algorithms segment a person according to a previously established model. Yan and Pollefeys [YP08] build a kinematic chain of an articulated object to segment articulated motion within non-rigid parts. Song et al. [SFP00] give a method based on learning an approximate probabilistic model of the joint positions and velocity of different body features. These methods are effective but more complicated for establishing a model. On the contrary, non-probabilistic methods are more simple and adaptive to many kinds of objects, i.e. they are not limited to human models. Among these methods, HOG-based methods [DTS06, FMR08, LD10] are the current state of art in person detection. Dalal and Triggs [DT05] use HOG to detect stationary people who are upright and fully or almost fully visible. By using linear and Gaussian-kernel SVMs as classifiers, they report an extensive experimental evaluation. HOG shows superior performance in separating the image patches into human and non-human. It is robust against pose and appearance variations of the pedestrians. Various modifications [LD10, FMR08] exist, which improve its performance. Having excellent detection results, HOG generates false positives on person like structures (e.g. billboards showing people). Additionally, HOG results are 2D (image plane) only. Based on HOG, Lin and Davis [LD10] use deformable part models and a latent SVM to improve the performance. Felzenszwalb et al. [FMR08] present an idea of matching a hierarchical part template tree to detect humans and estimate their poses. Dalal et al. [DTS06] also combined a human shape descriptor with optical flow to detect moving people from a video. This algorithm runs a detection window across the image at all positions and scales, which is time consuming. There has been a detailed survey on visual surveillance [HTWM04] and pedestrian detection [LSG10]. Both mainly consider static cameras for video recording.

Section 4.1 describes how we detect and track robustly. Section 4.2 shows experimental verification of our system. In section 4.3 the outcome of the experiments is discussed. Finally, in section 4.4 we conclude.

4.1 Robust Person Detection and Tracking

In this section, we present our combined detector and tracking method. Our method uses both, motion information and human shape information, to detect and track moving people. Figure 4.1 illustrates an overview of our system from a moving observer. We apply the standard HOG implementation by Dalal and Triggs [DT05]. We provide the whole images as input. So we can get also a false positive rate by the HOG. For tracking detected people over time, a tracking algorithm is required. As both - the person and the observer - are moving, tracking is quite difficult due to the background motion. We use Meanshift tracking [CM02, CRM03], which is a simple iterative procedure. Its principle bases on a similarity measure. It shifts each data point to the average of data points in its neighborhood. It is efficient for tracking of a large variety of objects, either rigid or with articulated motion, and with different color and/or texture patterns such as human bodies. As a third component, we combine the MSaM described in chapter 3) with HOG and Meanshift tracking.

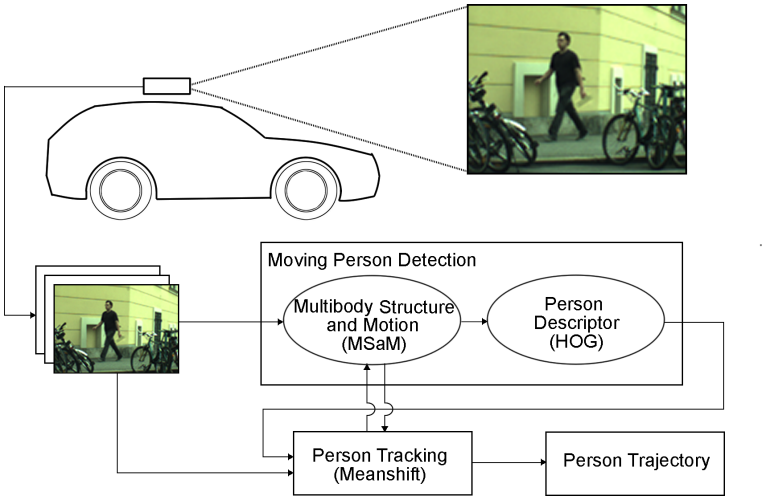


Figure 4.1: Graphical overview of our MSaM system for moving person tracking. The system can be divided into three main parts: video capture, person detection and person tracking

The system can be divided into three main parts: video capture, person detection, and person tracking. First, MSaM gives us information on moving objects. Then, HOG verifies if the moving object is a person. Finally, Meanshift tracking is

established, to track the moving person. This is a hybrid approach, because Meanshift tracking is established by the combination of HOG and MSaM and the output of these three is compared periodically. In case of divergence, i.e. HOG and/or MSaM do not match with the Meanshift tracking any longer, re-initialization of the hybrid tracker is required. Our main contributions are:

- The fast and robust person detector. Multibody moving object detection provides possible locations of people in 3D. These locations are searched for human shapes. This increases the speed of person detection. Firstly, it can reduce the searching time for a person. The human-shape descriptor (i.e. the HOG) is computed for this subarea only. Secondly, we know the scale because of MSaM. We can limit the scale-pyramid used in HOG to fewer levels.

- The mutual influence of moving object detection and tracking and person detection makes tracking more reliable. Many false positives detected by the HOG can be eliminated. The output of the hybrid tracking is fed back to the moving object detection (MSaM). There, this information is used to harvest more point features on the object. By this, we can find point features which were wrongly identified as background structure or were not clustered to the object by the motion clustering procedure (see section 3.4). These additional point features can be used to further improve the estimation of the moving person's trajectory.

4.1.1 Moving Person Validation

Moving objects are detected and tracked by MSaM. The output of MSaM is validated with HOG. Figure 4.2 illustrates a correct HOG detection. From MSaM, we know the distance from the observer (camera) to the person. Thus, we know which scale we can apply for the HOG. We cannot guarantee that the output of MSaM covers a complete person, only subparts may be detected instead. But, we can enlarge the MSaM regions on the image such that it covers the whole person. The size of the surrounding region can be chosen depending on the distance of the person to the observer. This avoids false positive detections by the HOG.

Figure 4.2: HOG Detection of a moving person

As the MSaM and HOG detection windows can differ in size massively, we cannot apply the PASCAL criterion here (refer to equation 4.4). We consider the overlap a_{val} of HOG and MSaM as correct match, if the overlap is larger than 50% of the smaller area of either HOG or MSaM (eq. 4.1). In most cases, the HOG area is larger, as MSaM mostly detects subparts of a person only.

$$a_{val} := \max\left(a_{MSaM}, a_{HOG}\right) > 0.5 \tag{4.1}$$

where

$$a_{MSaM} = \frac{area(B_{MSaM} \cap B_{HOG})}{area(B_{MSaM})} \tag{4.2}$$

$$a_{HOG} = \frac{area(B_{MSaM} \cap B_{HOG})}{area(B_{HOG})} \tag{4.3}$$

4.1.2 Supporting Structure by Feedback Control

Once an overlap of HOG and MSaM occurs, Meanshift tracking is initialized. We take the region within the bounding box of the HOG as input for Meanshift tracking. For the subsequent frames, we consider tracking successful, if either HOG or MSaM overlap with the Meanshift tracking for more than 50%. Otherwise, if for a cer-

tain amount of frames neither HOG nor MSaM match with the Meanshift tracking window, Meanshift tracking is stopped. In contrast to MSaM, Meanshift tracking provides 2D information only. By feeding back the Meanshift tracking information to MSaM, we are in the position to periodically inspect MSaM and Meanshift tracking. In case of major differences, person tracking is re-initialized.

This feedback routine has also advantages on the available point features. If MSaM overlaps with the Meanshift tracker, we can search for supporting structure in the overlap. We call every found stable point feature a supporting structure, if it is in the overlap of MSaM and Meanshift tracker and approximately at the same 3D depth as the object's reference point of the MSaM. With this routine, we gather more point features on the object, i.e. estimation of the person's trajectory will become more precise.

Figure 4.3 illustrates the output of our algorithm. The MSaM detection window is shown as yellow bounding box, the Meanshift tracking bounding box is shown in green. The reference point of the object and its Kalman prediction are shown as yellow and green cross respectively. The different types of MSaM point features on the object are shown in the colors red, magenta, and cyan. In the overlap of the MSaM and Meanshift tracking window several supporting structure point features are found (yellow).

4.2 Experiments

This section presents five selected experiments executed with our hybrid tracking system. These experiments span a range of challenges. Experiment VMG_Lab_01 shows a controlled experiment setup with two moving objects. The result demonstrates that our hybrid algorithm can suppress false HOG positives. Experiment VMG_Person_01 tracks a person with a rapidly moving observer. Here, the outcome of hybrid tracking improves the performance over individual HOG and MSaM. Experiment VMG_Person_02 shows a similar scene, but the person is moving towards the camera, which results in a change of scale. At the end of the sequence, the person is only partially visible. Again the good performance of the

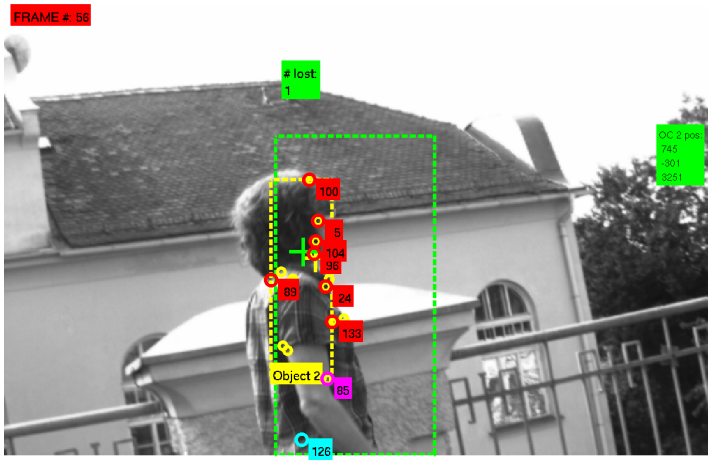


Figure 4.3: MSaM detection and Meanshift tracking of a moving person. Lost MSaM point features (cyan), active MSaM point features (red), inactive MSaM point features (magenta), supporting structure (yellow), MSaM bounding box (yellow), MSaM reference point (yellow cross), MSaM Kalman prediction (green cross), and Meanshift tracking bounding box (green).

hybrid tracking is shown. Experiment VMG_Person_03 is a special case; the person is far away and partly invisible behind a set of bicycles. Here, HOG performs much better than MSaM. Neglecting the PASCAL criterion for the hybrid tracking approach, the results are still promising. Experiment KIT_Seq_01 shows a moving person and a moving car. As expected, MSaM tracks the car. Using hybrid tracking, the car detection is neglected due to the combination of MSaM with HOG.

When referring to positive detections we consider the PASCAL criterion. This means, the correct detection requires an overlap a_o of the ground truth bounding box B_{gt} and predicted bounding box B_p above 50% and (ii) multiple detections of the same object are considered false detections.

$$a_o := \frac{area(B_p \cap B_{gt})}{area(B_p \cup B_{gt})} > 0.5 \tag{4.4}$$

The MSaM detections are not evaluated with the PASCAL criterion. As mentioned earlier, most of the detections contain only subparts of an object, depending on the available outlier point features. We render an MSaM detection correctly,

when an object fills at least 50% of the the detected region (eq. 4.1).

4.2.1 Experiment VMG_Lab_01

In this experiment we use the dataset VMG_Lab_01 (see section 2.3). MSaM tracks the moving objects (cup and cow pulled by a string) very well. HOG has no correct detections, as no person is moving in the scene. However, HOG detects 105 false positives in the background. The hybrid approach eliminates the false positives of the HOG. Table 4.1 shows the results. In line "Avg #M gain", the average amount of additional supporting features gathered by hybrid tracking is listed. Fig. 4.4 shows the 3D-output back-projected to the image plan. The MSaM detections of the moving objects are illustrated as yellow bounding boxes. The yellow bounding box represents the MSaM tracking result, the blue bounding box is a HOG detection.

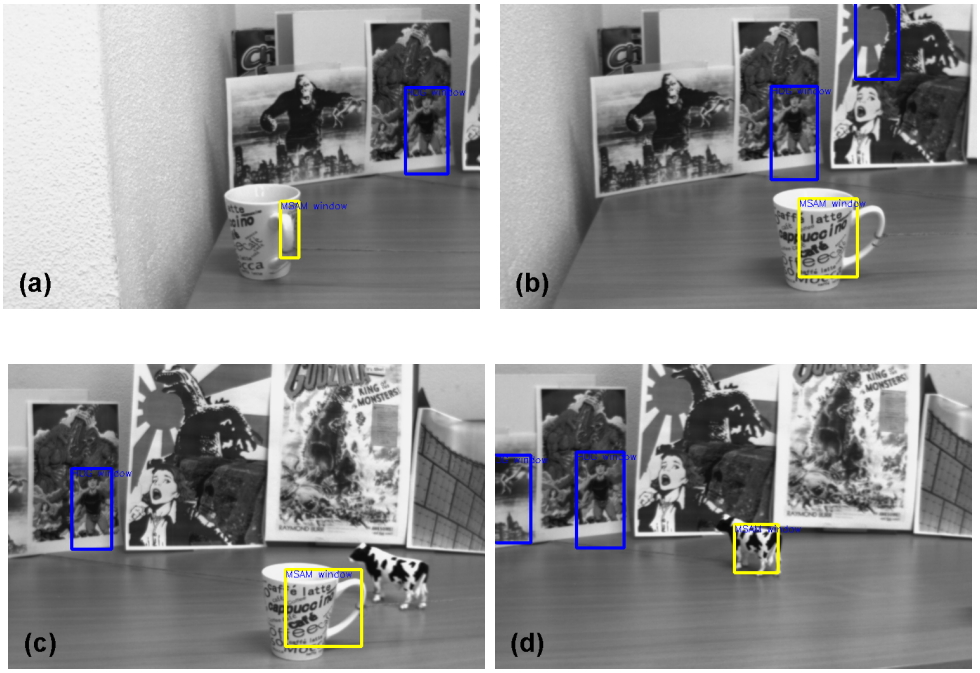


Figure 4.4: Experiment VMG_Lab_01: 3D-output back-projected to the image-plane. Bounding box of MSaM tracking (yellow); bounding boxes of HOG detections (blue); no hybrid tracking, as no moving person in scene.

Table 4.1: Experiment VMG_Lab_01: Quantitative Results.

	HOG	MSaM	Hybrid
Det. Rate	-	91.5%/54.5%	-
False Pos.	105	-/-	-
No Det.	-	8.5%/45.5%	-
Avg #M gain	-	-	-

4.2.2 Experiment VMG_Person_01

In this experiment we use the dataset VMG_Person_01 (see section 2.4). The results are shown in table 4.2. The HOG detection rate is rather low, as (i) the observer moves rapidly and (ii) the person is only partly in the scene. MSaM tracking is more reliable, but is below 70% due to motion blur and the lack of outliers on the person in the first 30% of the frames. Hybrid tracking seems to be worse than the MSaM tracking. This is due to the PASCAL criterion. The reqirements on the hybrid tracking are much higher compared to MSaM. Combining the false positives and the correct detecions, hybrid tracking would perform the same as MSaM. 14.1% of no detections are due to the Meanshift's limits on grayscale images and the too large HOG window on the initialization (a lot of background). With hybrid tracking, we get an average of 8.8 points per frame of additional point features. Fig. 4.5 shows the 3D-output back-projected to the image-plane. An overlap of the HOG and MSaM bounding boxes (fig. 4.5(a)) initializes the hybrid tracking window, illustrated as red bounding box (fig. 4.5(b)). In figure 4.5(c) HOG does not find a person, but the hybrid tracker is still tracking. In figure 4.5(d) the hybrid tracker lost the target, the deactivation is imminent.

Table 4.2: Experiment VMG_Person_01: Quantitative Results.

	HOG	MSaM	Hybrid
Det. Rate	17.2%	67.0%	42.4%
No Det.	80.8%	33.0%	14.1%
False Pos.	2	0	21
Avg #M gain	-	-	8.8

Figure 4.5: Experiment VMG_Person_01: 3D-output back-projected to the image-plane. 3D-output back-projected to the image-plane. Overlap of HOG and MSaM initializes hybrid tracking (red) **(a)**; hybrid tracking (red), MSaM tracking (yellow), and HOG detections (blue) **(b)**; no HOG detection **(c)**; hybrid tracking lost target (orange), deactivation of hybrid tracking is imminent **(d)**.

4.2.3 Experiment VMG_Person_02

In this experiment we use the dataset VMG_Person_02 (see section 2.5). Here, the person walks towards the observer. This results in a scale change of the person. The results are shown in table 4.3. MSaM tracks the person well. The HOG detection rate is again rather low. MSaM tracking is more reliable, as it does not refer to the PASCAL criterion. Hybrid tracking again seems to be worse than the MSaM tracking as it has higher requirements due to the PASCAL criterion. I.e. neglecting the PASCAL criterion for hybrid tracking would increase the detection rate. Then, hybrid tracking would outperform the MSaM approach.

Feeding back the hybrid tracking result to the MSaM, we get an average amount of 4.6 supporting structure points on the object. The MSaM tracker detects the head

of the person only. By HOG, we are able to establish the hybrid tracker (fig. 4.6(a). Figure 4.6(b) illustrates a false positive detection by HOG, a correct MSaM detection (yellow bounding box), and a false hybrid detections (red) according to the PASCAL criterion. In figure 4.6(c), only the hybrid tracker works, no MSaM nor HOG detection. Figure 4.6(d) shows a false positive detection of hybrid tracking according to the PASCAL criterion and a correct MSaM detection (yellow).

Table 4.3: Experiment VMG_Person_02: Quantitative Results.

	HOG	MSaM	Hybrid
Det. Rate	23%	79.4%	40.4%
No Det.	77%	20.6%	14.9%
False Pos.	47	-	71
Avg #M gain	-	-	4.6

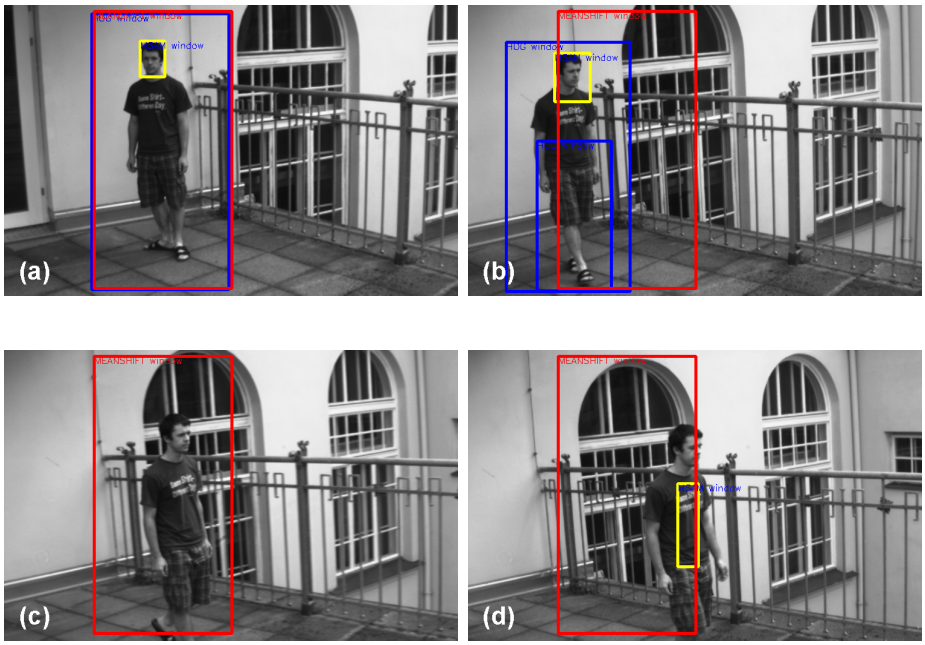


Figure 4.6: Experiment VMG_Person_02: 3D-output back-projected to the image-plane. MSaM (yellow), HOG (blue), and hybrid tracking (red) **(a)**; HOG false positive detection, correct MSaM detection (yellow), and false hybrid detections (red) according to the PASCAL criterion **(b)**; only hybrid tracking (red) works **(c)**, correct MSaM tracking (yellow) but false positive detection of hybrid tracking according to the PASCAL criterion **(d)**.

4.2.4 Experiment VMG_Person_03

In this experiment we use the dataset VMG_Person_03 (see section 2.6). The results are shown in table 4.4. The MSaM result is poor. The person is small and uniformly colored, i.e. very few outlier point features are found on the object. The HOG detection rate is very good, even when the person is partly occluded. Hybrid tracking seems to be worse than the MSaM tracking. Again, neglecting the PASCAL criterion, the result of hybrid tracking is similar to the good performance of the HOG. But in contrast to the HOG, hybrid tracking deals with 3D information. The average amount of 1.1 supporting structure points on the object can be explained be the low hybrid detection rate. First, only the HOG detection works (fig. 4.7(a)). In figure 4.7(b) HOG (blue) and MSaM (yellow) initialized the hybrid tracker (red). The person is re-detected by HOG and tracked by MSaM and the hybrid tracker (fig. 4.7(c)). In figure 4.7(d) no further MSaM tracking is possible. Additionally, HOG gives us multiple detections (blue) and the hybrid tracker lost the target (red).

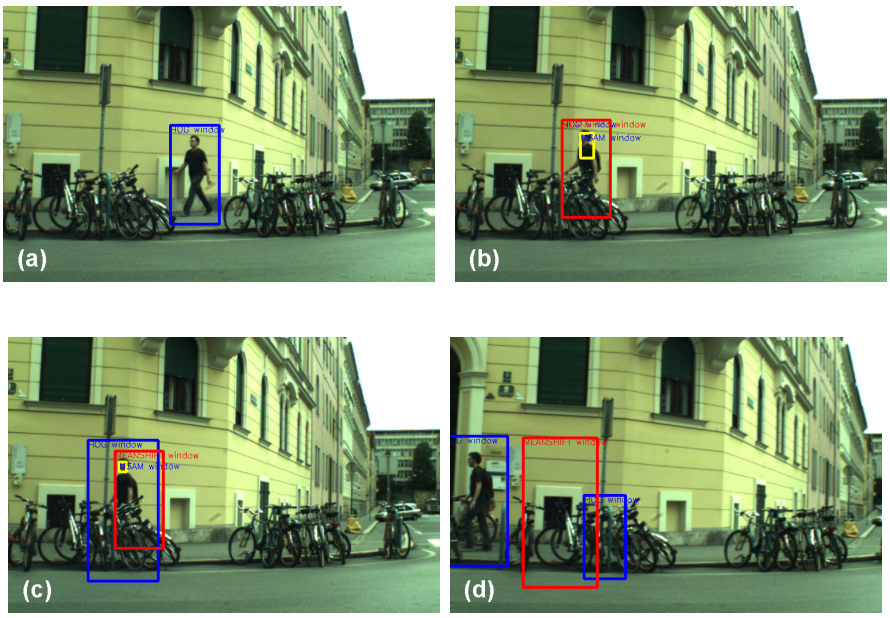


Figure 4.7: Experiment VMG_Person_03: 3D-output back-projected to the image-plane. Only HOG detection (blue) **(a)**; HOG (blue), MSaM (yellow), and hybrid detection (red) **(b)**; HOG (blue), MSaM (yellow), and hybrid tracking (red) works **(c)**, no further MSaM tracking possible, multiple HOG detections (blue), false hybrid tracking (red) **(d)**.

Table 4.4: Experiment VMG_Person_03: Quantitative Results.

	HOG	MSaM	Hybrid
Det. Rate	78.2%	16.4%	10.9%
False Pos.	30	0	52
No Det.	20%	83.6%	x
Avg #M gain	-	-	1.1

4.2.5 Experiment KIT_Seq_01

In this experiment we use the dataset VMG_KIT_Seq_01 (see section 2.7). The results are shown in table 4.5. In contrast to the other experiments, the moving person is not visible throughout the whole scene. Instead it is only visible in 99 of 180 frames. MSaM not only tracks the moving person, but also the moving car. First, the person is too small to be detected by MSaM, i.e. very few outlier point features are found on the object. HOG delivers a positive person match, even the person is very small. Due to the Pascal criterion, the detection rate of the hybrid tracking seems to be worse than HOG or MSaM alone. Again, neglecting it, 3D-hybrid tracking would perform in the range of MSaM and HOG. Compared to experiment 4, the average amount of the supporting structure points decreased to 0.7, explaining the low hybrid detection rate. First, MSaM detects the moving car (fig. 4.8(a)). Then, HOG identifies a person in the scene 4.8. The scale of the person is too small, not enough outliers can be detected on the person. Once the person is more prominent in the scene, MSaM and HOG detect the person. Meanshift tracking is initialized (fig 4.8(c).

Table 4.5: Experiment KIT_Seq_01: Quantitative Results.

	HOG	MSaM	Hybrid
Det. Rate	56%	41.5%/13.1%	-/9.6%
False Pos.	37	3	11
No Det.	40.5%	58.5%/86.9%	-/23.3%
Avg #M gain	-	-	-/0.7

Figure 4.8: Experiment KIT_Seq_01: 3D-output back-projected to the image-plane.

4.3 Discussion

Summing up all experiments, the following observations can be made:

- The MSaM's detection rate is typically higher than HOG's or hybrid's. As we cannot control, which parts of an object are detected by MSaM (texture), we cannot use the PASCAL criterion.
- The hybrid tracking provides 3D information. We can speed-up the HOG, as (i) we know the distance to the person (fewer pyramid levels) and (ii) we get a rough idea, where to search in an image (region of interest).
- The hybrid tracking provides important feedback for MSaM. We can investigate inliers in a larger subarea (HOG window / hybrid tracking window).

Knowing the distance, we find supporting structure for a person, which can help to improve the estimation of the person's reference point.

In chapter 3 it was explained, that MSaM can be used to detect and track rigid foreground motion. In this chapter however, MSaM detects and tracks moving people which can be modeled as articulated objects. I.e. MSaM identifies articulated foreground motion. This is possible, as the body of a person behaves similar to a rigid object, whereas arms or legs are connected in an articulated manner. Thus, most MSaM detections occur on the body and the head of a person.

4.4 Conclusion

We presented a moving person detection and tracking system. As tracking by a moving observer is a difficult task, we combined 3D algorithms with 2D descriptors and tracking algorithms. The system allows a moving observer and moving objects. As we use MSaM, we get 3D information on the scene, observer motion, and object motion.

By combining different components, we gain a mutual benefit. By combining the HOG with the MSaM tracker, we get 3D information of the person motion and eliminate false positive HOG detections. By feeding back the Meanshift tracking, we can harvest additional features on the object for improved MSaM performance. Our system deals with 3D and 2D information. As we know the 3D depth and the position on the image-plane, we can speed up HOG (fewer pyramid levels, image subarea validation).

Extensions to other categories are possible. The system is not limited to a human shape descriptor. Introducing different descriptors, the system can track different (or even multiple) categories.

5

Spatial Temporal Connectivity

SLAM or SaM algorithms are able to reconstruct the scene and estimate the pose of a moving observer only if the observer navigates within in a static scenery. Having a scenario with a moving observer and/or moving foreground objects (as introduced in chapter 3), the results of SLAM and SaM deteriorate. This is due to the lack of stable, stationary background points which are essential to estimate the pose and reconstruct the scene. The introduced MSaM algorithm (see chapters 3 and 4) identifies moving objects or regions which are part of moving objects due to outlier analysis. In the previous chapter we have shown how we can identify the full object by combing an additional 2D detector and tracker to the MSaM system. All - SLAM, SaM, and MSaM - require at least 50% inliers and use the whole inlier information for scene reconstruction and pose estimation. In this chapter, we analyze the inliers to determine several subsets and evaluate their properties as good features to track. Shi and Tomasi [ST94] state that by analyzing the appearance change over time of point features, one is able to identify good features to track. Such subsets then can be used to estimate the observer's pose without deteriorating results. Thus, subsets provide two major advantages: (i) false inliers (e.g. parts of a moving object have

not been identified) may be neglected with high probability and (ii) the amount of point features for pose estimation can be reduced.

We start by harvesting point features over time and connect them accordingly to point feature trajectories. We apply a spatial-temporal appearance descriptor to each point feature trajectory, which allows us to describe the appearance change of a point feature over time. We then evaluate this descriptor by histogram statistics evaluation methods and compare the resulting inlier subsets.

First, in section 5.1 we describe how we gather point features (or more generally: region of interests). Section 5.2 explains how a point feature is re-identified in subsequent frames and linked to a time-trajectory. Next, in section 5.3 we explain the Space-Time Appearance (STA) descriptor introduced by Brkic et al. [BPSK11]. The STA descriptor is used for appearance change information retrieval. Section 5.4 explains, how we evaluate the properties of the STA descriptor. Section 5.5 describes important extensions which are required to apply the STA descriptors to the generated point feature trajectories. Subsequently, section 5.6 contains experiments, where we select and compare good features to track derived by the evaluation methods described in section 5.4. In section 5.7, the outcome of the experiments is discussed. Finally, section 5.8 concludes with the properties of our method for harvesting good features to track by analyzing appearance change information of point features.

5.1 Feature Generation

In order to generate a spatial-temporal description of point features, we have to locate and track point features over time. We use three different methods for feature point detection: (i) the Harris corner detector [HS88], (ii) the Scale-Invariant Feature Tracker (SIFT) [Low99, Low04], and (iii) the Speeded Up Robust Features (SURF) detector [BETG08]. A comprehensive performance evaluation of local descriptors including SIFT and Harris points was done by Mikolajczyk and Schmid [MS05].

Harris Corner Detector

The Harris corner detector was introduced by Harris and Stephens [HS88] in 1988. It bases on an auto-correlation matrix which represents the structure of the local neighborhood of a certain position in the image. Depending on the eigenvalues of the auto-correlation matrix it is possible to differ between a uniform region (no significant eigenvalue), a contour (one significant eigenvalue), and an interest point (two significant eigenvalues). The benefit of the Harris corner detector is, that it is not necessary to compute the eigenvalues. Instead, for each pixel, one can compute the measure of the corner response. Let λ_1, λ_2 be the eigenvalues of the auto-correlation matrix $\boldsymbol{M}$. Then, the measure of corner response is computed by

$$R(x, y) = det(\boldsymbol{M}) - k(trace(\boldsymbol{M}))^2 \tag{5.1}$$

where

$$\begin{aligned} det(M) &= \lambda_1 * \lambda_2 \\ trace(M) &= \lambda_1 + \lambda_2 \\ \text{k} &= \text{empirical constant } (0.04 \geq k \geq 0.06) \end{aligned} \tag{5.2}$$

Followed by a non-maximal suppression process, a large R indicates a corner, whereas a negative R with a large magnitude indicates an edge. A small $|R|$ indicates a homogeneous region.

As the name implies, the Harris Corner detector does not provide any information but the position of an interest point.

Scale-Invariant Feature Transform (SIFT)

The Scale-Invariant Feature Tracker (SIFT) descriptor was first introduced by Lowe et al. [Low04]. Beside describing an interest point, the original paper suggests the detection of interest points by cascade filtering to minimize the costs for feature extraction. Using Lowe's SIFT, the main procedures are

1. Difference of Gaussian (DoG) for potential interest point detection of all scales

2. interest point localization, i.e. identifying location and scale of each interest point

3. identification of the interest point orientation(s)

4. descriptor generation by local image gradients at the selected scale around the interest point

The main benefits of SIFT are the scale and orientation-invariance (retrieved by steps 2 and 3). SIFT is also tolerant against local shape distortion and illumination changes (due to step 4).

Speeded Up Robust Features (SURF)

Speeded Up Robust Features (SURF) is a scale and rotation-invariant interest point detector and descriptor. SURF was presented in 2006 by Bay et al. [BTG06, BETG08]. Compared to SIFT, which relies on gradient information, SURF uses first-order Haar wavelet responses in both x and y direction. To reduce the computational costs, integral images are generated.

The value of an integral image $I_{\sum(\boldsymbol{x})}$ at position $\boldsymbol{x} = (x, y)^T$ is represented by the sum of all pixels within a rectangular region R starting with the upper left corner at the image origin $R_{UL} = (0, 0)^T$ and the lower right corner at the position $\boldsymbol{x} = (x, y)^T$ (see eq. 5.3).

$$I_{\sum(\boldsymbol{x})} = \sum_{i=0}^{i \leq x} \sum_{j=0}^{j \leq y} I(i, j) \tag{5.3}$$

SURF uses filters which increase in size, while the image remains at the original size. This differs from SIFT, where sub-sampled images are subtracted to retrieve the Difference of Gaussian (DoG). For interest point localization, SURF uses the Hessian matrix, i.e. it uses a second-order derivative filter. Additionally a 3D non-maximum suppression is applied on the local neighborhood and the neighboring scales.

5.2 Trajectory Generation

In order to analyze the appearance change of point features over time, we have to connect the independently detected point features at each frame t_i. For that purpose, in frame t_i we let one of the feature detectors introduced in section 5.1 detect the features in our image. We then apply the Kanade-Lucas-Tomasi feature tracker (KLT) [LK81, TK91, ST94, Bou00] to compute the estimated position in frame t_{i+1}. Then, in frame t_{i+1} again we use our feature detector. Once we got the set of features in frame t_{i+1}, we compare this set with the estimated position of the KLT tracker. If this position matches, we connect this feature from frame t_i to frame t_{i+1}, i.e. we created a trajectory of length 2. Figure 5.1 illustrates a set of point feature trajectories in a region of interest at a certain time step t_i.

Figure 5.1: Visualization of collected trajectories (blue lines) in a region of interest (blue box)

5.3 The Space Time Appearance Descriptor

The Space-Time Appearance (STA) descriptor collects appearance information over time. It was first introduced by Brkic et al. [BPSK11]. They propose two variants, the STA1 and the STA2 descriptor.

The STA1 descriptor divides the patch $p_j(t_i)$ into a grid of $m \times n$ cells. For each

grid cell, a histogram is calculated, which reflects the distribution of the image measurement for each frame t_i and over time. Each histogram consist of k bins, i.e. the spectrum of measured values is divided into k intervals. The individual histograms of each frame t_i are accumulated over time regarding a weighting function.

The STA2 descriptor relies on the STA1 descriptor. For each of the k bins of each of the individual (not yet accumulated) $m \times n$ histograms per frame t_i, the STA2 descriptor generates a histogram with l bins. As with STA1, this information is accumulated over time, i.e the STA2 descriptor models the distribution of each bin of the STA1 descriptor.

The following subsections explain the STA1 and STA2 descriptors with regard to our implementation.

5.3.1 STA1

The STA1 descriptor divides the input patch $p_j(t_i)$ into an $m \times n$ grid. For each grid cell, a one dimensional histogram with k bins is generated. The input data is is some image measurement at time t_i, where t are the individual time steps on the point feature trajectory. Over time t_i, the $m \times n$ histograms of the STA1 descriptor are accumulated according to a weighting function. We use the weighting function

$$hist_{t_i} = \frac{1}{t_i}((t_{i-1}) - \varepsilon) * hist_{t_{i-1}} + (1 + \varepsilon) * hist_{t_i})) \tag{5.4}$$

where

$$\varepsilon = w_{STA1} * (t_{i-1}). \tag{5.5}$$

Setting w_{STA1} to zero generates equally distributed histograms, a weight $0 < w_{STA1} \leq 1$ gives more impact on the recent STA1 histograms over time.

Figure 5.2 illustrates an image patch together with its STA1 histograms at a certain time step t_i. The STA1 grid contains 2×2 cells with 4 bins per histogram.

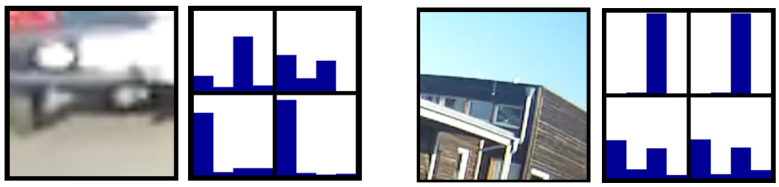


Figure 5.2: Two examples for a 2×2 STA1 descriptor of a SIFT interest point. $w_{STA1} = 0$, trajectory length: 28 frames

5.3.2 STA2

The STA2 descriptor relies on the STA1 descriptor (see chapter 5.3.1). It independently analyzes all k bins of the $m \times n$ histograms of the individual STA1 descriptor, i.e. it does not use the accumulated STA1 information. This results in $k \times m \times n$ histograms. The STA2 descriptor models each STA1 bin's distribution on a STA2 histogram with l bins. The number l of bins of a STA2 histogram is user-defined and can be an arbitrary value greater than zero. Any modification in the STA1 descriptor settings (e.g. number of cells, etc) has an impact on the STA2 descriptor. Over time, the STA2 descriptor is accumulated with equal weighting for each time step t_i.

Fig. 5.3 illustrates a sample SIFT point feature patch with its STA1 and STA2 descriptors. The STA2 descriptor analyzes each bin of each STA1 descriptor histograms. Same colors in the STA1 and STA2 descriptor histograms indicate their affiliation. E.g. for each bin in the yellow STA1 cell histogram, an individual STA2 histogram is generated, containing the bin's distribution over time.

5.4 STA Properties Evaluation

Once we create the STA descriptors for the patches $p(t_i)$, there are two ways of analyzing the data: (i) analyzing the accumulated descriptor at the last time step t_n of the trajectory, or (ii) analyzing the the accumulated descriptor of a patch $p_j(t_i)$ online at each time step t_i. For the latter, we require a certain minimum trajectory

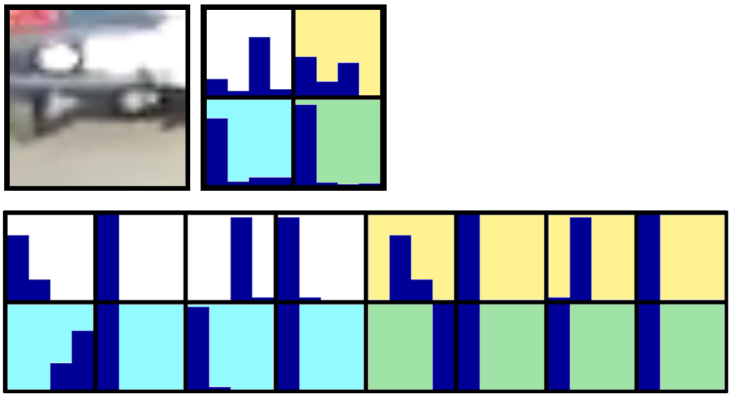


Figure 5.3: STA1 descriptor (top) and the according STA2 descriptor (bottom) of a SIFT interest point patch at the same time step as in fig. 5.2.

length before we can reliably analyze the accumulated descriptor. In our case, we require at least a minimum trajectory length n of 5 and 10 frames respectively, depending on the experiments. Any of the proposed evaluation tools is also applied on the STA1 and STA2 descriptor.

We analyze the STA data with three different histogram statistics evaluation methods, (i) the minimum entropy, (ii) the minimum variance, and (iii) the Lyapunov exponent. All evaluation methods are described in detail below. Each histogram of the STA1 and STA2 descriptor is inspected by the evaluation tools. As the STA2 descriptor of a point feature has $k \times m \times n$ histograms (a histogram for each bin k of $m \times n$ histograms in the STA1 descriptor), this results in $k \times m \times n$ values for each evaluation method. Evaluating the STA1 descriptor leads to $m \times n$ values. To achieve one representative value at the time step t_i per evaluation method, our implementation allows to select the minimum, the maximum, or the mean value of all histograms of a point feature STA descriptor at the time step t_i. In our experiments, for each evaluation method we uniformly compute the mean value over all STA histograms per frame t_i and let this mean value be the representative value for the appearance change of a point feature at the time step t_i.

5.4.1 Minimum Entropy

The origin of entropy leads back in 1948 to a paper by Shannon [Sha48]. In information theory, Shannon's entropy H is used to measure the average information content. It provides information on the average number of bits per symbol required for encoding. The entropy H is defined as

$$H(p_1, \ldots, p_n) = -\sum_{i}^{n} p_i \log_2 p_i. \tag{5.6}$$

where $p_i = P(X = i)$ is the probability the word i appears.

Applying the Minimum Entropy to STA Data

At each timestep t_i we apply the entropy computation to each grid cell of the STA1 descriptor. Thus, in case of an $m \times n$ grid per point feature, the descriptor consists of $m \times n$ histograms and we get $m \times n$ entropies per timestep t_i and point feature $p_j(t_i)$. The entropy representing the whole STA1 descriptor at a timestep t_i is gained by averaging the $m \times n$ entropies. For the STA2 descriptor, we proceed identically. However, $k \times m \times n$ grid cells have to be processed.

As both descriptors STA1 and STA2 are accumulated over time (see section 5.3, eq. 5.4), the impact of an appearance change of a point feature patch $p_j(t_i)$ is higher for a lower i (i.e. the point feature trajectory is shorter). I.e. when evaluating the appearance change online, any change contributes more when the trajectory is still short and contributes less when the trajectory grows. E.g. in case of equal weighting, at $t_i, i = 2$ the appearance information represents 50% of the information available, at $t_i, i = 100$ the information of the current frame represents a hundreth of the total available information. A constant appearance change variance $\sigma^2 = 0$ is a special case and has equal impact over time.

An STA's entropy value of 0 indicates, that no appearance change occurred over the last frames. The larger the entropy value, the more the appearance of the point feature changed over time. As we want to gather stable reliable point features for the pose estimation, point features with low entropies are favored. I.e., we search for n point features with the STAs providing the n lowest averaged entropies.

5.4.2 Minimum Variance

The variance is a frequently used tool in statistics and describes the expectation of spreading of a set of numbers. In other words, it measures the squared deviation of a random variable of its expected value. For discrete data series x_i with $i = 1 \dots n$, the variance σ^2 can be computed by

$$\sigma^2 = \frac{1}{n} \sum_{i=1}^{n} (x_i - \bar{x})^2 \tag{5.7}$$

where $\bar{x}$ represents the arithmetic mean of the data series x_i.

Applying the Variance to STA Data

At each timestep t_i we apply the variance computation to each grid cell of the STA1 descriptor. Thus, in case of an $m \times n$ grid per point feature, the descriptor consists of $m \times n$ histograms and we get $m \times n$ variances per timestep t_i and point feature $p_j(t_i)$. The variance representing the whole STA1 descriptor at a timestep t_i is gained by averaging the $m \times n$ variances. For the STA2 descriptor, we proceed identically. However, $k \times m \times n$ grid cells have to be processed.

An STA's variance value of 0 indicates, that no appearance change occurred over the last frames. The larger the variance, the more the appearance of the point feature deviates over time. In order to get stable and reliable point features for the pose estimation, point features with low variances are favored. I.e., we search for n point features with the STAs providing the n lowest averaged variances.

5.4.3 Lyapunov Exponent

The Lyapunov exponent [Lor63, ABK91, Sch88] describes the behavior of infinitesimally close trajectories. Originally it is explained as the average rate of divergence or convergence of two neighboring trajectories in the phase space. More generally, the Lyapunov exponent is a quantitative measure of the sensitive dependence on the initial conditions. Figure 5.4 illustrates the idea of the Lyapunov exponent.

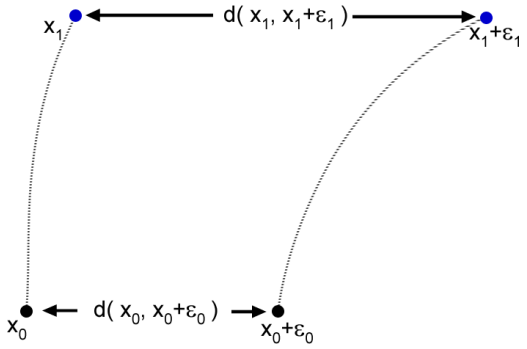


Figure 5.4: Basic idea of the Lyapunov exponent.

At $t = 0$ we have two points, x_0 and $x_0 + \epsilon_0$, separated by the distance

$$d(x_0, x_0 + \epsilon_0) = \epsilon_0. \tag{5.8}$$

Assuming one dimension only, at $t = 1$ we gain $x_n = f(x_{t-1})$, i.e. $t_1 = f(x_0)$. With this mapping we are able to compute x_1 and $x_1 + \epsilon_1$, illustrated as blue dots in fig. 5.4. The distance between these two points is

$$d(x_1, x_1 + \epsilon_1) = d(f(x_0), f(x_0 + \epsilon_0)) = \epsilon_1. \tag{5.9}$$

The Lyapunov exponent is defined by exponential growth, so we can rewrite eq. 5.9 as

$$d(x_1, x_1 + \epsilon_1) = d(f(x_0), f(x_0 + \epsilon_0)) = e^{\lambda t} d(x_0, x_0 + \epsilon_0). \tag{5.10}$$

We can rewrite the equation as

$$|f'(x_0)| = \frac{d(f(x_0), f(x_0 + \epsilon_0))}{d(x_0, x_0 + \epsilon_0)} = e^{\lambda t}. \tag{5.11}$$

For more than one iteration, we can rewrite eq. 5.11 as

$$|f'(x_i)| = \frac{d(f(x_i), f(x_i + \epsilon_i))}{d(x_i, x_i + \epsilon_i)}. \tag{5.12}$$

Then, the Lyapunov exponent λ is specified as

$$\lambda = \lim_{t\to\infty} \frac{1}{t} \sum_{i=1}^{t} \ln |f'(x_i)|. \tag{5.13}$$

The Relationship of Lyapunov and Entropy

Regarding Schuster [Sch88], we can use the Lyapunov exponent to measure the average loss of information. Imagine the range [0, 1] divided into n equal intervals. Assume, a point x_0 can occur in each interval with equal probability $1/n$. By observing the point x_0 distributed by equal probability, we gain the average information content

$$I_0 = -\sum_{i=1}^{n} \frac{1}{n} \log_2 \frac{1}{n} = \log_2 n \tag{5.14}$$

One can obviously see that eq. 5.14 is equal to eq. 5.6, which describes the entropy H. Decreasing n will reduce the information content I_0 and it will become zero for $n = 1$.

Let us introduce a linear map $f(x)$, which maps any point x_i linearly to $f(x_i)$. For a linear map $f(x) = k * x + d$, the first derivation $f'(x_i)$ is k for all x_i. Then it is shown by [Sch88] that the linear mapping function $f(x)$ scales each interval by the factor $|k|$. This change of the interval's size leads to a loss ΔI of the information content after the mapping (see eq. 5.15). The n intervals and their mapping by $f(x)$ to $\frac{n}{k}$ intervals is illustrated in fig. 5.5.

$$\Delta I = -\sum_{i=1}^{\frac{n}{k}} \frac{k}{n} \log_2 \frac{k}{n} + \sum_{i=1}^{n} \frac{1}{n} \log_2 \frac{1}{n} = -\log_2 a = -\log_2 |k| \tag{5.15}$$

Generalizing eq. 5.15, i.e. allowing partly-linear mapping functions $f(x)$ (i.e. the function is linear within the interval) and observing many iterations, the mean loss of the information content can be expressed as

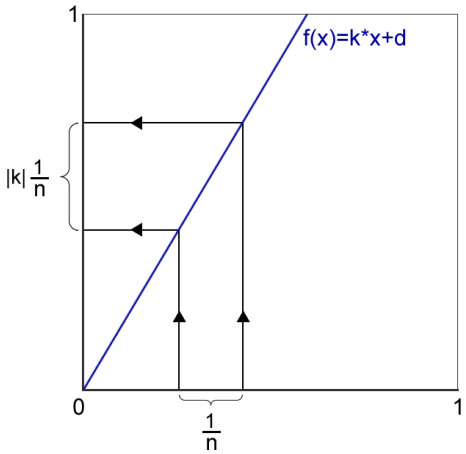


Figure 5.5: Change of interval size by a linear map (adapted from [Sch88]).

$$\overline{\Delta I} = -\lim_{t\to\infty} \sum_{i=0}^{t-1} log_2|f'(x_i)| \tag{5.16}$$

which is equal to eq. 5.13, the Lyapunov exponent.

Summing up, the entropy H_t represents the average information content at time t, while the Lyapunov exponent λ_t reflects the average change of the information content at time t.

Applying the Lyapunov exponent to STA Data

As with the entropy and the variance, we apply the Lyapunov exponent to the STA descriptors. However, a certain kind of input data has to be provided for the Lyapunov exponent. As the Lyapunov exponent describes the average change of the information content provided, we choose the entropy as input data. I.e., the Lyapunov exponent describes the average change of the entropy over time.

As we want one value for one STA descriptor of a point feature per time step t_i, we choose the entropy as described above as input data for the Lyapunov exponent. Over time we compute the Lyapunov exponent for each STA descriptor. As the Lyapunov exponent relies on a deviation, at least two frames are required for its

computation.

Similar to the entropy and variance, any STA descriptor with a Lyapunov exponent of zero indicates no appearance change occurred between t_{i-1} and t_i. In contrast to the entropy and variance, the Lyapunov exponent is able to distinguish between convergence and divergence. Negative values indicate convergence, i.e. the appearance change converges towards zero. Positive values indicates divergence, i.e. the appearance change from t_{i-1} to t_i is more severe than from t_{i-2} to t_{i-1}. As we work with accumulated information in both cases, STA1 and STA2, the Lyapunov exponent does not only compare the information at time step t_i with t_{i-1} but the current information with whole accumulated information available at the previous time step.

As with the entropy and the variance, we gather a subset of n point features, which provides stable point features. We prefer point features with a Lyapunov exponent of zero, as this means the point feature has a constant appearance. However, in case the size n of the required point feature subset is higher than the available point features with a Lyapunov exponent of zero, we search in the range $-\infty < r \leq 0$, where r starts at 0. This means, our first choice is a Lyapunov exponent of zero. However, if not enough point features fulfill this requirement we prefer slight convergence over divergence.

Once the Lyapunov exponents for all STA descriptors are computed at a certain time step t_i, we choose the n best point features. For the Lyapunov exponent this means,

5.5 Implementation Details

We developed a C++ application in combination with OpenCV 2.2. The program accepts two types of input: either interest points or bounding boxes (e.g. labeled groundtruth objects). In both cases a certain region of interest (ROI) is selected which represents the input to the STA descriptor. The application implements both descriptors, STA1 and STA2, which are applicable in real-time. The grid size, the number of bins, the weighting term w_{STA1}, and grid overlapping can be defined by

the user. The kind of image value used for the STA descriptor is also user-defined. Currently the application supports (i) grayscale, (ii) hue, (iii) saturation, and (iv) gradients as image values.

5.5.1 Trajectory Generation

The C++ application allows us to detect and track point features online over time by different detectors (Harris [HS88], SURF [BETG08], SIFT [Low99, Low04]) and generate the STA information. We implemented the optical flow by using the OpenCV method *calcOpticalFlowPyrLK()* [Bou00], which implements a pyramidal implementation of the Lucas Kanade Feature Tracker.

Beside the generation of point feature trajectories, the application can also handle any rectangular regions as input (e.g. ground truth annotated objects). In this case, the rectangular region is treated like a point feature patch. We only require an unique ID per region of interest. The application is able to generate trajectories of the rectangular regions offline. However, the bounding box information cannot be collected by detectors, i.e. they have to be manually generated or an existing labeled dataset can be used. In case of rectangular regions of interest, the STA descriptor then can be applied to the whole rectangular region.

5.5.2 Patch Adaption for STA descriptor

Once the trajectories are generated, at each time step t_i the dimension of the ROI (i.e. either the surrounding patch of an interest point or the groundtruth annotated object patches) has to be verified. This means, that we have to guarantee, that the grid specified by the STA1 fits into the patch. For that reason, if the STA1 grid does not perfectly fit into the region of interest, we resample the patch. However, to resample as little as possible we determine whether to shrink or grow the patch size. We calculate the resampled patch size by

$$\boldsymbol{size}_{resampled} \begin{cases} \boldsymbol{size} - \boldsymbol{mod}_{size} & \text{... if } \boldsymbol{mod}_{size} < \frac{\boldsymbol{grid}_{STA1}}{2} \\ \boldsymbol{size} - \boldsymbol{mod}_{size} + \boldsymbol{grid}_{STA1} & \text{... if } \boldsymbol{mod}_{size} >= \frac{\boldsymbol{grid}_{STA1}}{2} \end{cases} \tag{5.17}$$

where

$$\boldsymbol{mod}_{size} = \begin{bmatrix} \text{width} & (\text{mod } grid_{x_{STA1}}) \\ \text{height} & (\text{mod } grid_{y_{STA1}}) \end{bmatrix} \tag{5.18}$$

and

$$\boldsymbol{grid}_{STA1} = \begin{bmatrix} grid_{x_{STA1}} \\ grid_{y_{STA1}} \end{bmatrix} \tag{5.19}$$

contains the number of pixels of a grid cell in x and y direction. Once the resampled patch size is computed, the image patch is resized by bilinear interpolation using OpenCV's *resize()* method.

5.6 Experiments

In this experiment section, we analyze the appearance change information of point features by various evaluation methods. We want to exploit the similarities and dissimilarities of appearance change information calculated by entropy, variance, and the Lyapunov exponent.

By having the appearance change information gathered by the STA descriptor, we compute several subsets out of all point features:

- minimum entropy subset (E_L)
- minimum variance subset (V_L)
- Lyapunov exponent subset (L)

In addition to these three subsets, we also compute the

- maximum entropy subset (E_H), and the
- maximum variance subset (V_H).

The subsets contain n point features with the highest entropy and variance values respectively and are used to compare the results with the minimum subsets. A maximum Lyapunov exponent subset was computed too, but it hardly provided

common point features over two successive frames. I.e. most point features in this subset had very a short trajectory, as new appearing point features were favored by the maximum Lyapunov exponent. Thus, this subset is not comparable to the other five subsets listed above.

We run the subset generation multiple times, i.e. for each type of subset we generate different subset sizes. Particularly, for an experiment we generate subset sizes of 15, 20, 25, 50, 75, and 100 point features per frame. Depending on the scene (i.e. the amount of point features available in a frame), we generate only certain subset sizes. I.e., it makes no sense to find 50 point features with the lowest entropy if only 75 point features are available in a frame (this would lead to an intersection of 50% of lowest and highest entropies).

Next, we want to compare the generated subsets among each other. Thus, we generate the following intersections:

- intersection of the minimum entropy subset with the Lyapunov exponent subset (E_L/L)
- intersection of the minimum entropy subset with the minimum variance subset (E_L/V_L)
- intersection of the Lyapunov exponent subset with the minimum variance subset (L/V_L)
- intersection of the minimum entropy subset with the minimum variance subset and the Lyapunov exponent subset ($E_L/V_L/L$)
- intersection of the maximum entropy subset with the Lyapunov exponent subset (E_H/L)
- intersection of the maximum entropy subset with the maximum variance subset (E_H/V_H)
- intersection of the Lyapunov exponent subset with the maximum variance subset (L/V_H)

- intersection of the maximum entropy subset with the the maximum variance subset and the Lyapunov exponent subset($E_H/V_H/L$)

In the next subsections, three experiments are introduced. Each experiment contains a sequence with a moving observer. Throughout the sequence, we gain point features by two different salient point detectors, the Harris corner detector and the SIFT detector/descriptor. We generate point feature trajectories by connecting point features over time as described in section 5.2 and analyze the point feature appearance change by the STA descriptors. As we analyze the appearance change, we require a minimum trajectory length providing us appearance change information. For each experiment sequence we run the tests with a minimum trajectory length of 5 and 10 frames. We apply the histogram statistics evaluation methods described above (entropy, variance, Lyapunov exponent) on the STA descriptors to gain different subsets. Finally, a table shows the intersection sets evolved from intersecting the subsets.

As we provide just a single Lyapunov exponent subset (see subsection 5.4.3, preferring a stable behavior and favoring slight convergence), we expect a higher intersection rate of Lyapunov with the minimum subsets than with the maximum subsets. The input for the Lyapunov exponent subset is the entropy (i.e. it models the change of entropy). In case of the STA2 descriptor, which describes the average appearance change information, we expect that the minimum entropy subset has a large intersection with the Lyapunov exponent subset. As the minimum entropy implies little information loss, the corresponding variance is also small. As the Lyapunov exponent reflects the average information change, we expect the subset gathered by the Lyapunov exponent to be different from the entropy and variance subset in case of the STA1 descriptor. However, in case of the STA2 descriptor, which describes the average appearance change of point features, we are expecting a large intersection with the minimum entropy subset.

5.6.1 Experiment KIT_Seq_01

This experiment uses the dataset KIT_Seq_01 (see section 2.7). As we require monocular data in this experiment section, we use the data from the left camera only. We run this experiment with two different minimum trajectory lengths, 5 frames and 10 frames.

Table 5.1 contains the general run information of this experiment. Run 1 required a minimum trajectory length of 5 frames, run 2 had a minimum trajectory length of 10 frames.

Table 5.1: General information of experiment KIT_Seq_01. Min T Len: the minimum trajectory length required; Detector Type: either Harris corner detector or the SIFT detector/descriptor; Min # T: minimum amount of trajectories in a frame; Max # T: maximum amount of trajectories in a frame; Mean # T: average amount of trajectories throughout the scene.

	Min T Len	Detector Type	Min # T	Max # T	Mean # T
Run 1	5	Harris	93	448	296.70
		SIFT	97	485	340.02
Run 2	10	Harris	44	334	187.11
		SIFT	52	349	224.86

Run 1

In this run, only point feature trajectories with a minimum length of 5 are used. Table 5.2 and 5.3 show the intersection of the subsets in percent for different sized subsets retrieved from the STA1 descriptor. The tables 5.4 and 5.5 represent the counterpart for the STA2 descriptor. In table 5.2 the point features were detected by the Harris corner detector, in table 5.3 with SIFT. The first column indicates the subset size per frame, columns two to five contain several intersection information for the minimum entropy subset, the minimum variance subset, and the Lyapunov exponent subset. Columns six to nine contain the intersection information for the maximum entropy subset, the maximum variance subset, and the Lyapunov exponent subset.

Table 5.2: Experiment KIT_Seq_01: Intersection in % of the subsets retrieved from the STA1 descriptor. Point feature detector: Harris corner, minimum trajectory length: 5.

	E_L/L	E_L/V_L	L/V_L	E_L/V_L/L	E_H/L	E_H/V_H	L/V_H	E_H/V_H/L
15	5.86	0.77	6.13	0.15	6.17	0.00	5.79	0.00
20	8.45	1.41	8.05	0.26	8.16	0.00	8.02	0.00
25	10.67	1.40	10.97	0.25	10.64	0.00	9.98	0.00

Table 5.3: Experiment KIT_Seq_01: Intersection in % of the subsets retrieved from the STA1 descriptor. Point feature detector: SIFT, minimum trajectory length: 5

	E_L/L	E_L/V_L	L/V_L	E_L/V_L/L	E_H/L	E_H/V_H	L/V_H	E_H/V_H/L
15	4.44	0.23	5.25	0.00	5.13	0.00	4.10	0.00
20	5.83	0.75	7.59	0.06	7.39	0.00	5.98	0.00
25	7.79	0.90	9.22	0.09	9.40	0.00	7.70	0.00

Table 5.4: Experiment KIT_Seq_01: Intersection in % of the subsets retrieved from the STA2 descriptor. Point feature detector: Harris corner, minimum trajectory length: 5.

	E_L/L	E_L/V_L	L/V_L	E_L/V_L/L	E_H/L	E_H/V_H	L/V_H	E_H/V_H/L
15	44.02	69.12	38.35	36.40	2.99	18.24	3.52	0.69
20	41.78	69.54	35.55	33.16	4.77	23.53	5.83	1.32
25	41.89	69.45	34.64	32.23	6.00	27.26	7.36	2.28

Table 5.5: Experiment KIT_Seq_01: Intersection in % of the subsets retrieved from the STA2 descriptor. Point feature detector: SIFT, minimum trajectory length: 5.

	E_L/L	E_L/V_L	L/V_L	E_L/V_L/L	E_H/L	E_H/V_H	L/V_H	E_H/V_H/L
15	72.41	83.56	67.55	66.78	0.92	17.47	1.19	0.08
20	66.93	80.09	62.18	61.15	2.41	24.14	2.41	0.37
25	63.49	76.02	58.80	56.94	3.54	28.53	3.59	0.85

Run 2

Opposed to run 1, this run uses point feature trajectories with a minimum length of 10 only. The table 5.6 to 5.9 show the intersection of the subsets in percent for different sized subsets retrieved from the STA1 descriptor. The results in tables 5.6 and 5.7 are generated with STA1 subsets, the results in tables 5.8 and 5.9 with STA2 subsets.

Table 5.6: Experiment KIT_Seq_01: Intersection in % of the subsets retrieved from the STA1 descriptor. Point feature detector: Harris corner, minimum trajectory length: 10.

	E_L/L	E_L/V_L	L/V_L	E_L/V_L/L	E_H/L	E_H/V_H	L/V_H	E_H/V_H/L
15	12.31	0.75	12.23	0.12	12.11	0.00	11.40	0.00
20	15.98	0.71	16.39	0.09	16.18	0.03	15.06	0.03

Table 5.7: Experiment KIT_Seq_01: Intersection in % of the subsets retrieved from the STA1 descriptor. Point feature detector: SIFT, minimum trajectory length: 10.

	E_L/L	E_L/V_L	L/V_L	E_L/V_L/L	E_H/L	E_H/V_H	L/V_H	E_H/V_H/L
15	9.23	0.51	9.70	0.00	9.70	0.00	8.84	0.00
20	11.54	0.41	13.11	0.00	13.67	0.00	11.75	0.00

Table 5.8: Experiment KIT_Seq_01: Intersection in % of the subsets retrieved from the STA2 descriptor. Point feature detector: Harris corner, minimum trajectory length: 10.

	E_L/L	E_L/V_L	L/V_L	E_L/V_L/L	E_H/L	E_H/V_H	L/V_H	E_H/V_H/L
15	31.87	52.70	24.77	20.16	8.21	20.71	10.14	2.37
20	34.11	55.80	27.37	21.57	11.72	27.43	13.58	4.44

Table 5.9: Experiment KIT_Seq_01: Intersection in % of the subsets retrieved from the STA2 descriptor. Point feature detector: SIFT, minimum trajectory length: 10.

	E_L/L	E_L/V_L	L/V_L	E_L/V_L/L	E_H/L	E_H/V_H	L/V_H	E_H/V_H/L
15	47.38	59.84	42.05	38.03	6.00	19.25	5.56	1.07
20	45.09	59.56	39.79	34.32	9.23	27.10	8.17	2.66

Discussion

With a minimum point feature trajectory amount of 93 for the Harris corner detector and 97 for the SIFT (see table 5.1), in run 1 we were able to create subsets of a maximum of 25 point features per frame only. As we fetch the point features with the lowest and highest values according to an evaluation scheme (e.g. entropy), at least double of the point features required for a subset have to be available per frame. Obviously, run 2 provides less point feature trajectories than run 1 due to the longer minimum trajectory length requirement. Thus, in run 2 it was not possible to build subsets with more than 20 point features per subset. Otherwise, the lowest subset would overlap with the highest subset. Per frame, SIFT provides both a higher minimum trajectory amount and a higher maximum trajectory amount.

One observation which can clearly be made is, that the STA1 Lyapunov exponent intersection set with the STA1 maximum entropy subset (E_H/L) is about the same size as with the STA1 minimum entropy subset (E_L/L) (see tables 5.2, 5.3, and 5.6). The intersection set of the STA1 maximum entropy subset with the STA1 maximum variance subset (E_H/V_H) is always zero. Therefore, also the intersection of the maximum entropy subsets with the maximum variance subset and the Lyapunov exponent subset ($E_H/V_H/L$) is always zero, too. Additionally, the intersection of the STA2 minimum subsets with the Lyapunov exponent subset ($E_L/L, E_L/V_L, L/V_L, E_L/V_L/L$) seem to be constant independent of the subset size (see tables 5.4, 5.5, 5.8, and 5.9). However, the overlap is higher in case of SIFT. Additionally, these STA2 intersections of the minimum subsets are significantly higher than the STA2 intersections with the maximum subsets.

5.6.2 Experiment VMG_Bike_01

This experiment uses the dataset VMG_Bike_01, described in section 2.1. Again, the experiment consists of two runs, one with a minimum point feature trajectory length of 5 and one with a length of 10. Table 5.10 contains the general experiment information for both runs.

Table 5.10: General information of experiment VMG_Bike_01. Min T Len: the minimum trajectory length required; Detector Type: either Harris corner detector or the SIFT detector/descriptor; Min # T: minimum amount of trajectories in a frame; Max # T: maximum amount of trajectories in a frame; Mean # T: average amount of trajectories throughout the scene.

	Min T Len	Detector Type	Min # T	Max # T	Mean # T
Run 1	5	Harris	148	222	179.79
		SIFT	236	333	283.41
Run 2	10	Harris	75	119	94.23
		SIFT	130	187	164.34

Run 1

In this run, all point features are used with a point feature trajectory length of at least 5 frames. Tables 5.11 to 5.14 show the intersections in percent of the subsets.

Table 5.11: Experiment VMG_Bike_01: Intersection in % of the subsets retrieved from the STA1 descriptor. Point feature detector: Harris corner, minimum trajectory length: 5.

	E_L/L	E_L/V_L	L/V_L	E_L/V_L/L	E_H/L	E_H/V_H	L/V_H	E_H/V_H/L
15	9.81	0.48	9.14	0.19	9.24	0.00	8.00	0.00
20	12.21	0.50	11.93	0.29	12.00	0.00	11.43	0.00
25	14.80	0.40	14.86	0.29	15.03	0.00	13.66	0.00

Table 5.12: Experiment VMG_Bike_01: Intersection in % of the subsets retrieved from the STA1 descriptor. Point feature detector: SIFT, minimum trajectory length: 5.

	E_L/L	E_L/V_L	L/V_L	E_L/V_L/L	E_H/L	E_H/V_H	L/V_H	E_H/V_H/L
15	4.57	0.00	6.10	0.00	6.10	0.00	4.57	0.00
20	6.43	0.93	8.00	0.07	8.50	0.00	6.43	0.00
25	8.40	1.03	9.60	0.06	9.66	0.00	8.23	0.00
50	17.83	2.11	18.77	0.49	18.94	0.00	17.60	0.00
75	26.38	2.11	28.86	0.69	28.78	0.00	25.50	0.00
100	34.37	3.40	37.49	1.37	36.91	0.04	34.06	0.01

Table 5.13: Experiment VMG_Bike_01: Intersection in % of the subsets retrieved from the STA2 descriptor. Point feature detector: Harris corner, minimum trajectory length: 5.

	E_L/L	E_L/V_L	L/V_L	E_L/V_L/L	E_H/L	E_H/V_H	L/V_H	E_H/V_H/L
15	40.57	69.90	35.62	33.14	3.52	21.62	5.71	0.86
20	40.07	71.00	32.93	30.50	5.43	26.57	7.71	1.36
25	40.51	69.43	32.06	29.37	6.23	30.57	9.66	1.49

Table 5.14: Experiment VMG_Bike_01: Intersection in % of the subsets retrieved from the STA2 descriptor. Point feature detector: SIFT, minimum trajectory length: 5.

	E_L/L	E_L/V_L	L/V_L	E_L/V_L/L	E_H/L	E_H/V_H	L/V_H	E_H/V_H/L
15	83.05	93.14	82.48	81.81	0.00	26.95	0.10	0.00
20	79.07	81.57	76.14	74.71	0.07	32.36	0.14	0.00
25	76.17	71.03	69.54	66.80	0.63	36.34	0.40	0.11
50	65.77	78.14	56.00	54.49	3.20	54.57	4.46	1.69
75	63.18	77.83	52.99	50.21	7.56	66.06	11.77	5.35
100	62.56	79.56	54.50	50.44	15.50	74.41	21.80	12.29

Run 2

Opposed to the first run, only point features with a minimum trajectory length of 10 frames are used. Tables 5.15 to 5.18 show the intersection results.

Table 5.15: Experiment VMG_Bike_01: Intersection in % of the subsets retrieved from the STA1 descriptor. Point feature detector: Harris corner, minimum trajectory length: 10.

	E_L/L	E_L/V_L	L/V_L	E_L/V_L/L	E_H/L	E_H/V_H	L/V_H	E_H/V_H/L
15	17.95	0.10	16.62	0.10	17.23	0.00	17.64	0.00
20	23.54	0.62	20.54	0.15	21.54	0.00	21.31	0.00
25	28.12	1.91	26.95	0.62	28.37	0.00	25.85	0.00

Discussion

Again, the STA1 maximum entropy subset hardly ever intersects with the STA1 maximum variance subset subset (E_H/V_H) (see tables 5.11, 5.12, 5.15, and 5.16).

Table 5.16: Experiment VMG_Bike_01: Intersection in % of the subsets retrieved from the STA1 descriptor. Point feature detector: SIFT, minimum trajectory length: 10.

	E_L/L	E_L/V_L	L/V_L	E_L/V_L/L	E_H/L	E_H/V_H	L/V_H	E_H/V_H/L
15	7.28	1.03	10.67	0.00	10.97	0.00	7.18	0.00
20	9.54	1.00	13.08	0.00	13.54	0.00	9.69	0.00
25	13.72	0.86	16.00	0.06	16.55	0.00	12.98	0.00
50	29.20	1.26	32.31	0.31	31.85	0.00	28.43	0.00

Table 5.17: Experiment VMG_Bike_01: Intersection in % of the subsets retrieved from the STA2 descriptor. Point feature detector: Harris corner, minimum trajectory length: 10.

	E_L/L	E_L/V_L	L/V_L	E_L/V_L/L	E_H/L	E_H/V_H	L/V_H	E_H/V_H/L
15	37.74	61.13	28.51	23.59	8.21	28.92	11.28	1.85
20	41.23	58.85	31.77	24.69	11.85	35.62	15.92	4.31
25	44.12	58.52	35.88	27.08	16.12	42.77	20.12	6.46

Table 5.18: Experiment VMG_Bike_01: Intersection in % of the subsets retrieved from the STA2 descriptor. Point feature detector: SIFT, minimum trajectory length: 10.

	E_L/L	E_L/V_L	L/V_L	E_L/V_L/L	E_H/L	E_H/V_H	L/V_H	E_H/V_H/L
15	68.51	66.87	59.79	55.59	1.33	32.21	1.03	0.31
20	65.00	73.15	53.69	52.00	2.00	36.77	1.77	0.46
25	62.28	71.02	52.31	49.60	2.95	42.15	2.83	0.86
50	60.15	71.88	50.89	44.92	11.17	63.38	15.82	7.11

This leads to an intersection of 0% of the maximum entropy subsets with the maximum variance subset and the Lyapunov exponent subset ($E_H/V_H/L$). Also the intersection of the STA2 minimum subsets and the STA2 Lyapunov exponent subset ($E_L/L, E_L/V_L, L/V_L, E_L/V_L/L$) seem to be constant independent from the subset size (see tables 5.4, 5.5, 5.8, and 5.9). Again these intersections are significantly higher than the STA2 maximum subset intersections.

5.6.3 Experiment VMG_Bike_02

This experiment uses the dataset VMG_Bike_02 (see section 2.2). Table 5.19 contains the general experiment information for both runs.

Table 5.19: General information of experiment VMG_Bike_02. Min T Len: the minimum trajectory length required; Detector Type: either Harris corner detector or the SIFT detector/descriptor; Min # T: minimum amount of trajectories in a frame; Max # T: maximum amount of trajectories in a frame; Mean # T: average amount of trajectories throughout the scene.

	Min T Len	Detector Type	Min # T	Max # T	Mean # T
Run 1	5	Harris	67	227	139.41
		SIFT	115	258	177.17
Run 2	10	Harris	29	121	78.52
		SIFT	45	161	97.94

Run 1

Tables 5.20 to 5.23 show the intersection results for subsets containing point features with a minimum trajectory length of 5 frames.

Table 5.20: Experiment VMG_Bike_02: Intersection in % of the subsets retrieved from the STA1 descriptor. Point feature detector: Harris corner, minimum trajectory length: 5.

	E_L/L	E_L/V_L	L/V_L	E_L/V_L/L	E_H/L	E_H/V_H	L/V_H	E_H/V_H/L
15	11.84	3.97	12.00	0.57	11.66	0.00	10.91	0.00
20	15.82	3.72	15.29	0.73	14.71	0.00	14.80	0.00
25	19.41	4.24	19.36	1.06	18.61	0.00	18.68	0.00

Table 5.21: Experiment VMG_Bike_02: Intersection in % of the subsets retrieved from the STA1 descriptor. Point feature detector: SIFT, minimum trajectory length: 5.

	E_L/L	E_L/V_L	L/V_L	E_L/V_L/L	E_H/L	E_H/V_H	L/V_H	E_H/V_H/L
15	9.48	1.07	9.02	0.16	9.14	0.00	9.16	0.00
20	11.84	1.36	12.55	0.34	12.23	0.00	11.58	0.00
25	14.88	1.62	15.78	0.44	15.51	0.00	14.73	0.00
50	29.30	3.60	30.46	1.23	30.41	0.00	28.54	0.00

Table 5.22: Experiment VMG_Bike_02: Intersection in % of the subsets retrieved from the STA2 descriptor. Point feature detector: Harris corner, minimum trajectory length: 5.

	E_L/L	E_L/V_L	L/V_L	E_L/V_L/L	E_H/L	E_H/V_H	L/V_H	E_H/V_H/L
15	51.90	68.19	42.09	40.48	4.92	18.78	6.03	0.79
20	49.54	69.52	39.81	37.60	7.72	26.46	9.40	1.97
25	48.87	71.28	39.50	36.86	9.98	34.35	12.44	3.39

Table 5.23: Experiment VMG_Bike_02: Intersection in % of the subsets retrieved from the STA2 descriptor. Point feature detector: SIFT, minimum trajectory length: 5.

	E_L/L	E_L/V_L	L/V_L	E_L/V_L/L	E_H/L	E_H/V_H	L/V_H	E_H/V_H/L
15	81.86	92.43	80.43	80.00	0.36	33.97	0.36	0.11
20	77.50	85.73	74.68	73.76	0.94	39.01	1.22	0.43
25	73.96	80.39	69.03	67.67	2.03	44.04	2.72	1.02
50	64.52	76.88	54.70	52.10	10.81	63.71	15.62	7.53

Run 2

Tables 5.24 to 5.25 show the intersection results for subsets containing point features with a minimum trajectory length of 10 frames. For this run, we only provide results for the SIFT. We were not able to evaluate the Harris corner subsets, as at certain frames less than 30 point features were available. I.e., with a subset size of 15 point features at minimum an overlap of the lowest and highest subset would occur.

Table 5.24: Experiment VMG_Bike_02: Intersection in % of the subsets retrieved from the STA1 descriptor. Point feature detector: SIFT, minimum trajectory length: 10.

	E_L/L	E_L/V_L	L/V_L	E_L/V_L/L	E_H/L	E_H/V_H	L/V_H	E_H/V_H/L
15	16.93	0.81	16.86	0.28	16.96	0.00	16.52	0.00
20	22.94	1.07	22.89	0.47	22.23	0.00	22.21	0.00

Table 5.25: Experiment VMG_Bike_02: Intersection in % of the subsets retrieved from the STA2 descriptor. Point feature detector: SIFT, minimum trajectory length: 10.

	E_L/L	E_L/V_L	L/V_L	E_L/V_L/L	E_H/L	E_H/V_H	L/V_H	E_H/V_H/L
15	57.42	64.78	47.52	44.31	6.25	41.04	6.32	2.45
20	56.19	66.64	46.49	41.99	9.34	48.17	10.76	4.72

Discussion

The STA1 maximum entropy subset hardly ever intersects with the STA1 maximum variance subset (E_H/V_H). Also, there's no intersection of the STA1 maximum entropy subsets with the STA1maximum variance subset and the STA1 Lyapunov exponent subset ($E_H/V_H/L$). The STA2 minimum subsets and the STA2 Lyapunov exponent subset again seem to be constant independent from the subset size.

5.7 Summary

In this chapter we explained how point features are gained from image sequences and how point feature trajectories are created. Along a point feature trajectory, the appearance of a point feature may change. We showed, that by using the STA descriptor it is possible to describe these appearance changes. We introduced histogram statistics evaluation methods such as entropy and variance to evaluate the STA descriptors of point features. We explained the origin of the Lyapunov exponent and how it is related to the entropy. With three different methods (entropy, variance, and Lyapunov exponent) we analyzed the appearance change information collected by the STA descriptors and generated several subsets out of the whole point feature set. We analyzed the behavior of the subsets by looking at their intersection sets.

In the experiments sections 5.6.1-5.6.3, the following observations were made:

- There is no overlap of the STA1 maximum entropy subset and the STA1 maximum variance subset.
- The intersection of the STA1 minimum variance subset with the Lyapunov entropy subset (L/V_L) is similar to the intersection of the STA1 maximum

entropy with the STA1 maximum variance (E_H/V_H).

- The intersection set of the STA1 minimum entropy subset with the STA1 minimum variance subset and the STA1 Lyapunov exponent is almost empty.
- The intersection set of the STA1 maximum entropy subset with the STA1 maximum variance subset and the STA1 Lyapunov exponent is almost empty.
- The intersections of the STA2 minimum subsets and the STA2 Lyapunov exponent subset intersections are constant independent of the subset size.
- The intersections of the STA2 minimum subsets and the STA2 Lyapunov exponent subset are significantly higher than the intersections with the STA2 maximum subsets.

5.8 Conclusion

Depending on the STA descriptor type - STA1 or STA2 - the intersection sets are different. The STA2 minimum has an intersection with both the STA2 minimum variance subset and the Lyapunov exponent subset. In contrast, the STA1 minimum entropy subset has only a small intersection with the Lyapunov exponent, as the Lyapunov exponent describes the average change of the input data (i.e. entropy), while the entropy describes the average information content. I.e., the STA2 minimum subsets together with the Lyapunov subset contain many common point features. Using the subsets for the observer pose estimation, we are expecting similar results for these subsets. In contrast, no assumption can be made for the STA1 minimum subsets and the STA1 Lyapunov exponent subset, as the subsets contain different point features. For certain, the maximum subsets are a bad choice for gathering stable point features. Any of the maximum subsets contains different point features.

In the next chapter, we apply the different evaluation methods (entropy, variance, Lyapunov exponent) on real-world data. With the different subsets gained, we want to estimate the observer pose and compare it against each other.

6

MSaM plus STA

In chapters 3 and 4 it was shown how inlier and outlier information can be used to identify moving foreground objects. As mentioned, we cannot guarantee that a moving object is identified as a whole. Rather, it may happen that only a subpart of an object is detected. We solved this issue in chapter 4 by combining our MSaM system with a 2D detector and tracker. The introduced 2D detector is capable of a specific category of objects, i.e. we deployed an identifier for moving persons. Yet, our MSaM system works with any other detector too. The basic idea is to identify the whole object by the additional 2D detector in case that MSaM detects a sub-part of a moving object only.

Still we need a minimum of stationary background information which is required for pose estimation. At least 50% of stationary background information is required. If more than 50% of the available information is located on the dominant foreground motion, the pose estimation results will deteriorate. We address this problem by finding a reliable subset of features, which are in the stationary background and are supposed to have good properties for tracking. In 1994, Shi and Tomasi [ST94] came up with the question "What are good features to track?", mentioning the

problem of feature tracking. They state that even if a point feature patch contains highly textured content, it may not be well suited for tracking. Furthermore, they address the problem of virtual crossings. E.g., having a transmission line in the foreground and a flagpole in the back. Although they do not cross in the 3D real world there will be a crossing in a 2D image. Shi and Thomasi [ST94] explained that by analyzing the appearance change over time, one is able to distinguish between real and virtual crossings. For that purpose, they introduced a measure of feature dissimilarity, which reflects the change of appearance over time. They also stated that right point features are always those which make a tracker work best.

With the Space-Time Appearance (STA) descriptors by Brkic et al. [BPSK11] we are able to describe appearance changes over time. With the introduced evaluation methods on histogram statistics in chapter 5 we are in the position to identify several subsets out of the available point features. While we showed the similarities and dissimilarities of the different point feature subsets in the previous chapter, in this chapter these subsets are used for pose estimation. The results of the different subsets are compared with the computed observer poses of both, manually generated reference data solely containing stationary background point features and all point features identified as inliers by our MSaM system. We retrieve the point feature appearance change information by the histogram statistics evaluation of the STA descriptors (see chapter 5). This evaluation involves the minimum entropy, the Lyapunov exponent, and the minimum variance.

6.1 Experimental Setup

The experiments rely on the datasets used in chapter 3. Basically, MSaM is capable to distinguish between stationary background information (inliers) and moving foreground motion (outliers). In case of a subpart detection of a moving object by our MSaM system, some point features of the moving object might wrongly be classified as inliers. In order to have a reliable reference data set, wrong inliers are excluded manually from this set. We harvest appearance change information from both, all MSaM inliers and the reference data set and evaluate the data by the above

mentioned methods.

For each of the described data sets, the observer pose is estimated and compared against the pose estimated using the reference data. Basically, we use two approaches for pose estimation, a non-robust approach and a robust approach. The non-robust approach computes the pose by using all the available data per dataset followed by Bundle Adjustment for optimization. The robust approach additionally uses RANSAC to apply the best model for pose estimation before the data is passed to the Bundler.

The execution of each experiment requires the following steps:

(1) generation of the reference data,

(2) robust pose estimation with the reference data,

(3) robust pose estimation with all MSaM inliers,

(4) non-robust pose estimation with the reference data,

(5) non-robust pose estimation with all MSaM inliers for comparison,

(6) subset generation from the reference data by STA evaluation,

(7) subset generation from all MSaM inliers by STA evaluation,

(8) robust pose estimation with the STA subset data,

(9) non-robust pose estimation with the STA subset data.

6.1.1 Generation of the Reference Data

We build on the data retrieved by MSaM introduced in chapter 3. First, any outliers identified by MSaM are neglected. To assure that no moving foreground object contributes to the stationary background data (e.g. in case an object is not detected entirely), any remaining inliers located on a moving foreground object are manually removed.

6.1.2 Robust Pose Estimation with the Reference Data

From the manually generated reference data, the observer's pose is generated robustly. We start by selecting the best point features regarding to their stereo-correspondences. For that purpose, a 2D-homography RANSAC robustly fits an appropriate homography-model and its supporting point features. By that it is ensured, that point features with correct stereo-correspondences are selected only. We use the 2D-homography algorithm RANSAC by Kovesi [Kov]. The point features selected by the 2D-homography RANSAC are passed as input to a 3D-pose RANSAC. The 3D-pose RANSAC is a modified version of Kovesi's RANSAC [Kov]. It robustly fits a pose model per frame, i.e. we gain the relative observer motion per frame. Furthermore, as a post processing step, the obtained observer poses are optimized with the robust Bundle Adjustment implementation by Klein et al. [KM07]. The resulting observer motion is used as reference, i.e. any other experimental results are compared with this data.

6.1.3 Robust Pose Estimation with all MSaM inliers

We proceed as described in subsection 6.1.2, i.e. we select the 2D point correspondences returned by the 2D-homography RANSAC after robust model fitting, compute the pose with the point features derived from the 3D-pose RANSAC, and apply the robust Bundle Adjustment implementation. However, eventually appearing wrong inliers are not removed from the data set as described in 6.1.1, i.e. all MSaM inliers are used as input. In all the experiments, the outcome is compared with the reference observer poses, which are derived as described in subsection 6.1.2.

6.1.4 Non-Robust Pose Estimation with the Reference Data/all MSaM inliers

In contrast to the procedure in subsections 6.1.2 and 6.1.3, the 2D-homography RANSAC and the 3D-pose RANSAC are skipped in this case. Either the reference data or all MSaM inliers is used as input. We triangulate all point correspondences

and compute the pose with all 3D point features appearing in two subsequent frames. Then, the robust Bundler by Klein et al. [KM07] is applied to achieve the results. The outcome is compared to the reference observer poses, which are derived as described in subsection 6.1.2.

6.1.5 Subset Generation from Reference Data

As explained in chapter 5, the appearance change information of point features is collected over time with the STA descriptors. This data is evaluated by the minimum entropy, the Lyapunov exponent, and the minimum variance. All three evaluation methods are applied to both, STA1 and STA2 descriptors. Here, we analyze the STA descriptors of the reference data generated as described in subsection 6.1.1 and generate subsets of the best n point features with each STA descriptor evaluation method. The amount n of taken point features per subset is scene dependent. It depends on the amount of the available point features which correlate with texture available in the scene and the image dimensions.

6.1.6 Subset Generation from All MSaM Inliers

We proceed as described in subsection 6.1.5. Different subsets are generated by performing STA histogram statistics evaluation with the minimum entropy, the Lyapunov exponent, and the minimum variance. However, these subsets are generated from the entire MSaM inlier data.

6.1.7 Robust Pose Estimation with the STA Subset Data

By generating the subsets, twelve different subsets are gained:

- Minimum entropy subset on STA1 and STA2, applied to the reference data as well as to all MSaM inliers.
- Lyapunov exponent subset on STA1 and STA2, applied to the reference data as well as to all MSaM inliers.

- Minimum variance subset on STA1 and STA2, applied to the reference data as well as to all MSaM inliers.

For each subset, the observer pose is estimated as described in subsection 6.1.2. Solely the input data differs, i.e. each subset of point features generated by the STA evaluation methods is used as input.

6.1.8 Non-Robust Pose Estimation with the STA Subset Data

The robust pose estimation with each of the twelve subsets is similar to the non-robust estimation described in subsection 6.1.4. One subset per run is used as input data. Neither the 2D-homography RANSAC nor the 3D-pose RANSAC is applied. After all, the robust Bundle Adjustment implementation by Klein et al. [KM07] is applied to the data.

6.2 Experiments

The experimental setup requires three parameters to set. The first parameter specifies the maximum distance a point feature is allowed to be distanced from the observer. For all experiments we set this distance threshold to 100 m. The second parameter sets the threshold for the homography estimation by RANSAC [Kov]. As the 2D-coordinates are normalized so that their mean distance is $\sqrt{2}$ from the origin, this threshold is a relative value. We set this parameter to 0.1 for all experiments. The third parameter represents a threshold for the 3D-pose RANSAC. The used 3D-pose RANSAC is adaptive, i.e. if not enough inliers are found, the algorithm adaptively increases this threshold. Thus, this threshold represents the hardest constraint on retrieving a pose model. If not enough point features support this model (due to the hard constraint) the threshold is increased and the RANSAC again tries to retrieve an appropriate model. By that it is ensured that the 3D-pose RANSAC is applicable to different scenes with different reconstruction qualities. For

all experiments this threshold is set to 25 mm. In general, all three parameters are set equally for all experiments.

In this section, four selected experiments are shown. The first experiment is an indoor setup to illustrate the basic proceedings and results. Experiments 2, 3 and 4 cover street sequences with one moving observer and various moving foreground objects. We use the generated reference data, the MSaM inlier data, and the data gained by the various STA evaluation methods introduced in chapter 5 to estimate the observer pose and compare it to reference observer motion computed with the reference data. We expect the robust pose estimation by STA evaluated data to provide similar results as the estimated poses derived from the reference data. Furthermore, the non-robust pose estimation by STA data is expected to outperform the non-robust pose estimation with all MSaM inliers. To validate the observer poses retrieved by the reference data, the results of experiments 2, 3, and 4 are compared with the GPS data provided by the Karlsruhe dataset [Gei].

As each of the experiments consists of comprehensive data, only the first experiment contains all figures. For the other experiments, the figures are attached in the appendix.

6.2.1 Experiment VMG_Lab_01

This experiment uses the stereo dataset VMG_Lab_01 (see section 2.3). The scene consists of 180 frames. However, the reconstruction starts at frame 5 due to the subsets derived by STA evaluation. Point features have to have a certain minimum trajectory length to provide reliable appearance change information. As mentioned in subsection 6.1.5, the subsets have a certain size of n which depends on the experiment. For this experiment, each subset has a size $n = 50$, i.e. in each frame the 50 most relevant point features according to the selected STA histogram statistics evaluation are chosen. I.e., for the minimum entropy subset, the 50 point features with the lowest entropy are selected.

Figure 6.1 illustrates the estimated observer motion for the reference data (left) and for all MSaM inliers (right).

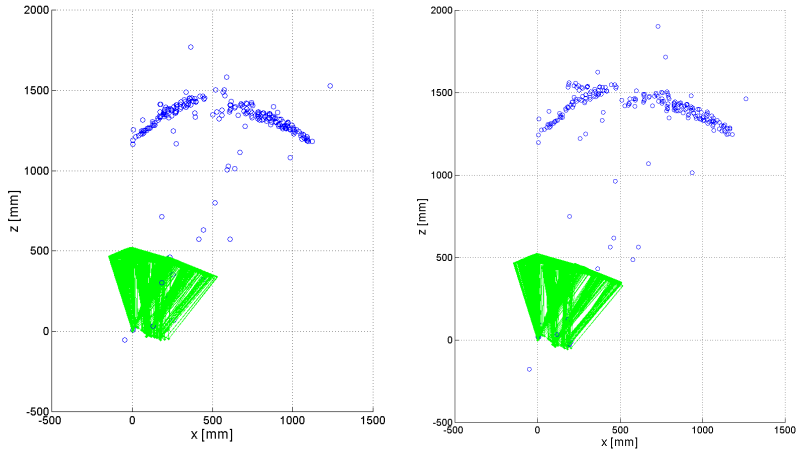


Figure 6.1: ExperimentVMG_Lab_01: Robust pose estimation with the reference data only (left) and with all MSaM inliers (right). Estimated camera poses (green triangles), estimated structure (blue circles).

This experiment contains 16 evaluations. First, the observer pose is computed from the reference data and all MSaM inliers robustly (see fig. 6.1) and non-robustly, which leads to 4 evaluations. Second, the observer pose is computed with data gathered by analyzing the reference data with the minimum entropy, the Lyapunov exponent, and the minimum variance on both, the STA1 and STA2 descriptors of the point features. This leads to six evaluations. Finally, step 2 is repeated on all MSaM inliers, again leading to 6 evaluations.

This sequence is a controlled setup from the lab. Thus, the observer does not move rapidly and only within a couple of centimeters. Furthermore, the observer moves sidewards most of the time. As MSaM detects most of the outliers in this scene, we are expecting the robust observer motion estimations from the reference data and all MSaM inliers to be very similar (see fig. 6.1). Also, all robust pose estimations from any STA subset should deliver similar results compared to the reference data. Additionally, we are expecting the non-robust pose estimations of any subset retrieved from STA evaluation as well as all MSaM inliers to be reliable yet a little bit more imprecise.

The experimental evaluation is structured as follows: first, a subsection with all

evaluations on the reference data gives detailed results on the pose estimation of each STA derived subset computed robustly and non-robustly. In general, for each evaluation the mean distance difference compared to the reference observer pose (derived by the reference data) as well as the variance of the distance difference are shown. While the mean distance shows the average distance to the reference observer pose, the variance reflects the stability of the observer pose estimation compared to the reference observer pose. I.e., a large variance implies a very different observer motion behavior compared to the reference observer pose. Second, a subsection contains all evaluations on all MSaM inliers. This section again contains all STA derived subsets. However, they were generated by analyzing the MSaM inliers instead of the reference data. All the evaluations are again compared with the reference data.

6.2.1.1 Reference Data Evaluation

Table 6.1 contains the average common point feature set of the minimum entropy subset, the Lyapunov exponent subset, and the minimum variance subset in two subsequent frames for STA1 and STA2. Table 6.2 contains the mean L2 norm pose distances in m of the reference data, the minimum entropy subset, the Lyapunov exponent subset, and the minimum variance subset compared to the reference data. Each subset provides four measures: each evaluation was applied on STA1 and STA2 and were computed robustly and non-robustly. Table 6.3 shows the variance of the distances. Again the poses retrieved from the the reference data as well as the three STA subsets derived by the minimum entropy, the Lyapunov exponent, and the minimum variance are compared to the reference data. As in table 6.2, the STA subsets have four evaluations, depending on STA1 or STA2 evaluation as well as on robust or non-robust computation.

Figure 6.2 illustrates the non-robust pose estimation results of the STA2 minimum entropy subset, the STA2 Lyapunov exponent subset, and the STA2 minimum variance subset. One can clearly see the relation between variances of distances (see table 6.3) and the pose estimation results. For small variances, the result deteriorate. Larger variances of distances indicate that the pose estimation failed.

As the observer poses retrieved by the robust computation with the reference

data is identical to the reference observer poses computed as described in subsection 6.1.2, both the mean distance in table 6.2 and the variance in table 6.3 are zero. This holds for all experiments.

Figures 6.3 to 6.5 illustrate the robustly reconstructed observer pose by the minimum entropy, the Lyapunov exponent, and the minimum variance. As expected, all robust pose estimations provide very similar observer pose estimations. Solely the Lyapunov exponent subset applied to STA1 does not give any result. This is due to insufficient common point features in two subsequent frames required for pose estimation. I.e., with this subset it was not possible to estimate the observer pose over the entire sequence.

Table 6.1: Experiment VMG_Lab_01: Average common point feature sets of the minimum entropy subset, the Lyapunov exponent subset, and the minimum variance subset in two successive frames for STA1 and STA2.

	Entropy	Lyapunov	Variance
STA1	40.49	-	40.46
STA2	39.46	23.46	40.47

Table 6.2: Experiment VMG_Lab_01: Mean L2 norm pose distance between non-robust/robust STA1 and STA2 evaluation of the reference data and the reference data in m.

	Reference Data	Descriptor Type			
			Entropy	Lyapunov	Variance
Non-robust background	1.53	STA1	0.10	-	0.11
		STA2	0.10	1.79	0.26
Robust background	0	STA1	0.02	-	0.15
		STA2	0.01	0.02	0.27

Table 6.3: Experiment VMG_Lab_01: Variance of distances between non-robust/robust STA1 and STA2 evaluation the reference data and the reference data in m^2.

	Reference Data	Descriptor Type			
			Entropy	Lyapunov	Variance
Non-robust background	225.13	STA1	1.26	-	0.16
		STA2	0.28	1278.91	149.61
Robust background	0	STA1	0.18	-	0.15
		STA2	0.10	0.27	0.19

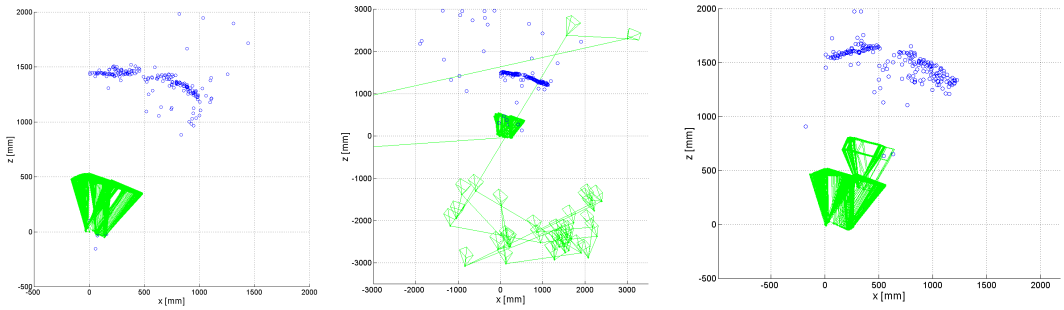


Figure 6.2: Experiment VMG_Lab_01: Non-robust pose estimation results of the STA2 minimum entropy subset (left), the STA2 Lyapunov exponent subset (center), and the STA2 minimum variance subset (right). Estimated camera poses (green triangles), estimated structure (blue circles). The non-robust pose estimation with STA2 Lyapunov exponent subset and the STA2 minimum variance subset failed.

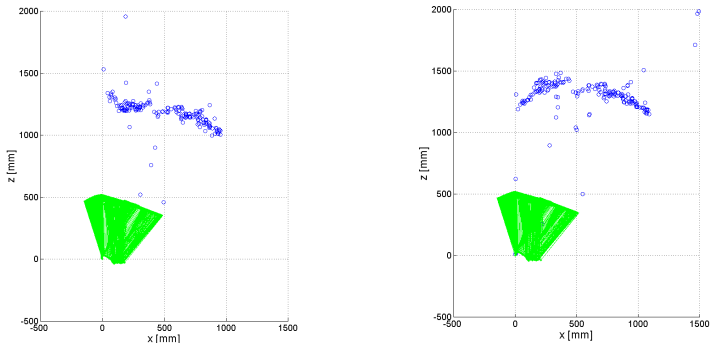


Figure 6.3: Experiment VMG_Lab_01: Robustly estimated pose with the STA1 minimum entropy (left) and the STA2 minimum (right) subset generated from the reference data. Estimated camera poses (green triangles), estimated structure (blue circles).

6.2.1.2 All MSaM Inliers Evaluation

Table 6.4 contains the average common point feature set of the minimum entropy subset, the Lyapunov exponent subset, and the minimum variance subset in two subsequent frames for STA1 and STA2. Table 6.5 contains the mean L2 norm pose distances in m of all MSaM inliers compared to the reference data, the minimum entropy subset, the Lyapunov exponent subset, and the minimum variance subset. Again, the subsets provide four measures: each evaluation was applied on STA1 and

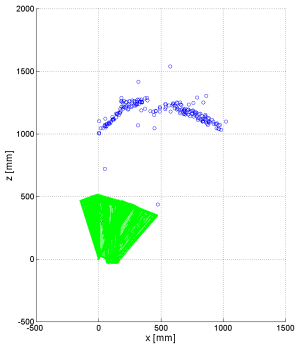


Figure 6.4: Experiment VMG_Lab_01: Robustly estimated pose with the STA2 Lyapunov exponent subset generated from the reference data. Estimated camera poses (green triangles), estimated structure (blue circles). The evaluation of the STA1 Lyapunov exponent subset failed.

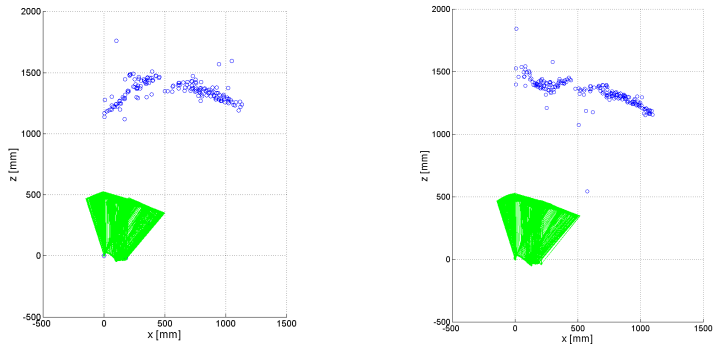


Figure 6.5: Experiment VMG_Lab_01: Robustly estimated pose with the STA1 minimum variance (left) and the STA2 minimum variance (right) subset generated from the reference data. Estimated camera poses (green triangles), estimated structure (blue circles).

STA2 and were computed robustly and non-robustly. Table 6.6 shows the variance of the distances. The estimated poses generated with the three STA subsets derived by all MSaM inliers, the minimum entropy, the Lyapunov exponent, and the minimum variance are compared to the reference data. As in table 6.5, the STA subsets have four evaluations, depending on STA1 or STA2 evaluation as well as on robust or non-robust computation.

In contrast to the reference data evaluation, the robust computation with all

MSaM inliers is not equal to the reference observer poses (see subsection 6.1.2). I.e., the set of all MSaM inliers contains point features which are falsely identified as inliers. The reference data does not contain these point features.

Table 6.4: Experiment KIT_Seq_01: Average common point feature sets of the minimum entropy subset, the Lyapunov exponent subset, and the minimum variance subset in two successive frames for STA1 and STA2.

	Entropy	Lyapunov	Variance
STA1	31.27	-	26.19
STA2	28.45	16.35	29.45

Table 6.5: Experiment VMG_Lab_01: Mean L2 norm pose distance between non-robust/robust STA1 and STA2 evaluation of all MSaM inliers and the reference data in m.

	All MSaM Inliers	Descriptor Type			
			Entropy	Lyapunov	Variance
Non-robust background	0.49	STA1	0.18	-	0.02
		STA2	0.35	0.02	2.22
Robust background	0	STA1	0.02	-	0.02
		STA2	0.01	0.01	0.02

Table 6.6: Experiment VMG_Lab_01: Variance of distances between non-robust/robust STA1 and STA2 evaluation of all MSaM inliers and the reference data in m^2.

	All MSaM Inliers	Descriptor Type			
			Entropy	Lyapunov	Variance
Non-robust background	5.17	STA1	9.24	-	0.19
		STA2	59.22	0.20	24164.58
Robust background	0.01	STA1	0.18	-	0.25
		STA2	0.11	0.11	0.15

Figure 6.6 illustrates the non-robust pose estimation results of the STA2 minimum entropy subset, the STA2 Lyapunov exponent subset, and the STA2 minimum variance subset. Again, the relation between the variances of distances (see table 6.6) and the pose estimation results can be clearly seen. For small variances, the result deteriorate. Larger variances of distances indicate that the pose estimation failed.

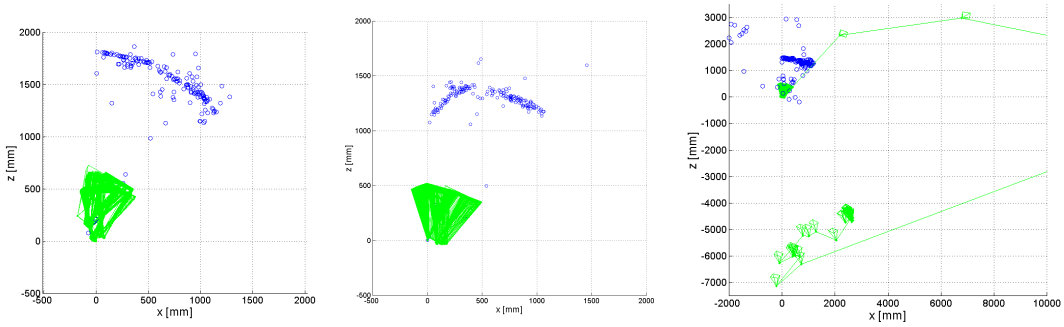


Figure 6.6: Experiment VMG_Lab_01: Non-robust pose estimation results of the STA2 minimum entropy subset (left), the STA2 Lyapunov exponent subset (center), and the STA2 minimum variance subset (right). Estimated camera poses (green triangles), estimated structure (blue circles). The non-robust pose estimation with minimum variance subset and the minimum entropy subset fails.

As expected, all robust pose estimations provide very similar observer pose estimations. They are similar to those gained by the reference data. At a first glance, the mean differences of the non-robust pose estimations (table 6.5) seem to yield similar results compared to those of the reference data (table 6.2). The minimum entropy subset and the variance subset provide a better pose estimation when applied on the reference data (table 6.3). In contrast, the STA2 Lyapunov exponent subset is able to reconstruct the observer pose when applied on all MSaM data, while it fails in case of the reference data. But one has to take this result with care, as only an average of 16.35 common point features were available in two subsequent frames throughout the scene. The STA1 Lyapunov exponent subset failed again, as not enough common point features were available. This is due to insufficient common point features in two subsequent frames required for pose estimation. Figures 6.7 to 6.9 illustrate the robustly reconstructed observer pose by the minimum entropy, the Lyapunov exponent, and the minimum variance.

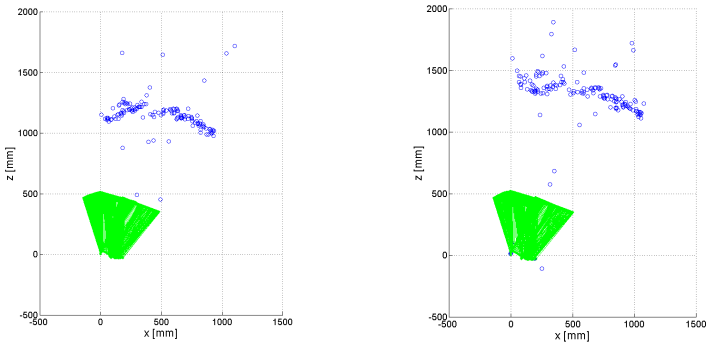


Figure 6.7: Experiment VMG_Lab_01: Robustly estimated pose with the STA1 minimum entropy (left) and the STA2 minimum entropy (right) subset generated from all MSaM inliers. Estimated camera poses (green triangles), estimated structure (blue circles).

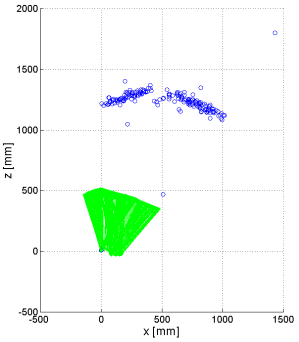


Figure 6.8: Experiment VMG_Lab_01: Robustly estimated pose with the STA2 Lyapunov exponent subset generated from the reference data. Estimated camera poses (green triangles), estimated structure (blue circles). The evaluation of the STA1 Lyapunov exponent subset failed.

6.2.2 Experiment KIT_Seq_01

This experiment uses the dataset KIT_Seq_01 (see section 2.7). The complete scene consists of 180 frames. As the subsets are generated by analyzing appearance change over time, a certain minimum information is required. In our case, we require the appearance change information of a point feature for at least 5 frames to start to evaluate, i.e. we start at frame 5. The STA subset size is 100 per frame, i.e. the

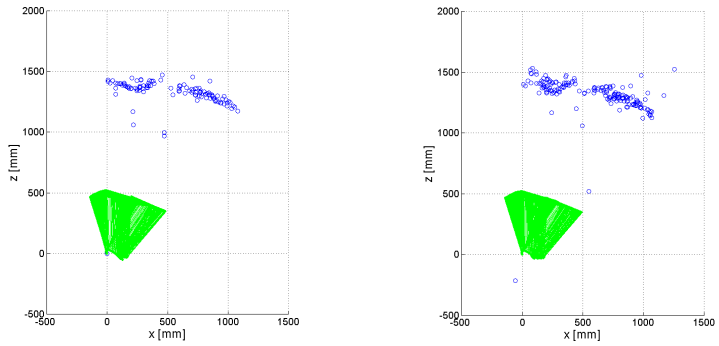


Figure 6.9: Experiment VMG_Lab_01: Robustly estimated pose with the STA1 minimum variance (left) and the STA2 minimum variance (right) subset generated from all MSaM inliers. Estimated camera poses (green triangles), estimated structure (blue circles).

100 best point features according to a specific STA histogram evaluation method are chosen per frame. E.g. for the minimum entropy, the 100 point features with the lowest entropy are chosen.

The reference observer motion is solely computed on point features which are located on the stationary background, i.e. the manually generated reference data. Figure 6.10 illustrates the estimated observer motion with the reference data (left) and all MSaM inliers (right). As some of the MSaM inliers are located on moving objects, the scene reconstruction diverges from the scene reconstructed with the reference data.

Fortunately, this dataset provides metric GPS data which can be used for comparison of the visual odometry data. The GPS coordinate system is a right-hand coordinate system, our visual odometry coordinate system is a left-hand coordinate system. Additionally, the cameras are mounted with a pitch of $-4.6°$ and the axes are not aligned properly, i.e. converting the GPS coordinate system to a left-hand coordinate system and aligning solely the origin of the coordinate systems is insufficient. Rather, the rotation ΔR between the two coordinate systems has to be computed, including the pitch of $-4.6°$ of the cameras. Figure 6.11 shows the X/Y view (left) and X/Z view (right). The reference observer pose (retrieved from the reference data) is illustrated in green, the metric GPS information (aligned at the

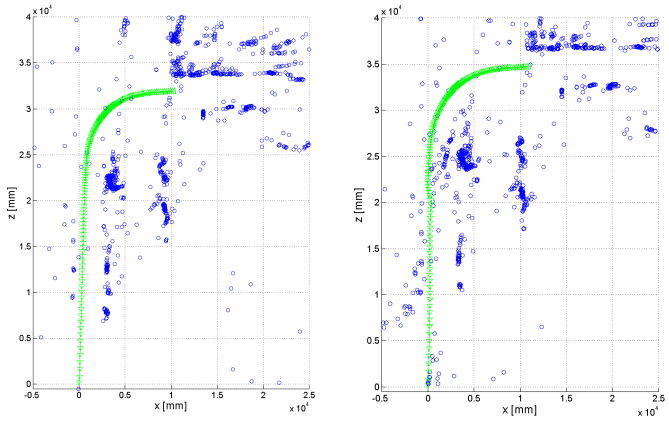


Figure 6.10: Experiment KIT_Seq_01: Robust estimation of the observer pose with the reference data (left) and all MSaM inliers (right). Estimated camera poses (green triangles), estimated structure (blue circles).

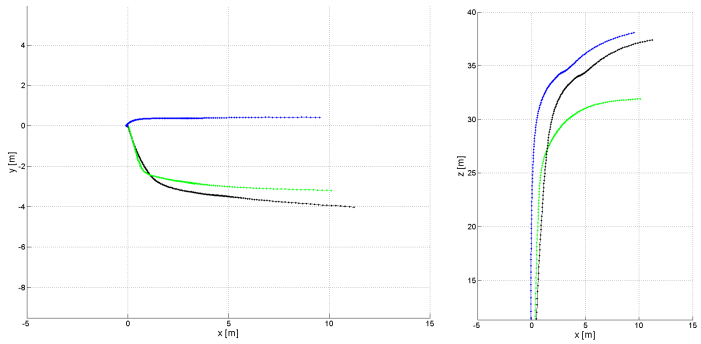


Figure 6.11: Experiment KIT_Seq_01: GPS reference data (blue), GPS reference data aligned with visual odometry coordinate system (black), stationary background reference data (green). X/Y view (left), X/Z view (right).

origin of our visual odometry coordinate system) is shown in blue. The aligned and rotated GPS information is shown in black.

As stated on the Karlsruhe dataset [Gei] homepage, the GPS data is not always as precise as it should be. Comparing the pictures of the sequence with the GPS data, the GPS data should show an observer motion turn of approximately 90°. We believe, that around $z = 35m$ the GPS data deteriorate. This assumption is supported by a saddle point at $x = 5m$, $z = 34m$ (see fig. 6.11, right, black line).

Table 6.7 contains the mean distance between the reference observer pose and the GPS data as well as the variance of the difference between the observer pose reference data and the GPS data. The low variance value indicates a similar shape of both GPS and visual odometry.

Table 6.7: Experiment KIT_Seq_01: Comparison of GPS coordinates with the reference data.

Mean L2 norm pose distance in m	Variance of distances in m^2
3.49	2.07

6.2.2.1 Reference Data Evaluation

Table 6.8 contains the average common point feature set of the minimum entropy subset, the Lyapunov exponent subset, and the minimum variance subset in two subsequent frames for STA1 and STA2. The average common point feature set is similar for the minimum entropy subset and the minimum variance subset. The Lyapunov exponent subset provides fewer common point features in two successive frames. I.e., in each frame, the Lyapunov exponent prefers point features with short trajectories.

Table 6.9 contains the mean L2 norm pose distances in m of the reference data and the three subsets data compared to the reference data. The non-robust pose estimation with all point features of the reference data fails (indicated by the high mean L2 norm value). Obviously, some point features with wrong stereo correspondences falsify the pose estimation process. Table 6.10 shows the variance of the distances. The variance of distances indicates that any non-robust pose estimation provides deteriorate results. The STA1 minimum variance subset fails to estimate the observer pose.

With the reference data, all subsets recover the shape of observer motion. In terms of mean distance difference, the robust computation with the minimum entropy subset on STA1 and the Lyapunov exponent subset on STA1 performs best. The variance of the STA1 Lyapunov exponent subset in table 6.10 is larger than the variance of the STA1 minimum entropy subset. This effect is also visible in the pose

Table 6.8: Experiment KIT_Seq_01: Average common point feature sets of the minimum entropy subset, the Lyapunov exponent subset, and the minimum variance subset in two successive frames for STA1 and STA2.

	Entropy	Lyapunov	Variance
STA1	49.05	19.25	49.91
STA2	52.66	22.93	49.48

Table 6.9: Experiment KIT_Seq_01: Mean L2 norm pose distance between non-robust/robust STA1 and STA2 evaluation the stationary background data and the reference data in m.

	Reference Data	Descriptor Type			
			Entropy	Lyapunov	Variance
Non-robust background	$7.29*10^{10}$	STA1	9.16	2.98	45.79
		STA2	14.37	5.90	12.86
Robust background	0	STA1	0.35	0.72	2.36
		STA2	2.06	2.58	1.32

Table 6.10: Experiment KIT_Seq_01: Variance of distances between non-robust/robust STA1 and STA2 evaluation the stationary background data and the reference data in m^2.

	Reference Data	Descriptor Type			
			Entropy	Lyapunov	Variance
Non-robust background	225.13	STA1	9000.91	275.60	$1.26*10^{6}$
		STA2	2599.61	4691.93	3969.24
Robust background	0	STA1	7.31	111.62	778.48
		STA2	340.62	509.38	132.35

estimation result (see fig. A.1 and A.1). The lower the variance in table 6.10, the better the pose estimation. One can also see, that the minimum variance subset of STA1 performs worse than the minimum variance subset of STA2. This is valid for both, the mean L2 norm in table 6.9 and the variance in table 6.10.

The non-robust computation of the observer poses with the reference data fails. Compared to the reference data, the non-robust computation with any subset provides significantly better results. Still these results are imprecise (high variance, see table 6.10). For both Lyapunov exponent subsets, but especially for the STA1 Lyapunov exponent subset, the average common point feature set is very low. One can observe, that in case of non-robust observer pose estimation the Lyapunov exponent

subsets yields better results than the minimum entropy subset, which gives the best results in case of robust observer pose estimation.

In case of robust pose estimation, the minimum entropy subsets outperform the Lyapunov exponent subsets due to the higher common point feature set within two subsequent frames. The robust pose estimation with any subset provides a very low mean difference distance to the reference data. The best results in terms of mean difference distance and variance of the distance are provided by the STA1 minimum entropy, the STA1 Lyapunov exponent, and the STA2 minimum variance subset.

Figures A.1 to A.3 illustrate the robustly reconstructed observer pose with the minimum entropy, the Lyapunov exponent, and the minimum variance.

6.2.2.2 All MSaM Inliers Evaluation

Table 6.11 contains the average common point feature set of the minimum entropy subset, the Lyapunov exponent subset, and the minimum variance subset in two subsequent frames for STA1 and STA2. Table 6.12 contains the mean L2 norm pose distances in m of all MSaM inliers and the three subsets. Table 6.13 shows the variance of the distances. Figures A.4 to A.6 illustrate the robustly computed observer pose by the minimum entropy, the Lyapunov exponent, and the minimum variance.

Table 6.11: Experiment KIT_Seq_01: Average common point feature sets of the minimum entropy subset, the Lyapunov exponent subset, and the minimum variance subset in two successive frames for STA1 and STA2.

	Entropy	Lyapunov	Variance
STA1	45.41	16.98	42.73
STA2	51.87	20.97	48.45

In contrast to experiment VMG_Lab_01, which had a controlled setup, this experiment highlights discrepancies between the observer pose estimation on the reference data and on all MSaM inliers. I.e., the non-robust computation of the observer motion with subsets of all MSaM inliers fails. Only the STA1 minimum entropy subset provides a usable observer pose estimation in terms of the observer motion shape. However, the variance of the distance (see table 6.13) indicates irregularities

Table 6.12: Experiment KIT_Seq_01: Mean L2 norm pose distance between non-robust/robust STA1 and STA2 evaluation of all available data and the reference data in m.

	All MSaM Inliers	Descriptor Type			
			Entropy	Lyapunov	Variance
Non-robust background	$2.70*10^{11}$	STA1	3.04	7.71	114.77
		STA2	3.33	4.57	1.88
Robust background	2.18	STA1	1.22	4.11	6.98
		STA2	4.41	5.23	4.72

Table 6.13: Experiment KIT_Seq_01: Variance of distances between non-robust/robust STA1 and STA2 evaluation of all available data and the reference data in m^2.

	All MSaM Inliers	Descriptor Type			
			Entropy	Lyapunov	Variance
Non-robust background	$1.28*10^{24}$	STA1	850.67	5355.93	$5.32*10^{6}$
		STA2	1886.03	1412.04	182.15
Robust background	527.48	STA1	299.46	1512.08	4665.73
		STA2	5061.32	2962.07	2008.61

in the motion. Robustly computed, the STA1 minimum entropy subset performs best again (see fig. A.4, left). It is the only subset, which provides useful data on the observer's pose. However, the variance of the STA1 minimum entropy (see table 6.13) shows some motion irregularities.

Obviously, more outliers than in experiment VMG_Lab_01 remained in the set of MSaM inliers. With the robust observer pose estimation, the subsets allow to recover a similar observer motion shape. However, the variances of the distances (see table 6.13) show a major deviation from the reference data.

6.2.3 Experiment KIT_Seq_02

This experiment uses the dataset KIT_Seq_03 (see section 2.8). The sequence consists of 85 frames. As in experiment KIT_Seq_01, a certain minimum appearance change information is required to have reliable information at hand. I.e., while appearance change information collection starts at frame 1, the STA histogram statistics evaluation starts at frame 5. Again, the STA subset size is 100 per frame, i.e. the 100

best point features according to a specific STA histogram evaluation method are chosen per frame.

The reference observer motion is again estimated by using point features located on the stationary background, i.e. the reference data. Figure A.7 illustrates the estimated observer motion retrieved by the reference data (left) and all MSaM inliers (right). Figure A.8 shows the X/Y view (top) and X/Z view (bottom) of the visual odometry observer poses (green) as well as the metric GPS information (blue) and the aligned and rotated GPS information (black).

In contrast to experiment KIT_Seq_02 the GPS data seem to be more accurate in this sequence. Table 6.14 contains the mean distance between the reference observer pose and the GPS data as well as the variance of the difference between the observer pose reference data and the GPS data. The variance value close to zero indicates a very similar shape of both GPS data and visual odometry.

Table 6.14: Experiment KIT_Seq_02: Comparison of GPS coordinates with the stationary background reference data.

Mean L2 norm pose distance in m	Variance of distances in m^2
0.18	0.01

6.2.3.1 Reference Data Evaluation

Table 6.15 contains the average common point feature set of the minimum entropy subset, the Lyapunov exponent subset, and the minimum variance subset in two subsequent frames for STA1 and STA2. Table 6.16 contains the mean L2 norm pose distances in m, table 6.17 shows the variance of the distances. While the non-robust pose estimation with all point features of the reference data fails, all subsets allow a reliable non-robust observer pose estimation. With the robust pose estimation, the observer pose estimation is possible with the reference data and all subsets.

Figures A.9 to A.11 illustrate the robustly computed observer pose by the minimum entropy, the Lyapunov exponent, and the minimum variance.

Table 6.15: Experiment KIT_Seq_02: Average common point feature sets of the minimum entropy subset, the Lyapunov exponent subset, and the minimum variance subset in two successive frames for STA1 and STA2.

	Entropy	Lyapunov	Variance
STA1	74.94	14.72	62.91
STA2	76.00	36.14	70.78

Table 6.16: Experiment KIT_Seq_02: Mean L2 norm pose distance between non-robust/robust STA1 and STA2 evaluation the stationary background data and the reference data in m.

	Reference Data	Descriptor Type			
			Entropy	Lyapunov	Variance
Non-robust background	$3.76*10^9$	STA1	0.18	0.24	0.19
		STA2	0.16	0.28	0.08
Robust background	0	STA1	0.04	0.10	0.08
		STA2	0.03	0.05	0.06

Table 6.17: Experiment KIT_Seq_02: Variance of distances between non-robust/robust STA1 and STA2 evaluation the stationary background data and the reference data in m^2.

	Reference Data	Descriptor Type			
			Entropy	Lyapunov	Variance
Non-robust background	$5.74*10^{22}$	STA1	29.91	35.37	36.58
		STA2	10.54	35.33	1.65
Robust background	0	STA1	2.00	25.39	7.90
		STA2	0.49	1.45	3.09

6.2.3.2 All MSaM Inliers Evaluation

Table 6.18 contains the average common point feature set of the minimum entropy subset, the Lyapunov exponent subset, and the minimum variance subset in two subsequent frames for STA1 and STA2. Figures A.12 to A.14 illustrate the robustly computed observer pose by the minimum entropy, the Lyapunov exponent, and the minimum variance.

Table 6.19 contains the mean L2 norm pose distances in m, table 6.20 shows the variance of the distances. The non-robust estimation with the STA2 minimum variance subset performs best. The non-robust pose estimation with all MSaM inliers as well as the STA1 minimum entropy subset fails. The non-robust estimation with

the STA1 Lyapunov exponent subset and the STA1 minimum variance subset deteriorates. In case of a robust pose estimation, the STA2 minimum variance subsets performs best, followed by the STA2 and the STA1 entropy subsets. The robust pose estimation with the STA1 Lyapunov exponent subset fails. The robust estimation with the STA2 Lyapunov exponent subset and the STA1 minimum variance subset deteriorates.

Figures A.12 to A.14 illustrate the robustly computed observer pose by the minimum entropy, the Lyapunov exponent, and the minimum variance.

Table 6.18: Experiment KIT_Seq_02: Average common point feature sets of the minimum entropy subset, the Lyapunov exponent subset, and the minimum variance subset in two successive frames for STA1 and STA2.

	Entropy	Lyapunov	Variance
STA1	74.46	12.91	58.59
STA2	75.14	31.24	69.90

Table 6.19: Experiment KIT_Seq_02: Mean L2 norm pose distance between non-robust/robust STA1 and STA2 evaluation of all available data and the reference data in m.

	All MSaM Inliers	Descriptor Type			
			Entropy	Lyapunov	Variance
Non-robust background	$3.80*10^{9}$	STA1	$3.80*10^{9}$	0.62	0.80
		STA2	0.23	0.59	0.16
Robust background	0.02	STA1	0.08	6.80	0.15
		STA2	0.08	0.10	0.04

Table 6.20: Experiment KIT_Seq_02: Variance of distances between non-robust/robust STA1 and STA2 evaluation of all available data and the reference data in m^2.

	All MSaM Inliers	Descriptor Type			
			Entropy	Lyapunov	Variance
Non-robust background	$5.84*10^{22}$	STA1	$5.84*10^{22}$	1286.09	773.76
		STA2	46.66	347.35	18.87
Robust background	0.02	STA1	7.79	$1.13*10^{5}$	2448.05
		STA2	5.03	1621.97	1.47

6.2.4 Experiment KIT_Seq_03

This experiment uses the dataset KIT_Seq_03 (see section 2.9). It consists of 99 frames. As with all experiments in this section, we start the STA histogram statistics evaluation at frame 5, having reliable appearance change information of at least 5 frames. The STA subset size is 100 per frame, i.e. the 100 best point features according to a specific STA histogram evaluation method are chosen per frame.

The reference observer motion is again estimated by using point features located on the stationary background, i.e. the manually generated reference data. Figure A.15 illustrates the estimated observer motion of the reference data (i.e. the reference observer motion, left) and all MSaM inliers (right).

Figure A.16 shows the X/Y view (top) and X/Z view (bottom) of the visual odometry observer poses (green) as well as the metric GPS information (blue) and the aligned and rotated GPS information (black).

As with experiment KIT_Seq_02, the GPS data of this sequence seem to be accurate. Table 6.21 contains the mean distance between the reference observer pose and the GPS data as well as the variance of the difference between the observer pose reference data and the GPS data. The variance value indicates a difference between the observer motion shape retrieved from the GPS data and visual odometry. Indeed, the GPS data shows a similar shape compared to the visual odometry observer motion. However, the diverging motion lengths in direction of the z-axis cause the variance value of $14.65m^2$ (see table 6.21). Due to the low common point features within two subsequent frames, in this experiment the reference observer pose estimation seem to deteriorate through the sequence.

Table 6.21: Experiment KIT_Seq_03: Comparison of GPS coordinates with the stationary background reference data.

Mean L2 norm pose distance in m	Variance of distances in m^2
4.64	14.65

6.2.4.1 Reference Data Evaluation

Table 6.22 contains the average common point feature set of the minimum entropy subset, the Lyapunov exponent subset, and the minimum variance subset in two subsequent frames for STA1 and STA2. Table 6.23 contains the mean L2 norm pose distances in m, table 6.24 shows the variance of the distances. Any non-robust observer pose estimation fails. Additionally, the robust poses estimation deteriorates for all subsets and the reference data. The best results are achieved with the robust minimum entropy subsets, but the retrieved observer motion shape diverges from the reference observer motion. However, all subsets allow a more accurate non-robust pose estimation than achieved with the whole reference data as input for the non-robust pose estimation. It seems, that the data collected in the scene is worse than in experiment 3. Especially the right turn of the moving observer around frame 46 seem to make pose estimation hard due to the fast motion and the low frame rate (only 10 frames per second).

Figures A.17 to A.19 illustrate the robustly computed observer pose by the minimum entropy, the Lyapunov exponent, and the minimum variance.

Table 6.22: Experiment KIT_Seq_02: Average common point feature sets of the minimum entropy subset, the Lyapunov exponent subset, and the minimum variance subset in two successive frames for STA1 and STA2.

	Entropy	Lyapunov	Variance
STA1	42.66	19.85	37.98.59
STA2	44.83	23.34	39.40

Table 6.23: Experiment KIT_Seq_03: Mean L2 norm pose distance between non-robust/robust STA1 and STA2 evaluation the stationary background data and the reference data in m.

	Reference Data		Descriptor Type		
			Entropy	Lyapunov	Variance
Non-robust background	$1.28*10^{13}$	STA1	151.27	57.80	159.10
		STA2	145.66	719.27	63.11
Robust background	0	STA1	7.37	9.07	11.05
		STA2	6.21	24.62	8.80

Table 6.24: Experiment KIT_Seq_03: Variance of distances between non-robust/robust STA1 and STA2 evaluation the stationary background data and the reference data in m^2.

	Reference Data		Descriptor Type		
			Entropy	Lyapunov	Variance
Non-robust background	$1.01*10^{29}$	STA1	$5.45*10^{6}$	$3.35*10^{5}$	$1.63*10^{7}$
		STA2	$6.36*10^{6}$	$1.66*10^{9}$	$2.63*10^{6}$
Robust background	0	STA1	18606.37	11490.28	200001.03
		STA2	29695.52	$1.07*10^{6}$	47727.22

6.2.4.2 All MSaM Inliers Evaluation

Table 6.25 contains the average common point feature set of the minimum entropy subset, the Lyapunov exponent subset, and the minimum variance subset in two subsequent frames for STA1 and STA2. Table 6.26 contains the mean L2 norm pose distances in m , table 6.27 shows the variance of the distances. All subsets fail to reconstruct the observer pose estimation. But the non-robust subsets provide significantly better results than using all MSaM inliers or even the reference data as input for the non-robust pose estimation. As with the reference data, the STA1 minimum entropy subset performs best and achieves similar results even with all MSaM inliers as input. Thus, with the STA1 minimum entropy subset one achieves the most similar motion shape compared to the reference observer pose. However, one can clearly see, that on two positions, the estimated pose jumps considerably (see fig. A.20, left).

Figures A.20 to A.22 illustrate the robustly computed observer pose by the minimum entropy, the Lyapunov exponent, and the minimum variance.

Table 6.25: Experiment KIT_Seq_02: Average common point feature sets of the minimum entropy subset, the Lyapunov exponent subset, and the minimum variance subset in two successive frames for STA1 and STA2.

	Entropy	Lyapunov	Variance
STA1	43.27	18.09	34.67
STA2	42.25	21.96	39.63

Table 6.26: Experiment KIT_Seq_03: Mean L2 norm pose distance between non-robust/robust STA1 and STA2 evaluation of all available data and the reference data in m.

	All MSaM Inliers		Descriptor Type		
			Entropy	Lyapunov	Variance
Non-robust background	$2.16*10^{12}$	STA1	148.07	41.78	135.75
		STA2	91.55	53.30	90.18
Robust background	4.58	STA1	6.56	14.09	17.33
		STA2	12.02	31.23	6.72

Table 6.27: Experiment KIT_Seq_03: Variance of distances between non-robust/robust STA1 and STA2 evaluation of all available data and the reference data in m^2.

	All MSaM Inliers		Descriptor Type		
			Entropy	Lyapunov	Variance
Non-robust background	$3.04*10^{27}$	STA1	$5.95*10^{6}$	$1.62*10^{6}$	$1.02*10^{7}$
		STA2	$2.69*10^{6}$	$7.29*10^{7}$	$2.43*10^{6}$
Robust background	6617.99	STA1	9057.05	69791.42	$1.54*10^{5}$
		STA2	$1.32*10^{5}$	$9.74*10^{5}$	13244.73

6.3 Summary

In this chapter we evaluated the outcome of the observer pose estimation in several ways. The stationary background data were manually selected out of the MSaM inlier data and appearance-change-based point feature subsets were generated, initially introduced in chapter 5. We showed, that taking out subsets of both, the reference data and all MSaM inlier data (which may contain false inliers), observer motion estimation is possible in many cases.

Two different estimation procedures were introduced: a robust motion estimation and a non-robust motion estimation. The robust observer pose estimation uses RANSAC algorithms for model fitting and global Bundle Adjustment for optimizing the result in a post-processing step. The non-robust motion estimation, solely uses Bundle Adjustment. The non-robust observer pose estimation shows, that any point feature subset generated by appearance change analysis gives a more precise result than using either the whole reference data or all the MSaM inliers as input. The non-robust observer pose estimation failed most of of the time with both the reference

data and all MSaM inliers. For the robust observer pose estimation, our results showed a wider range of possibilities. For two out of four experiments very similar results were gained. One experiment gave similar results, however depending on the subset, the result deteriorate from the reference data. Finally, one experiment allowed us to robustly estimate the observer poses with the whole reference data, but failed to give similar results with either one of the subsets or all MSaM inliers.

6.4 Conclusion

Focusing on the subsets, the robust STA1 minimum entropy subset provided the most reliable data for observer pose estimation. As the Lyapunov exponent uses the entropy as input, we expected the STA1 Lyapunov exponent subset to achieve similar results as the STA2 minimum entropy subset. We assumed this, as the STA2 descriptor describes the distribution of each bin of the STA1 descriptor, i.e. it models an entropy change over time. The Lyapunov exponent models the change of information per definition (first order derivative). The data evaluation with the Lyapunov exponent did not meet our expectations, as the STA1 and STA2 Lyapunov exponent subsets performed worse than the entropy subsets.

As shown in the experiments, the Lyapunov exponent subset has always fewer common inter-frame point features than the minimum entropy and minimum variance subset. I.e., applying the Lyapunov exponent on all point features and retrieving the best n point features, the point features may vary significantly in two successive frames. On the other hand, performing a non-robust pose estimation, the STA1 Lyapunov exponent provides similar results than the minimum entropy (except in experiment 6.2.2, all MSaM inliers evaluation).

The bad performance of the robust pose estimation with the Lyapunov exponent subsets is due to the model selection algorithm (RANSAC) performed by the robust pose estimation approach. While the non-robust pose estimation approach uses all point features provided by a subset for pose estimation, the model selection algorithm selects only those point features, which fit to the selected model. I.e., too little information is provided for the model generation by the Lyapunov exponent

subset.

To address the Lyapunov exponent issue, the information gathering of the appearance change (i.e. the subset generation) has to be modified. Instead of analyzing the point feature trajectories online, one can use the point features' whole trajectories for subset generation. I.e., the subset generation can be performed in a post processing step, having all the appearance change information at hand. By that, the low common point feature set in two successive frames can be increased for the Lyapunov exponent subset. Within the scope of this thesis, there have only been investigations on the online appearance change analysis.

With the proposed online appearance change handling of point features, we can state:

- The STA minimum entropy subsets perform best for tracking good features to track.
- The results of the STA2 minimum variance subsets are similar to the STA minimum entropy subsets.
- The STA1 minimum variance subset is not suited for identifying good features to track.
- The Lyapunov exponent subsets are inapplicable for pose estimation (due to the online appearance change handling).
- But: any subset provides significantly better results than using all MSaM inliers or even the reference data as input for the non-robust pose estimation.

7 Discussion

7.1 Summary

In this thesis we presented a practical MSaM system with various extensions. First we explained, how to extend a SaM/SLAM to MSaM. We showed, that this extension can be done by a geometry-based approach. For certain tasks (e.g. re-identifying lost point features), we introduced a descriptor-based solution. By extending SaM/SLAM to MSaM, our system is able to identify independent foreground motion. Second, we extended our MSaM to classify the foreground motion. We modularly added a 2D detector and tracker for persons. Together with a feedback control system, the 2D person detector and tracker module was able to communicate with MSaM. This extension enriches our MSaM in two ways: (i) certain classes of moving objects can be identified and classified and (ii) in case of a sub-part detection of a moving object, the whole object can be identified. Third, we analyzed the stationary background information. By using a descriptor relying on appearance change information, we evaluated this information by certain histogram statistics evaluation methods. We gained various subsets of point features out of the all point features.

We used these subsets for observer pose estimation and compared the outcome to the reference data, i.e. all point features located on the stationary background. We found out, that the observer pose estimation works with few stable, stationary point features too, provided that the visual odometry data does not contain major error sources (e.g. motion blur, low frame rate). For each implementation or extension we provided experiments, where we tested the particular development.

7.2 Limitations

The introduced MSaM relies on point features. I.e., if point features cannot be established in images due to several reasons (bad lighting conditions, uniformly colored environment, etc.), the estimation of the observer pose and the scene reconstruction will fail. A map generation introduced by Klein and Murray [KM08] would lead to satisfying results in case of a few point features only. However, this approach is limited to stationary scenes only. We are analyzing the 3D information available from the scene. For plenty of foreground motion types our approach achieves good results. We are not explicitly handling actions like merging and splitting of multiple foreground motion objects. I.e, we are not able to provide a motion model for such actions. However, identifying the splitting and merging is possible due to the 3D information, we just cannot predict it. Another point to be discussed here, is not a limitation of the system itself, but more a limitation of the thesis. Most of the image sequences processed are stereo. I.e. one gains access to the 3D information by epipolar geometry. It would be nice to test the system on monocular sequences too (in fact, some monocular sequences were processed). This does not necessarily mean to change the deployed MSaM. It would be sufficient to modify or replace the SaM algorithm on which the MSaM is built. Indeed, the routine for point feature correspondences between two images has to be rewritten.

7.3 Impact on the State-of-the-Art

Our MSaM benefits from processing full 3D information. In contrast, Ozden et al. [OSG10] process 2D information for foreground motion clustering only. They claim, that 3D information processing for the model hypothesis generation would not work in real time. In fact, their approach contains a comprehensive foreground motion model predictor. With that, they are able to predict actions like merging and splitting. Having full 3D information available, we cannot predict actions, but are able to handle them.

By identifying independent foreground motion, one could think of replacing the training phase of a semi-supervised approach by observing and classifying motion by MSaM. MSaM identifies the independently moving objects. Then, appearance information could be collected and classified automatically, e.g. with random ferns [OCLF10].

The task of identifying good features to track in my opinion is a very essential and important one. As Shi and Tomasi [ST94] stated, appearance change information of point features can provide useful hints for recognizing good features to track. As shown in this thesis, it is possible to retrieve certain point features due to their appearance change information and estimate the observer pose with them. However, one has to keep in mind that the appearance change information is highly related to the quality of an image sequence (e.g. frame rate, resolution, etc.).

A

Appendix

To provide a clear structure for the reader in chapter 6, figures of coherent data were moved to the appendix. While the first experiment contains all figures, the remaining three experiments refer to the respective figures in the appendix. The figures show the reconstructed scene and the estimated observer motion, either estimated with all available point features or with certain subsets retrieved from three different evaluation methods. Other figures compare available GPS information with the estimated observer motion.

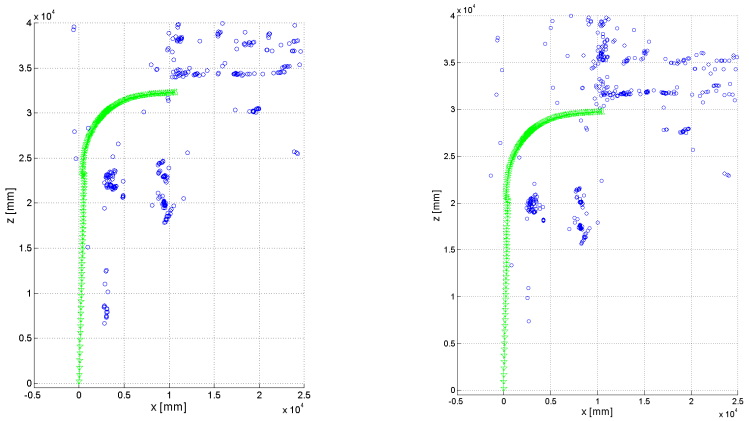


Figure A.1: Experiment KIT_Seq_01 (reference data): Robustly estimated pose with the STA1 minimum entropy (left) and the STA2 minimum entropy (right) subset generated from the reference data. Estimated camera poses (green triangles), estimated structure (blue circles). Of all subsets, the estimation with the STA1 minimum entropy (left) performs best.

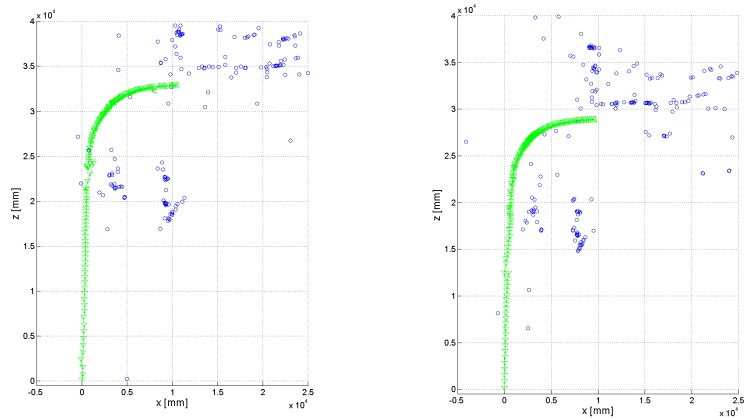


Figure A.2: Experiment KIT_Seq_01 (reference data): Robustly estimated pose with the STA1 Lyapunov exponent (left) and the STA2 Lyapunov exponent (right) subset generated from the reference data. Estimated camera poses (green triangles), estimated structure (blue circles). Only the STA1 minimum variance has worse results than the STA2 Lyapunov exponent (right).

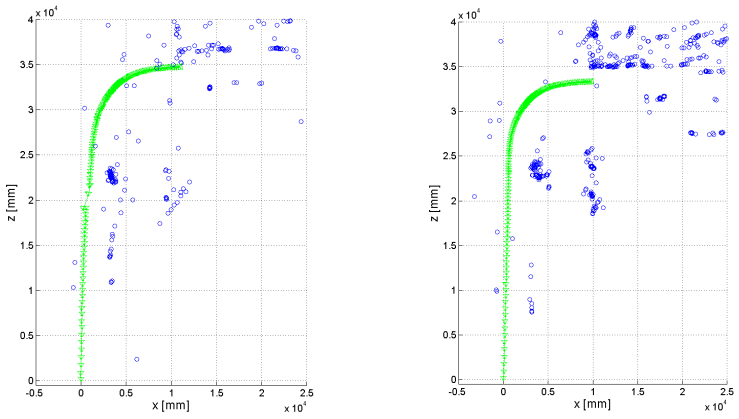


Figure A.3: Experiment KIT_Seq_01 (reference data): Robustly estimated pose with the STA1 minimum variance (left) and STA2 minimum variance (right) subset generated from the reference data. Estimated camera poses (green triangles), estimated structure (blue circles). Of all subsets, the estimation with the STA1 minimum variance (left) performs worst. The STA2 minimum variance (right) provides the second best results after the STA1 minimum entropy.

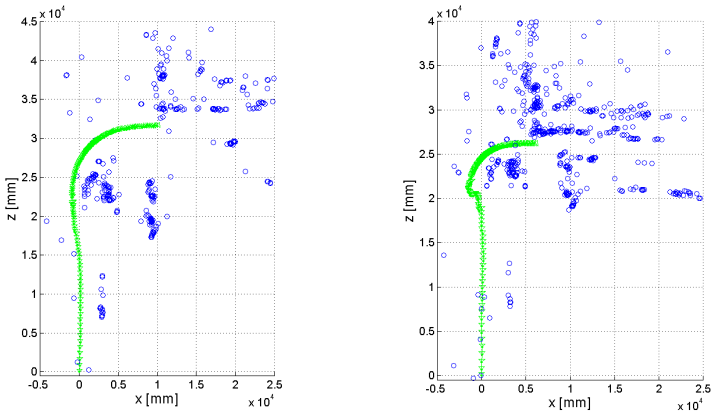


Figure A.4: Experiment KIT_Seq_01 (all MSaM inliers): Robustly estimated pose with the STA1 minimum entropy (left) and the STA2 minimum entropy (right) subset generated from all MSaM inliers. Estimated camera poses (green triangles), estimated structure (blue circles). Only with the STA1 minimum entropy (left) the pose estimation provides an acceptable - but still deteriorated - result.

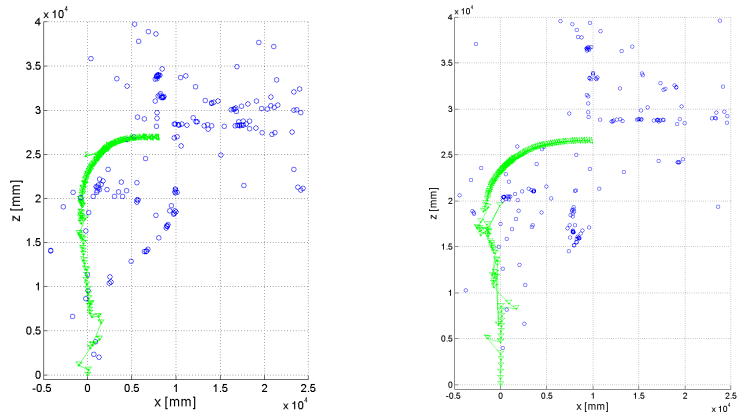


Figure A.5: Experiment KIT_Seq_01 (all MSaM inliers): Robustly estimated pose with the STA1 Lyapunov exponent (left) and the STA2 Lyapunov exponent (right) subset generated from all MSaM inliers. Estimated camera poses (green triangles), estimated structure (blue circles). With both subsets, the pose estimation fails.

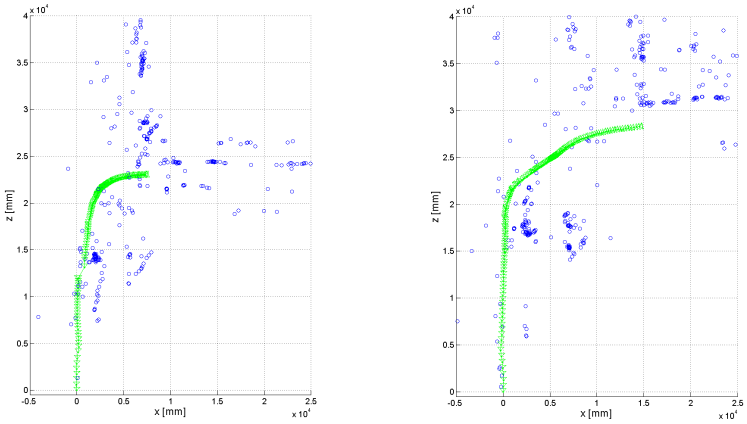


Figure A.6: Experiment KIT_Seq_01 (all MSaM inliers): Robustly estimated pose with the STA1 minimum variance (left) and the STA2 minimum variance (right) subset generated from all MSaM inliers. Estimated camera poses (green triangles), estimated structure (blue circles). With both subsets, the pose estimation fails.

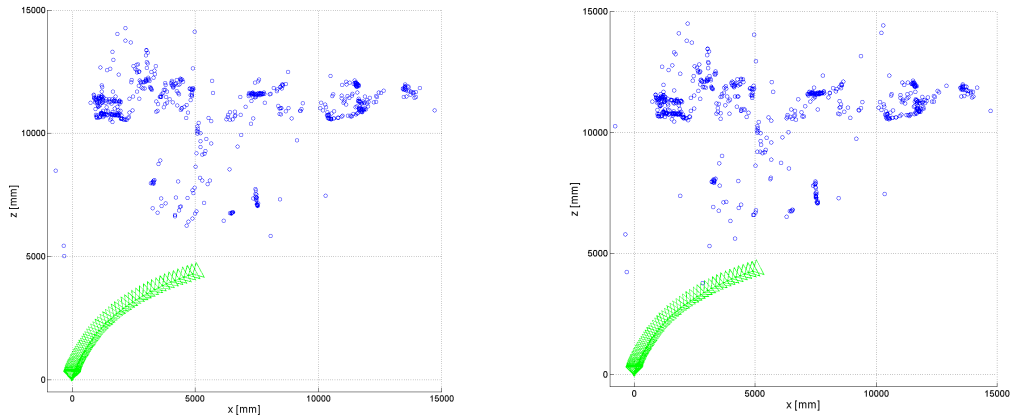


Figure A.7: Experiment KIT_Seq_02: Robust pose estimation with the reference data only (left) and with all MSaM inliers (right). Estimated camera poses (green triangles), estimated structure (blue circles).

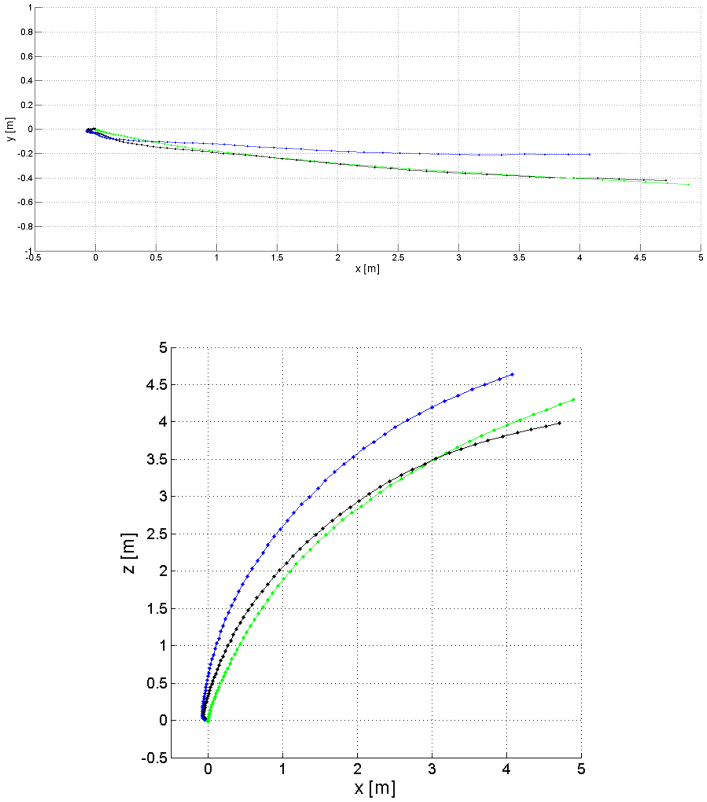


Figure A.8: Experiment KIT_Seq_02: GPS reference data (blue), GPS reference data aligned with visual odometry coordinate system (black), stationary background reference data (green). X/Y view (top), X/Z view (bottom).

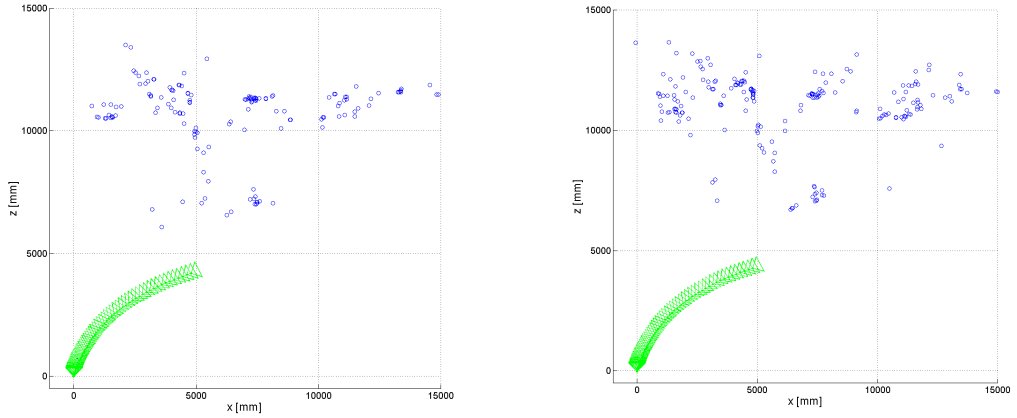


Figure A.9: Experiment KIT_Seq_02 (reference data): Robustly estimated pose with the STA1 minimum entropy (left) and the STA2 minimum entropy (right) subset generated from the reference data. Estimated camera poses (green triangles), estimated structure (blue circles). The STA2 minimum entropy (right) achieves the best result, followed by the STA1 minimum entropy (left), which performs similar to the STA2 Lyapunov exponent (see fig. A.10 right).

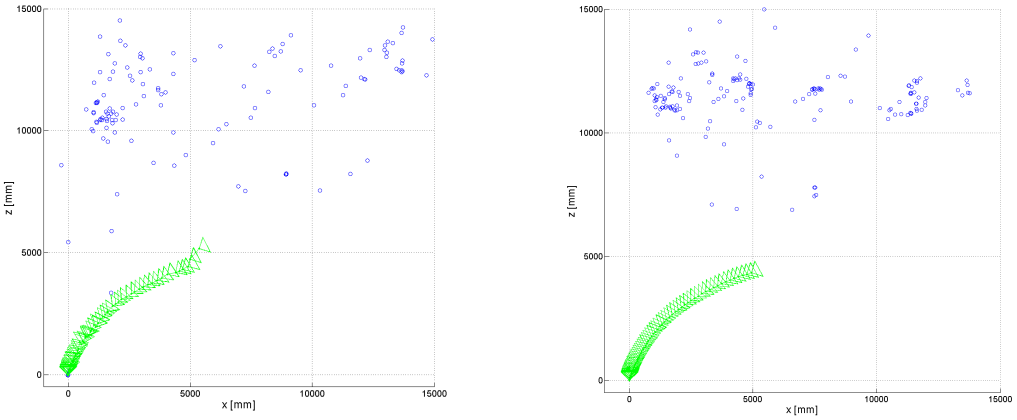


Figure A.10: Experiment KIT_Seq_02 (reference data): Robustly estimated pose with the STA1 Lyapunov exponent (left) and the STA2 Lyapunov exponent (right) subset generated from the reference data. Estimated camera poses (green triangles), estimated structure (blue circles). The STA1 Lyapunov exponent provides a slightly deteriorated result, while the STA2 Lyapunov exponent performs similar to the STA1 minimum entropy (see fig. A.9 left).

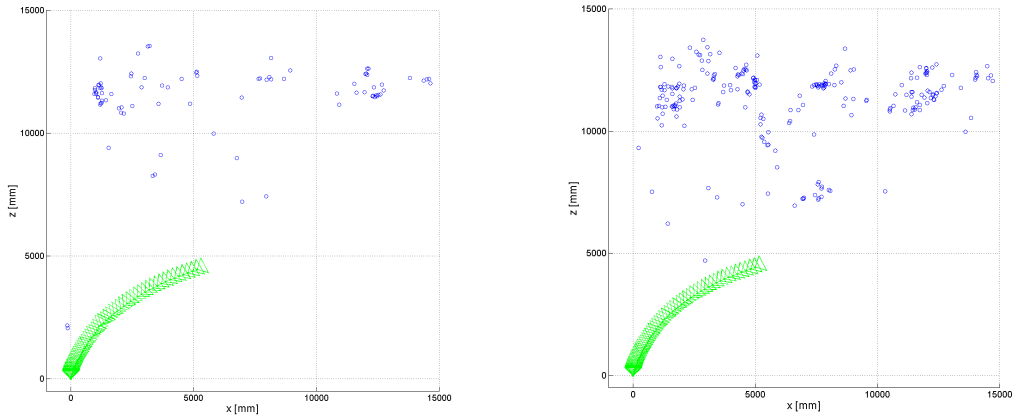


Figure A.11: Experiment KIT_Seq_02 (reference data): Robustly estimated pose with the STA1 minimum variance (left) and the STA2 minimum variance (right) subset generated from the reference data. Estimated camera poses (green triangles), estimated structure (blue circles). Even if both minimum variance subsets provide useable results, the results are worse than those of the minimum entropy subsets (see fig. A.9) and the Lyapunov exponent subsets (see fig. A.10).

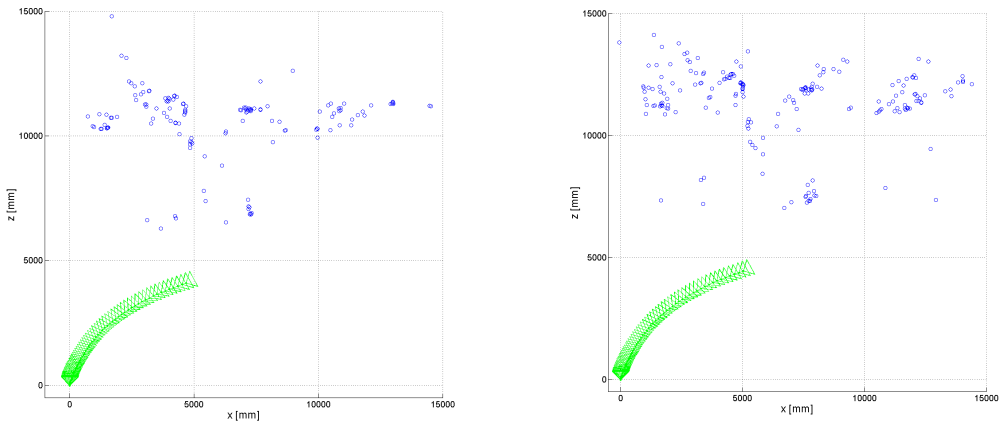


Figure A.12: Experiment KIT_Seq_02 (all MSaM inliers): Robustly estimated pose with the STA1 minimum entropy (left) and the STA2 minimum entropy (right) subset generated from all MSaM inliers. Estimated camera poses (green triangles), estimated structure (blue circles). Robust pose estimation with the STA2 minimum entropy (right) is more accurate as with the STA1 minimum entropy (left), which provides a slightly deteriorated result. The STA2 minimum variance subset (see fig. A.14 right) provides the most accurate pose estimation result.

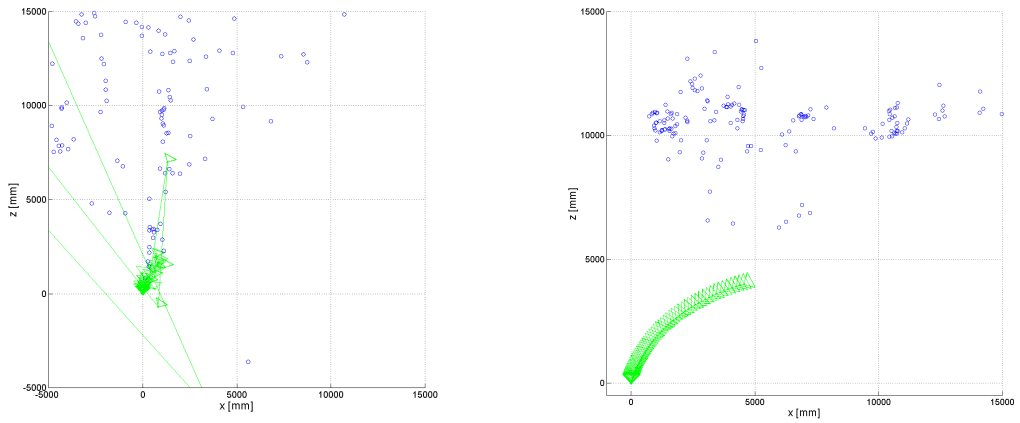


Figure A.13: Experiment KIT_Seq_02 (all MSaM inliers): Robustly estimated pose with the STA1 Lyapunov exponent (left) and the STA2 Lyapunov exponent (right) subset generated from all MSaM inliers. Estimated camera poses (green triangles), estimated structure (blue circles). Robust pose estimation with the STA1 Lyapunov exponent (left) fails. The result of the STA2 Lyapunov exponent subset (right) deteriorates slightly.

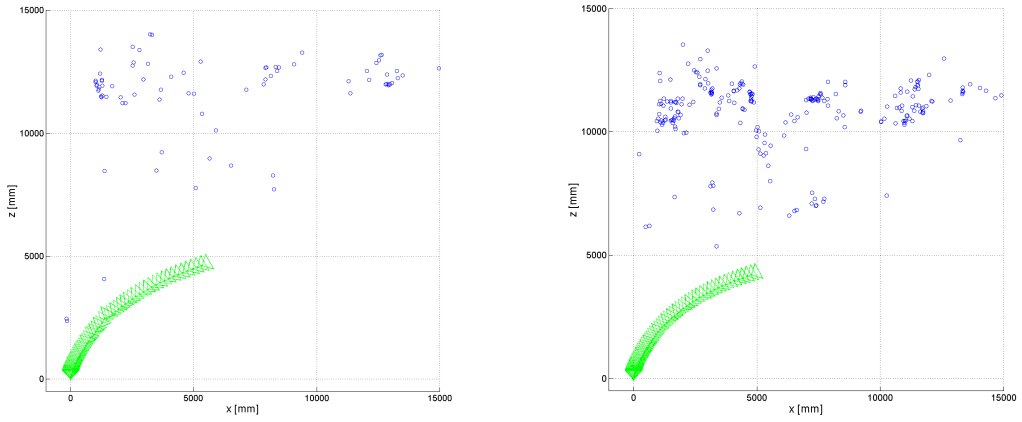


Figure A.14: Experiment KIT_Seq_02 (all MSaM inliers): Robustly estimated pose with the STA1 minimum variance (left) and the STA2 minimum variance (right) subset generated from all MSaM inliers. Estimated camera poses (green triangles), estimated structure (blue circles). The STA2 minimum variance subset (right) provides the best result. The STA1 minimum variance subset (left) performs similar to the STA2 Lyapunov exponent subset (see fig. A.13 right); both results deteriorate slightly.

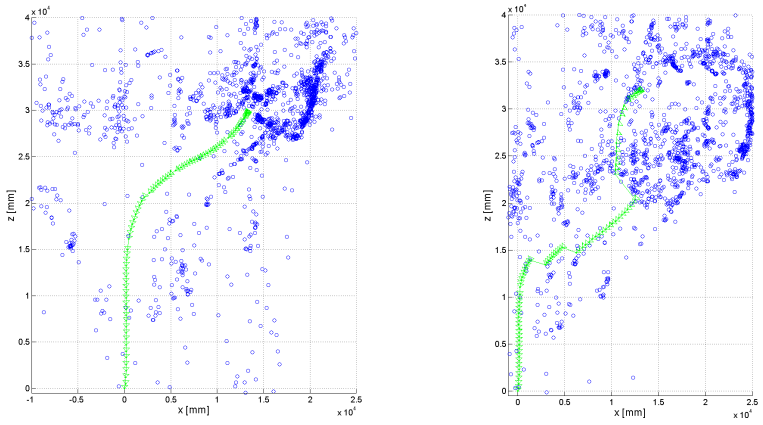


Figure A.15: Experiment KIT_Seq_03: Robust pose estimation with the reference data only (left) and with all MSaM inliers (right). Estimated camera poses (green triangles), estimated structure (blue circles). The pose estimation with all MSaM inliers (right) fails. The result of the pose estimation with the reference data (left) deteriorates.

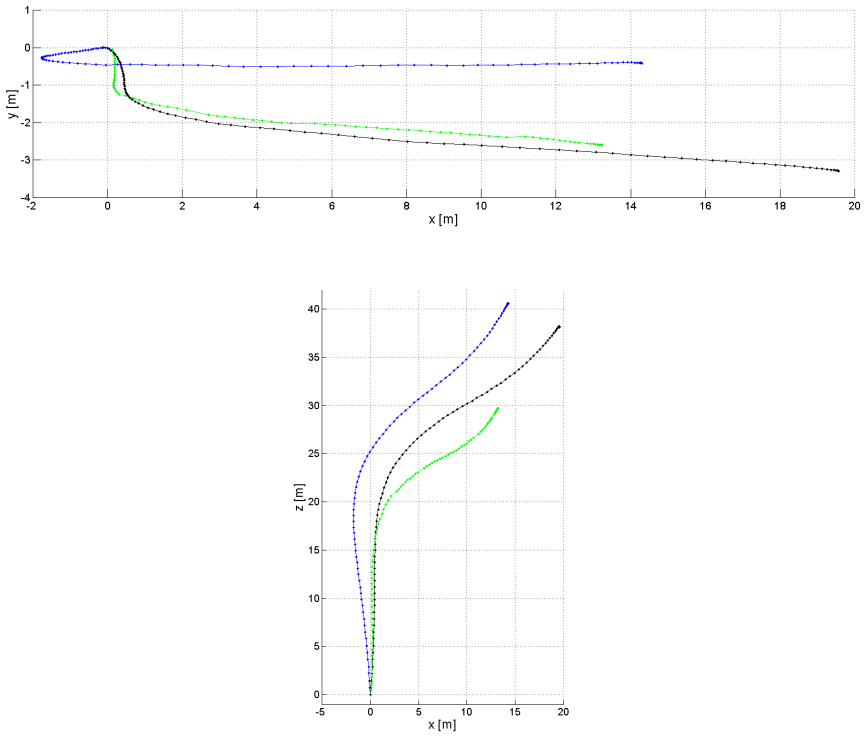


Figure A.16: Experiment KIT_Seq_03: GPS reference data (blue), GPS reference data aligned with visual odometry coordinate system (black), stationary background reference data (green). X/Y view (left), X/Z view (right). The result of the pose estimation with reference data (green) deteriorates; it has a similar shape but a different length on the z-axis).

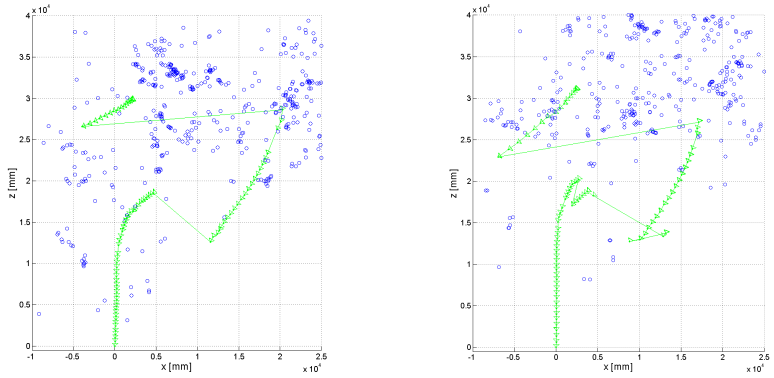


Figure A.17: Experiment KIT_Seq_03 (reference data): Robustly estimated pose with the STA1 minimum entropy (left) and the STA2 minimum entropy (right) subset generated from the reference data. Estimated camera poses (green triangles), estimated structure (blue circles). The pose estimation fails with both subsets. However, the STA1 minimum entropy subset (left) provides the best result compared to all other subsets. One can clearly see the two positions, where the estimated pose jumps considerably.

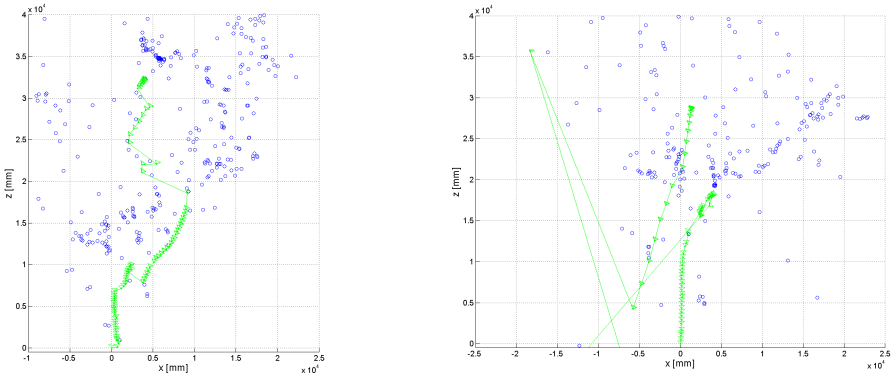


Figure A.18: Experiment KIT_Seq_03 (reference data): Robustly estimated pose with the STA1 Lyapunov exponent (left) and the STA2 Lyapunov exponent (right) subset generated from the reference data. Estimated camera poses (green triangles), estimated structure (blue circles). The pose estimation fails with both subsets.

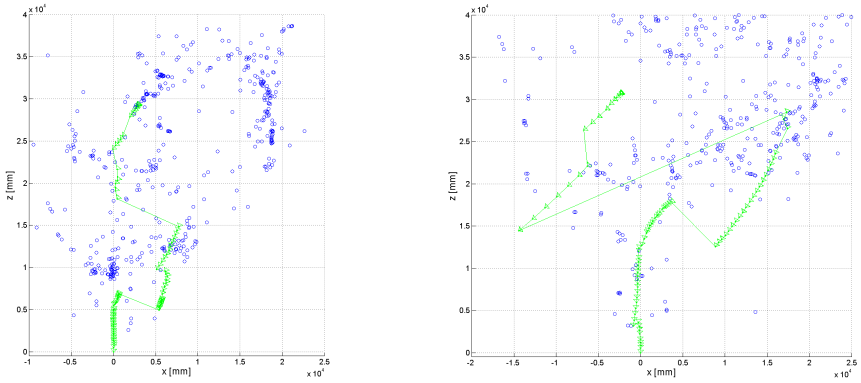


Figure A.19: Experiment KIT_Seq_03 (reference data): Robustly estimated pose with the STA1 minimum variance (left) and the STA2 minimum variance (right) subset generated from the reference data. Estimated camera poses (green triangles), estimated structure (blue circles). The pose estimation fails with both subsets.

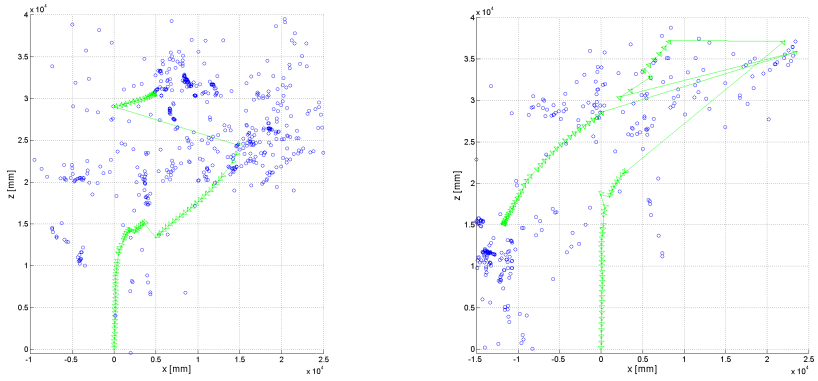


Figure A.20: Experiment KIT_Seq_03 (all MSaM inliers): Robustly estimated pose with the STA1 minimum entropy (left) and the STA2 minimum entropy (right) subset generated from all MSaM inliers. Estimated camera poses (green triangles), estimated structure (blue circles). The pose estimation with both subsets fail. However, The STA1 minimum entropy subset (left) provides together with the STA2 minimum variance subset (see fig. A.22 right) similar results than the STA1 minimum entropy subset (see fig. A.17 left) on the reference data.

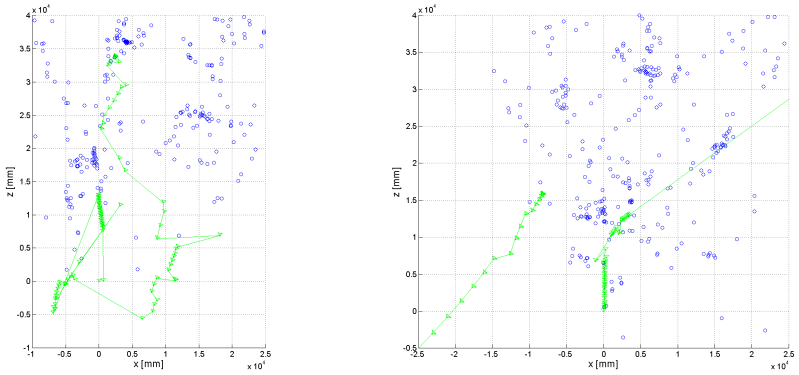


Figure A.21: Experiment KIT_Seq_03 (all MSaM inliers): Robustly estimated pose with the STA1 Lyapunov exponent (left) and the STA2 Lyapunov exponent (right) subset generated from all MSaM inliers. Estimated camera poses (green triangles), estimated structure (blue circles). The pose estimation with both subsets fail.

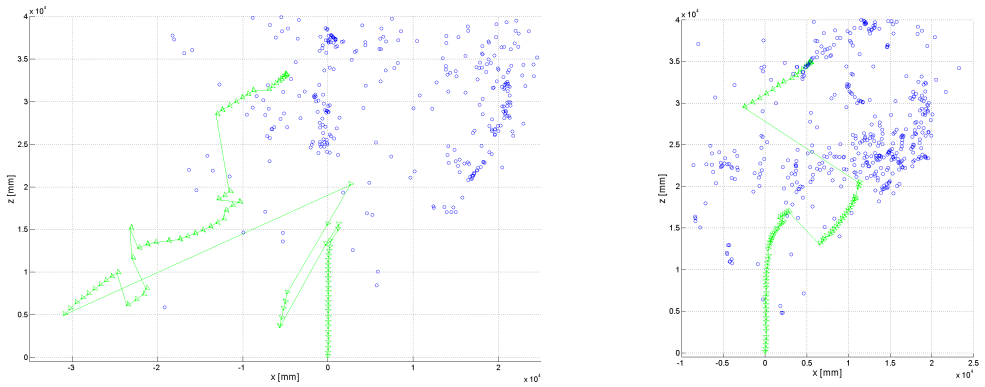


Figure A.22: Experiment KIT_Seq_03 (all MSaM inliers): Robustly estimated pose with the STA1 minimum variance (left) and the STA2 minimum variance (right) subset generated from all MSaM inliers. Estimated camera poses (green triangles), estimated structure (blue circles). The pose estimation with both subsets fail. However, The STA2 minimum variance subset (right) provides together with the STA1 minimum entropy subset (see fig. A.20 left) similar results than the STA1 minimum entropy subset (see fig. A.17 left) on the reference data.

List of Figures

List of Tables

Listings

Bibliography

[ABK91] Henry D. I. Abarbanel, Reggie Brown, and M. B. Kennel. Lyapunov exponents in chaotic systems: Their importance and their evaluation using observed data. *International Journal of Modern Physics*, 5:1347–1375, 1991.

[AD03] Adnan Ansar and Kostas Daniilidis. Linear pose estimation from points or lines. *PAMI*, 25:578 – 589, 2003.

[ADP12] Henrik Aanæs, Anders Lindbjerg Dahl, and Kim Steenstrup Pedersen. Interesting interest points: A comparative study of interest point performance on a unique data set. *IJCV*, 97:18–35, 2012.

[BETG08] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. *Computer Vision and Image Understanding (CVIU)*, 110:346–359, 2008.

[BF09] S. Blunsden and R. B. Fisher. The behave video dataset: ground truthed video for multi-person. `http://groups.inf.ed.ac.uk/vision/BEHAVEDATA/INTERACTIONS/`, 2009.

[Bou00] Jean-Yves Bouguet. Pyramidal implementation of the lucas kanade feature tracker description of the algorithm, 2000.

[BPSK11] Karla Brkić, Axel Pinz, Sinisa Segvić, and Zoran Kalafatić. Histogram-based description of local space-time appearance. In *SCIA*, 2011.

[BTG06] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *In ECCV*, pages 404–417, 2006.

[Cal] Caltech. Caltech pedestrian dataset. `http://www.vision.caltech.edu/Image_Datasets/CaltechPedestrians/`.

[CK95] João Costeira and Takeo Kanade. A multi-body factorization method for motion analysis. In *ICCV*, pages 1071–1076, 1995.

[CK98] João Paulo Costeira and Takeo Kanade. A multibody factorization method for independently moving objects. *IJCV*, 29:159–179, 1998.

[CM02] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *PAMI*, 24:603–619, 2002.

[CRM03] Dorin Comaniciu, V. Ramesh, and Peter Meer. Kernel-based object tracking. *PAMI*, 25:564–577, 2003.

[Dal05] N. Dalal. Inria person dataset. `http://pascal.inrialpes.fr/data/human/`, 2005.

[DRCB05] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. In *VS-PETS*, October 2005.

[DRMS07] Andrew J. Davison, I. Reid, N. Molton, and O. Stasse. Monoslam: Real-time single camera slam. *PAMI*, 29:1052–1067, 2007.

[DT05] Navneet Dalal and Bill Triggs. Histogram of oriented gradients for human detection. In *CVPR*, 2005.

[DTS06] Navneet Dalal, Bill Triggs, and Cordelia Schmid. Human detection using oriented histograms of flow and appearance. In *ECCV*, 2006.

[EG09] Markus Enzweiler and Dariu M. Gavrila. Monocular pedestrian detection: Survey and experiments. *PAMI*, 31:2179–2195, 2009.

[ELSvG08a] Andreas Ess, Bastian Leibe, Konrad Schindler, and Luc van Gool. A mobile vision system for robust multi-person tracking. In *CVPR*, 2008.

[ELSvG08b] Andreas Ess, Bastian Leibe, Konrad Schindler, and Luc van Gool. A mobile vision system for robust multi-person tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'08)*, 2008.

[FH75] Keinosuke Fukunaga and Larry D. Hostetler. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Trans. on Information Theory*, 21:32–40, 1975.

[FM95] Olivier Faugeras and Bernard Mourrain. On the geometry and algebra of the point and line correspondences between n images. In *IEEE International Conference on Computer Vision*, 1995.

[FMR08] Pedro F. Felzenszwalb, David McAllester, and Deva Ramanan. A discriminatively trained, mulitscale, deformable part model. In *CVPR*, 2008.

[FZ00] Andrew W. Fitzgibbon and Andrew Zisserman. Multibody structure and motion: 3-d reconstruction of independently moving objects. In *In European Conference on Computer Vision*, pages 891–906. Springer-Verlag, 2000.

[Gei] Andreas Geiger. Karlsruhe dataset. `http://cvlibs.net/datasets.html`.

[GN01] Michael D. Grossberg and Shree K. Nayar. A general imaging model and a method for finding its parameters. In *ICCV*, 2001.

[GRU10] Andreas Geiger, Martin Roser, and Raquel Urtasun. Efficient large-scale stereo matching. In *Asian Conference on Computer Vision*, Queenstown, New Zealand, November 2010.

[Har94] Richard Hartley. Lines and points in three views: A unified approach. In *ARPA94*, pages 1009–1016, 1994.

[Har97] Richard Hartley. In defense of the eight-point algorithm. *PAMI*, 19:580–593, 1997.

[Hor87] Berthold K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America A*, 4(4):629–642, 1987.

[HS88] Chris Harris and Mike Stephens. A combined corner and edge detector. In *Alvey Vision Conference*, pages 147–152, 1988.

[HTWM04] W. Hu, T. Tan, L. Wang, and S. Maybank. A survey on visual surveillance of object motion and behaviors. *Trans. on Systems, Man, and Cybernetics*, 34:334–352, 2004.

[KGL10] Bernd Kitt, Andreas Geiger, and Henning Lategahn. Visual odometry based on stereo image sequences with ransac-based outlier rejection scheme. In *IEEE Intelligent Vehicles Symposium*, San Diego, USA, 2010.

[KM07] Georg Klein and David Murray. Parallel tracking and mapping for small AR workspaces. In *ISMAR*, 2007.

[KM08] Georg Klein and David Murray. Improving the agility of keyframe-based slam. In *ECCV*, 2008.

[Kov] Peter Kovesi. Matlab and octave functions for computer vision and image processing. `http://www.csse.uwa.edu.au/~pk/research/matlabfns/`.

[LD10] Zhe Lin and Larry S. Davis. Shape-based human detection and segmentation via hierarchical part-template matching. In *PAMI*, 2010.

[LH81] Hugh Christopher Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133 – 135, 1981.

[LK81] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *International Joint Conference on Artificial Intelligence*, pages 674–679, 1981.

[LKSV07] Ting Li, Vinutha Kallem, Dheeraj Singaraju, and René Vidal. Projective factorization of multiple rigid-body motions. In *CVPR*, 2007.

[LKZF10] Qingshan Luo, Xiaodong Kong, Guihua Zeng, and Jianping Fan. Human action detection via boosted local motion histograms. *Mach. Vision Appl.*, 21:377–389, April 2010.

[Lor63] Edward N. Lorenz. Deterministic nonperiodic flow. *Journal of the Atmospheric Sciences*, 20:130–141, 1963.

[Low99] David Lowe. Object recognition from local scale-invariant features. In *ICCV*, Corfu, 1999.

[Low04] David Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60:91–110, 2004.

[LP07] I. Laptev and P. Perez. Retrieving actions in movies. In *Proc. ICCV*, pages 1–8, 2007.

[LSCG08] Bastian Leibe, Kondrad Schindler, Nico Cornelis, and Luc Van Gool. Coupled object detection and tracking from static cameras and moving vehicles. *PAMI*, 30:1683–1698, 2008.

[LSG10] D. M. Lopez, A. D. Sappa, and T. Graf. Survey of pedestrian detection for advanced driver assistance systems. *PAMI*, 32:1239–1258, 2010.

[MS05] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 27(10):1615–1630, 2005.

[ND10] Richard A. Newcombe and Andrew J. Davison. Live dense reconstruction with a single moving camera. In *CVPR*, 2010.

[Nis03] David Nistér. Preemptive ransac for live structure and motion estimation. In *Proceedings of the Ninth IEEE International Conference on Computer Vision - Volume 2*, ICCV '03, pages 199–206, Washington, DC, USA, 2003. IEEE Computer Society.

[Nis04] David Nistér. An efficient solution to the five-point relative pose problem. *PAMI*, 26:756–770, 2004.

[NNB04] David Nistér, Oleg Narodistky, and James Bergen. Visual odometry. In *CVPR*, pages 652–659, 2004.

[OCLF10] M. Ozuysal, M. Calonder, V. Lepetit, and P. Fua. Fast keypoint recognition using random ferns. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 32, pages 448–461, 2010.

[OSG10] Kemal Ozden, Konrad Schindler, and Luc Van Gool. Multibody structure-from-motion in practice. *PAMI*, 32:1134–1141, 2010.

[PET] Dataset - pets: Performance evaluation of tracking and surveillance. `http://www.cvg.rdg.ac.uk/slides/pets.html`.

[Sch88] Heinz Georg Schuster. *Deterministic Chaos: An Introduction*. VCH Verlagsgesellschaft mbH, 2., rev. ed. edition, 1988.

[SFP00] Y. Song, X. Feng, and P. Perona. Towards detection of human motion. In *CVPR*, 2000.

[SFS09] D Scaramuzza, F Fraundorfer, and R Siegwart. Real-time monocular visual odometry for on-road vehicles with 1-point ransac. In *Proc. of The IEEE International Conference on Robotics and Automation (ICRA)*, May 2009.

[Sha48] Claude E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 623–656, 1948.

[SiW95] Amnon Shashua and M ike Werman. International conference on computer vision (iccv). In *ICCV*, 1995.

[SK52] John Greenless Semple and G. T. Kneebone. *Algebraic Projective Geometry*. Oxford University Press, 1952.

[SLC04] Christian Schuldt, Ivan Laptev, and Barbara Caputo. Recognizing human actions: A local svm approach, 2004.

[SS02] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms, April 2002.

[SSP08] Gerald Schweighofer, Sinisa Segvic, and Axel Pinz. Online/realtime structure and motion for general camera models. In *IEEE Workshop on Applications of Computer Vision*, 2008.

[SSW08] Konrad Schindler, David Suter, and Hanzi Wang. A model-selection framework for multibody structure-and-motion of image sequences. *IJCV*, 79:159–177, 2008.

[ST94] Jianbo Shi and Carlo Tomasi. Good features to track. In *CVPR*, pages 593–600, 1994.

[SUW06] Konrad Schindler, James U, and Hanzi Wang. Perspective n-view multibody structure-and-motion through model selection. In *In Proc. of ECCV*, pages 606–619, 2006.

[TK91] Carlo Tomasi and Takeo Kanade. Detection and tracking of point features. Carnegie Mellon University Technical Report, 1991.

[Tri95] Bill Triggs. Matching constraints and the joint image. In *International Conference on Computer Vision*, 1995.

[VF08] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. `http://www.vlfeat.org`, 2008.

[WKR07] Brian Williams, Georg Klein, and Ian Reid. Real-time slam relocalisation. In *ICCV*, 2007.

[WKR11] Brian Williams, Georg Klein, and Ian Reid. Automatic relocalization and loop closing for real-time monocular slam. *PAMI*, 33:1699–1712, 2011.

[YP06] Jingyu Yan and Marc Pollefeys. A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate. In *ECCV*, pages 94–106, 2006.

[YP08] Jingyu Yan and Marc Pollefeys. A factorization based approach for articulated nonrigid shape, motion, and kinematic chain recovery from video. *PAMI*, 30:865–887, 2008.