TECHNISCHE UNIVERSITÄT MÜNCHEN
Lehrstuhl für Echtzeitsysteme und Robotik

Visual Tracking of Multiple Humans with Machine Learning based Robustness Enhancement applied to Real-World Robotic Systems

Suraj Nair

Vollständiger Abdruck der von der Fakultät der Informatik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzender:	Univ.-Prof. Dr. Bernd Radig
Prüfer der Dissertation:	1. Univ.-Prof. Dr.-Ing. habil. Alois Knoll
	2. Prof. Dr. Dieter Fox, University of Washington/USA

Die Dissertation wurde am 06.02.2012 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 29.08.2012 angenommen.

Abstract

This thesis presents a novel and robust vision-based 3D multiple human tracking system. It is capable of automatically identifying, labelling and tracking multiple humans in real-time even when they occlude each other. The primary contribution is a methodology to improve the robustness of the human tracking system and demonstrate its integration into real-world scenarios. The proposal is a system consisting of 2 stages, 1. a vision based human tracking system using multiple visual cues with a robust occlusion handling module, and 2. a machine learning based module for intelligent multi-modal fusion and self adapting the system towards drastic changes in lighting conditions. The function of the intelligent fusion module is to perform an on line analysis of image parameters that influence the performance of the tracker. According to this analysis, optimal weights are generated for each visual modality, determining its contribution in the current scene.

The thesis also proposes a novel approach to validate the 3D multiple human tracking system through zero-error ground truth data. Further, it proposes and demonstrates that the author's work can be easily integrated into a variety of distributed robotic systems being used in real world applications. The main focus of this thesis is in the area of *Human-Robot Interaction*, which requires real-time and precise information of the human positions to guarantee the safe interaction.

Zusammenfassung

Die vorliegende Dissertation beschreibt ein neuartiges und robustes bildbasiertes System zur dreidimensionalen Positionsverfolgung mehrerer Menschen. Es ist fähig, mehrere Personen automatisch in Echtzeit zu identifizieren, zu markieren und zu verfolgen - selbst im Fall von gegenseitigen Verdeckungen. Den primären Beitrag der Arbeit bildet eine Methodik zur Verbesserung der Robustheit bildbasierter Verfolgung von Menschen und eine Demonstration der Umsetzung in realistischen Szenarien.

Das vorgeschlagene System besteht aus zwei Stufen. Die erste ist ein bildbasiertes System zur Verfolgung von Personen unter Verwendung multipler visueller Wahrnehmungselemente mit einem robusten Modul zur Verdeckungsbehandlung, die zweite eine Einheit zur intelligenten multi-modalen Fusionierung und Selbstanpassung des Systems an drastische Beleuchtungsänderungen unter Zuhilfenahme maschinellen Lernens. Die Aufgabe des intelligenten Fusionierungsmoduls ist die Durchführung einer Online-Analyse von Bildparametern, die Einfluss auf die Performanz des Verfolgungsalgorithmus haben. Entsprechend dieser Analyse werden optimale Gewichte für jede visuelle Modalität berechnet, durch die ihr Beitrag an der Positionsermittlung für die momentane Szene bestimmt wird. Die Arbeit stellt auÃerdem einen neuen Ansatz zur Validierung des dreidimensionalen Personenverfolgungsalgorithmus mit Hilfe von fehlerfreien Messdaten vor. Weiterhin wird die Möglichkeit der einfachen Integration des vorgeschlagenen Systems in eine Vielzahl verteilter, praktisch eingesetzter Robotiksysteme dargelegt. Der Schwerpunkt der Arbeit liegt im Bereich der Mensch-Roboter-Interaktion, welche präzise Informationen der menschlichen Positionsdaten in Echtzeit benötigt, um eine sichere Zusammenarbeit zwischen Mensch und Roboter zu gewährleisten.

Acknowledgements

There are a number of individuals I would like to thank in gratitude to the support shown during the course of this thesis. These include my friends, colleagues and family. There are a few individuals who deserve a special mention.

I would like to thank Prof. Dr. Alois Knoll for providing me with the opportunity to conduct my research in his esteemed group. I would like to thank all my colleagues at Robotics and Embedded Systems and especially the OpenTL team for their intellectual support.

I would also like to thank Monika Knuerr, Gisela Hibsch and Amy Bücherl for their support and co-operation in the administrative tasks. In addition I would also like to thank them for creating a healthy and friendly atmosphere to work in.

A special thanks to Dr. Emmanuel Dean, Sebastian Klose and Philipp Heise with whom I have been working recently on diverse research topics. A special thanks to my colleague, Dr. Emmanuel Dean and my wife, Sonal Tavkar for proof reading the thesis. Lastly and most importantly I would like to thank my wife, parents and sister for their support and patience during the course of the thesis.

Contents

List of Figures

Chapter 1

Introduction

Over the past few decades robotic technology has developed rapidly [9, 10, 11]. Researchers have been investigating, with immense success, to systematically accommodate robots within the human space and possess cognitive abilities. Robots came into existence with the primary purpose of serving the auto mobile and automation industry [12, 13, 14, 15, 16, 17, 18, 19]. Fig. 1.1 shows a typical industrial robot arm in an car manufacturing assembly line. As technology progressed, they evolved further with abilities to perform complex tasks. This facilitated their use in other service domains such as monitoring, servoing, transportation etc. Robots were deployed in power plants, bio-tech labs, surgical environments, etc. in order to improve efficiency and enable tasks in environments where humans safety was of concern. Fig. 1.2 is an example of a service robot [20] in a bio-tech lab. As robots evolved further, they reached a superior state of intelligence. In the recent years, researchers have achieved immense success in accommodating robots within the human space [21, 22, 23, 24, 25]. Contrary to the past, where robots and humans did not share a common workspace, today the focus is more towards human robot interaction, both in the industrial and social domain. Social robotics is a new area, in which research is aimed towards developing robots to be a part of the daily lives of humans. This requires integration of modern sensors and sophisticated algorithms in order to sample information and understand the composition of the surrounding environment in real-time. Collaborative tasks between humans and robots demands a high degree of safety, both for humans and robots [9]. The robot requires to be aware of the actions of the humans and their activities in real-time in order to execute its task safely.

Computer vision [26] has served as an important tool in imparting intelligence to robot systems [27]. This thesis focuses on an area of computer vision, involving localisation and

Figure 1.1: Industrial robots in a car assembly plant.

tracking of humans in a shared robotic workspace. This chapter will discuss the role of computer vision and human motion tracking in robotics and their shortcomings when applied to real-world scenarios. On the basis of these arguments, the objectives and contribution of this thesis will be formulated.

1.1 Computer Vision for Sensor Driven Robotics

One of the major factors that resulted in robotics technology reaching newer heights is computer vision. Visual sensors integrated into robot systems and computer vision algorithms have paved the way for intelligent sensor driven robotics [28, 29, 30, 31, 32, 33]. Computer vision in itself is a wide area, and involves a large collection of algorithms to analyse the environment by using visual sensors. These algorithms interpret important events in the environment, which are further used by the robot to make correct decisions and execute the required tasks. A number of computer vision libraries exist providing application programming interfaces [34, 2, 35]. These libraries are engineered to provide a collection of computer vision algorithms, image processing techniques and sensor interfaces. Using the resources available, different computer vision applications can be developed.

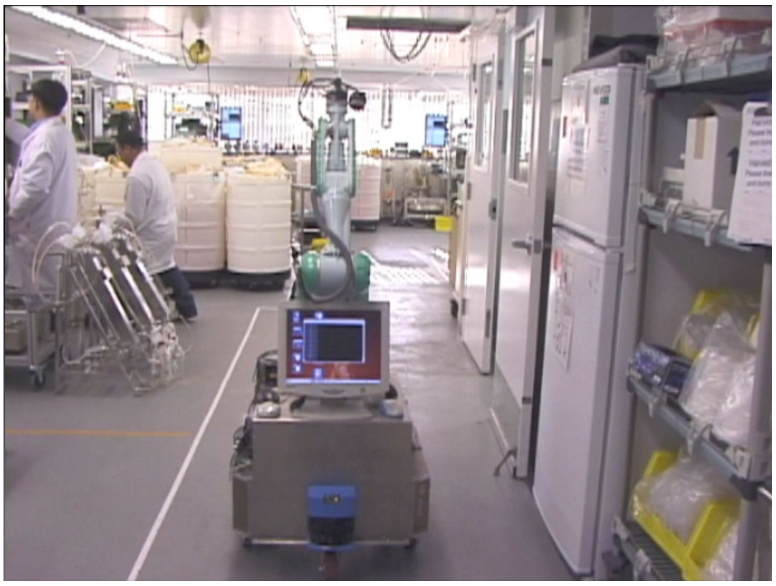


Figure 1.2: A mobile robot with an industrial arm in a bio-tech laboratory [1].

1.1.1 Visual Tracking

Visual tracking is one of the key areas within computer vision and involves localisation and tracking of a subject or subjects of interest. The subjects can be objects, humans, features within objects, etc. The pose of the targets are computed in 2D or 3D and depends on how many degrees of freedom the target is modelled with. Several algorithms have been developed to perform model based tracking where pre-defined models of the targets are generated offline and used for tracking using visual information obtained from a one or more cameras. The target model can be as simple as a rigid cube or a complete articulated human with many degrees of freedom [36]. Tracking of deformable surfaces is also possible which has found use case in augmented reality [37].

1.1.2 Tracking Humans in Shared Workspaces

This area of computer vision deals with the problems of tracking a single or multiple humans in one or more camera views. It can be applied to a variety of applications such as surveillance, animation, virtual-augmented reality, human robot interaction etc. Different types of human tracking systems exist depending on the methodology, target modelling, visual sensors used, etc. [38]. Some model the target as a simple box, while some use a complete 3D CAD model describing the shape and articulation together with the appearance. This depends on the main goal of the tracker. For some cases a simple target representation is enough to achieve the task. Variety of algorithms exist for tracking and retrieving the target's pose. The pose returned by the object could be either 2D or 3D depending on the state representation.

1.1.3 Requirements in Real-world Scenarios

Vision based tracking systems find use cases in a wide range of real-world applications. However, they are required to be extremely robust in order to perform within the specifications and safety regulations [39, 40, 41, 42]. This is also true for human tracking systems. Robustness is a key factor for their safe deployment in real-word applications. The system should consistently perform, within specifications, for all possible scenarios that the application might encounter. Therefore, robustness of any human tracking system is of primary importance in real-world scenarios.

1.1.4 Shortcomings of Existing Systems

A large number of vision based human tracking systems exist and will be highlighted in Chapter. 2. In this section the common problem faced by most of these systems when used in real-world scenarios will be discussed.

Developing a robust human tracking system, which will perform with consistent robustness in all possible scenarios is a challenging task. The robustness depends on multiple factors such as visual modalities, data fusion, sensor parameters, workspace conditions, etc. The key factor affecting robustness is degradation in the quality of information from the sensors. The primary reason for this is abrupt changes in the tracking environment due to external factors. Such a situation is highly probable since tracking environments do not necessarily have static conditions and they could change without prior information. A typical example is when the lighting condition in the tracking environment changes. Such a change affects the image quality acquired from the cameras. Under such circumstances, if the tracking algorithm uses a visual cue such a colour, it could fail when the lighting condition changes or fluctuates.

There are other factors which could affect visual cues such as intensity edges, optical flow etc. Hence, relying on a single visual cue is not optimal. Multi-modal fusion is an option, but also has dis-advantages. The performance of each visual modality depends on certain parameters. Abrupt changes in different parameters in the tracking environment can influence different modalities in different ways. Therefore, when a certain modality or a set of modalities are performing with the desired robustness and if a specific modality fails, it can cause the whole system to fail. Hence, static multi-modal fusion can under-perform in dynamic environments. Furthermore, certain modalities are suitable only under specific conditions but they reduce the robustness of the system in less favourable conditions.

Another important issue is system integration. Most human tracking systems perform impressively when used alone but when integrated into larger distributed systems for e.g. robotic systems, their robustness level reduces. This is due to the physical constraints introduced by the robotic system causing the environment to become more dynamic. For e.g when tracking humans in a workspace occlusion handling becomes tougher when the humans are occluded by a robot in a camera view and the position of the robot is not tracked. Even if the position of the robot is tracked the loss of visual information of the human target in the camera view cannot be avoided.

Many impressive human tracking system exists with robustness claims. However, most of them do not provide experimental evaluation under situation of claimed robustness. The system described in [43] demonstrates impressive 3D tracking results for multiple humans and claims to be invariant to lighting conditions but there are no concrete experiments conducted with respect to changing light to validate this claim. [44] demonstrates another interesting stereo based human tracking system using a 3D model. Their system shows impressive performance but does not show any evaluation under dynamically changing scenarios. In addition these systems do not shows how robust their performance is when integrated with larger systems in order to find suitable use cases. This test is important as most vision system find use cases within larger distributed systems. Chapter. 2 will discuss these issues in greater detail.

The main contributions of this thesis is a general purpose human tracking system with the primary objective of robustness enhancement in order to facilitate its use in a real-world robotic systems. Integration into real-world robotic systems and validating the robust performance of the system in dynamical environments is also an important contribution. The next section will formulate the core contributions of this thesis.

1.2 Thesis Proposal

The primary contribution of this thesis are as listed below:

- 3D Multiple Human Detection and Tracking System
- Robustness Enhancement by Machine Learning

- Validation of Different Robustness Aspects
- General Purpose Interface for Stand Alone Operation
- Integration into Real-World Robotic Systems

The following subsection introduces the contributions in brief.

1.2.1 3D Multiple Human Detection and Tracking System

It is a vision based system, capable of automatically detecting human targets within a defined tracking area. After detecting the targets, the motion of each target is tracked with a 3D pose in real-time. It uses a combination of visual sensors and computer vision algorithms to perform this task. The system has an ability to resolve multiple target occlusions in real-time and maintain individual trajectories, provided the targets do not leave the designated tracking area. The occlusion handling system guarantees robust tracking under circumstances of mutual occlusion in multiple camera views.

1.2.2 Robustness Enhancement by Machine Learning

A machine learning based approach is introduced to train and classify lighting conditions in the tracking environment. Since lighting conditions are highly influential in robust visual tracking, its classification helps the tracker to take important decisions to maintain the robustness. Depending on the classification, intelligent multi-modal fusion of two visual cues is performed. The optimal proportion in which the visual cues should be fused in order to achieve the desired robustness is computed. This approach improves the robustness of the tracking system in terms of self adaptability to changing tracking conditions.

Although classifying lighting conditions is useful in multi-modal fusion, it also finds an important use-case in robust pre-processing of camera images, such as background segmentation. Sudden changes in lighting conditions can be detected and the background model can be updated using this approach. The background model update is not trivial in presence of foreground targets. This thesis introduces an approach to identify such situations and update the background model in presence of foreground targets under changing lighting conditions. This exercise improves the robustness of the system to a great extent. Thereby, a general purpose human tracking system suitable for a wide range of applications and environments can be realized.

1.2.3 Validation of Different Robustness Aspects

The system is validated through a set of experiments. Each experiments is unique and validates the robustness of the system under a specific scenario. In this manner the performance of the system in different scenarios are evaluated. A novel approach to generate zero error ground truth has been developed which could be used as a benchmarking tool. Thereby, the system is validated for each robustness claim made.

1.2.4 General Purpose Interface for Stand Alone Operation

It serves as an interface between the human tracking system and other systems within a distributed framework. It is implemented using TCP/IP sockets. The interface is designed in a way to facilitate stand alone operation and thereby providing plug and play functionality. Within a distributed system, different systems can connect to the multiple human tracking system and retrieve information of the targets. The protocol is designed to provide a variety of functionalities depending on the information required. For e.g. a particular system might require only the information regarding a certain target while another would want data concerning all targets. Furthermore, its also possible to retrieve specific information such as ids, velocity, position, trajectory etc. for each target. The interface also allows other systems to remotely control the operation of the multiple human tracking system.

1.2.5 Integration with Real-World Robotic Systems

The thesis demonstrates how the system is designed in order to facilitate easy integration into a variety of real-world robotic systems. As mentioned before, the 3D multiple human tracking system is a stand alone system and can be interfaced with larger systems through generally purpose communication interface rather than complete integration within. The performance of the system is validated through demonstration scenarios in application such as visual servoing of humans for automatic cameraman in indoor applications, stero-scopic rendering in virtual reality scenarios, human interaction with virtual objects in augmented reality, 3D position based visual servoing of multiple humans and close range safe human-robot-object interaction.

In addition to the primary contributions the thesis also proposes two independent visual tracking systems used as complimentary systems within specific robotic demonstrators. The following subsections will introduce in Chapter. 6.

1.3 Discussion

In this chapter a overview into the evolution in the area of robotics was discussed. It highlighted the importance of computer vision in the area of robotics with a special focus on tracking humans in a shared robotic workspace. It emphasised the deficiencies and limitations of such systems when used in real-world applications. This thesis proposes a human tracking system with a focus on robustness enhancement and integration into real-word robotic systems, to improve its performance and reliability. In the course of the thesis, each aspect of the proposed system will be discussed in detail.

1.4 Thesis Contents

The contents of this thesis are distributed over several chapters, each addressing a specific phase in the thesis work. Following the introduction, already discussed in *Chapter 1*, the state of the art in vision based human tracking systems is described in *Chapter 2*. This highlights different methodologies and their advantages and limitations. Thereby, the motivation to develop a general purpose and robust human tracking system is justified. *Chapter 3* introduces the novel general purpose human tracking system and its modular construction, following the design principles of the *OpenTL Library* [2, 36]. The *OpenTL Library* is introduced, to which the author has been an active contributor. The human tracking system is discussed in detail with extensive discussion on each building block. Further, in *Chapter 4*, a novel approach to improve the robustness of the system in dynamic environments using machine learning is introduced and discussed in detail. *Chapter 5* describes the experiments performed to validate and benchmark the system in different scenarios. These experiments were performed in real and simulated environments. Following the experiments, *Chapter 6* describes the integration of the human tracking system into several real-word robotic applications which were tested, verified and demonstrated. These include laboratory, public exhibitions and industrial set-ups. It also introduces a vision based system, a 2D face tracking system and a 3D object tracking system, that is used in two of the demonstration scenarios. Finally, *Chapter 7* summarises the conclusions drawn from this thesis, followed by the possible improvements in different aspects of the system through future work.

Chapter 2

Prior Art

This chapter addresses the state of the art in vision based human detection and tracking. It provides an insight into existing systems in this area and classifies them on the basis of the methodology. With respect to the state of the art, the main contribution of this thesis is formulated thereafter.

2.1 Human Detection and Tracking

There is a considerable amount of literature in the area of human detection and tracking, also referred as person or people tracking [45] [46] [47], [48]. These approaches differ on different aspects such as number of targets, modelling, pose estimation, tracking methodology, sensor etc. In this chapter different types of human tracking systems will be highlighted. Within each type, a set of relevant systems will be discussed. The main focus will be on multi camera based human tracking systems since the contribution of this thesis is in the similar area. Due to the large number of approaches and systems in each area of human tracking, a comprehensive study of each system is out of the scope of this chapter. In addition, the main focus of this thesis is to address the robustness issues of vision based human tracking systems in a general set-up. It highlights the factors that affect the robustness of a wide range of vision based systems. Therefore, in this chapter the relevant approaches will mentioned and credited, but they wont be analysed critically. Instead, a general common problem formulation will be introduced which could benefit a variety of systems.

2.1.1 Single Target Trackers

These systems are capable of detecting and keeping track of a single target. Many popular systems for single target tracking are based on colour histogram statistics [49, 50, 51, 52] and employ a pre-defined shape and appearance model throughout the whole task.

In particular, [51] uses a standard particle filter with colour histogram likelihood with respect to a reference image of the target, while [50] improves this method by adapting the model online to light variations, which, however may introduce drift problems in presence of partial occlusions. The same colour likelihood is used by the well-known *mean-shift* kernel tracker [52].

The person tracking system [49] employs a complex model of shape and appearance, where colour and shape blobs are modelled by multiple Gaussian distributions, with articulated degrees of freedom, thus requiring a complex modelling phase, as well as several parameter specifications.

2.1.2 Sliding Window Techniques

In the sliding window approach the image is scanned at relevant positions and scales in order to detect people. There are two components involved, which are the features and the classifier [53]. The feature provides appearance information and the classifiers determine the possibility of a successful detection of the person within the window. These methods are computationally exhaustive but advances in GPU computing have led to the possibility of real-time performances. The system demonstrated by [54] is a good example. A variety of features and classifiers have been used within this approach. The most commonly used features are Haar wavelets, shape contexts and histogram of oriented gradients [53]. The classifiers that are used are mainly SVM and AdaBoost.

[55] uses a combinations of Haar features and a polynomial SVM classifier. They use a dense representation using wavelets with scales of 16 and 32 pixels with a 75% overlap. Three different types namely vertical, horizontal and diagonal allow the encoding of low frequency changes in contrast. The feature vector for a 64×128 pixel detection window is 1326 dimensions.

[56] uses histogram of oriented gradients as a feature. They compute the image derivatives in x and y direction. Using these gradients a cell histogram is generated. The histogram blocks are groups of cells with dimensions 2×2 with an overlap of one cell both in the x and y directions. The final feature vector consists of normalized block histograms with a dimension of 3780 for a 64×128 detection window [53].

[57] and [58] use shape contexts as a feature descriptor. This feature descriptor is based on intensity edges. These edges are extracted using the Canny edge detector. The edges are stored in a log-polar histogram [53]. In the sliding window, search dense sampling is performed. The overall length of all feature descriptors for one test window is 3024 [53].

2.1.3 Part Based Models

[53] gives a good insight into a variety of approaches which rely on part based methods for human tracking. The models used in these approaches consist of two parts. The first part comprises of features which model different parts of the humans. The second part models the human body topology [53]. A variety of part based models have been proposed such as upright models [57] and articulated models [53].

[59] models the human in the form of a template consisting of different parts. The head, torso, upper and lower arms forming the unique parts. They use probability estimation techniques to detect and track people in 2D. Certain assumptions regarding the constraints on the states of the pose are introduced and justified with respect to motion primitives [59]. The likelihood computation is performed based on colour histograms. Their system is capable of handling occlusions in the 2D scene. The system is claimed to perform between 10 and 13 fps on a 2.66 GHz Core Duo PC.

2.1.4 Stereo Trackers

In the recent past, several systems have been developed to track humans using multiple cameras in both un-calibrated and stereo-calibrated fashion [48].

[48] provides a systematic mention of approaches [60, 61, 62, 63, 64], which use un-calibrated cameras and homography to perform people tracking. The homography constraint in each camera view is used to compute the multiple projection of the principle axis of the target to the ground [48]. The approaches stated above use the homography constraints in different ways. [63] uses the homography constraint within a particle filter framework with an appearance model for segmentation and matching of each hypothesis [48].

[48] use a slightly different approach. They make use of a combination of the perspective geometry and the homography constraints from each camera view. This information is fused to check for the presence of people in each camera view. Background subtraction is used for segmenting the camera images for foreground information. The information from the multiple cameras are used to solve occlusions.

An approach based on fusing information from multiple cameras and using a sequential Bayesian filter for each camera to estimate the state of the tracker is presented in [65]. In their approach a hypothesis is assigned to one of the cameras and each camera has a different number of hypothesis attached to it. The contribution obtained from each camera keeps changing, depending on the prior information computed and the measurement data [65] obtained from the cameras.

[66] present another multi-target tracking system using multiple cameras. Their approach is focussed on a self-configuring camera network consisting of cameras with a pan-tilt. The cameras keep a track of the targets and adjust their parameters with respect to each other in order to obtain high resolution information of each target by collaboration. It also helps the cameras keeping focus on the tracking area [66]. They use the Kalman-Consensus filter [67] to perform the tracking of the targets [66]. Due to the continuous collaboration between the cameras, their configuration keeps changing during the tracking process due to changing pan and tilt settings [66].

Another multi camera approach is presented by [68]. However, in their approach, the cameras are sparsely distributed and have non overlapping views. They model the appearance of the targets and use a probabilistic framework to do the tracking. In order to minimize the chances of occlusions, the cameras are mounted on the ceiling looking downwards. The human object model is divided into three regions along the principle axis. The camera image is preprocessed for background subtraction and foreground blob extraction. The foreground blob is further used for feature extraction and used in the probabilistic framework.

[69], presents a multi camera approach to track people in cluttered scenes by performing occlusion handling. They use colour models with background subtraction and a Bayesian estimation for tracking. They also propose a region based stereo algorithm, which is claimed to be capable of detecting a 3D point in an object if its region in the image is known [69]. Their algorithm runs at a rate of 5 seconds per frame on Pentium 2 Xeon 4 Mhz PC.

Further, [70] proposes a multi-view, multi-hypothesis system to track people on a ground plane. They assume that the people could be occluded. They use colour based models and ground plane homography to estimate the target positions [70]. They use iterative segmentation with a particle filtering framework to tackle the large state space and reduce computational costs [70].

[71] presents a slightly different approach of multi view tracking of people. They use multiple cameras but do not detect or track targets from any single camera or a pair of cameras [71].

Instead, they use information in combination from all view which is projected back to each camera view. They do not require a calibrated set up. They use planar homographic occupancy constraints for likelihood computation [71]. This is used to resolve occlusions. This fusion method also models scene clutter using the Schmieder and Weathersby clutter measure [72, 73, 74, 75, 71].

[76] presents an approach to detect and track humans using stereo cameras. They combine background segmentation with Kalman filtering supported by colour information of the targets. The background segmentation is used to detect targets where the background is modelled using a height map. A face detector assists the background segmentation process to detect human targets. The cameras are positioned to see the faces of the targets. The tracker uses a Kalman filter and colour information to track the targets when they are too close to each other. The system can track targets up to a distance of 5 meters with a cycle time of 40 ms.

Multiple people trackers [77, 78, 79], have the common requirement of using a very little and generic off-line information concerning the person's shape and appearance, while building and refining more precise models (colour, edges, background) during the on-line tracking task. This unavoidable limitation is due to the more general context with respect to single-target tracking, for which instead specific models can be built off-line.

The work presented in [80], uses a template based approach. This method uses about 4,500 templates to match pedestrians in images. The Chamfer distance measure is used for similarity measure.

[43] combines target occupancy in the ground plane with colour and motion models to track people in continuous video sequences. This approach requires heuristics to rank the individual targets to avoid confusing them with another.

[81] also presents a vision based system to track human in multiple cameras. They use an approach to find the limit on the field of view for one camera which is visible in the other cameras. The FOV constraint is further used to disambiguate between correspondences.

[82] introduces a vision based 3D system for tracking people in a smart room. They use a calibrated camera system within a distributed framework. Each camera runs on a dedicated PC. Each PC computes the foreground region in its camera scene. They use a model adaptive background model to obtain the foreground information from each camera. The detected foreground regions are sent to a tracking agent which computes the locations of people from the detected regions. [82] uses a probabilistic framework to do the tracking. They use two different approaches, namely best-hypothesis heuristics and multi-hypothesis tracker [82]. They test

their approach using a sequence containing two people walking in a conference room, recorded by three cameras. Their results suggest that both approaches show comparable performances.

Another multi-camera approach in human motion tracking is presented by [83]. They use grey scale images from multiple fixed cameras to perform the tracking. They use multivariate Gaussian models to estimate the closest matches of humans between consecutive image frames obtained from the cameras [83]. The algorithm runs in real-time.

The system proposed by [84] is aimed at tracking human motion with key focus on occlusions. They use a multi-view approach to combat occlusions and articulated motion. Each camera view is independently processed on an individual computer [84]. This processing step uses a predictor-corrector filter [84] to weigh the re-projections of the 3D pose estimates with the observed image motions [84]. The data from the correction step is provided as an input to a Bayesian network. Within the Bayesian network, the observations from the different cameras are fused together in order to resolve the independent relations and confidence levels [84]. An additional Kalman filter is used to update the 3D state estimates. The system was tested on a sequence with multiple people in motion. They claim from their experiments that their method yields better results as compared to data fusion techniques based on averaging.

[85] present a multi camera people tracking system using Bayesian filtering based modality fusion. They also resolve occlusion using multiple views. The cameras are un-calibrated and widely arranged. They employ a modality fusion technique based on the approach by [86]. Their approach works in two modes, namely single camera tracking in which subjects are matched between consecutive frames and multi camera co-operative tracking where the subjects are matched using information gained across cameras. [85] claims that the system can also be used to track and follow multiple subjects through the field of views of different cameras.

[87] presents a stereo camera based people tracking system. They address the problem of tracking in rooms where the camera cannot be mounted high enough. Due to this there is a high probability of occlusions. They propose a method to project the 3D voxels on the tracking floor and thereby track their peaks for the purpose of ignoring view changes due to low camera mounting. They also provide forces among trackers to handle occlusions [87].

[88] presents another stereo cameras based people tracking system. It is a real-time system to track humans over a wide area. Each individual camera detects and tracks the targets in its view. A multi-camera fusion module combines tracks of a single target in all view to a global track [88]. They use stereo segmentation to track multiple humans in cluttered scenes. Camera

hand-off is performed through ground based fusion. The system was tested on a 12 camera system.

2.1.5 Human Pose Trackers

Extensive research has been conducted in the area of human body pose tracking. In these systems, the goal is to compute the articulated pose of the human body. The complexity of the pose in terms of degrees of freedom varies according to the methodology used and the manner in which the human is modelled. Survey by [45] provides the initial developments into this area. Since the recent past, depth cameras with a wide range of accuracy and cost are available, leading to their use in human pose tracking [89, 30, 90]. [91, 90] are few of the many systems using such hardware.

One of the most impressive systems in this area has been [92]. Their approach computes the 3D pose of the human body using a single depth image. The depth image camera of the Kinect gaming platform is used. The method involves training of a large data set of human pose represented in the form of depth maps which is further used to classify the human pose. The training set models the human pose, body shapes and also clothing information. The training images are generated synthetically with close resemblance to the real world. The training is performed on a deep randomized decision forest. The algorithm runs at 200 fps. The robust performance of the system depends on the range of the depth image sensor which in this case is approximately 3 to 5 meters. Therefore, the use of such sensors in large workspace is limited.

2.1.6 Human Tracking in Robotic Scenarios

The ultimate goal of vision based people tracking systems is application scenarios. Robotics finds many use for such systems. A variety of single and multiple people tracking systems have been used in robotic systems to achieve various tasks.

[93] presents a vision based person tracker deployed on a mobile robot. They use colour based blob detection combined with contour detection. The colour based part provides a rough estimation and edge matching and a contour based model refines the result [93]. The system was tested on a real robot in a natural indoor environment. The task of the robot was to track and follow the person.

[94] presents a people detection and tracking system within a service robotic scenario. They combine multiple sensor modalities in a general tracking framework [94]. They use learned motion patterns to track the motion trajectories. A particle filtering approach is used combining

laser scanner and camera image data. The observed trajectories are generalized using self organizing maps [94]. These generalized patterns are used to predict the target's position. Their experiments show how multi-modality improves robustness of the system.

[95] presents another people tracking system applied to mobile robots. The system tracks fast moving people in outdoor environments. The tracking is performed through a laser scanner and an omnidirectional camera. They demonstrate two approaches. The first method uses visual tracking and performs well under slow speed of targets. However, its performance degrades under higher speeds. The second approach uses a combination of laser scanner and the camera and performs well under dynamic conditions.

2.2 Motivation of the Thesis

The investigation into the state of the art confirms the extensive research being conducted in the area of human tracking with several impressive systems. However, there exists a lot of scope in terms of robustness enhancement. Many systems claim to be robust to dynamic environments but they do not validate this claim with experiments in such conditions. For e.g. [61] demonstrate a impressive multi-camera people tracking system with a probabilistic occupancy map. They claim that their system is robust to light changes but there is not experimental results to prove this. Similar claims are made by [63], where the target is modelled to incorporate the illumination changes. However, there are no experiments in such conditions to validate the tracker performance. In addition, the most of the above mentioned systems are tested under stand alone conditions. In these tests the tracker is not integrated into complete vision solutions. It is important that a system is also tested when it is integrated into a larger system for e.g a robotic system since they introduce physical constraints.

The main goal of this thesis is to improve these systems in two ways. The first part is to develop and complete human tracking system capable of automatically detecting and tracking humans. The second part is to focus on the issues which effect the robustness of the system when integrated into real-world dynamic environments. The thesis uses a novel approach in the form of a bank of Bayesian Particle Filters combined with Intelligent Multi-modal fusion of two visual modalities through machine learning. The novelty of the approach is the positive impact on robustness in dynamic environments through occlusion handling, machine learning based adaptation to drastically changing lighting conditions, intelligent multi-modal fusion and the possibility of adapting individual Bayesian filters to the behaviour of their targets. Another important contribution of the thesis is a methodology to validate the systems performance through experiments targeting different aspects of the system. The literature review confirmed that in multi-camera systems for 3D human tracking there is not standardised benchmarking technique as compared to pedestrian detection systems [96]. Different systems evaluate their results in different ways. For e.g. [97] introduce an impressive system which can track multiple people but the experiment is conducted using a single target and moreover the tracking results are recorded by making the person stand at predefined locations. The tracker is not evaluated when the target is in motion which is a very important test to evaluate the tracking accuracy. The reason being, for stationary targets the tracker converges better as compared to moving targets. Experiments with multiple targets interacting with each other is also essential but missing. [70] test their system with multiple targets in indoor and outdoor conditions. The ground truth is generated using manual marking schemes which induces its own errors. The tracker evaluation using this ground truth does not provide an error free analysis of the tracking accuracy. Therefore, in this thesis a novel method to generate zero error ground truth through 3D modelling and simulated animations has been implemented. Using zero error ground truth data, the system has been evaluated in different aspects. The same experiments have also been conducted in real-world scenarios.

In addition, the thesis also focusses on a general purpose interface in order to facilitate easy integration into different robotic systems. In order to validate this claim 5 real-world robotic demonstrations scenarios have been constructed and tested.

To summarize, the main limitations of vision based human tracking system operating in real-world scenarios are:

- Dynamic behaviour of the environment such as changing lighting conditions
- Tracking speed
- Occlusion in case of multiple targets
- Easy systems integration into real-world scenarios

The motivation for this doctoral thesis is based on the limitations presented so far. The main goal of this thesis is:

- To improve the robustness of the system in dynamic environments
- Provide real-time capabilities irrespective of the number of targets tracked

- Handle occlusion in case of multiple targets
- Evaluate the system in different scenarios
- Provide a stand alone system architecture for easy integration with larger robotic systems

Chapter 3

Multiple Human Tracking System

3.1 Overview

In this chapter a detailed description of the real-time multiple human tracking system is provided. The system architecture and tracking methodology are presented.

The multiple human tracking system uses visual information from multiple cameras in order to detect humans within a pre-defined workspace and thereafter, tracks their motion in real time. The workspace is also referred as the tracking area. The system automatically detect humans when they enter the tracking area. Following the detection, it keeps track of the 3D position of each target, thereby providing an estimate of their motion trajectories. The detection process operates independent of the tracking allowing detection of new targets when they enter the tracking area. The visual tracking process is not interrupted by the detection of new targets. Furthermore, one of the main features of this novel system is that its speed is not reduced when the number of targets increases.

The system architecture is inspired by the *OpenTL* library [2, 36, 98] for rigid and articulated object tracking. The author is an active contributor to its research and development. The following sections provide detailed discussions about various aspects of the system. It will start with a discussion about *OpenTL* followed by engineering methodology and functional pipeline of the 3D multiple human tracking system.

3.2 OpenTL Library for Rigid and Articulated Object Tracking

OpenTL is a general-purpose software library for real-time, model-based and marker-less tracking of targets. It allows tracking of single or multiple targets with different degrees of freedom using single or multiple cameras and different visual modalities. It supports a wide variety of computer vision algorithms and systems. The software architecture is layered in the form of an object-oriented software library. Each visual modality is described under a common abstraction. A variety of data association and fusion schemes can be instantiated, providing modularity, scalability and parametrisation. This library is a powerful tool for developers in the computer vision and robotic communities.

3.2.1 Library Architecture

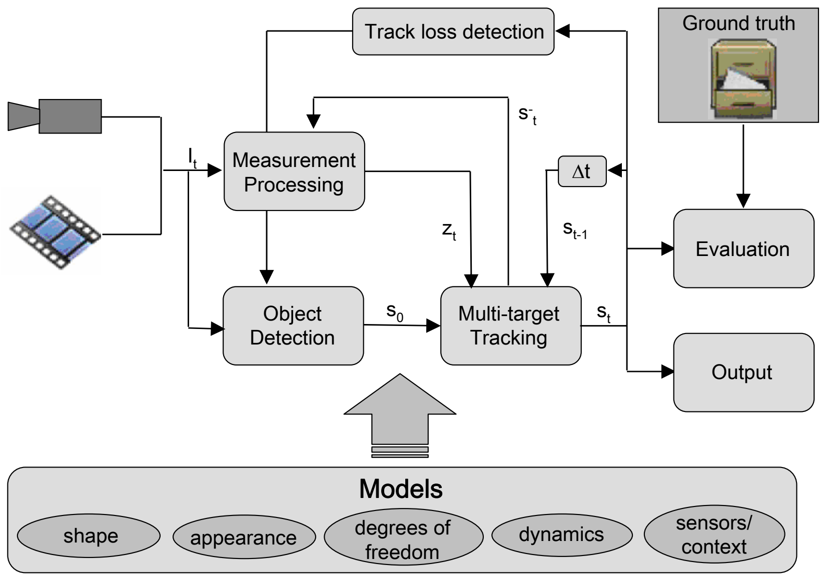


Figure 3.1: High-level flow diagram for a general visual tracking system [2].

Fig. 3.1 illustrates the processing modules within the *OpenTLlibrary*. They can be categorized as:

- Models: It consists of the off-line information available about the targets. It can also include sensors models, background models etc.
- Tracking pipeline: Consists of input devices, pre-processing layers, measurement processing (image processing, data association and fusion), Bayesian estimation, post-processing and visualization of different parameters of the system.
- Complimentary modules: These includes object detection which provide useful data for initialization, target loss detection and benchmarking estimates with ground truth.

3.2.2 Functional Architecture of the Library

The architecture of the library is organized in functional layers. Fig. 3.2 shows the abstraction, from base utility and data structures to the application:

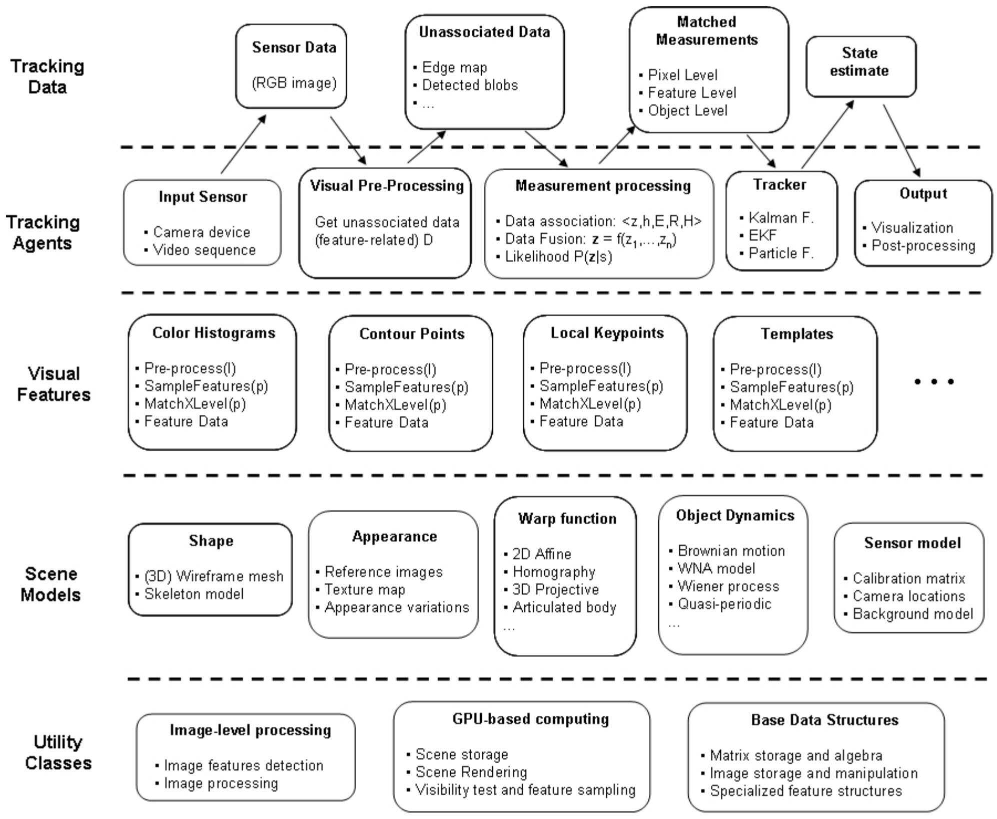

Tracking Data
Sensor Data (RGB image)
Unassociated Data • Edge map • Detected blobs • ...
Matched Measurements • Pixel Level • Feature Level • Object Level
State estimate
Tracking Agents
Input Sensor • Camera device • Video sequence
Visual Pre-Processing Get unassociated data (feature-related) D
Measurement processing • Data association: <z,h,E,R,H> • Data Fusion: z = f(z1,...,zn) • Likelihood P(z|s)
Tracker • Kalman F. • EKF • Particle F.
Output • Visualization • Post-processing
Visual Features
Color Histograms • Pre-process(I) • SampleFeatures(p) • MatchXLevel(p) • Feature Data
Contour Points • Pre-process(I) • SampleFeatures(p) • MatchXLevel(p) • Feature Data
Local Keypoints • Pre-process(I) • SampleFeatures(p) • MatchXLevel(p) • Feature Data
Templates • Pre-process(I) • SampleFeatures(p) • MatchXLevel(p) • Feature Data
• • •
Scene Models
Shape • (3D) Wireframe mesh • Skeleton model
Appearance • Reference images • Texture map • Appearance variations
Warp function • 2D Affine • Homography • 3D Projective • Articulated body ...
Object Dynamics • Brownian motion • WNA model • Wiener process • Quasi-periodic ...
Sensor model • Calibration matrix • Camera locations • Background model
Utility Classes
Image-level processing • Image features detection • Image processing
GPU-based computing • Scene storage • Scene Rendering • Visibility test and feature sampling
Base Data Structures • Matrix storage and algebra • Image storage and manipulation • Specialized feature structures

Figure 3.2: The layered class architecture of the OpenTL library [2].

In particular, the library has the following layers as described in [2]:

1. Tracking Data: It consists of the signal obtained from the sensor, pre-processed data depending on the visual modality, tracker measurement data and residuals obtained from the observation model. It can also contain the multi-target posterior state distribution.

2. Tracking Agents: They consist of the input devices, pre-processing modules, visual feature processing (sampling, matching and data fusion), likelihood computation, Bayesian tracker, output visualization and post-processing.

3. Visual Feature Processing: It is a common abstraction for the visual modalities involving pre-processing, sampling from the model, warp in image space, data association, residual and covariance computation. Each class contains storage for off-line and on-line model features and intermediate processing results.

4. Object Models: It consists of the off-line available information from the target object. This includes the shape, appearance, degrees of freedom for pose parameters and related warp functions with Jacobian computations. It can also have the object dynamics, sensor and context models.

5. Utility Classes: These include model independent low level functions for image processing, data storage, basic algebra and image manipulation functions. It also includes GPU-assisted scene rendering tools and visibility testing of geometric primitives under a given camera view.

The next sections will provide a detailed description of the multiple human tracking system. It starts with the methodology of engineering, followed by the system architecture. Further, the tracking methodology is discussed followed by the user and communication interfaces.

3.3 System Methodology

The methodology adopted to engineer the 3D multiple human tracking system can be highlighted as:

- Modular Construction
- Each Module has Specific Processing Task
- General Purpose Interface among Modules
- Offline Configuration of Individual Module Parameters

- Online Configuration of Specific Parameters through User Interface
- General Purpose Interface to other External Applications

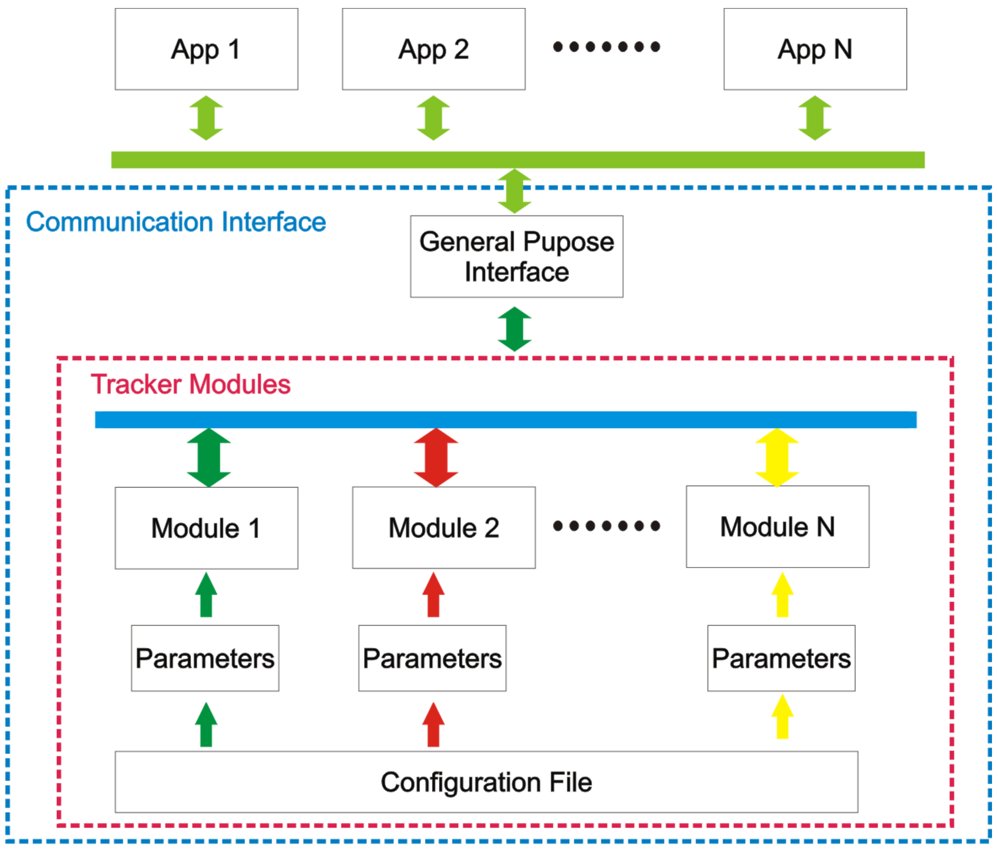


Figure 3.3: Methodology of Engineering.

Fig. 3.3 shows how the system is engineered. It is designed using a modular approach. Different functional aspects of the system are designed in the form of individual modules. Each specific module is independent from the other in terms of functionality and performs a specific processing task. The outputs from individual modules serve as inputs to other modules. The dependency between modules is realized by a well defined general purpose interface. If one module undergoes an update, upgrade or restructuring, the dependent modules do not require modification. This also allows analysis and optimization at a modular level. Each module has its own parameter container. The parameters of each module can be configured offline through a unified system configuration file which is parsed on start-up wherein, each individual parameter container is initialized. In addition, certain specific system parameters can be tuned online through an user interface.

An important design principle of this system is a general purpose communication interface to external applications in a standalone manner. Multiple applications can communicate with this system and exchange information allowing development of larger real-world application such as vision driven robotic systems in a convenient manner.

3.4 System Architecture

This section will provide a detailed insight into the construction and function of the system. The section is organised into subsections, each dedicated to a certain aspect. To begin with, the general hardware set-up is introduced. This is followed by target modelling and visual modalities used by the tracker. Finally, the justification for a Bayesian framework is provided.

3.4.1 Visual Input Sensors

The visual sensors used are cameras capable of streaming raw *RGB* images at rate of 25 - 50 fps. The cameras can be USB or Firewire. They are mounted on the ceiling in a stereo configuration, sharing a common view. The stereo set-up requires a minimum of two camera in order to estimate the 3D position of the targets. The more the number of cameras, better is the accuracy of tracking depending on their placement. However, there is a limit on the PC hardware in terms of how many cameras can be supported while maintaining the required data transfer rate. In the current set-up 4 cameras of the same type are used which stream raw *RGB* 444 images of resolution (752×480). The system is scalable with respect to the number of cameras. Adding a new camera requires no modifications to the software architecture. The cameras operate in streaming mode where images are written into the memory continuously. When a request arrives for an image update, the latest image from the buffer is returned to the image stack from where it can be accessed by different modules of the system.

3.4.2 Target Modelling

The target, representing a human, is modelled to represent its shape and appearance. The target's shape is modelled as a simple 3D rectangular box approximating to the dimensions of an average sized human. The appearance model is required for visual modality processing. For e.g. colour modalities extract information from the colour distribution data. Therefore, when

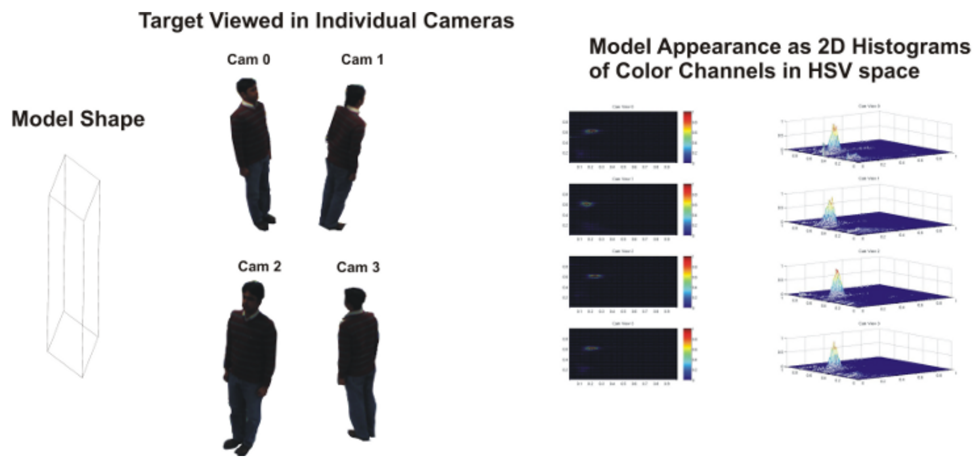

Target Viewed in Individual Cameras
Cam 0
Cam 1
Cam 2
Cam 3
Model Shape
Model Appearance as 2D Histograms of Color Channels in HSV space

Figure 3.4: The figure illustrates how the target is modelled in the form of its shape and appearance. It shows the target viewed in each individual camera view. To the left the shape model is shown in the form of a rectangular cube with dimensions of an average sized human. The right hand side of the image shows the appearance model in the form of 2D joint probability histograms of the target's colour distribution in the HSV colour space [3]. The appearance model is generated for each individual camera view.

a target is detected its appearance model is generated in the form of a 2D joint probability histogram [99, 2] of its colour distribution. The HSV colour space is used to build the histograms, since this colour space distinguishes colour information from the intensity in an image. All camera views are used to generate the appearance model.

The tracker holds a 3D state-space representation of the target's pose, given by a translation (x, y, z) within the tracking area. This pose form provides 3 degrees of freedom to the target which forms the basis for modelling its dynamics.

Fig. 3.4 illustrates the target modelled in the form of its shape and appearance. The target dynamics is modelled using the Constant White Noise Acceleration ($CWNA$) motion model [100, 2, 36].

3.4.3 Visual Modalities

The system employs multi-modal fusion of two different visual modalities complimenting each other for robust tracking of the the human targets. These modalities are colour histograms and optical flow. The contribution of the modalities to the tracker is obtained through an intelligent multi-modal fusion module using machine learning. This module will be described in detail in Chapter 4.

The colour modality represents the tracker measurement in the form of colour distribution through 2D joint probability histograms. The histograms are computed in the regions where a hypothesis is generated in order to weight the observation. On the other hand optical flow [2, 34, 101, 102] represents the tracker measurement in the form of pixel displacements in the visual scene. It computes the apparent motion of objects, edges and surfaces projected on the camera image. The system uses a technique know as total variation described in [102] to compute the optical flow. Similar to the colour modality, analysis of the optical flow vectors in the projected hypothesis region provide a good measure for comparison with the apparent velocity of the target hypothesis.

3.4.4 Bayesian Tracking

The tracker uses a bank of sampling-importance-resampling particle filters [103, 104, 79, 105] working on a 3D motion model, appearance model and optical flow. Each target is associated with a unique particle filter. A particle filter is chosen over the more conventional Kalman Filtering techniques [106], because the tracker needs to be highly robust in dealing with multi-modal likelihoods due to cluttered background. The particle filter provides the sequential prediction and update of the respective 3D $states = (x, y, z)$.

Particle filters are usually computationally intensive. A bank of particle filters increase computation cost with every new target. In order to achieve real-time performance in this system, a global particle set is maintained and distributed evenly among the bank of particle filters. Hence, if a new target enters the tracking area, the system instantiates a new particle filter and redistributes the global particle set evenly among the updated filter bank, keeping the computation cost constant. In other words, the computational speed is not affected when the number of targets increases. This is feasible because, when the number of targets increase in the tracking area, their mobility reduces and the number of particles needed to track a target can be reduced. The system relies on the particle filter bank approach over the the more conventional $MCMC$ filter [107] for handling multiple targets since the later maintains a global motion model, where as a bank of particle filters allows the system to learn and control the motion model of each target individually. This is important when the targets exhibit different dynamical behaviours, which is the case in real scenarios.

Tracking multiple targets requires occlusion handling between targets in each camera view. This is important since, when a target occludes another target in a camera view, that particular camera should be excluded during the likelihood computation for the targets which are occluded.

The reason being that, the region sampled will contain measurement data only for the target which occludes the other targets. However, for successfully obtaining the 3D pose of each target, it is necessary that the features be visible in at least 2 or more camera views. This system handles occlusions between targets in real-time using an occlusion query module. This module is also used during the target detection phase because, when the system models the target, the appearance information should be sampled only from the camera views in which the target is visible. This module will be discussed in greater detail in the sections to come.

3.5 Tracking Pipeline

Fig. 3.5 describes the complete pipeline of the tracking system [108]. Each module is discussed in detail in the subsections below.

3.5.1 Image Acquisition

The sensor images are obtained from the cameras. Each camera provides a raw RGB 444 image. The cameras can operate in a free streaming and on-request mode. In the former, the acquired images are updated in regular time intervals while in the later the camera updates an image stack when requested for an update.

3.5.2 Pre-processing of Sensor Images

Each sensor image I_{id}, obtained from the image acquisition system, where the index id corresponds to the USB camera index, undergoes a two stage pre-processing. In the first step background segmentation is performed on each camera image using a static background model. The background subtracter uses the Gaussian mixture model approach described in [109, 110, 111]. The background segmented image from each camera is then converted from RGB to HSV for the colour-based likelihood.

$$Ibg_{id} = bgSub(I_{id}) \tag{3.1}$$

$$z_{id}^{colour} = rgb2hsv(Ibg_{id}) \tag{3.2}$$

$$z_{id}^{flow} = rgb2flow(I_{id}) \tag{3.3}$$

The pre-processed images are available at both stages since the on line target detection module only requires the background segmented image, whereas the tracker requires the pre-processed

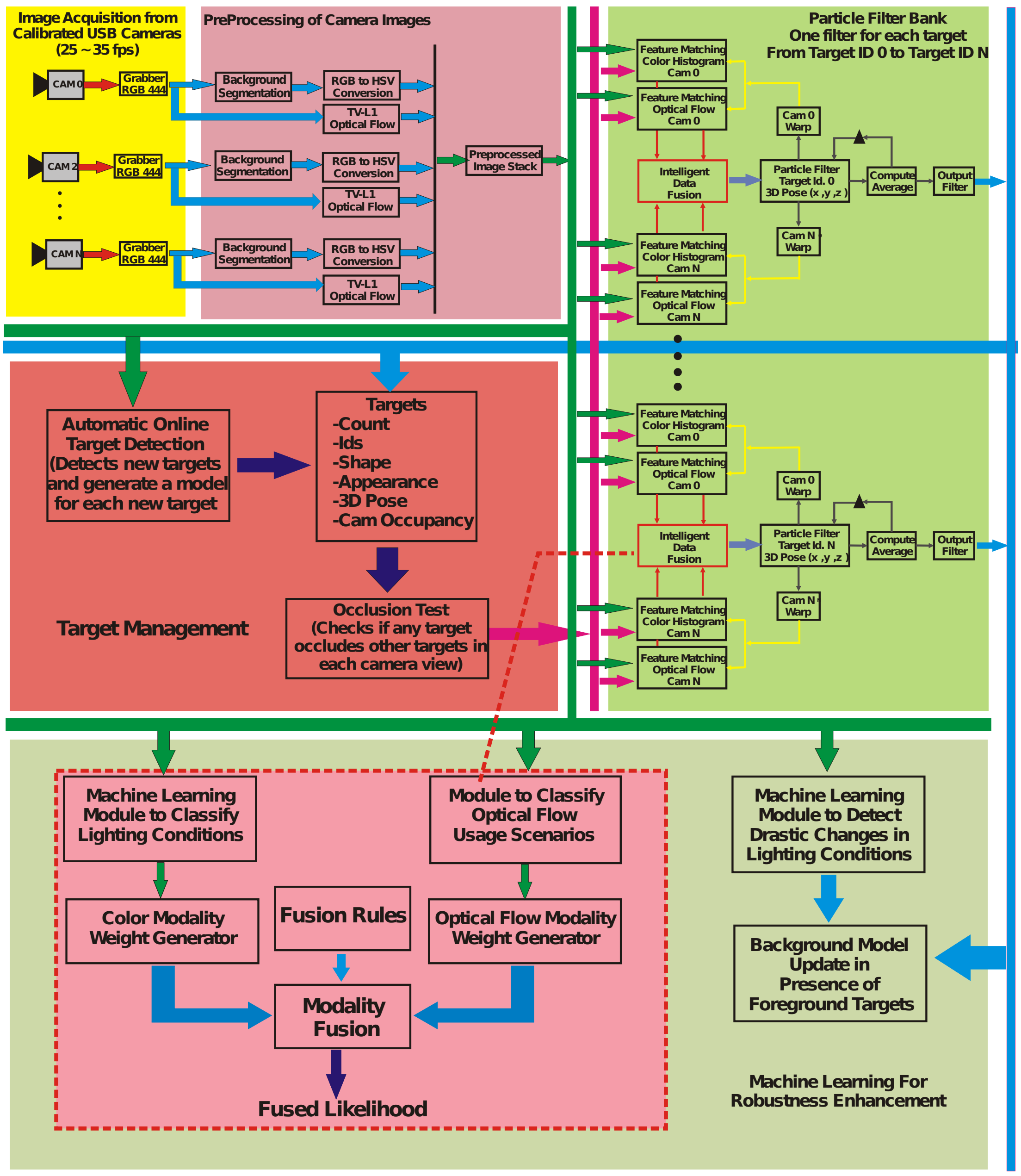


Image Acquisition from Calibrated USB Cameras (25 ~ 35 fps)
CAM 0
Grabber RGB 444
CAM 2
Grabber RGB 444
CAM N
Grabber RGB 444
PreProcessing of Camera Images
Background Segmentation
RGB to HSV Conversion
TV-L1 Optical Flow
Background Segmentation
RGB to HSV Conversion
TV-L1 Optical Flow
Background Segmentation
RGB to HSV Conversion
TV-L1 Optical Flow
Preprocessed Image Stack
Particle Filter Bank
One filter for each target
From Target ID 0 to Target ID N
Feature Matching Color Histogram Cam 0
Feature Matching Optical Flow Cam 0
Intelligent Data Fusion
Feature Matching Color Histogram Cam N
Feature Matching Optical Flow Cam N
Cam 0 Warp
Particle Filter Target Id. 0 3D Pose (x ,y ,z)
Compute Average
Output Filter
Cam N Warp
Feature Matching Color Histogram Cam 0
Feature Matching Optical Flow Cam 0
Intelligent Data Fusion
Feature Matching Color Histogram Cam N
Feature Matching Optical Flow Cam N
Cam 0 Warp
Particle Filter Target Id. N 3D Pose (x ,y ,z)
Compute Average
Output Filter
Cam N Warp
Automatic Online Target Detection (Detects new targets and generate a model for each new target
Targets -Count -Ids -Shape -Appearance -3D Pose -Cam Occupancy
Occlusion Test (Checks if any target occludes other targets in each camera view)
Target Management
Machine Learning Module to Classify Lighting Conditions
Module to Classify Optical Flow Usage Scenarios
Color Modality Weight Generator
Fusion Rules
Optical Flow Modality Weight Generator
Modality Fusion
Fused Likelihood
Machine Learning Module to Detect Drastic Changes in Lighting Conditions
Background Model Update in Presence of Foreground Targets
Machine Learning For Robustness Enhancement

Figure 3.5: The figure illustrates the block diagram of the multiple tracking system. It consists of the image acquisition module, target detection module, occlusion handling module, the tracker based on a particle filter bank and machine learning based multi-modal fusion and background update module.

image obtained after both stages are performed. The optical flow processing is performed on the raw sensor image I_{id}.

3.5.3 On Line Target Detection

This module automatically detects targets when they enter the tracking area by performing a scan along the tracking floor area using the target model. At each location in the scan, the probability of a possible target occupancy is computed using the background segmented image. The number of foreground pixels are computed within the 2D region obtained by warping the 3D pose of the target model on to the respective camera images. Regions occupied by existing targets are excluded from the scan. If foreground occupancy observed in each camera view is beyond the desired threshold, a target is registered with an initial 3D pose corresponding to the scan location. The threshold has to be defined by the user. In most cases a threshold of 0.7 (i.e 70%) is well suited. However, this threshold should be tuned depending on the quality of the background subtraction module since sometimes the foreground segmentation contains holes due to shadows resulting in lowered occupancy. Fig. 3.6 illustrates the process. The green boundary is the detection area and the blue boundary is the tracking area. The green dots show the floor area scanned and the rectangular boxes show the successful detection of targets as a result of the occupancy analysis. The left image show an empty scene and the scanned locations. The image on the right shows two targets successfully detected in the scene.

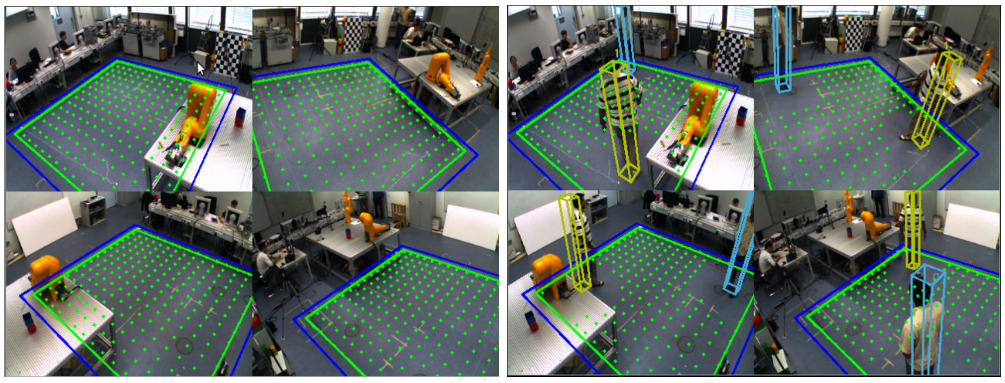


Figure 3.6: On-line Target detection by scanning the tracking volume and computing the occupancy map of the targets.

An occlusion test is performed at the target location in order to identify the cameras in which the target is completely visible. Using this information the shape and appearance model of the target is generated. The shape consist of a 3D rectangular cube with dimension of a normal sized human ($0.2m \times 0.3m \times 1.8m$) which can be modified if required. The appearance model consists of 2D histograms of the targets in the HSV colour space. Only cameras in which the target is visible are selected. If the target is occluded in a camera view, the appearance model generation in that view is postponed until the target is visible again in that camera view. In order to register a target, it is required to be visible in at least 2 camera views. The target data consists of namely: 1. unique target ID, 2. initial 3D pose, 3. shape data 4. appearance data 5. occlusion test data, 6. current 3D pose and 7. velocity.

3.5.4 Occlusion Testing

This module is very important to ensure robust tracking of multiple humans. It determines if a target is occluded by other targets in a particular camera view. This information is essential during target detection and tracking since 2D regions in each camera view are used by the detection module to compute the probability of target occupancy and by the tracker for the likelihood estimations. These regions are obtained by warping the 3D pose of each target under consideration on to each camera image (id= 0,....,M). Each target is defined by a 3-dimensional container box comprised by 8 vertices

$$V_n(t) = \left\{ v_j \in \mathbb{R}^3 \mid j = 0, 1, ..., 7 \right\} \tag{3.4}$$

where, v_j is the j^{th} vertex of target shape model defined in Cartesian space for the state $s(t)$. These vertices are projected on each camera as follows:

$$S_n(t) = \left\{ r_j \in \mathbb{R}^2 \mid r_j = K[R \mid T]\, v_j,\ v_j \in V_n(t) \right\} \tag{3.5}$$

where, $S_n(t)$ is a set of the projected vertices of the target n. K, R, and T describe the camera model.

Then, we define $d_n(t)$ as the Oriented Bounding Box (OBB) of $S_n(t)$.

$$l_n(t) = \left\{ (x, y) \in \mathbb{R}^2 \mid (x, y) \in d_n(t) \right\} \tag{3.6}$$

The geometric meaning of $l_n(t)$ is all the pixels from the area of the OBB $d_n(t)$. When a target occludes other targets in a certain camera view, the bounding boxes overlap making the appearance data visible only for the un-occluded targets. Under such situations, the information from these 2D regions should not be sampled for the occluded targets. The occlusion test detects such situations by checking the occupant percentage of the bounding box of each target. Lack of this information can result in false tracking of targets, since when a target is occluded in a camera view its bounding box will contain appearance information of the target which it is occluded by. If this information is used, it will return a low likelihood, although the hypothesis was optimal.

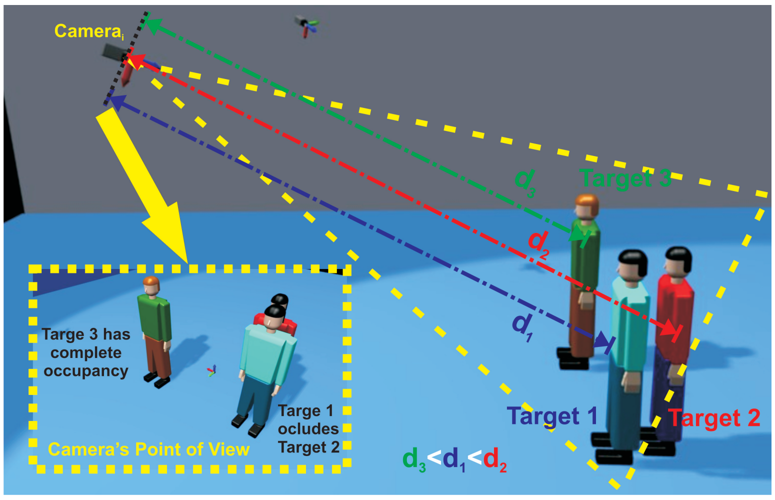


Figure 3.7: The figure illustrates the occlusion test system. The left part of the figure shows a scene in a camera view with 3 Targets, where Target 1 occludes Target 2. The right part of the figure shows how the occlusion is detected by rendering the targets with respect to their distance from the camera. The target closest to the camera is rendered first.

Fig. 3.7 illustrates the *Occlusion Test System*. This system considers all the targets and computes their occupancies in each camera image. For each camera view the euclidean distance from the camera to each target is computed. The target farthest from the camera is rendered first on the camera image. This is followed by the remaining targets, with the closest target rendered last. Once all targets are rendered an overlap test is conducted between the rendered regions. If a target occludes another target it will overlap the rendered region of that particular target. For any target, if the rendered region visible is above a the visibility threshold, it is considered to have a good occupancy in the current view of this particular camera. The visibility threshold is set by the user. This threshold can be initially set by manual observation of human motions in the camera views. If the number of targets are large compared to the tracking area, the threshold can be lowered due to possibilities of multiple occlusions. The default value chosen is 70% of the warped 3D region and can be tuned by the user until the optimal value is obtained. Thereby, the tracker considers only the camera views in which the target is not occluded to be used for the likelihood test. For a target to be tracked successfully, it is required to be visible in at least in 2 or more camera views in order to estimate its 3D pose. Chapter 5 validates the importance of the occlusion test module through experiments.

3.5.5 Tracker

The tracker performs the primary task of keeping track of all the targets in real-time, once they have been registered by the target detection system. In order to achieve this, the tracker uses a bank of Sampling-Importance-Resampling based Particle filters [103]. Each target is equipped with its own particle filter. Each filter uses a motion model [2, 10] and a 3D translation state. The visual modalities used are 2D colour histograms and optical flow. The likelihood estimation is performed by distance computation of each modality with respect to the reference and thereafter performing multi-modal fusion. For each hypothesis, the likelihood is computed for each camera view, in turn computing an average likelihood. Camera views in which targets are not visible are dropped during the likelihood computation for the respective targets through the occlusion test.

Particle filters are computationally expensive and hence in order to obtain real-time performance from a bank of particle filters, a common global particle count is maintained which is distributed evenly among the filters. This distribution depends on the number of targets. When a target is added or removed from the target list, the number of particles allocated to each filter is updated. Hence, if N_p is the global particle count and n_p is the number of particles allocated to each filter, then

$$n_p = \frac{N_p}{N}, \tag{3.7}$$

where, N is the number of targets. This approach is well suited since, when the number of targets increase within the tracking area, their mobility reduces and hence the number of particles needed to track them can be reduced. The following sub-sections provide detailed descriptions of the functioning of the particle filter.

- Tracker Prediction: The particle filter generates several prior state hypotheses s_t^i from the previous distribution $(s^i, w^i)_{t-1}$ through a prediction model, which is more precisely called as the motion model. Different motion models exist which define how the target state evolves in time. These models vary depending on the dynamics of the target. For example, the motion model for a car will be different from the motion model for a human, which in turn will vary from the motion model describing a pendulum. The Brownian, CWNA and oscillatory motion model are the most common motion models used in computer vision [36]. In this system the constant velocity model was tested. The Brownian model is given by,

$$s_t^i = s_{t-1}^i + v_t^i, \tag{3.8}$$

with v a zero-mean Gaussian white noise of pre-defined covariance in the (x, y, z) state variables. The motion model can be controlled during the course of tracking by learning the motion of the target. Deterministic re-sampling strategy over the previous weights w_{t-1}^i is also employed.

On the other hand, the constant velocity model is represented as follows,

$$s_t^i = s_{t-1}^i + \dot{s}_{t-1}^i \tau_i + \frac{1}{2} v_t^i \tau_t^2 \tag{3.9}$$

where, $\dot{s}_{t-1}^i$ is the velocity and its constant over time, plus a random acceleration v_t^i. τ is the sampling interval [36].

For each generated hypothesis, the tracker asks for computation of the likelihood values $P(z^{col}, z^{flow}|s^i)_n$ after projecting every hypothesis on to each camera image. Since, the system also uses optical flow as a visual modality, information regarding the target velocity is essential. Therefore, when the optical flow modality is used within the particle filter framework, the motion model used is $CWNA$.

- Likelihood Computation:

  The global likelihood is computed by fusing the distance measure obtained from the colour histogram and optical flow modalities in each camera view.

  The object model defining the target's shape is projected on the pre-processed image of each camera at the predicted hypothesis s_t^i using the intrinsic and extrinsic parameters of the respective cameras. The underlying H and S colour pixels are collected in the respective 2D histogram $q\left(s_t^i\right)$, that is compared with the reference q^* through the Bhattacharyya coefficient [112, 51],

$$B_m\left(q_i\left(s\right), q_i^*\right) = \left[1 - \sum_N \sqrt{q_i^*\left(n\right) q_i\left(s, n\right)}\right]^{\frac{1}{2}}, \tag{3.10}$$

where the sum is performed over the $(bin \times bin)$ histogram bins (in the current implementation, $bin = 10$). The computation is done for each camera. B_m is the colour likelihood (not under a Gaussian model).

For the optical flow modality the estimated velocity of the target component is used as a reference. The directional component of this vector is compared with the optical flow vectors obtained from the pre-processing stage. The flow vectors are obtained in the image space. Therefore, the velocity vector of the targets is required to be projected on the image plane of the respective cameras. Thereafter, this projected vector is compared with each pixel flow vector within the warped 2D region of the hypothesis of the particle filter similar to the process in the case of the colour histogram modality. A distance measure is computed by comparing each flow vector $f\left(s_t^i\right)$ within the 2D region with the projected reference velocity vector f^* as follows,

$$F_m\left(f_i\left(s\right), f_i^*\right) = \left[1 - \sum_N \sqrt{f_i^*\left(n\right) f_i\left(s, n\right)}\right]^{\frac{1}{2}} \tag{3.11}$$

F_m is the resulting optical flow likelihood (not under a Gaussian model) for an individual hypothesis.

Once the colour histogram and optical flow likelihoods are estimated for an individual hypothesis, they are fused to obtain a global likelihood. The fusion depends on different factors which define the proportion in which the two modalities should contribute to the global likelihood. The intelligent multi-modal module, which will be described in Chapter 4, generates the weights W_{col} and W_{flow}. The global likelihood for the hypothesis s_t^i is then given by

$$P(z^{fused}|s_t^i) = B_m(z^{col}|s_t^i)W_{col} + F_m(z^{flow}|s_t^i)W_{flow} \tag{3.12}$$

Thereafter, the global likelihood for the hypothesis s_t^i is evaluated under a Gaussian model in the overall residual

$$P(z^{global}|s_t^i) \propto exp(-\sum_M \log\left(P(z^{fused}|s_t^i)_m^2/\lambda\right)) \tag{3.13}$$

with given covariance λ and can be tuned by the user depending on the quality of matching required for convergence.

- Hypothesis Penalty to Improve Occlusion Handling: Tracking multiple people in a small workspace is not trivial due to high probabilities of multiple occlusion of targets in multiple cameras. The situation becomes more complicated if two or more targets have similar appearance and the targets are moving at a short distance with respect to each other. For example, tracking 4 people in a room of size 5×5 meters becomes difficult when the targets are moving close to each other.

  In order to handle such situations, the particle filter is equipped with a hypothesis likelihood penalty function. The primary objective of this function is to penalize any hypothesis of a particular target, if another target is present at that location or at a short distance D_h specified as a parameter by the user. The penalty function operates in two modes, hard penalty mode and linear penalty mode. In the hard penalty mode, if the distance between the generated hypothesis and any other target is less that D_h, the particle filter immediately sets the likelihood for that hypothesis to zero. This ensures that the in the next prediction step, the probability of an hypothesis generated in this region will be lowered due to the re-sampling. On the other hand, in the linear penalty mode the particle filter computes the likelihood for the hypothesis and thereafter penalizes it by a factor inversely proportional to the minimum distance from each other target. The following equation describes the penalty function.

$$H_{pf} = \frac{d_{min}}{F} \tag{3.14}$$

  where, $F \neq 0$ is the constant factor chosen by the user, describing the strength of the penalty, d_{min} is the minimum distance of the hypothesis with respect to all other targets and H_{pf} is the resulting penalty factor. These two parameters are chosen by the user through manual observations of the target motions. For eg. in situations where there are many targets in a relatively small area, d_{min} can be lowered and vice versa. Selection of F depends on the degree of impact required. The resulting likelihood after applying the penalty is as follow.

$$P(z^{global_{pf}}|s_t^i) = P(z^{global}|s_t^i) \times H_{pf}|s_t^i \tag{3.15}$$

- Computing the Estimated State:

The average state $\overline{s}_t$

$$\overline{s}_t = \frac{1}{N}\sum_i w_t^i s_t^i \tag{3.16}$$

is computed and the three components $(\overline{x}, \overline{y}, \overline{z})$ are returned. In order to reduce the jitter in the output, the average pose can be smoothed using an exponential filter.

- Refining the Motion Model: The covariance matrix defines the distribution of the hypothesis during the prediction phase. It is set as a parameter during the initialization phase with respect to some prior knowledge regarding the motion expected within the tracking area. However, the velocity of the targets can vary between a certain higher and lower limit. In order to improve the performance of the tracker the motion model is tuned online to the current motion behaviour of the target. The knowledge obtained from the estimated velocity of a target can be used as a cue in order to tune the parameters in the covariance matrix such that the prediction model adapts to the target's motion pattern. For example, if initially the target is moving with a low velocity, the covariance can be set in order to obtain a compact distribution around the average pose of the target. If the target velocity increases, the covariance parameters can be increased for a wider spatial spread of the hypothesis. This allows the tracker to successfully track the target under changing velocities.

3.6 Graphical User Interface

It is a high level graphical user interface (GUI) allowing the user to monitor and control the complete system. It allows the user to perform basic tasks such as start, stop and standby. It provides essential debug information in the form of visual feedback of the sensor images during the different levels of processing. It also allows the user to log the different parameters of the system for further debugging. The GUI provides information about the current state of the system along with the debug data associated with that particular state. The user can configure the different modules of the system with possibilities of modifying some parameters online. The configuration files for the cameras, detector and tracker can be generated using the GUI.

The *GUI* is designed using the *QT* library [113]. It mainly performs the task of sending information from the user to the system and providing the necessary debug information to the user. It can be easily detached from the system if not required. Fig. 3.8 illustrates the graphical user interface used as a front end tool to control the system.

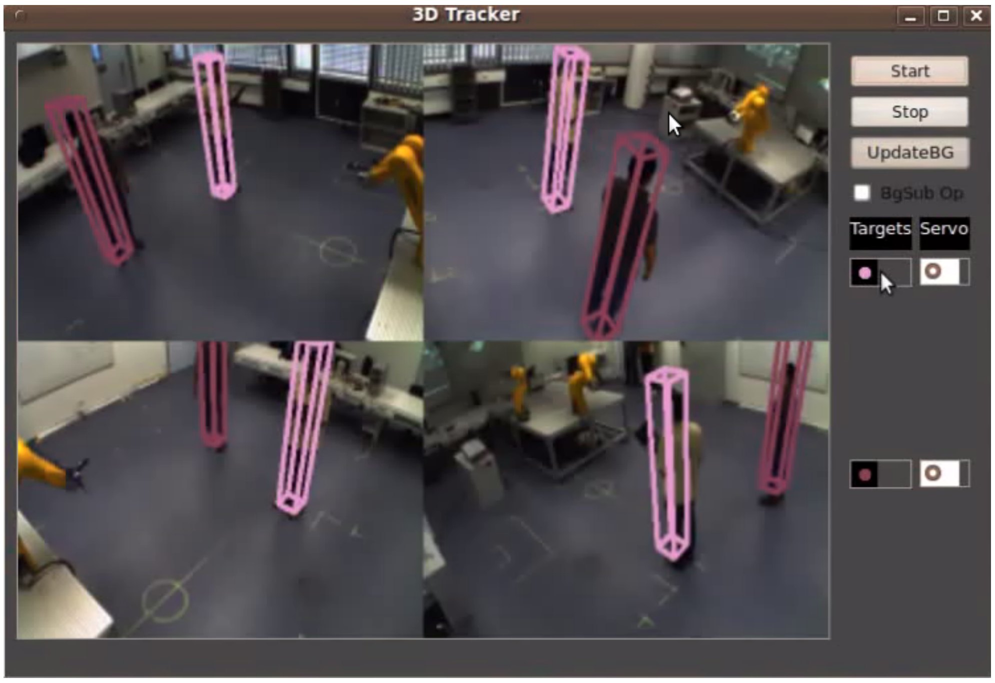


Figure 3.8: Graphical user interface for controlling the system and visualizing the tracking results. It represents the various control options and the views from all cameras.

3.7 Communication Interface

The communication interface is an important tool using which the multiple human tracker communicates with external applications. This interface helps in the development of complete applications using the tracker.

Fig. 3.9 illustrates the building block of the communication interface module. It is a general purpose system based on TCP/IP sockets [114]. The multiple human tracking system contains a socket server operating on a specific port. Multiple external clients can connect to this server and retrieve the required information from the tracker. This information is mainly related to the targets being tracked but can also consist of the state of the system. The clients can also send control information to the tracker. For e.g. if the tracker functions are required to be controlled remotely by an external GUI.

The protocol is implemented in a manner such that information regarding different aspect of the target motion can be transferred to the clients in a seamless way. The protocol starts with the client sending a command ID. On receiving this information, the server transfers it to the function mapper. This module performs the task of calling the right function depending

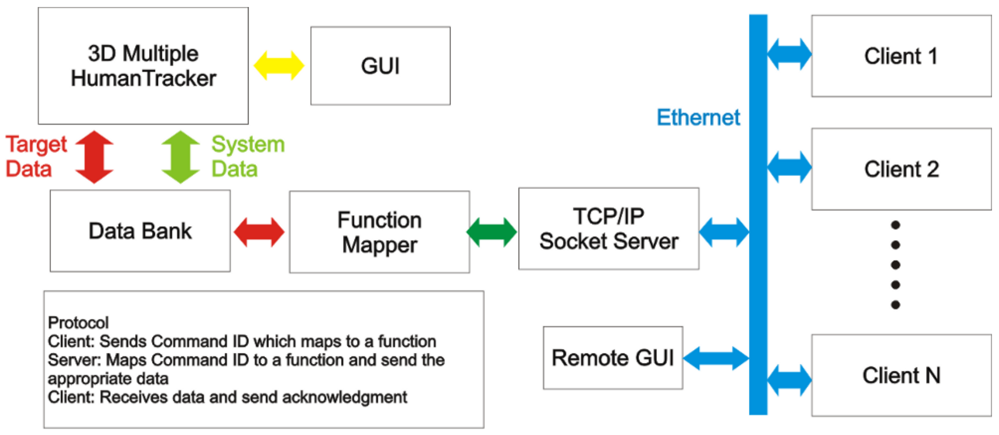

3D Multiple HumanTracker
GUI
Target Data
System Data
Data Bank
Function Mapper
TCP/IP Socket Server
Ethernet
Client 1
Client 2
Client N
Remote GUI
Protocol
Client: Sends Command ID which maps to a function
Server: Maps Command ID to a function and send the appropriate data
Client: Receives data and send acknowledgment

Figure 3.9: Communication Interface using TCP/IP.

on the command ID received. The corresponding function packs the necessary data from the data bank which stores all the necessary information regarding all targets and the system state. The packed data is sent to the client through the socket server. The data generated depends on the function. This can be the complete information of a single target, specific targets or all targets. It can also be a certain specific data concerning a single target for e.g. id, position, velocity etc.

In this manner, multiple external system can exchange data easily with the multiple human tracking system operating in a stand alone manner. The communication medium used is Ethernet [115]. Wireless data transfer is also possible through WiFi. In systems where Ethernet latency has to be kept low, gigabit Ethernet can be used.

3.8 Discussion

This chapter provided a detailed insight into the 3D multiple human tracking system in terms of its system architecture and tracking pipeline. The next chapter will introduce the novel approach of robustness enhancement through machine learning. It will provide a detailed discussion of the intelligent multi-modal fusion module, detection of drastic changes in lighting conditions and update of the background model in presence of fore ground targets under such scenarios.

Chapter 4

Machine Learning to Enhance Tracking Robustness

4.1 Overview

Chapter 3 provided a detailed analysis of the multiple human tracking system. The system uses a multi-modal fusion of two visual modalities namely, colour statistics and optical flow. This chapter introduces an approach to improve the robustness of the system using machine learning techniques [116].

As discussed before, multi-modal fusion of visual modalities has its advantages and disadvantages. The modalities used can assist each other to enable robust tracking. However, each modality has its own limitations and can fail in certain tracking scenarios. In such circumstances the modality contributes in a negative way to the overall tracking estimation, thereby increasing the probability of error and a possible target loss. In the next sections of this thesis, the effect of varying lighting conditions on the colour modality and background subtraction are discussed. Thereafter, the variations in the optical flow information with target motion behaviour is discussed. This is followed by a novel approach based on machine learning to overcome the effect of varying lighting conditions on the performance of the system. Further, scenarios which affect the use of optical flow as a modality are introduced. Finally, the intelligent mutli-modal fusion module is discussed.

4.2 Colour Statistics Quality under Changing Lighting Conditions

Colour statistics, in the form of joint probability 2D colour histograms [36, 10], are well suited as a visual modality under static lighting conditions [36, 10]. However, if the lighting conditions change, the performance of the colour modality is affected. In theory, the HSV colour space separates colour and intensity information, but in the real-world there is a shift in the colour distribution when the intensity changes beyond a certain extent. This mainly occurs due to the non-linear behaviour of the camera sensor noise. Therefore, distribution of colour histograms of the same regions of an image vary with lighting conditions.

The performance degradation of the colour modality depends on the amount of change in lighting conditions. Up to a certain limit, the colour modality, although affected by lighting changes, continues to provide good estimates for the likelihood function, although with a lower degree of accuracy. However, when there are drastic changes in the lighting conditions the colour modality could fail, thereby affecting the performance of the system. This situation usually occurs when the light distribution in the tracking scene changes to very dark, due to insufficient light sources, or gets saturated due to excessive lighting.

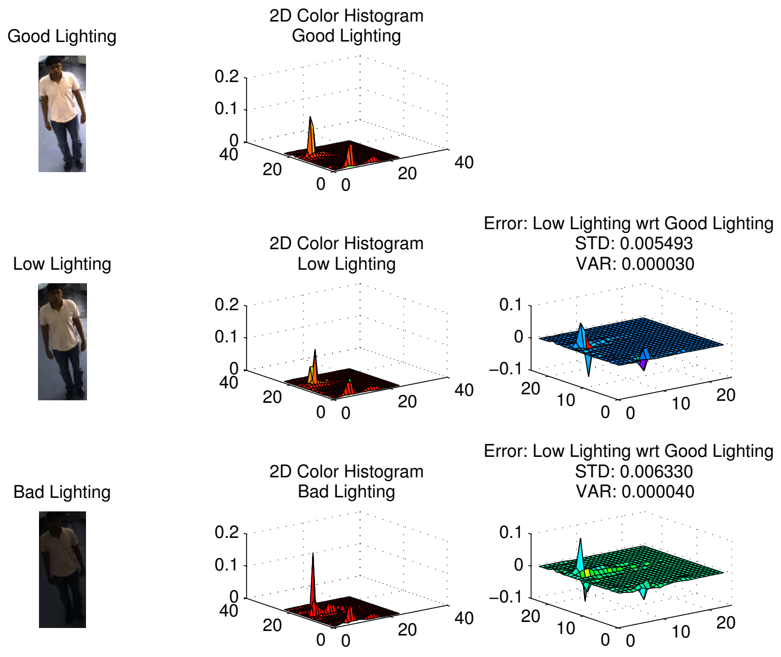


Figure 4.1: Effect of changing lighting conditions on colour statistics of the appearance of the human model.

Fig. 4.1 demonstrates the effect of light change on the appearance model of the human. Three different lighting conditions are presented, where the first one is the reference under which the target is modelled for his or her appearance. The reference case is generated under good lighting conditions, while in the following two cases the intensity of light is lowered in two steps. The 2D histograms under each case are plotted as shown. The histograms of the low intensity lighting are compared with that of the reference and the error is plotted. It can be observed that as the intensity of light is lowered, the error between the histograms of the same region increases. Similar behaviour can be proved when the intensity of light is increased until saturation.

4.3 Background Subtraction Quality under Changing Lighting Conditions

Background subtraction is an important pre-processing method in many visual tracking and motion capture systems [117, 118]. Various background subtraction techniques exist based on intensity, colour, edges etc. The most widely used techniques rely on image intensity information. Most background subtraction techniques rely on a background model which is generated or trained during the initialisation phase of the system [110].

Background models are sensitive to changing light. Drastic changes in lighting conditions affect the background model, thereby hampering the processing of the camera images for background segmentation. There are systems which deal with this problem by periodically updating the background model. This approach is trivial for empty scenes but fails in the presence of foreground targets. An update under such circumstances will include the targets into the background model resulting in tracking loss. There are other approaches such as [119], which make the background subtraction algorithm robust to light changes but perform well only up to a certain level of light change. [120] presents a background subtraction technique that is robust towards drastic changes in lighting, but its computationally intensive and does not guarantee real-time performance. There are fast background subtraction systems utilizing the *GPU* but lack the required robustness under drastic change in lighting [121, 122].

4.4 Optical Flow Quality depending on Target Motion Behaviour

Optical flow provides information of regions with motion activity within the scene. The motion vectors provide scale and direction information which is used to perform matching with the motion vector of the projected hypotheses. Although this is useful information, under certain circumstances it could lead to ambiguity.

The quality of information from the optical flow processing, used to track multiple humans in a limited workspace, depends on the motion behaviour of the targets. A static target provides no information for the optical flow processing and can cause the likelihood matching function to fail since the hypotheses return the same residuals from the target region and background regions due to the lack of motion vectors. Under such circumstances the use of optical flow can be hazardous and can impact the performance of any complimenting visual modality. Fig. 4.2 illustrates this scenario. It can be observed that the optical flow information is available only for the target in motion, whereas the stationary target does not return any optical flow data.

Optical flow processing provides valuable information for likelihood computation when two or more targets are moving in different directions, especially at a short distance from each other. Under such situations, optical flow processing becomes a valuable tool in order to compliment the colour modality. It is useful when two targets having similar appearance are crossing close to each other. In this situation, if a hypothesis of one target falls on the image region occupied by another target, it will return a high colour likelihood since the targets are similar in appearance. The optical flow modality although, will produce a low likelihood due to the difference in the motion vectors of the two targets. The reason being that, the constant velocity motion model produces a major proportion of the hypothesis along the velocity component of the current state of the target. Fig. 4.3 illustrates this behaviour. It can be observed that even though the two targets have similar appearance , the information produced by the optical flow processing is distinct due to the different directions of motion.

In contrary to the above situation, when two targets with similar appearances are moving close to each other and in the same direction, the optical flow modality could be ambiguous since the targets have the same flow vectors. Fig. 4.4 illustrates this scenario. It can be observed that the targets have similar appearance and are moving in the same direction, thereby producing similar flow vectors.

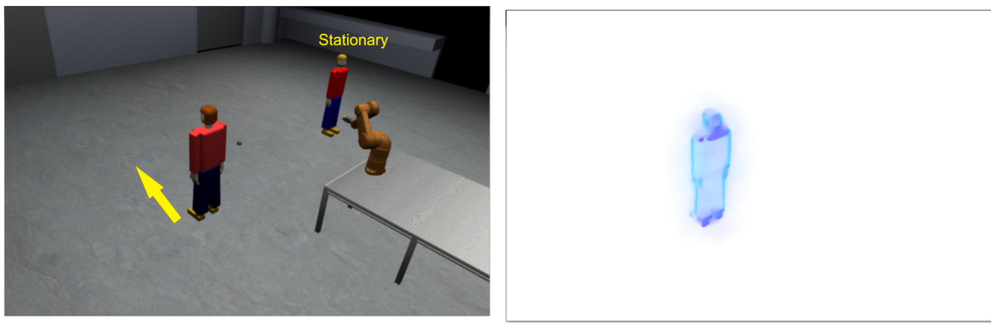


Figure 4.2: Optical flow processing results when one target is stationary and the other is moving

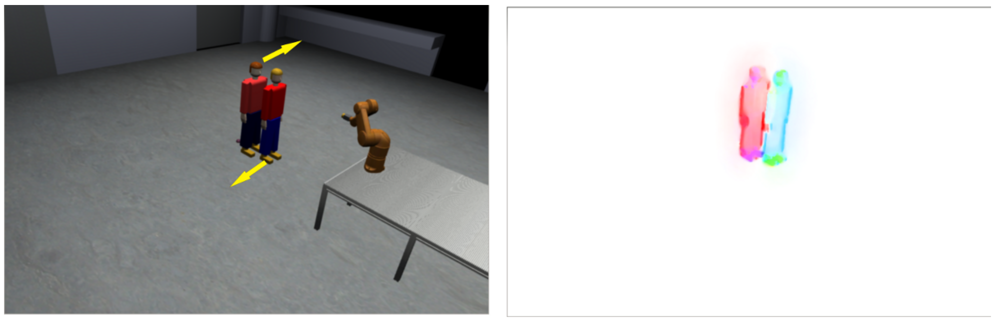


Figure 4.3: Optical flow processing results when the two targets are moving close to each other but in opposite directions

4.5 Machine Learning to enhance Visual Modality Performance

The performance of the system is optimal when the two visual modalities are fused in the right proportion depending on their information quality. The information quality depends on the tracking scenario. In this section, a supervised machine learning method is introduced in order to optimize the performance of the colour histogram modality under situations of sudden light changes causing performance degradation of this modality. Using machine learning a model is trained which allows to detect sudden changes in lighting conditions. With this information the contribution of the colour modality towards the global likelihood can be adjusted to obtain a optimal multi-modal fusion in the Bayesian tracking.

4.5.1 Supervised Machine Learning

It is a technique where a function defining a model is generated through a large set of supervised training data set. This model can be further used for classification of new inputs. The quality

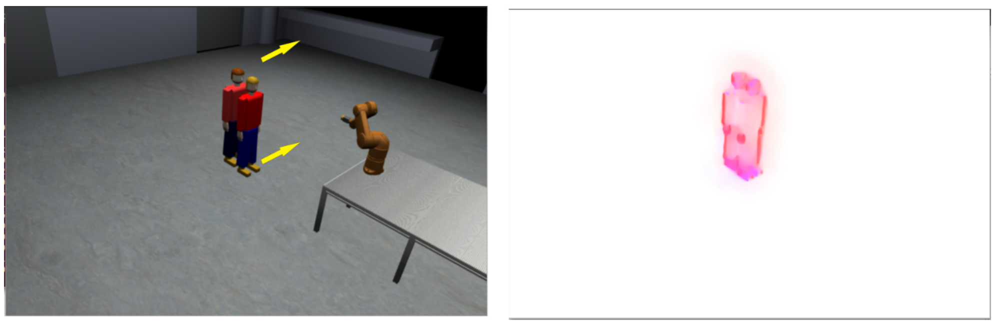


Figure 4.4: Optical flow processing results when the two targets are moving close to each other and in the same direction

of classification depends on the quality of the input data set. A popular supervised machine learning tool is a Support Vector Machine [4, 123, 124, 125, 126, 127, 128, 129]. It is used to analyse and identify patterns in data which can be further used for classification. In its standard form, the support vector machine takes a set of input data and predicts to which binary class it belongs. It is therefore a non-probabilistic binary linear classification engine.

4.5.1.1 General Formulation

Support vector machines work in two phases namely training and classification. In the training phase it uses a large set of training data which is labelled either automatically or manually. It builds up a model using this training data such that when a new training example comes in, it is able to place it into the right category. The model represents these examples as points in space such that different categories are divided by wide and clear gap. To be more precise, support vector machines build up hyperplanes of high dimensional space [4]. This space is used as a tool for classification of incoming data. Each data point is viewed as a p dimensional vector. A hyperplane which has the largest distance to the nearest training data is considered and is called as a functional margin. Larger the margin, lower is the error during classification. It uses a high dimensional space since it is observed that when problems are represented in a finite dimensional space, the sets which need to be discriminated are not linearly identifiable. A higher dimensional space makes this identification much simpler. Another reason is that, in a higher space cross products can be computed easily with respect to the variables in the original space, thereby reducing computational costs.

The cross products in the higher dimensional space is represented by a kernel function $K(x, y)$. The hyperplanes in the large space are represented as a set of points whose inner product with a vector in that space is constant. These vectors are chosen to be linear combinations of the parameters α_i of images of feature vectors. The points x in the feature space, mapped into the hyperplane are defined as:

$\sum_i \alpha_i K(x_i, x) = c$, where c is a constant [4]

As $K(x, y)$ reduces with increase in y further from x, elements in the sum measure the degree of closeness of the point x to its corresponding data base point x_i. Thereby, the sum of kernels can be used to measure the relative closeness of each test point to the data points.

A general formulation [4] of the training process can be given as follows:

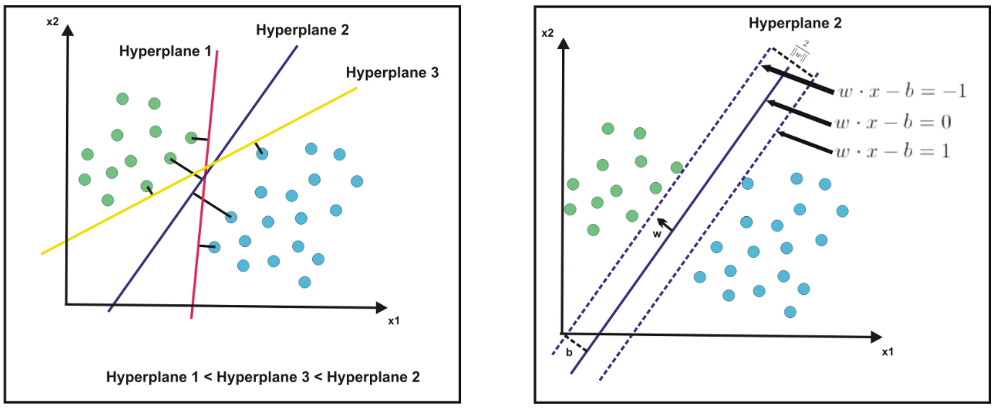


Figure 4.5: SVM Classifier: Left: Hyperplanes separating the two sets of data, Hyperplane 1 and 3 do not divide the data sets with a high margin as Hyperplane 2 does, Right: Hyperplane with the highest margin selected with thresholds defined for each class. The data points on the margin are called support vectors [4].

Considering a training data set D with n points,

$D = (x, y) \mid x_i \in \mathbb{R}^{'}, y_i \in \{-1, 1\}_{i=1}^{n}$

y_i represents the class to which the data point x_i belongs, i.e -1, 1.

The hyperplane with the maximum margin dividing the data points with $y_i = -1$ from the ones with $y_i = 1$ satisfies the following expression.

$w \cdot x - b = 0$ [4]

where, the vector w is a normal vector to the hyperplane and the parameter $b/\left|w\right|$ determines the offset of the hyperplane from the origin of the normal vector w. Here, w and b are chosen such that the distance between the parallel hyperplanes is maximum, defined as

$w \cdot x - b = -1$ and $w \cdot x - b = 1$ [4]. In order to avoid data points falling into the margins constrained are imposed to satisfy $w.x - b >= -1$ for x_i of the first class, or $w \cdot x - b <= 1$

for x_i [4] of the second class. Fig. 4.5 illustrates the possible hyperplanes and how the most optimum hyperplane is selected.

There are other variants of the forms of support vector machines such as primal form, dual form which are described in the literature [4].

4.5.1.2 Multi Class Support Vector Machines

In their general form, support vector machines provide a binary classification of data. Hence there are only two classes. However, the classification can be extended to data sets representing multiple classes. For this purpose, the multi class support vector machine is used. It works on the principal of reducing a multi-class problem into multiple binary classification problems. For each sub problem, a binary classifier is modelled, which provides higher values to the data points falling into the positive class and lower values to that belonging to the negative class.

4.5.2 Supervised Machine Learning to Model Lighting Conditions

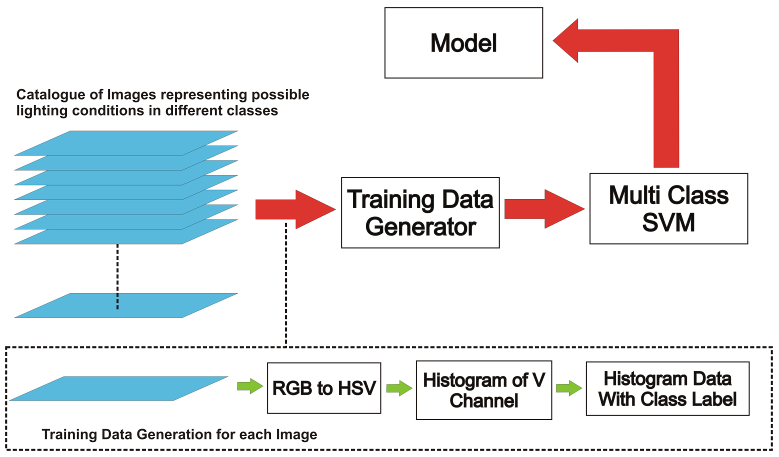

Model
Catalogue of Images representing possible lighting conditions in different classes
Training Data Generator
Multi Class SVM
RGB to HSV
Histogram of V Channel
Histogram Data With Class Label
Training Data Generation for each Image

Figure 4.6: Building blocks of the SVM trainer for lighting conditions

Fig. 4.6 illustrates the building blocks of the support vector machine based training module for lighting conditions. It consists of a large set of training samples in the form of images obtained from the cameras. The training samples are collected from real-scenes and simulated environments. The large set of images are generated for each individual class. There is no upper limit to the number of training samples. The quality of the model improves with the number of training samples available, provided they are grouped in the right classes. Each training sample is processed to obtain the training data. Once the training data is available, it is used by the

SVM training module to generate a model based on the classes in which the training data were grouped. The process of generating the training data from the training samples and the model generation process is discussed below followed by the real-time classification engine.

In this system the lighting conditions are modelled into three distinct classes representing its quality. The classes are:

1. Insufficient lighting: This class represents a wide range of scenarios in which the intensity of light available is below the requirements. The training data for this class consists of a wide variety of images in which the intensity of light is below a certain threshold.

2. Good lighting: The training data in this class consists of a large set of images in which the intensity of light is sufficient and homogeneously distributed over the entire image.

3. Saturated lighting: This class consists of different images in which the intensity of light is too high resulting in saturation of the entire or certain regions of the image.

Each of the distinct classes accommodate a large set of possible image samples satisfying its constraints. The grouping of these samples into their respective classes is carried out by analysing the histogram of the intensity channel. This process is carried out in three stages as show in fig. 4.9.

1. RGB to HSV Colour Space Conversion: In this step the input image is converted from the RGB colour space to the HSV colour space. HSV colour space [3] is preferred over RGB for intensity analysis because, in the HSV colour space the colour and intensity information are represented independent of each other. The H and the S channels provide the hue and saturation of the colour while the V channels represents the intensity. Thereby, analysis of the intensity distribution in the image become simpler. This transformation can be expressed as follows: $I_{hsv} = rgb2hsv(I_{rgb})$

   where, I_{rgb} and I_{hsv} represents the input image sample in the RGB and HSV colour space.

2. Histogram Computation: Once the input image sample is transformed to the required colour space, a N bin histogram of the V channel is computed representing the intensity distribution. The number of bins are usually 256, but can be reduced to a lower value in order to have a more compact representation. This operation can be represented as follows:

$n = \sum_i^N H_i$

where, n is the total number of image pixels and H is the histogram. The histogram is further normalized using n.

$H^N_{i_{norm}} = \sum_i^N \frac{H_i}{n}$.

3. Labelling: Once the normalized N bin histogram is ready, with automatic analysis or manual observation a class label is generated for the sample. The class label together with the histogram data forms one training data sample for the multi-class support vector machine.

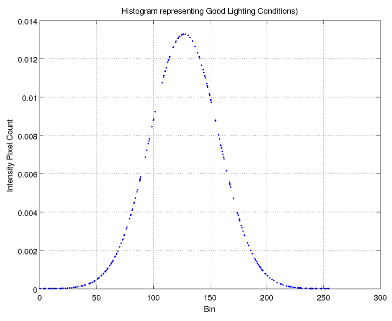


Figure 4.7: The figure illustrates good lighting conditions by observing the histograms of the intensity channel.

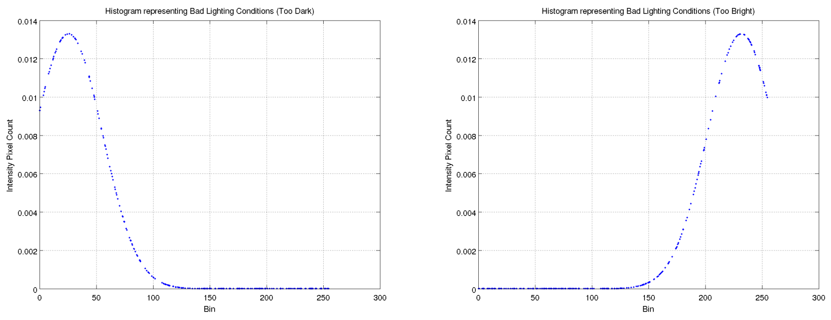


Figure 4.8: The figure illustrates bad lighting conditions by observing the histograms of the intensity channel.

Fig. 4.7 and 4.8 show examples of histograms belonging to the three classes. In fig. 4.7, the histogram is evenly distributed around the intensity value of 128 which is half of the white point saturation and is considered to be a good region for illumination. Similarly, histograms having similar distributions within a region around this value is considered to belong to class of good lighting. The span of this region is defined by a threshold generally obtained from manual observations.

Fig. 4.8 illustrates the histogram distributions for bad lighting conditions. The left part shows an example of the histogram for insufficient light where most of the histogram is distributed in the near regions of the intensity values. Under certain circumstances, the scene is under lit with dark regions. The right part shows an example histogram under saturated lighting conditions. In this situation, due to excessive intensity of light, the histogram is concentrated in the far region of the intensity values. Similar to the selection of histograms for good lighting conditions, the regions which accommodate insufficient lighting and saturated lighting conditions are determined by manual observations.

The training data set should include all possible scenarios of lighting conditions belonging to each class. This makes the training data set very large, with no upper limit. The larger the training data, greater is the accuracy of the trained model. In this system, initially around 4000 images of each class were used to generate the training data. This makes the total training data set to consist of 12000 data samples. These samples were generated using camera images obtained from the real scene and also from 3D simulations of the entire scene where the lighting conditions can be controlled. The 3D virtual environment used for this purpose will be discussed in greater detail in the chapters to proceed.

4.5.3 Classification of Lighting Conditions in Tracking Environments

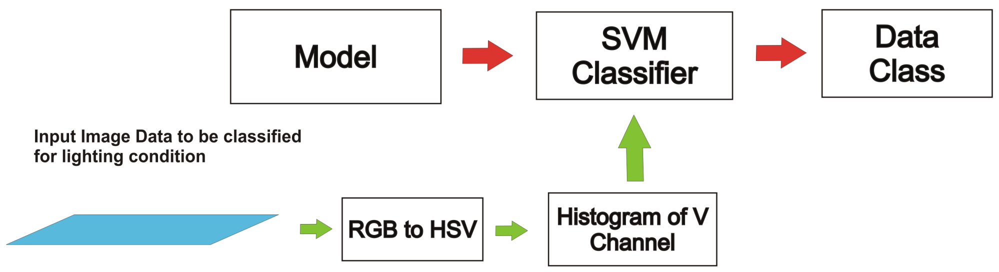


Figure 4.9: Building blocks of the SVM classifier for a lighting conditions

Fig. 4.6 illustrates the building blocks of the support vector machine based lighting conditions classifier which uses the model generated by the SVM trainer described in Section. 4.5.2. The multi class support vector machine generates the learnt model using the training data set.

This model is further used to classify incoming images for their lighting conditions. Similar to the training process, input data is generated for the support vector machine classifier. This data consists of the intensity histograms of the input image. It is the input to the classifier which computes the class to which the lighting conditions of the image belongs.

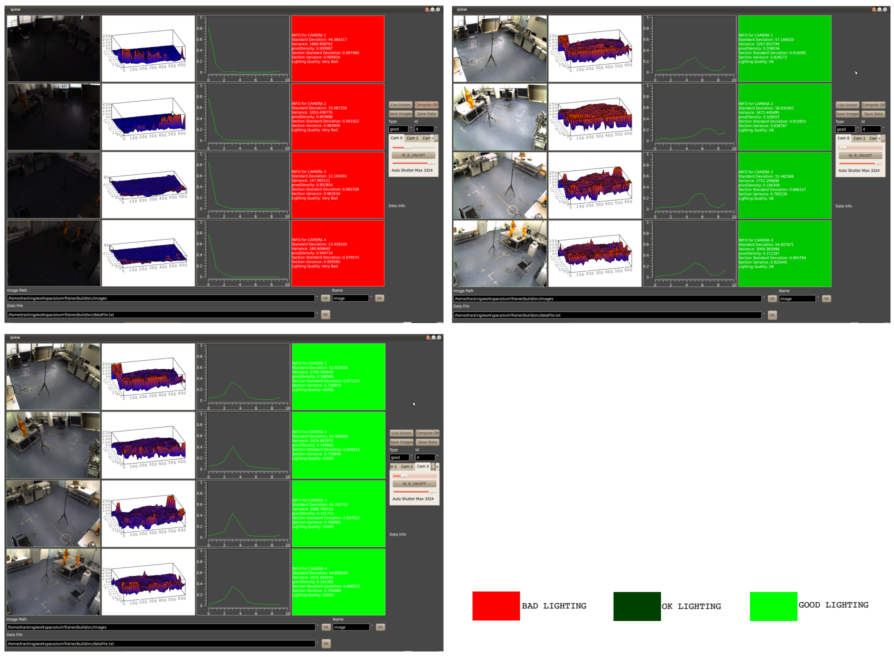


Figure 4.10: On-line classification of the lighting condition is obtained for each camera view using a trained model. The first column in each image represents images from each camera, the second column shows the intensity distribution, the third column illustrates the intensity histogram and the fourth column provides the classification results from the SVM classifier.

Fig. 4.10 illustrates the test conducted for the on-line classification of lighting conditions. The model is able to classify and associate the current lighting conditions in the camera views to their respective classes. The experiments and demonstrations described in Chapters. 5 and 6 have a SVM based lighting conditions trained for lighting conditions of classes *Bad* and *Good*. However, in this experiment the SVM model was trained for three classes of the lighting conditions in order to validate the multi-class classification. The classes are *Bad*, *OK* and *Good*, which can be further extended depending on the requirements. The first image shows a very dark view in each camera, which is successfully recognized by the classifier as bad lighting. The second image shows an improved set of lighting conditions. However, there exists some specular reflections, creating bright stops. The classifier identifies this as OK lighting conditions. The third image shows better lighting with respect to the other images and is classified as Good lighting conditions. The results show the ability of the classifier to successfully classify lighting conditions into their respective classes. With larger training sets, it is possible to achieve better classification covering more classes.

4.5.4 Background Model Update in the Presence of Foreground Targets

In section 4.1, issues associated with background segmentation, when the lighting conditions undergo drastic changes, were introduced. Background subtraction, being an important part of the system, should be robust to changes in lighting conditions. Section 4.5.2 covered the process of successfully detecting drastic changes in lighting conditions, using a trained classifier based on support vector machines. However, only detecting the light change is not sufficient. Updating the background model is required, since when the lighting conditions change, the background in a given scene changes. The background model update is trivial in case of an empty scene, but it becomes a complex task when there are foreground targets in the scene being tracked. Under such circumstances, only regions which do not accommodate any fore ground targets can be updated instantly. However, the foreground target regions need to be updated in an intelligent manner, such that the information from the target reagions are not sampled to be included into the updated background model.

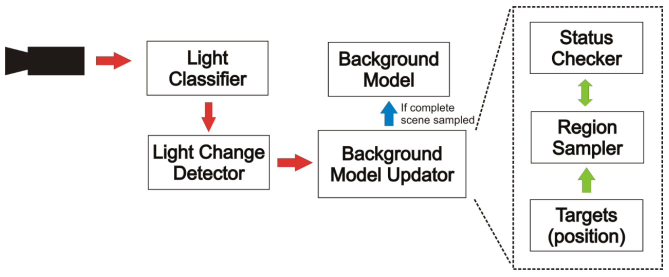


Figure 4.11: Background model update in presence of foreground targets.

This section proposes an approach to update the background model under changing lighting conditions while foreground targets are present in the scene. It exploits the fact that during the course of tracking the targets will move and expose the regions previously occluded by them. The occluded regions can be included into the background model once they are visible due to target motion. The assumption that the target will move is valid because, if the they do not move, then the tracker only needs to perform an extremely small local search to keep track of the target which does not require information from the background subtracter. Fig. 4.11 illustrates the Background Model Update Procedure (BMUP) under changing lighting conditions and in the presence of foreground targets. The BMUP comprises of three main parts:

Fig. 4.11 illustrates the background model update procedure under changing lighting conditions and in the presence of foreground targets.

- **Light Classifier:** determines which class the lighting conditions in the current camera image belongs to.
- **Light Change Detector:** continuously reads the classification result from the *Light Classifier* and compares it with the classification results of the previous instance and thereby detects drastic changes in lighting conditions.
- **Background Model Updater:** updates the background model when it is notified about a light change event by the *Light Change Detector*. It uses the target positions, region sampler and the status checker modules. If number of targets $N = 0$, the background model is updated with the image I_{id}. If $N > 0$, from each target position the occupancy region L_{id} of each camera (id=0,...,M) is obtained. This is given by:

$$L(t) = \bigcup_{j=1}^{N} l_j(t) \tag{4.1}$$

where l_j is given by Eq. 3.6.

This is the area that can not be included in the reference image for the new background model, and needs to be included when exposed. The current area for the reference image is initialised as:

$$D(t_0) = (A \cap L(t_0))^c \tag{4.2}$$

Then the background image is initialized,

$$I_{ref} = \{I(x,y) \mid x, y \in D(t_0)\} \tag{4.3}$$

where $A = \{(x,y) \mid x = 1,2,...,width,\ y = 1,2,...,height\}$. The non updated regions are updated in time when the targets are in motion, thereby exposing the previously hidden regions. This is computed in the form:

$$h_L(t) = (L(t-1) \setminus (L(t) \cap L(t-1))) \tag{4.4}$$

where h_L is the new exposed pixels in the current frame. Then the background image is updated using these pixels as follows:

$$I_{ref} = \{I(x,y) \mid x, y \in h_L(t)\} \tag{4.5}$$

Finally, the current area at time t is updated as below:

$$D(t) = D(t-1) \cup h_L(t) \tag{4.6}$$

$D(t)$ is updated until $|D(t)| = |A|$.

When the background update process is initiated the tracker changes the behaviour of its processing modules. This step is necessary since during the background model update process, the background segmentation process is suspended. Since the background segmentation image is not available, the tracker suspends the new target detection process temporarily and also modifies the sensor data pre-processing steps. Instead of generating the HSV image from the background segmented image, the tracker uses a mask to highlight only the local regions surrounding each target and suppresses the rest to $(0,0,0)$. Once the background model update is complete, the tracker activates the background subtraction module, target detection and initial pre-processing phases.

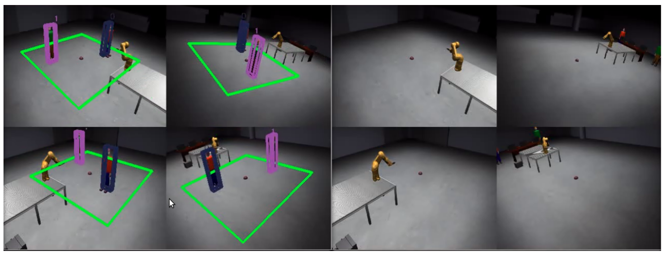


Figure 4.12: Background model update in presence of foreground targets.

Figs. 4.12, 4.13, 4.14 and 4.15 provide a step by step illustration of the background update process in a simulated environment. Validation in real-world scenarios will be discussed in Chapter. 5. Fig. 4.12 shows the current scene in each camera view with two targets which are

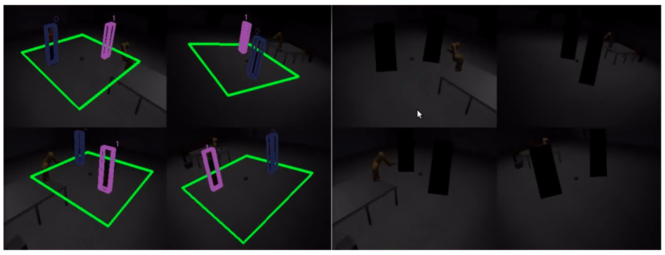


Figure 4.13: Background model update in presence of foreground targets.

being tracked by their respective trackers. In fig. 4.13 the lighting condition changes from good to very dark. This event is detected and the background model update process is initiated.

It can be observed in fig. 4.13 that the new background image generation is instantiated and the regions which are not occupied by the foreground targets are updated, while the regions which are occluded by the targets are left un-updated. Further in fig. 4.14 it can be observed that the targets move, resulting in further update of the background image with regions exposed, due to the target motion.

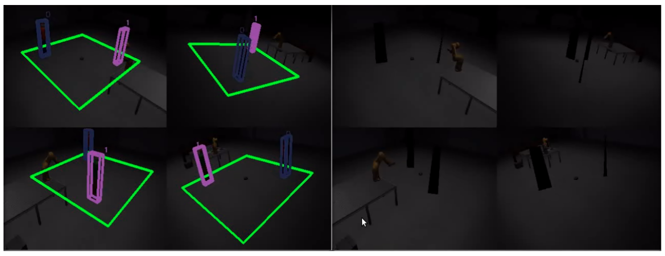


Figure 4.14: Background model update in presence of foreground targets.

Finally, in fig 4.15 every pixel in the background image is updated which is further used to update the background model of each camera view with its respective updated background image. The tracker then re-instantiates the background subtraction process.

4.6 Modality Weight Generation for Multi-modal Fusion

Intelligent multi-modal fusion of the colour and optical flow modality is achieved by generating individual weights for the respective modalities by analysis of the tracking scene. During tracking, the two modalities contribute towards a global likelihood value for the particle filter. The goal is to produce the weights which decide the individual contributions of the two modalities.

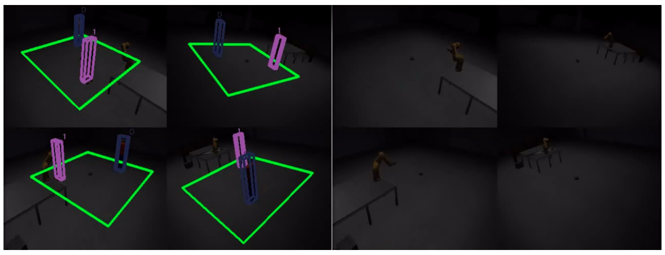
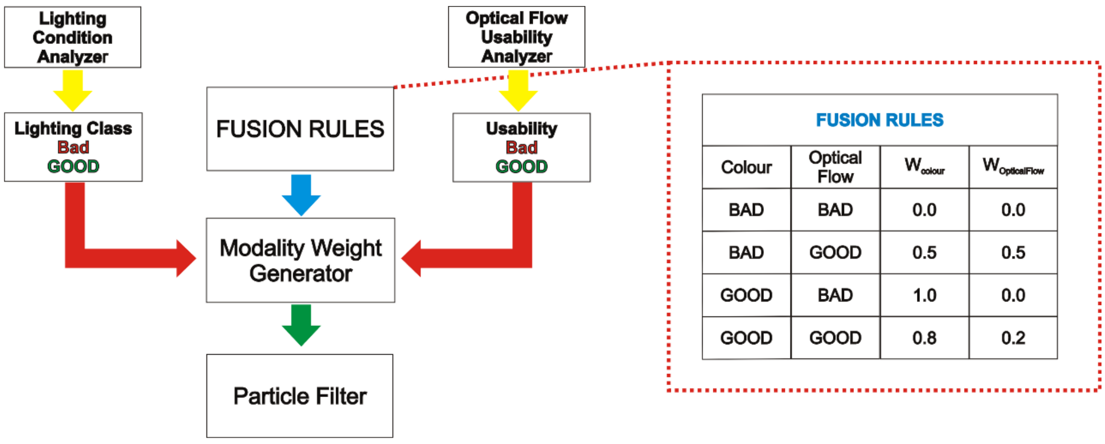


Figure 4.15: Background model update in presence of foreground targets.

While tracking, the scene is analysed to estimate the right contribution in which the colour and optical flow modality be fused. The analysis is performed individually for the respective modalities. The colour modality is analysed by checking the quality of light through the machine learning based lighting conditions analysis module. The optical flow modality is analysed through the optical flow suitability analyser, which uses the tracked trajectories of all the targets to decide the degree of usability of optical flow information for each individual target.


Lighting Condition Analyzer
Lighting Class
Bad
GOOD
FUSION RULES
Modality Weight Generator
Particle Filter
Optical Flow Usability Analyzer
Usability
Bad
GOOD
FUSION RULES

Colour	Optical Flow	W_{colour}	$W_{OpticalFlow}$
BAD	BAD	0.0	0.0
BAD	GOOD	0.5	0.5
GOOD	BAD	1.0	0.0
GOOD	GOOD	0.8	0.2

Figure 4.16: Intelligent fusion module to generate weights for the individual modalities through scene analysis.

Fig. 4.16 describes the module performing the task of generating the weights for the individual visual modalities through scene analysis. This module consists of two scene analysis units, each analysing the usability of the individual modality in the current scene. The usability of the modalities can be represented in the form of classes. The class categories can be divided into two or more types. In this example, a simple binary classification is used, labelled as Bad and Good. The class with label *Bad* indicates that the suitability of the corresponding modality in the current scene is low and the influence of its contribution towards the global likelihood be lowered. On the other hand, the class labelled *Good* indicates that the corresponding modality is well suited in the current scene and should contribute to the global likelihood generously. The usability classes can be extended to more categories such as $(Bad, OK, Good)$ or further into $(VeryBad, Bad, OK, Good, VeryGood)$. The class identification units for each modality depend on their respective scene analysers.

4.6.1 Quality Classification for Colour Histograms

The colour modality usability analyser uses the machine learning based lighting conditions classifier described in Section 4.5.2, in order to determine the usability class. The lighting condition classifier identifies if the current scene lighting is Good or Bad. This information is mapped directly by the usability analyser. If the lighting condition is Bad then the usability of the colour modality is set to Bad. On the other hand, if the lighting is Good then the colour modality usability is also set to Good.

4.6.2 Quality Classification for Optical Flow

The optical flow usability check unit analyses the tracked trajectories of the targets and their motion parameters in the current scenes as described in Section 4.4, in order to identify the correct usability class with respect to the current scene. To summarize the discussion of Section 4.4, the optical flow usability class falls in the category Bad for a particular target when:

- The target is stationary or moving with a velocity below a certain threshold V_{st}.
- The target is moving closer than a defined threshold d_{min} to another target, and the absolute difference of its optical flow direction component with respect to the the other target is below a certain threshold θ_{min}. This indicates that the two targets are moving roughly in the same direction, resulting in similar flow vectors.

On the other hand, the optical flow usability class is considered to be Good when,

- The target is moving with a velocity higher than the defined threshold V_{st}, and at a distance greater than d_{min}, with respect to all other targets in the scene.
- The target is at a distance less that d_{min} to another target or targets. It has a velocity component higher than V_{st} and the absolute difference of its optical flow direction component with that of each other target is greater than θ_{min}.

The threshold chosen depends on the dimensions of the workspace and the intended behaviour of the humans within the workspace. For example, the human motion patterns in a large workspace will be less confined as compared to a smaller workspace. Similarly, the motion patterns in a lab environment will be different from that in a industrial environment. In this demonstration scenario the workspace span was 5×5 meters containing between 2 to 5 humans. The thresholds selected were,

- $V_{st} = 25mm/sec$
- $d_{min} = 500mm$
- $\theta_{min} = 25degrees$

4.6.3 Multi-modal Fusion depending on Classification

Once the usability classes of the respective visual modalities are known for the current scene, this information is supplied to the modality weight generator. This module uses the results of the usability check units together with a rule based fusion approach to generate the optimum weights for the individual visual modalities. The rule based fusion technique is constructed through a fixed set of rules defined by the user. These rules specify the combination of weights to be assigned to the two modalities for each possible combination of classes. Fig. 4.16 illustrates a simple fusion rule data-bank for the binary classes consisting of Bad and Good labels. As mentioned above, these classes can be extended to a wider range, along with a more dense rule data-bank. An example of such and extended data-bank for class labels $(Bad, OK, Good)$ is shown below,

$Colour$	$OpticalFlow$	W_{color}	W_{opFlow}
Bad	Bad	0.0	0.0
Bad	OK	0.4	0.6
Bad	$Good$	0.5	0.5
OK	Bad	1.0	0.0
OK	OK	0.7	0.3
OK	$Good$	0.5	0.5
$Good$	Bad	1.0	0.0
$Good$	OK	0.9	1.0
$Good$	$Good$	0.8	0.2

Once the individual weights for the individual modalities summing up to 1.0 are obtained, they are fused in order to obtain a global likelihood. The fusion operation is performed for every hypothesis generated by the particle filter and for each camera view. When both modalities are un-suitable for tracking, the tracker declares a target loss and instantiates the target recovery mechanisms in the form of re-detection.

The mathematical representation of the complete fusion procedure is formulated below:

$$U_i^{colour} = L_{svm}(I_{cam_i}) \tag{4.7}$$

$$U_{i_{tid}}^{opFlow} = A_{opflow} T_{tid} \tag{4.8}$$

$$U_{fused} = (U_i^{colour}, U_{i_{tid}}^{opFlow}) \tag{4.9}$$

$$(W_{colour}, W_{opFlow}) = R(U_{fused}) \tag{4.10}$$

$$L_{filter_h} = W_{colour} L_{colour} + W_{opFlow} L_{opFlow} \tag{4.11}$$

where, U^{colour_i} is the usability class for the colour modality in the i^{th} hypothesis, L_{svm} is the machine learning based lighting condition classifier and I_{cam_i} is the current image from the camera. $U_{i_{tid}}^{opFlow}$ is the usability class for the optical flow modality in the i^{th} hypothesis for the target with id *tid*. A_{opflow} is the function which performs the optical flow usability check on the motion parameters of the current target given by T_{tid}. U_{fused} forms the combined set of usability classes for the colour and optical flow modality together obtained from the usability class checks. (W_{colour}, W_{opFlow}) are the unique weights for the two modalities using the fusion rule data-bank R. Finally, L_{filter_h} is the global likelihood computed through the weighted fusion of the colour and optical flow likelihoods given by L_{colour} and L_{opFlow} respectively, with their generated weights.

As mentioned in Chapter. 3, the modular construction of the system allows easy configuration of individual processing modules. The optical flow processing modality introduces considerable processing load on the system while contributing to the enhanced robustness. Therefore, in situation when environment is known to be fairly static and the number of targets expected is low, the optical flow modality can be switched off in order to boost the trackers speed. However, this should be avoided in presence of targets with similar appearances and increased mutual occlusions. Similarly, under situations of very bad lighting conditions, the colour histogram module can be turned off while relying solely only on optical flow. This measure should be adopted only in situations of complete degradation of colour information from the cameras due to very dark lighting conditions, since even under reduced lighting conditions cameras provide colour information with reduced quality which could still assist the tracker up to a certain level of benefit.

4.7 Discussion

This chapter introduced and discussed the robustness enhancement modules of the multiple human tracking system. The machine learning techniques for dynamic background model update under changing lighting conditions and intelligent multi-modal fusion were formulated. In the next chapter, experiments validating the functioning of the system with respect to different aspects will be discussed in detail.

Chapter 5

Experiments

In this chapter the experiments conducted to evaluate the multiple human tracking system are discussed. As discussed in Chapter. 2, a unified benchmarking framework is lacking in the area of stereo multiple human tracking. Existing approaches perform tests in different ways. In order to evaluate any vision based system comparisons of the results with ground truth is essential. In the context of human tracking, the ground truth data represents the actual positions of the human targets in each frame. This data, when available, can be used to evaluate the accuracy of the tracker. Most of the existing systems such as [61], use manual techniques to generate ground truth questioning the accuracy of the method. Methods such as [97], conduct experiments to compute tracking accuracy using stationary targets at predefined locations, but not by comparing their trajectories which is an important measure. Therefore, in thesis a general purpose technique to generate zero error ground truth which can be used to evaluate different aspects of the system has been presented. In the following sections, the process of generating ground truth data and its use in experiments to evaluate the accuracy of the tracker are discussed.

5.1 Zero Error Ground Truth Generation

In the context of human tracking a variety of approaches can be adopted for generating ground truth data. These methods are either manual, semi-automatic or automatic [130, 96, 131, 132]. In the manual approach, positions of the humans are marked manually using reference points of features in the images. In the semi-automatic approach the human targets are forced to follow predefined paths and the images generated are synchronised with these paths either automatically or by manual observation. In the automatic methods, typically external trackers using sensors other than cameras are installed. Most commonly, the targets are required to carry an active marker or sensor in close contact. The data obtained from these trackers are used as ground truth.

Although the above mentioned methods provide ground truth data, they cannot guarantee accuracy since they themselves have a certain tolerance. The manual techniques are dependent on the accuracy of the annotations. Similarly, the automatic approaches also depend on the accuracy of the external tracker which has its inherent tolerance. Under such circumstances, the results from the visual tracking system is compared with ground truth data which itself is not accurate. This can lead to errors in benchmarking and therefore is not completely reliable. Therefore, the multiple human tracking system discussed in this thesis was evaluated for its correctness by generating ground truth using a novel approach which guarantees accuracy.

In order to generate ground truth without inherent errors, the test environment was modelled in 3D in its completeness using Blender [133]. A 3D model of the complete lab environment was reproduced with a high degree of detail in terms of dimensions and appearance. Every object was modelled to its actual dimensions. The robot system was modelled in terms of its kinematics and dynamics. The 4 cameras used in the tracking system were reproduced with exact intrinsic and extrinsic parameters as compared to the real scene. The virtual environment was also equipped with light sources similar to the ones used in the lab environment. With the complete environment, designed in 3D, the only missing entities were the human targets. They were modelled using simple *Playmobil* [134] models. These models are simple in terms of human kinematics and dynamics. Since the object model used in the tracker is a rigid rectangular cube, the *Playmobil* models make a good choice. The appearance of these models can be designed as required.

Once the 3D model of the entire lab environment including the human targets are available, it is possible to simulate the motion of the human targets within the virtual scene. For each target, the trajectories can be planned and simulated. Hence it is possible to simulate the motion of multiple targets within the virtual scene as close to the real-world excluding real human dynamics. This implies that the human targets move with 2 *DOF* and an additional degree of freedom for rotation along the *Z axis*, which is sufficient considering the dynamics of the target model used. With these resources an animation can be created, where human targets move freely within the scene, obeying the predefined trajectories and velocities. During the animation the intensity of the light array can be changed either synchronously or individually. Once the animation is ready it can be rendered using the perspective of the 4 cameras. Since the cameras are modelled exactly as in the real scene, they produce an animation with the exact perspective. Another feature of this system is that it can emulate targets in different conditions, such as same colour and appearance to test the modality fusion.

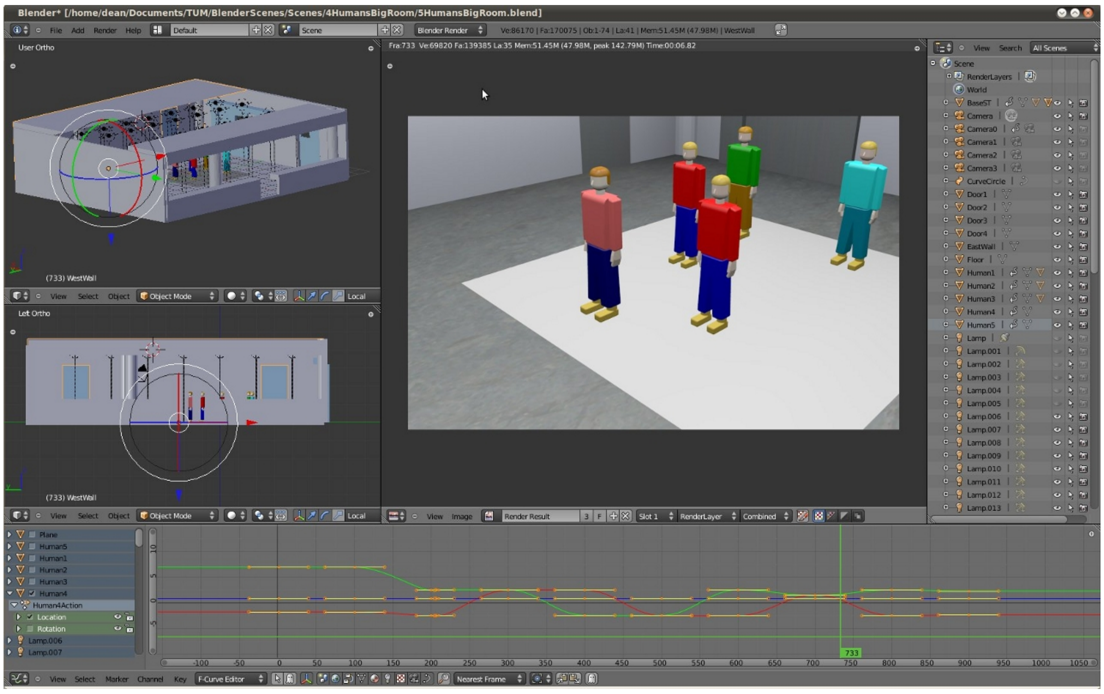

Blender* [/home/dean/Documents/TUM/BlenderScenes/Scenes/4HumansBigRoom/5HumansBigRoom.blend]
User Ortho
Left Ortho
(733) WestWall
Object Mode
Render Result
F-Curve Editor
Nearest Frame

Figure 5.1: The lab environment modelled in 3D.

Fig.5.1 shows a snapshot of the 3D model of the complete lab from one perspective. It consists of the robot system mounted on the table, the shared workspace, the cameras on the ceiling and the human targets. The light array is present but not rendered. In addition, the operator's workspace, which includes the control computers is also modelled. Although the motion trajectories of the humans can be simulated in blender, it cannot be directly used. The rendered animation from the four cameras within the blender model is further recorded in the form of videos. The videos obtained for each camera are used as in input to the multiple human tracker and the tracked trajectories are recorded. By using a python script within blender the real trajectories of the human targets are extracted. Once the blender generated trajectories and the trajectories tracked by the multiple human tracker are available, they can be compared in order to compute the accuracy of the tracker.

Fig. 5.2 presents the simulation of human motion within the virtual lab environment. The top left image shows two human targets being simulated and the rendered images obtained from

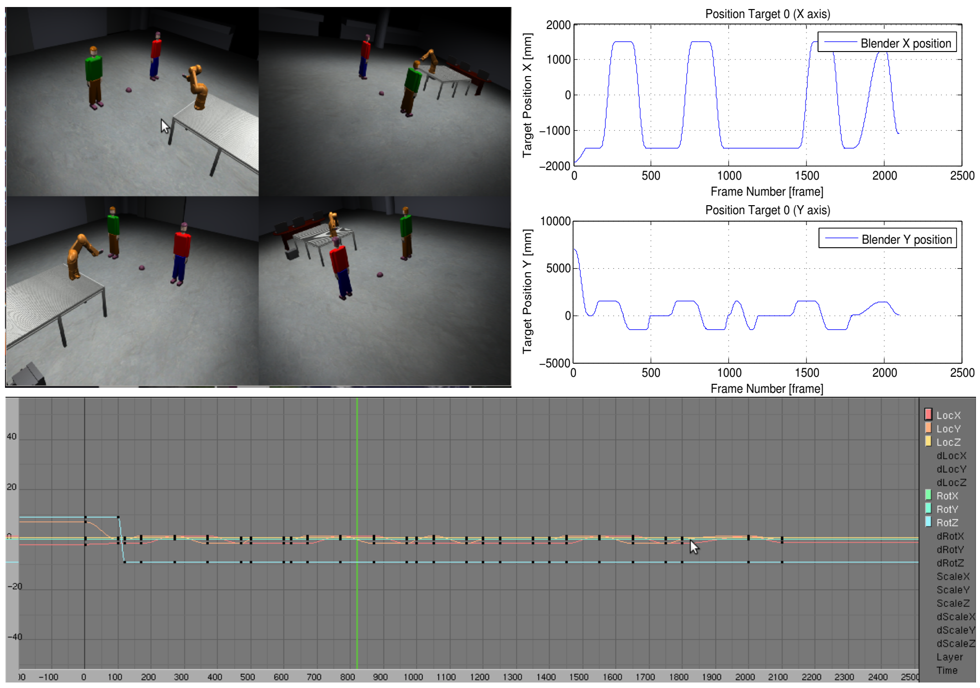


Figure 5.2: Human target simulation in the blender model of the lab environment.

the 4 cameras. The images obtained from the entire animation sequence are used for the tracker evaluation. The lower left image shows the trajectories of one the targets generated through blender. The motion of the target is modelled with 6 degrees of freedom each represented individually along the entire time-line of the animation. The top right image illustrates the trajectory data extracted from the trajectories generated through blender to suite the format used for the tracker evaluations. This data includes the (x, y, z) position of the human targets at each step in the time-line.

The experiments were conducted in the virtual and real environment and were focussed on evaluation of the following aspects:

- **Tracking accuracy of trajectories in the presence of multiple targets:** In these experiments the system was evaluated for its tracking accuracy. This includes target detection, target exit and trajectories of multiple targets. The experiments were carried out with 3, 4 and 5 targets. In the experiment with 5 targets, two targets have exact similar appearance and move very close to each other.

- **Robustness of tracking and background model update under drastically changing lighting conditions:** Here, the system is evaluated for its tracking robustness under drastically changing lighting conditions. It validates the performance of the background update module in presence of foreground targets.

- **Robustness of occlusion handling in presence of targets with similar appearance:** This experiment shows the accuracy and robustness of the tracker under multiple occlusions. The tests are performed under situation where targets have exact similar appearance and move very close to each other in opposite and also in the same directions. An experiment is also conducted in a situation where all targets are dressed in black which is an unfavourable for colour based tracking.

- **Positive impact of multi-modal fusion under drastically changing lighting conditions:** Here, the positive impact of multi-modal fusion of colour histograms and optical flow towards robust tracking under drastically changing lighting conditions is evaluated. It shows how the intelligent tuning of modality weights depending on the tracking scenario improve the performance of the particle filter.

The experiments conducted in the simulated environment use the videos generated by simulating humans within the blender modelled environment of the entire workspace. The videos are accompanied by the ground truth trajectories of the humans rendered in the four cameras. After verification of the tracker in the virtual environment with respect to different aspects, the same experiments were conducted in the real-environment to prove the usability of the system. The following sections will describe the tests conducted and the results obtained. The experiments start with testing the tracker with 4 targets and analysing the accuracy of the tracked trajectories. This is followed by a similar experiment but with 5 targets with close motion and some targets having similar appearance. Thereafter, the system is tested in the real environment with 3 targets in a constrained area. The next set of experiments are focused on the performance of the system under sudden and drastic illumination changes. This is evaluated both in the simulate and real environment, Thereafter, the importance of the occlusion handling module is verified using two targets of similar appearance and moving very close to each other. This experiment is extended to the real environment where three similarly dressed targets interact very closely with physical contact. These targets are dressed in black colour which makes colour based tracking very complex. Finally, the impact of the intelligent multi-modal fusion of the quality of feature matching under changing lighting conditions are presented. Each experiment is accompanied by a video demonstration link.

5.2 Tracking Accuracy of Trajectories in the Presence of Multiple Targets

The experiments start with testing the tracker with 4 targets and analysing the accuracy of the tracked trajectories. This is followed by a similar experiment but with 5 targets with close motion and some targets having exact similar appearance. Finally, a test in the real environment is presented.

5.2.1 Tracking with 4 Targets in the Scene

In this experimental set-up the system was tested in the simulate environment with 4 humans. The tracked positions of each target was compared with the available ground truth data. The error between the two sets of data was computed with statistical measures. The experimental set-up consists of 4 input videos, one from each camera. The videos were created using the virtual environment described in section 5.1. In the scene, 4 targets were simulated to produce human-like motion. The targets enter the scene two at a time and move within the workspace with different trajectories. These trajectories were pre-planned and consist of key-frames. Interpolating between key-frames, the actual positions of the targets in each frame were extracted. This data forms the ground truth. The human models were rendered with the 4 camera perspectives using the trajectory information.

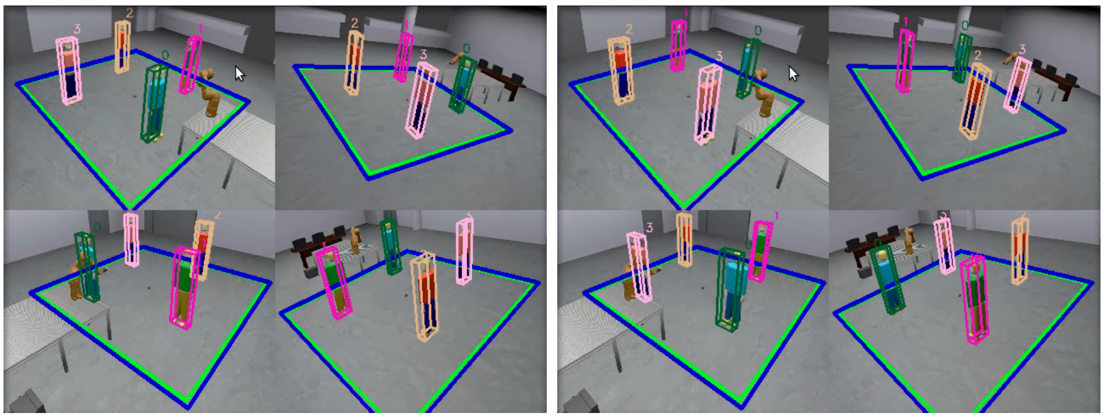


Figure 5.3: Illustration of the tracking system with 4 targets in the scene.

Fig. 5.3 illustrates the tracking of 4 targets in two unique frames in the input sequence. The tracker keeps track of each target throughout the complete motion sequence. The boxes around the targets represent the estimated 3D position of the targets by the tracker. These boxes are rendered in each camera perspective. The position of the individual targets tracked by the tracker were recorded for each frame. Once the tracked positions for each target over the complete video sequence were obtained, it was compared with the ground truth data.

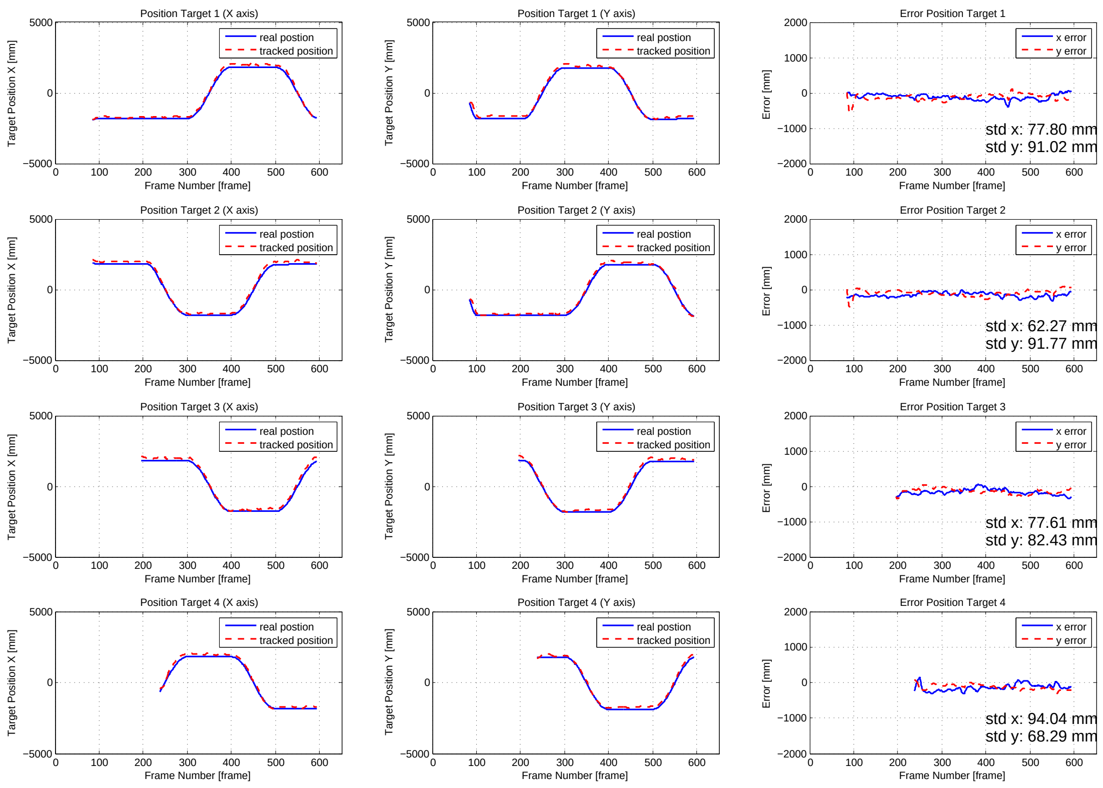


Figure 5.4: Experiment results of the comparisons between the tracked positions of the 4 targets in each frame and the ground truth data. Each row represent the experimental results for an individual target. The first column in each row shows the X co-ordinates of the tracked trajectories (in red colour) and the actual trajectory obtained from the ground truth (in blue colour). The second column in each row shows the Y co-ordinates of the tracked trajectories (in red colour) and the actual trajectory obtained from the ground truth (in blue colour). The third column in each row shows the plots of the error between the tracked and the actual trajectories in X (blue colour) and Y (red colour) axes. The standard deviation of the error in X and Y axes are also shown in the plots.

Fig. 5.4 shows the results from the experiment. The results are illustrated for each target separately. For each individual target, the computed trajectories are plotted along with the actual trajectories obtained from the ground truth. The trajectories are plotted for the X and Y axis separately as shown. Thereafter, the estimated and the actual trajectories are analysed in order to compute the error between them. The rightmost columns in Fig. 5.4 show the error plots and the standard deviation of error in each axis for each target. It can be observed that the tracking error is in the range of 5 cm to 10 cm in a area as large as 6×6 $meters$, which outperforms popular systems like [43]. A complete video demonstration is available under [135] or `http://www.youtube.com/watch?v=cJ8sDGKAcac`.

5.2.2 Tracking with 5 Targets in the Scene with Close Motion

This experiment is similar to the one described in Section 5.2.1 but contains 5 targets instead of 4. Two out of the five targets have exactly the same appearance and during the motion sequence move close to each other. This experiments evaluates the tracking results in such a scenario where the target count is high resulting in larger probabilities of occlusions and at the same time presence of targets with similar appearance resulting in increased tracking complexity.

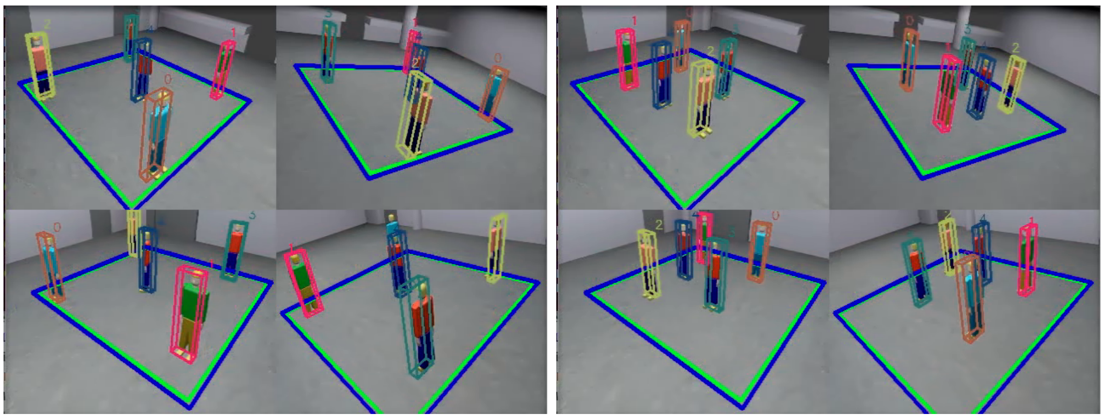


Figure 5.5: Illustration of the tracking system with 5 targets in the scene moving close to each other and two targets having similar appearance.

Fig. 5.5 illustrates the successful tracking of 5 targets within the tracking scene. The snapshots are obtained from two different frames within the tracking sequence. The left image shows the targets detected and tracked in the four camera views. The right image shows the tracker keeping track of the targets successfully even when there are moving very close to each other resulting in multiple occlusions.

Fig. 5.6 illustrates the plots of the tracked trajectories along with the actual trajectories obtained from the ground truth generator. The right most column represents the error computed in the X and Y directions. It can be seen that the standard deviation of the error computed

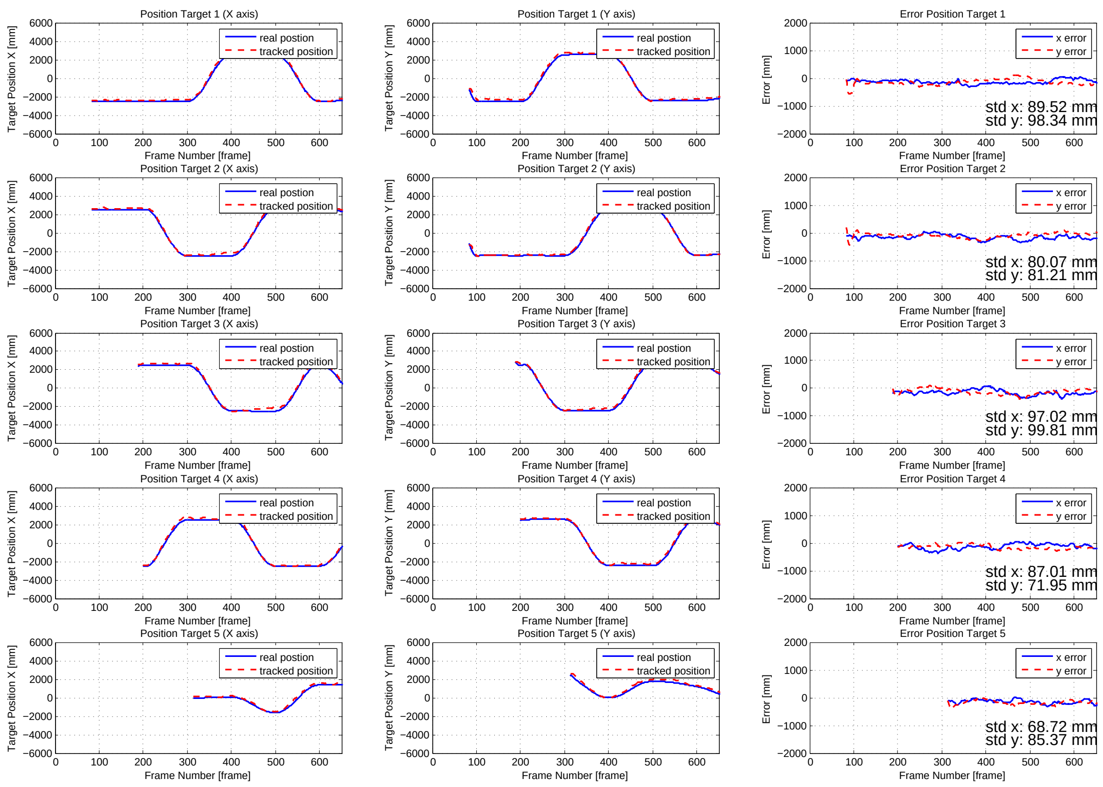


Figure 5.6: Experiment results of the comparisons between the tracked positions of the 5 targets in each frame and the ground truth data. Each row represent the experimental results for an individual target. The first column in each row shows the X co-ordinates of the tracked trajectories (in red colour) and the actual trajectory obtained from the ground truth (in blue colour). The second column in each row shows the Y co-ordinates of the tracked trajectories (in red colour) and the actual trajectory obtained from the ground truth (in blue colour). The third column in each row shows the plots of the error between the tracked and the actual trajectories in X (blue colour) and Y (red colour) axes. The standard deviation of the error in X and Y axes are also shown in the plots.

over the entire sequence is below 10 *cm* even under increased numbers of simultaneously mutual occlusions in multiple cameras. The tracker keeps good track of the targets with similar appearance even when they are moving close to each other. A complete video demonstration is available under [136] or `http://www.youtube.com/watch?v=g0HbIYap-hw`.

5.2.3 Tracking with 3 Targets in a Real Scene and Constrained Area

The previous sections validated the performance of the system in a simulated environment under various validation scenarios. The availability of ground truth data makes this possible. However, in the real world scenario actual ground truth data is not available. Therefore, the system was tested in a crude manner. The tracking area in the actual laboratory was highly constrained. The possibility of mounting the cameras on the ceiling was limited to 3 *meters* and the tracking areas covered was 3×3 *meters*. The experiment was performed with 3 targets in the scene. In order to get an estimate of the trajectories, a fixed path was marked in the tracking area and the humans were asked to move along this path.

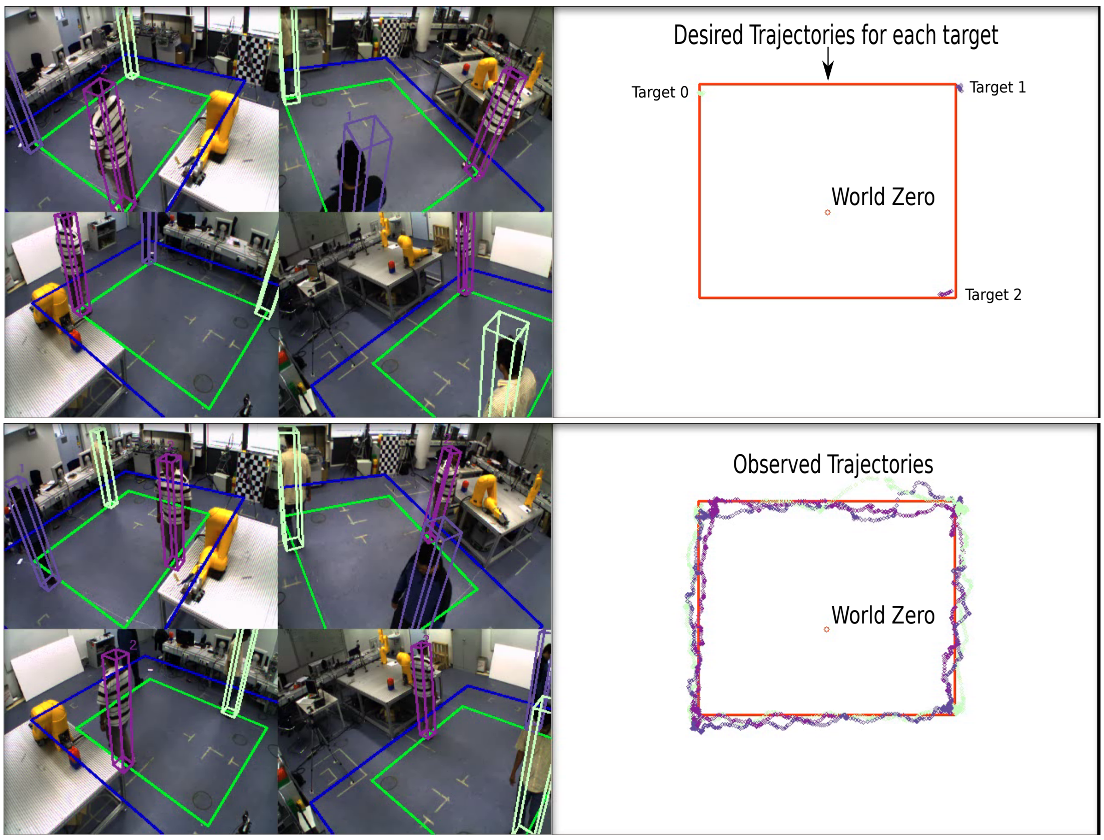


Figure 5.7: Experiment results in real world environment with 3 targets. The first row shows the three targets at their starting positions and the path to be followed marked in red towards the right. The second row shows the tracker tracking the targets and the motion trajectories generated.

Fig. 5.7 illustrates the results obtained. In the first row it can be observed that the targets are at their initial position and being tracked. Towards the right, a red box can be observed, which represents the path the targets are supposed to move on when observed from the top view. The second row shows the targets being tracked after they have moved. To its right the generated trajectories have been plotted which approximate the desired shape. The trajectory of each target is plotted and is similar to the colour id set by the tracker. The results cannot be mathematically evaluated since the ground truth data is not available, and the humans themselves induce an error while moving along the desired path. The goal of this experiment is to show that the tracker can track multiple targets and the results obtained are as desired in the real-world scenario.

A complete video demonstration is available under the link [137] or `http://www.youtube.com/watch?v=wePVQ7cXB9c`. The demonstration shows how the system tracks the targets. In the video, after completing the desired trajectories, the targets interact with each other by moving close to each other resulting in multiple occlusions. The tracker keeps a good track of all the targets even under these situations.

The experiments conducted in this section validate the claim that the multiple human tracking system cab automatically detect and track human targets within a predefined tracking area. It also proves that the tracker keeps a good track of the motion trajectories of each individual targets even under multiple occlusions. The next section will validate the robustness of the system under drastically changing lighting conditions.

5.3 Robustness of Tracking and Background Model Update under Drastically Changing Lighting Conditions

These experiments were conducted initially in the simulated environment and further in the real environment. The goal was to validate the robust performance and adaptability of the system towards drastically changing lighting conditions. Thereby, validating the claim that the system can be used in dynamic environments.

5.3.1 Experiment in Simulate Environment

This experiment was carried out to validate the performance of the tracker under drastic changes in lighting conditions. The machine learning based background update module used for this purpose was discussed in Section 4.5.4. To summarise, this module is responsible for detecting drastic changes in lighting conditions and updating the background model in the presence of foreground targets. During this process the pre-processing of the camera images for background segmentation is suspended. Therefore, the background update module has to ensure that the tracker is able to keep track of the targets until the new background model is generated. As illustrated earlier, figs, 4.12, 4.13, 4.14 and 4.15 provide a step by step illustration of the entire process.

The experiment was carried out using two targets simulated in the scene. The sequence was 1600 frames long and each image was added with Poisson noise to model the camera noise. The two targets move freely in the scene under good lighting conditions. At frame 1000 the lighting conditions in the scene change drastically. The new lighting condition falls into the class *Bad* as classified by the machine learning based module for lighting condition analysis. The background update process is activated until the new background model is generated. The tracked trajectories during the entire sequence are recorded and plotted along with the ground truth. Fig 5.8 shows the plotted results. A small glitch can be observed at frame 1000 when the lighting conditions change. The right most column represents the error plot in the X and Y directions respectively. An overshoot in the error in the part of the sequence where the light change occurred can be observed in the error plots but the tracker recovers immediately. From the standard deviation of the errors in each direction it can be observed that the average error is still below 10 *cms*, validating the performance of the background update module. A complete video demonstration is available under [138] or `http://www.youtube.com/watch?v=Jm2D1jUsuFM`.

5.3.2 Experiment in Real Environment

In this experiment the same lab scene described in Section 5.2.3 is used. The system is tested for its performance under drastic changes in lighting conditions in a real environment. The system requires to detect such events and update the background model in presence of foreground targets.

Fig. 5.9 illustrates the experiment conducted. It can be observed in the first row, the tracker successfully tracks the two targets in the scene. The background model used is shown on the right. The second row represents the frame in which the light changes drastically. The light change is detected by the lighting conditions classification module. The pre-processing for background segmentation is suspended and the tracker uses local statistics to keep track of the targets. The background model update is instantiated following which, regions exclusive of target occupancy are updated. During the course of tracking, while the targets exhibit motion, the not-updated regions of the background are updated as they become visible. Finally, the

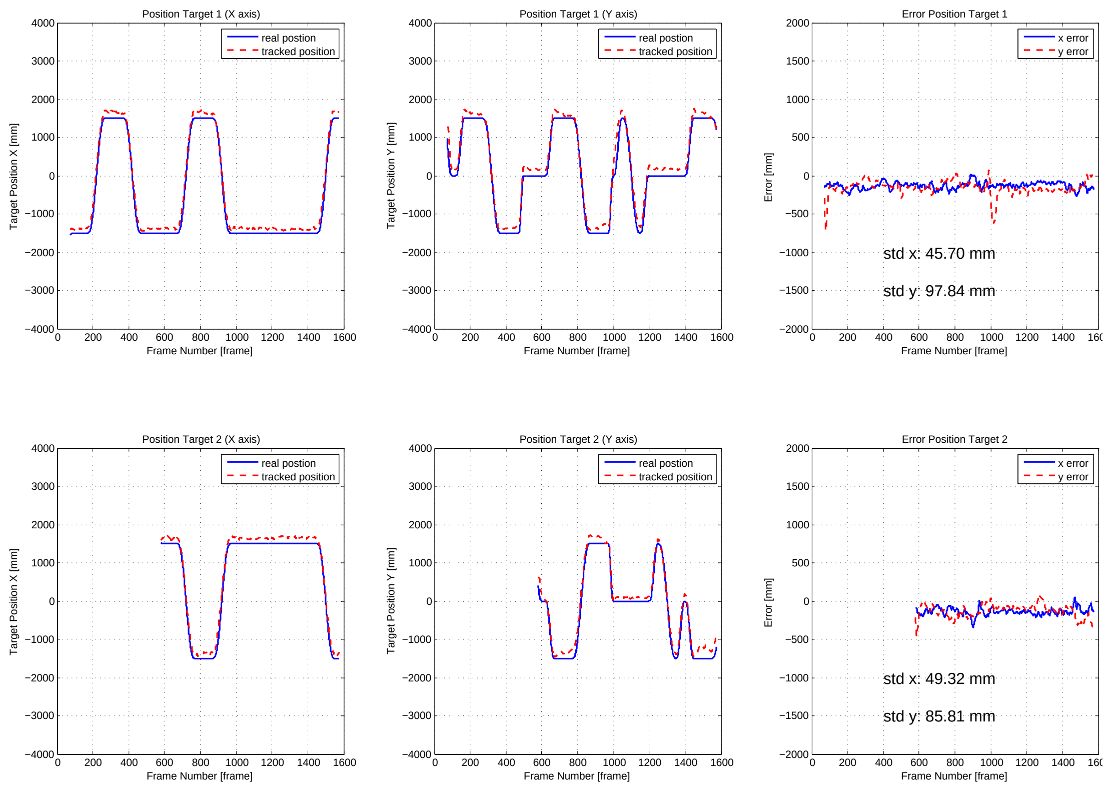


Figure 5.8: Illustration of the tracking system with 2 targets (where Target 1 enters at frame 150 and Target 2 enters at frame 600) and drastic changes in lighting conditions during the tracking sequence. The first column in each row show the X co-ordinates of the tracked trajectories (in red colour) and the actual trajectory obtained from the ground truth (in blue colour). The second column in each row show the Y co-ordinates of the tracked trajectories (in red colour) and the actual trajectory obtained from the ground truth (in blue colour). The third column in each row shows the plots of the error between the tracked and the actual trajectories in X (blue colour) and Y (red colour) axes. The standard deviation of the error in X and Y axes are also shown in the plots.

complete background model is updated in each camera view and the tracker successfully tracks all the targets. The experiment validates the approach. The primary assumption for this method is that the targets should exhibit motion in order to expose occluded background regions. This is a reasonable assumption since the probability of motion is high due to the application requiring human interaction with robots. Secondly, if the targets do not move, the prediction step is trivial and local statistics are sufficient. A complete video demonstration is available under [139] or `http://www.youtube.com/watch?v=yZHCXgdDf14`.

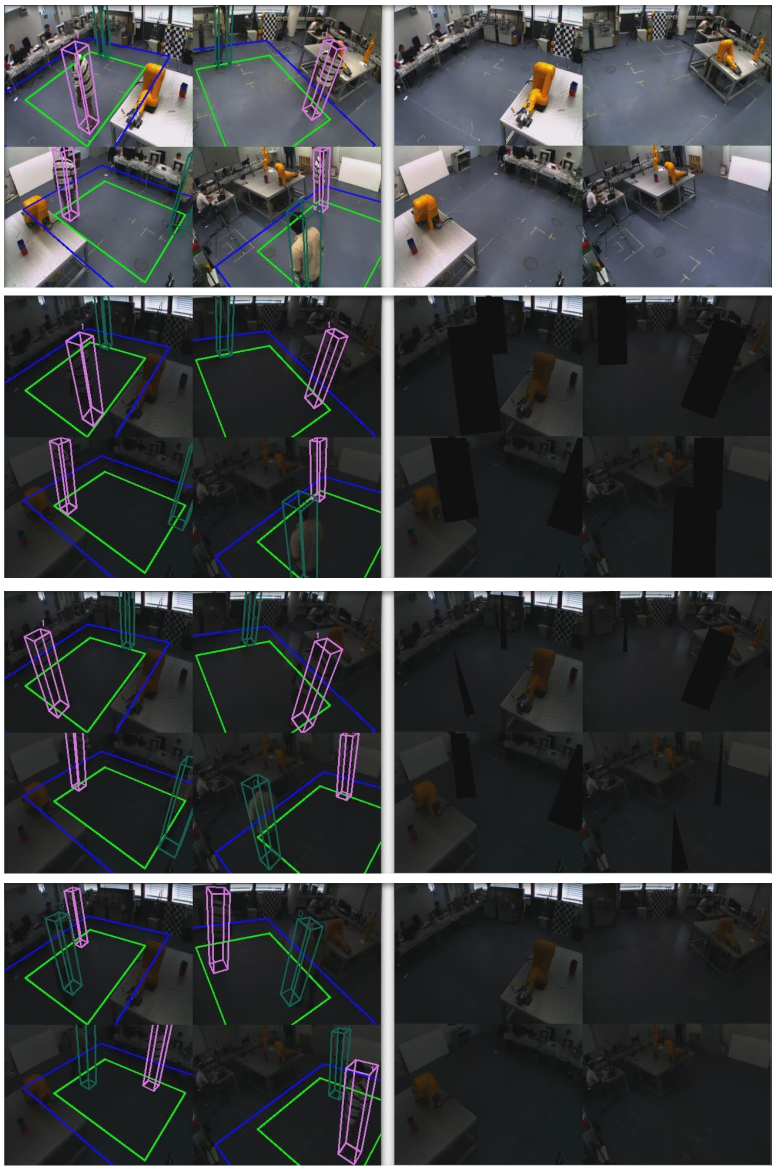


Figure 5.9: Experiment results in real world environment under drastically changing lighting conditions. The first row shows the tracking results to the left and the background model to the right. The left image in the second row illustrates the change in lighting conditions and the right image shows the detection of this situation. The third and fourth row show how the new background model is built while the targets move in the scene and the successful tracking of the targets under unfavourable tracking conditions.

The two experiments conducted validate the robust performance of the system under drastically changing lighting conditions. It shows how the system adapts to the dynamically changing environment while keeping a good track of all the targets being tracked. The next section validates the performance and importance of the occlusion handling module.

5.4 Robustness of Occlusion Handling in Presence of Targets with Similar Appearance

The occlusion handling module and its importance in robust tracking during mutual occlusion of targets was discussed in Section 3.5.4. This section demonstrates an experiment to validate the importance of the occlusion handling module in a special situation where most trackers will tend to lose the target. As before, the experiments were carried in both simulated and real environment.

5.4.1 Occlusion Handling with Similar Targets and Very Close Motion

The scenario consists of two targets with exactly the same appearance. The two targets enter the tracking area and move very close to each other. The distance between the targets when they cross each other is less than 10 *cm*. The motion of the targets are simulated in a straight line where they cross each other. In order to produce an environment as close to the real world, Poisson noise [140, 141, 142] representing the camera model was added in each image. To increase the tracking complexity, the motion trajectory of the two targets is initially in opposite directions along a straight line and thereafter, along the same line and in the same direction. Under such circumstances the tracking process becomes very complicated since the two targets have the same appearance, move close to each other and in the same direction. As discussed in Section 4.4, when the targets are moving in the opposite directions, the optical flow modality can save the feature matching process. However, when the two targets move in the same direction and very close to each other, both the colour and optical flow modality can fail. Under such circumstances, using the spatial information of the targets within the occlusion handling module, robust tracking is achieved.

Fig. 5.10 shows the tracker's output for the motion sequence. The left half shows the tracker tracking the targets while they move move in opposite directions. The targets move in a straight line and when they cross each other, the distance between them is less than 10 *cms*. The right half illustrates the successful tracking of the targets even when they move in the same direction and very close to each other.

Fig. 5.11 presents a detailed analysis of the trajectory comparisons between the tracker and the ground truth data. The first two columns show the plots of the tracked trajectories

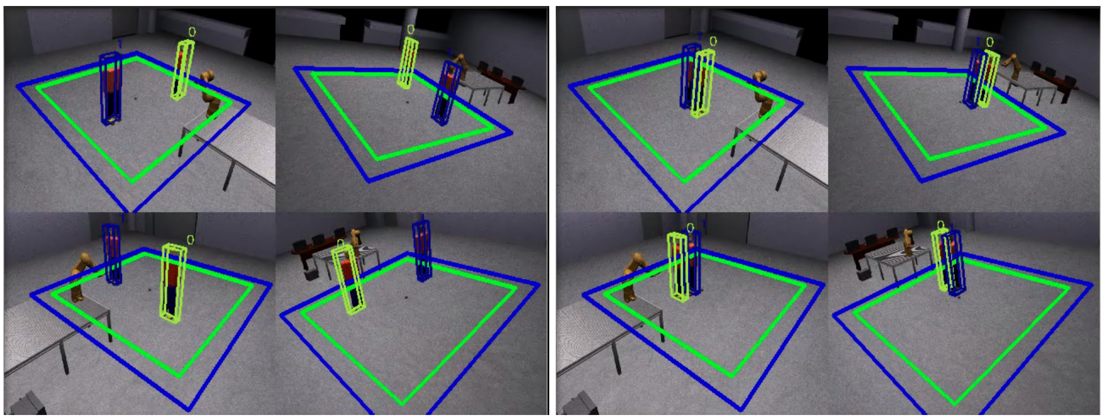


Figure 5.10: Illustration of the tracking system with 2 targets in the scene. The targets have exactly the same appearance. They move very close to each other. They move initially in opposite directions along a straight line as shown in the left image. Thereafter, they move in the same direction and very close to each other as show in the right image.

and the ground truth in the X and Y directions for each target. The characteristics of the motion between the two targets discussed earlier is validated from the plotted trajectories. The rightmost column represents the error plots in the X and Y directions. It can be observed that there is a sharp overshoot in the error in the dominant direction of motion when the targets start moving in the same direction. The tracker recovers in a few frames. This occurs due to the fact that both targets have exactly the same appearance which can cause failure in other system. In this system due to the presence of the occlusion handling module, the tracker recovers and tracks the two targets successfully even under a high degree of ambiguity in the feature matching. The average error is appeared to be twice as much as normal due to the overshoot, but in other parts of the sequence the error is still lower than 10 *cms*. A complete video demonstration is available under [143] or `http://www.youtube.com/watch?v=0vi7vtXIBxU`.

5.4.2 Occlusion Handling with 3 Targets and Very Close Motion and Dressed in Black Clothing

This experiment demonstrates the robustness of the tracker in a very difficult real-world scenario. It consists of 3 targets interacting very close to each other with physical contact. To make it more difficult for the tracker, all the targets are dressed in black clothing, which is unfavourable for colour based tracking since black and white colours do not provide any information in the colour space.

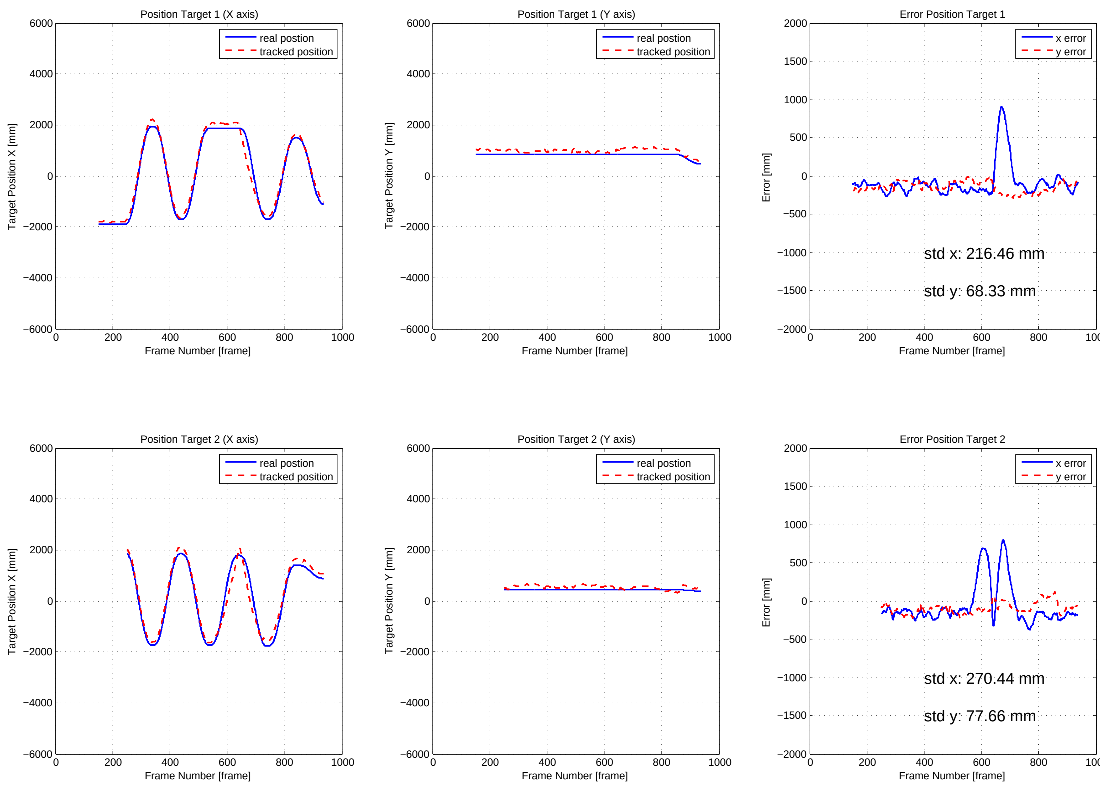


Figure 5.11: Experiment results between the tracked positions with respect to the ground truth of 2 Targets (where Target enters at frame 180 and Target 2 enters ate frame 220) with exactly similar appearance and very close motion. Each row represent the experimental results for an individual target. The first column in each row show the X co-ordinates of the tracked trajectories (in red colour) and the actual trajectory obtained from the ground truth (in blue colour). The second column in each row show the Y co-ordinates of the tracked trajectories (in red colour) and the actual trajectory obtained from the ground truth (in blue colour). The third column in each row shows the plots of the error between the tracked and the actual trajectories in X (blue colour) and Y (red colour) axes. The standard deviation of the error in X and Y axes are also shown in the plots.

Fig. 5.12 and the video sequence [144] or `http://www.youtube.com/watch?v=3ChUOW1QHoc` provides a good illustration of the trackers performance. It can be observed that the tracker successfully tracks all the three targets. The targets are tracked even when they are in physical contact resulting in multiple occlusions. The tracker keeps good track of all the 3 targets through the entire sequence. There are some frames in which the tracker tends to deflect from the target. This occurs due to multiple occlusions and some targets are out of the field of view

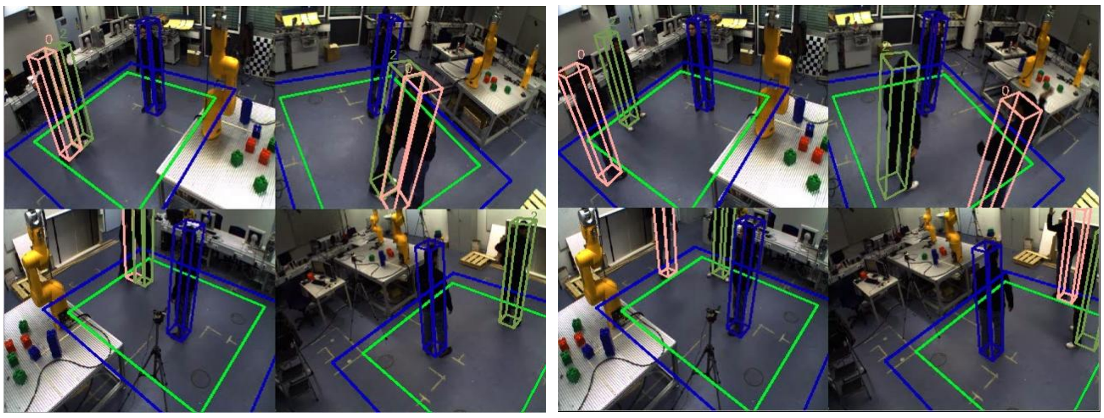


Figure 5.12: Tracking 3 targets with very close motion and similar black clothing

of certain cameras. Under such circumstance a target can be occluded in two view and be out of the field of view of one camera. This means that it is only visible in one camera which makes it difficult for the tracker to maintain the track. It can be observed that the tracker deflects from one target but recovers immediately as soon as it visible in at least two cameras.

The two experiments discussed in this section validate the importance and robust performance of the occlusion handling module. Due to this module the system performs robustly even in tracking scenarios where ambiguity is very high. In the next section the experiments to validate the contribution of the intelligent multi-modal fusion technique for improved performance of the tracker are discussed.

5.5 Positive Impact of Multi-modal Fusion under Drastically Changing Lighting Conditions

This experiment validates the use of intelligent multi-modal fusion in order to improve the performance of the tracker. It shows how the tracker likelihood is improved due to the fusion in difficult tracking scenarios. The experiment plots the feature matching distance from the colour and optical flow modalities. This data was recorded in a sequence from good lighting conditions and another sequence from bad lighting conditions. Further, the likelihood values computed with and without multi-modal fusion are plotted for analysis.

Fig. 5.13 demonstrates the results of the experiment. It can be observed that quality of lighting in the second scene is very bad as compared to the first one. Comparing the distance plots in the two scenes it can be observed that, the feature matching distance of colour his-

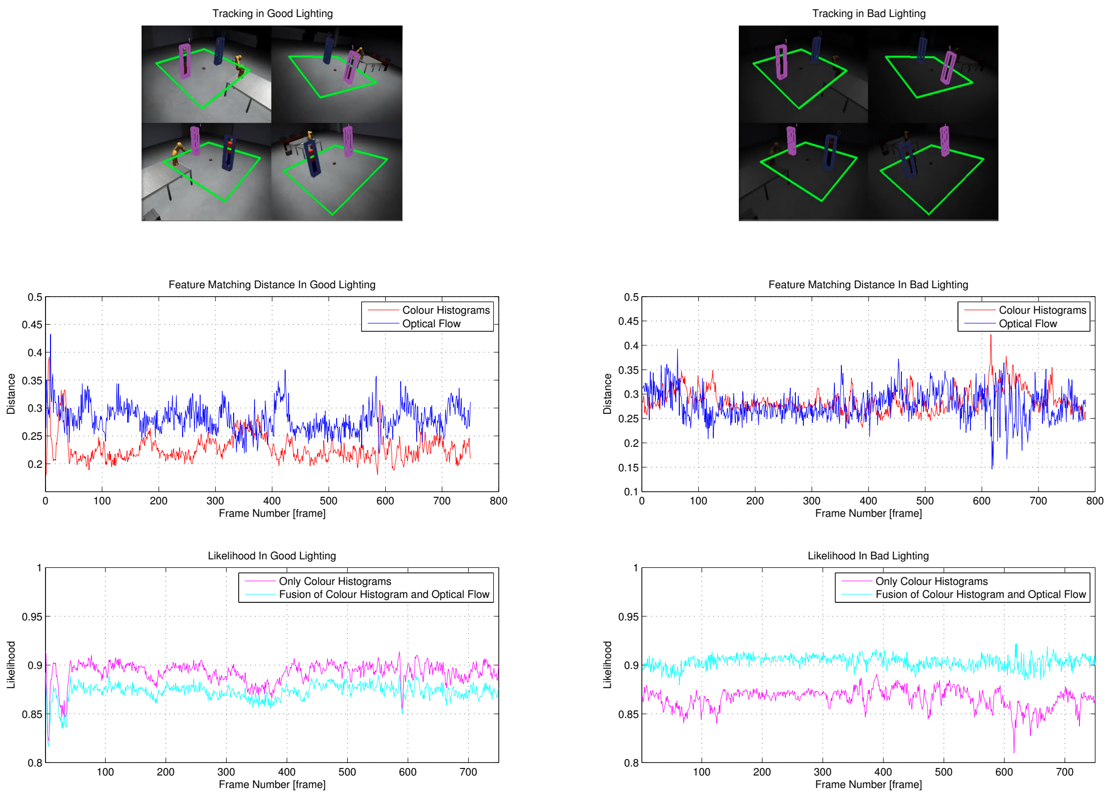


Figure 5.13: Experiment results for multi-modal fusion. The two images on the top show the tracking environments with good and bad lighting conditions. The second row shows the plot of the feature matching distance for colour (red) and optical flow (blue) modalities in each tracking environment. The third row shows the plots of the likelihood values computed with (light blue) and without (magenta) multi-modal fusion.

togram degrades in the bad lighting condition while the optical flow distance remains fairly constant. This shows the robustness of optical flow and the sensitivity of the colour distributions to changing lighting conditions. This also proves that optical flow information is a good supporting feature to be used under such circumstances. This claim is validated further in the likelihood plots obtained for the two lighting conditions. In each of the two scenarios, two likelihood plots are generated. The first represents the likelihoods computed using only colour information followed by likelihood computed through multi-modal fusion of colour and optical flow information. It can be observed that the likelihood without fusion degrades in bad lighting conditions while the likelihood with multi-modal fusion is much stronger. This is possible due to the machine learning module which detects the light change and adjusts the weights of the two modality accordingly to boost the fused likelihood. Initially the colour modality and optical flow were weighted 0.8 and and 0.2 respectively. After light change, in this particular experiment the weights of colour and optical flow were changed to 0.3 and 0.7 respectively. Therefore, the optical flow modality was more dominant in the bad lighting conditions due to its robustness toward changing lighting conditions. This proves that under bad lighting conditions the multi-modal fusion of colour and optical flow ensures robust and stable tracking results.

In order to validate the intelligent multi-modal fusion approach, an experiment was conducted, considering various aspects of the fusion process. The system was tested on scenarios described earlier such as abrupt changes in lighting conditions, stationary targets, targets moving very close to each other in the opposite as well as similar directions. In such scenarios, the intelligent multi-modal fusion is demonstrated by tuning the contribution of the individual visual modalities.

Fig. 5.14 illustrates the experimental results using the sequence introduced in Fig. 5.10. The tracked velocities and positions of each individual target is plotted. Together with this information, the contribution of the colour histogram and optical flow modalities are plotted for a single hypothesis (in this case the first). The plots show how the contribution of the modalities change depending on the motion behaviour of the targets with respect to each other. It can be observed that the optical flow contribution lowers when the target velocities are close to zero, indicating no motion. Similarly, the optical flow contribution is lowered when the distance between the targets gets lower than the threshold of 1 meter and the direction component of the velocities is less than a threshold of 50 degrees.

Fig. 5.15 demonstrates the intelligent multi-modal fusion using the sequence illustrated in figs. 4.12, 4.13, 4.14 and 4.15. This sequence contains two targets interacting with each other under drastic changes in lighting conditions. The experiment shows how the modality contributions change when the lighting conditions and motion behaviour of the targets change. It can be observed that when the lighting conditions degrade the contribution of the colour histogram modality is reduced and that of the optical flow is increased. Under these conditions the optical flow modality is further checked and suppressed under unfavourable condition such as stationary targets, targets moving close to each other in the same direction and during the background model update process.

The three experiments discussed in this section successfully validate the importance of the intelligent multi-modal fusion module in assisting towards robustness enhancement of the multiple human tracking system.

5.6 Discussion

This chapter formulated the experiments conducted in order to validate the multiple human tracking system. It introduced a methodology to generate ground truth through 3D modelling and simulations. Further, the system was validated through a set of experiments, both in virtual and real-world scenarios. Each experiment was aimed towards a certain aspect of the system to be validated. In the following section, the integration of the system into multiple real-world robotic scenarios will be discussed. Five demonstration scenarios will be introduced validating the generality of the system and its easy integration into diverse robotic applications.

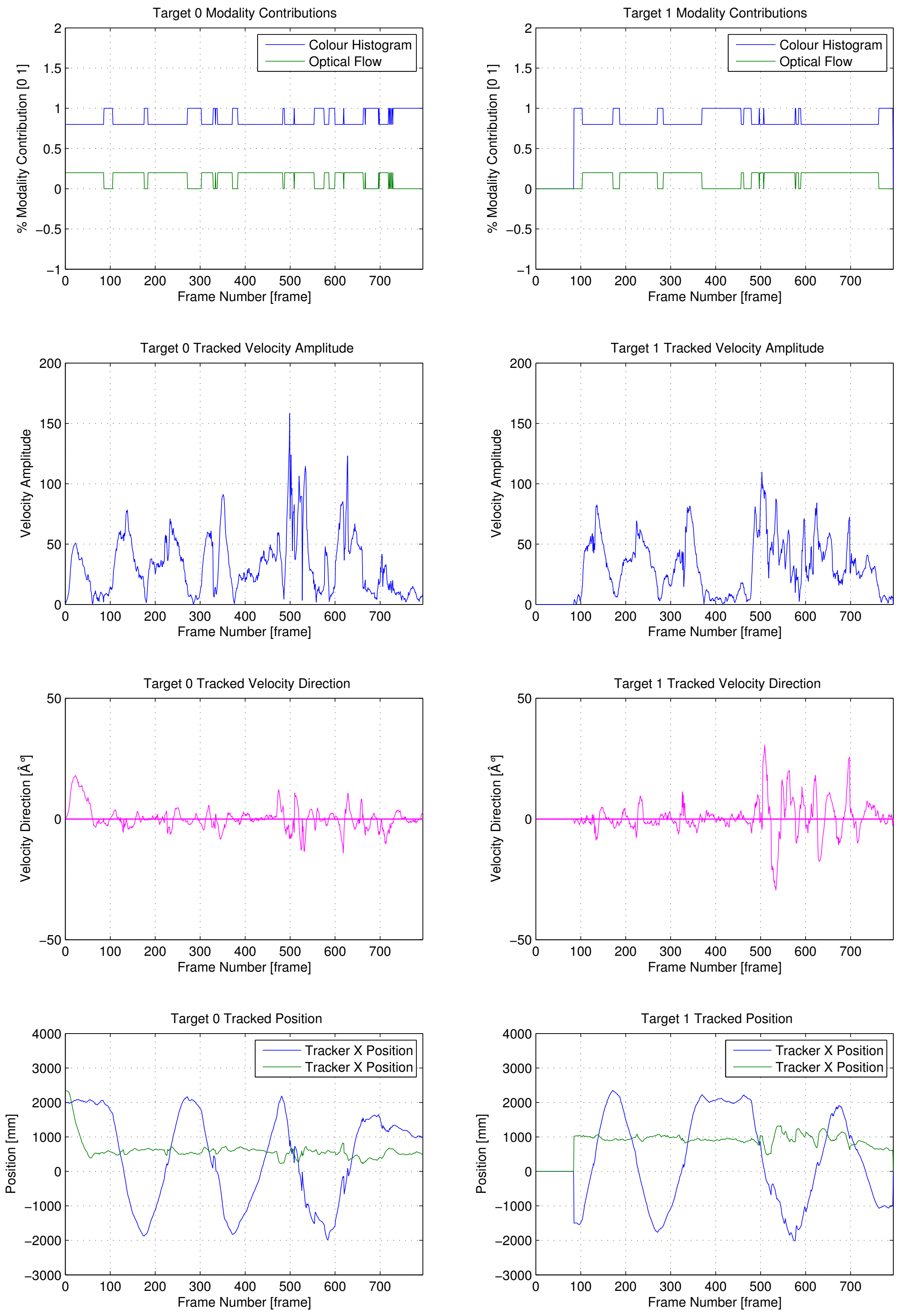


Figure 5.14: Experiment results for dynamic multi modal fusion with targets representing different motion behaviours. The scene consists of two targets, though their entry frame is different. Target two enters much later in frame 85. The first row represents the modality contributions of a single hypothesis in the sequence. The second row represents the velocity amplitude of the target. The third row represents the velocity directions. The fourth row represents the tracked positions.

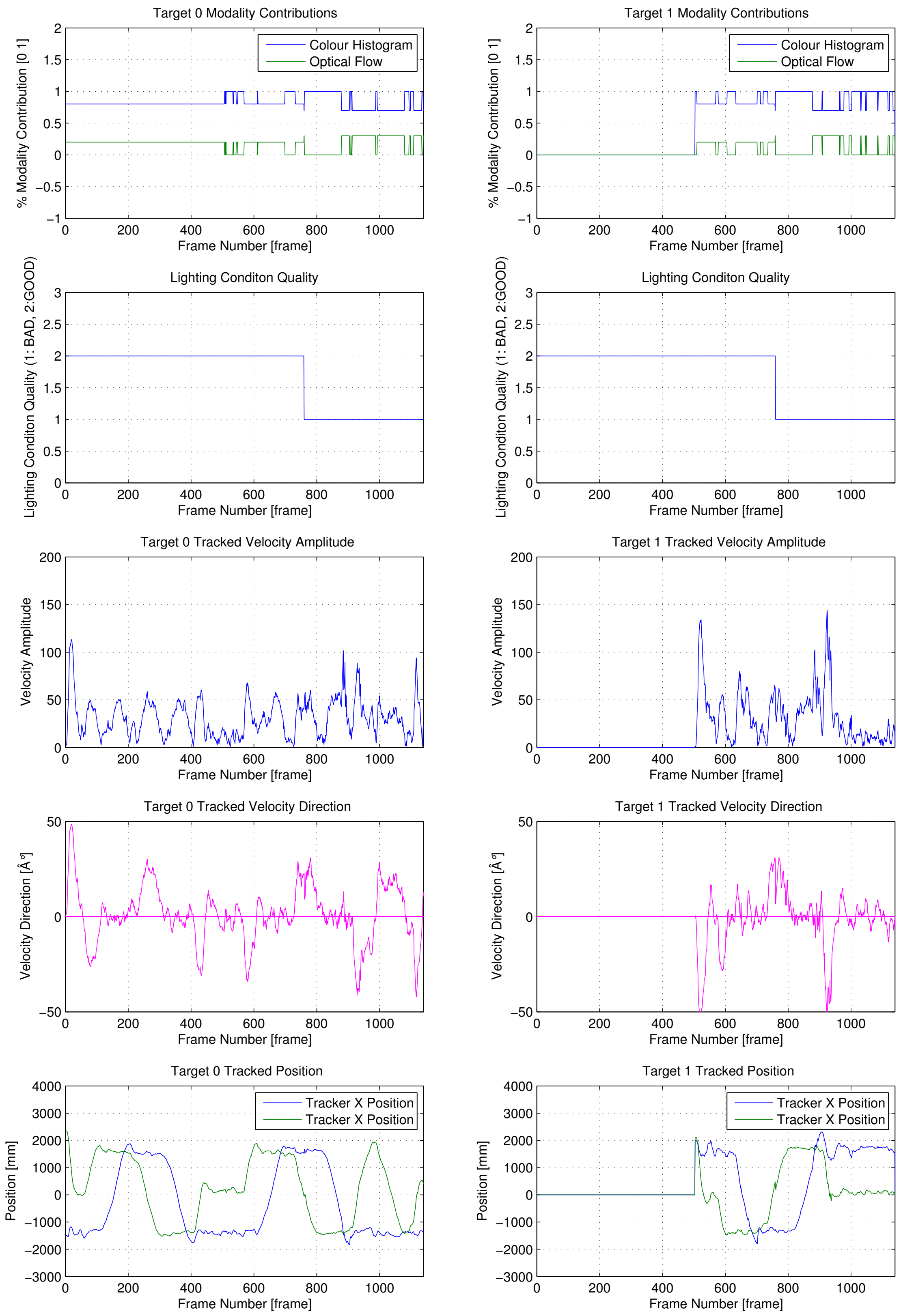


Figure 5.15: Experiments results in dynamic multi modal fusion under changing lighting conditions.

Chapter 6

Integration with Real-World Robotic Systems

6.1 Overview

In the preceding chapters, research and development of a multiple human tracking system was discussed. The primary focus was robustness enhancement using machine learning techniques. Various aspects of the system were evaluated through experiments in simulated and real environments. This chapter will focus on demonstrating and validating the usability of the system in real-world applications applied to robotic systems. A variety of applications will be discussed, demonstrating the easy and systematic integration of the multiple human tracking system with larger distributed systems involving robots. In addition, complementary vision systems which were developed as supporting modules in certain demonstrations will be introduced.

6.2 Multiple Human Tracking and Face Tracking for Visual Servoing using Robots

6.2.1 Overview

The multiple human tracking system combined with a face tracking system developed by the author, helps achieve visual servoing of human in indoor environments using robots. The system itself is very general and can be applied to a variety of applications. One such application is tracking moderators in a studio room to realize an automatic cameraman. The studio could be standard, virtual or virtual-augmented [10].

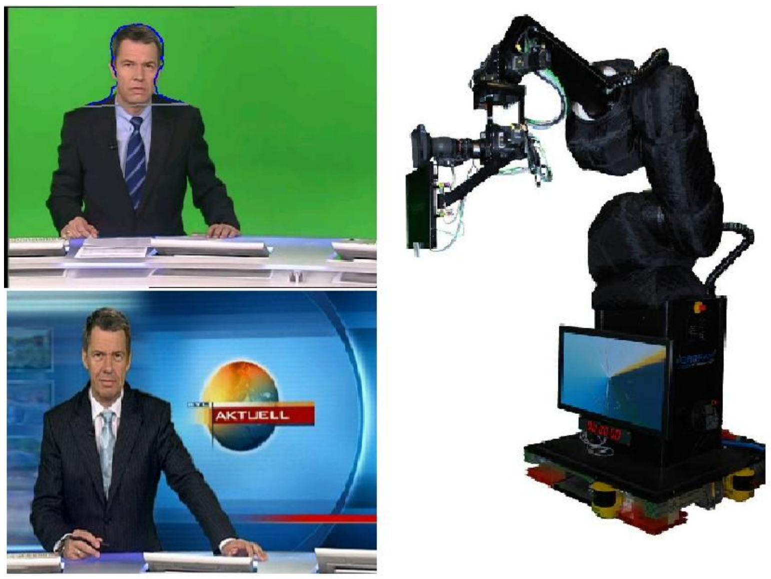


Figure 6.1: A virtual set for daily news broadcasting events. The upper and lower left pictures show the virtual studio. The right shows our robot camera system for broadcast automation. (*image courtesy: RTL Television Studio Köln, Germany, and Robotics Technology Leaders GmbH*).

Virtual TV studios have gained immense importance in the broadcasting area over the past years, and are becoming the mainstream way of broadcasting in the future. The present day technology also allows broadcasters to have virtual objects inside the virtual scene. A typical camera configuration for motion control consists of a pedestal housing a pan tilt unit. These systems have limitations in terms of degrees of freedom, motion smoothness and high costs of the external sensor-based tracker, used to recover the 3D pose of the camera.

Industrial robot arms can perform precise manipulation of TV cameras with high repeatability in large workspaces, using many degrees of freedom. Moreover, the main advantage of using a robotic system is that the 3D camera pose can be obtained free of cost through the robot kinematics, thereby eliminating the need for external trackers.

Robotic automation in TV studios can be pushed to a new high by imparting intelligence to the robot system. With this aim, a 2D face tracking system and the 3D multiple human tracking system have been integrated into a larger robotic system. It consists of robot driven cameras along with a wide range of hardware and software modules used in TV studios. The system is able to localize the moderator and keep her/him within the screen while sitting or freely walking inside the studio.

Another important feature is the automatic positioning of the moderator during different run-down scenes according to a pre-determined region of interest. For example, when switching from a scene with the moderator in the centre to one where visual graphics need to be rendered, it is necessary to hold the moderator on the left (or right) part of the scene. This can be achieved using the tracking results with almost no need for human intervention. The evolution of this system has been precisely documented in [10], [145] and [11], showing its integration into a distributed robot system called *RoboKam* [146].

The system consists of two parts: 1. a 2D face tracker, operating on each robot, localizes the moderator in position (x, y), scale h and rotation θ within the field of view of the TV camera, allowing the robot to hold the moderator in a desired region in the scene, with the desired zoom and focus.

2. The multiple human tracker which localizes the target over the entire studio floor w.r.t. 3D translation (x, y, z). The 3D tracker performs important tasks namely:

- Initialize each robot camera, so that they can bring the target in the respective fields of view independently of their initial positions
- Re-initialize the face trackers in case of a target loss, to recover their 2D locations
- Initialize zoom and focus control of each robot camera

In the face trcker, the target is modelled as a fixed omega-shape along with a frontal picture of the person's appearance.

Figure 6.2 gives an overview of the complete system. Each tracker is integrated into the robotic camera system through a modular middle ware[5], which was developed for communication and configuration of studio devices.

It is possible to have more than one robot camera in the same studio. However, there exists only a single common overhead tracking system. The overhead tracker uses ceiling mounted fire wire cameras in a stereo set-up, in order to compute the 3D pose of the moderator. These cameras are calibrated with respect to a common *zero point* for the individual intrinsic and extrinsic parameters.

For the 2D face tracker a Kalman filter [106] is used with a likelihood working on the contracting curve density *CCD* modality. The *CCD* algorithm [147, 148] provides a contour tracking modality, based on separation of local colour statistics. In an object tracking context, separation takes place between the object and the background regions, across the screen contour projected from the shape model onto a given camera view, under a given pose hypothesis (prediction).

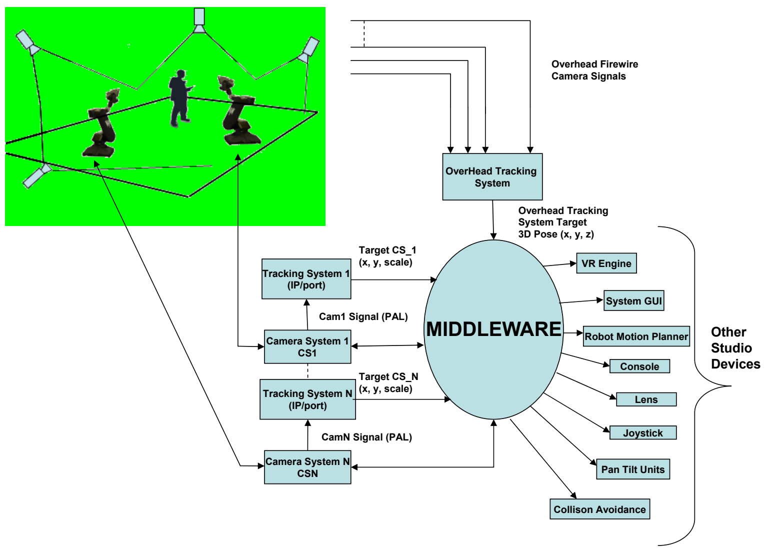


Figure 6.2: Block diagram of the system architecture. The middle ware [5] integrates multiple robotic camera devices and tracking systems. A multiple human tracking system (overhead tracker) is used to initialize the system at start-up and in cases of target loss. Since each camera device is calibrated to a given world point, the tracking information can be fused. The middle ware also connects to the virtual reality engine, system configuration modules, the operator GUI, the motion planning module etc.

Concerning computational resources, software for each camera system runs on a separate PC and obtains the TV camera picture through a frame grabber. The multiple human tracking system, also referred as the overhead tracker, uses stereo Fire-wire cameras with wide angle lenses for covering the complete studio floor.

6.2.2 2D Face Tracking of Moderators

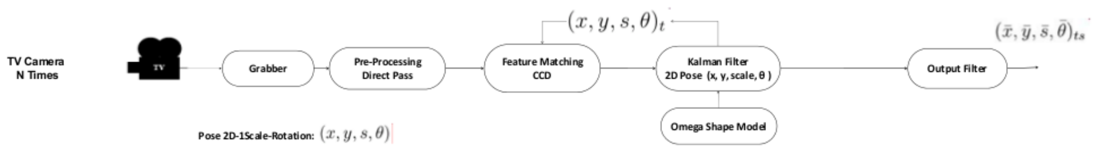

TV Camera N Times
TV
Grabber
Pre-Processing Direct Pass
Feature Matching CCD
$(x, y, s, \theta)_t$
Kalman Filter 2D Pose (x, y, scale, θ)
Omega Shape Model
$(\bar{x}, \bar{y}, \bar{s}, \bar{\theta})_{ts}$
Output Filter
Pose 2D-1Scale-Rotation: (x, y, s, θ)

Figure 6.3: Steps of the 2D person tracking pipelines: the top pipeline uses a 2D pose with 4 degrees of freedom: x, y, scale and rotation; A Kalman filter incorporating a feature-based contracting curve density matching as the underlying objective function/modality; This approach is based on an omega shape model.

Similar to the 3D multiple human tracker, the tracking pipeline is designed and implemented following an architecture inspired by the *OpenTL* Library [2]. Fig. 6.3 describes the complete pipeline.

The person tracker holds a state-space representation of the 2D model pose, given by a planar translation (x, y), scale h and rotation θ of the respective model in the image plane. The sample points on the contour model itself are not deformable but transform with respect to the model pose. The Kalman filter provides the sequential prediction and update of the respective *2D states* $s = (x, y, h, \theta, \dot{x}, \dot{y}, \dot{h}, \dot{\theta})$.

6.2.2.1 Pre-processing

The sensor data for the person tracker is obtained from the TV camera in a raw PAL format which is further converted to RGB 444 through a frame grabber. In CCD, the pre-processing step is absent, since the colour data of the image is directly used by the feature matching module, in order to collect local statistics and optimize the pose. Therefore, the pre-processing function merely copies the input image to a local storage for the other modules.

6.2.2.2 Tracker prediction

The Kalman filter generates a prior state hypothesis s_t^- from the previous state (s_{t-1}) by applying a Brownian [149] motion model. The Brownian motion model although very simple, is suitable as it provides a Gaussian distribution v_t around the current state which helps keeping track of the target when it is not moving much as well when it exhibits motion at acceptable speeds. If the target moves fast then, more sophisticated motion models are required, but this situation normally does not arise in TV Studio environments.

$$s_t^- = s_{t-1} + v_t; \tag{6.1}$$

6.2.2.3 CCD Likelihood for Feature Level Matching

In the original CCD algorithm [147], local areas for collecting colour statistics are given by regions around each contour sample position, on each side of the contour. In order to simplify the computation, as suggested in [147], first points are sampled along the respective normals, statistics are collected, and afterwards each statistic is *blurred* with the neighbouring ones (Fig. 6.4). This is fully equivalent to the initial process, but much more computationally convenient.

From each contour position h_i, foreground and background colour pixels are collected along the normals n_i up to a distance L, and local statistics up to the 2^{nd} order are estimated as:

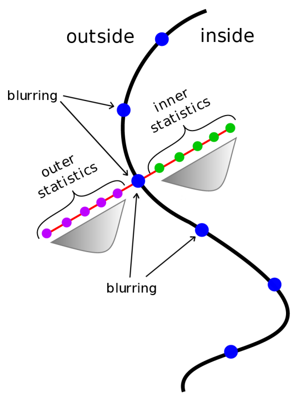


Figure 6.4: Contracting curve density tries to maximize the separation of colour statistics of a given inside and outside object region. Therefore the algorithm samples pixels along the normals for collecting local colour statistics.

$$
\begin{aligned}
\nu_i^{0,B/F} &= \sum_{d=1}^{D} w_{id} \\
\nu_i^{1,B/F} &= \sum_{d=1}^{D} w_{id} I(h_i \pm L\bar{d}n_i) \\
\nu_i^{2,B/F} &= \sum_{d=1}^{D} w_{id} I(h_i \pm L\bar{d}n_i) I(h_i \pm L\bar{d}n_i)^T
\end{aligned}
\tag{6.2}
$$

with $\bar{d} \equiv d/D$ the normalized contour distances, where the $\pm$ sign is referred to the respective contour side (for the background region), and image values I are 3-channel RGB colours. The local weights w_{id} decay exponentially with the normalized distance $\bar{d}$, thus giving a higher confidence to observed colours near the contour. For detailed explanation of equations please refer to [147].

Single-line statistics are thereafter *blurred* along the contour, providing statistics distributed on local areas

$$
\tilde{\nu}_i^{o,B/F} = \sum_j \exp(-\lambda |i-j|) \nu_j^{o,B/F}; \; o = 0, 1, 2. \tag{6.3}
$$

and finally normalized

$$\begin{aligned}\bar{I}_i^{B/F} &= \frac{\tilde{\nu}_i^{1,B/F}}{\tilde{\nu}_i^{0,B/F}} \\ \bar{R}_i^{B/F} &= \frac{\tilde{\nu}_i^{2,B/F}}{\tilde{\nu}_i^{0,B/F}}\end{aligned}$$

in order to provide the two-sided, local RGB means $\bar{I}$ and (3×3) covariance matrices $\bar{R}$.

The second step involves computing the residuals and Jacobian matrices for the Gauss-Newton pose update. For this purpose, observed pixel colours $I(h_i + L\bar{d}n_i)$ with $\bar{d} = -1, ..., 1$, are classified according to the collected statistics Eq. 6.4, under a fuzzy membership rule $a(x)$ to the foreground region

$$a(\bar{d}) = \frac{1}{2}\left[erf\left(\frac{\bar{d}}{\sqrt{2}\sigma}\right) + 1\right] \tag{6.4}$$

which becomes a sharp $\{0, 1\}$ assignment for $\sigma \to 0$; pixel classification is then accomplished by mixing the two statistics accordingly

$$\begin{aligned}\hat{I}_{id} &= a(\bar{d})\bar{I}_i^F + (1 - a(\bar{d}))\bar{I}_i^B \\ \hat{R}_{id} &= a(\bar{d})\bar{R}_i^F + (1 - a(\bar{d}))\bar{R}_i^B\end{aligned} \tag{6.5}$$

and colour residuals are given by

$$E_{id} = I(h_i + L\bar{d}n_i) - \hat{I}_{id} \tag{6.6}$$

with measurement covariances $\hat{R}_{id}$.

Finally, the $(3 \times n)$ derivatives of E_{id} can be computed by differentiating Eq. 6.6 and Eq. 6.4 with respect to the pose parameters s[1]

$$J_{id} = \frac{\partial \hat{I}_{id}}{\partial s} = \frac{1}{L}(\bar{I}_i^F - \bar{I}_i^B)\frac{\partial a}{\partial \bar{d}}\left(n_i^T \frac{\partial h_i}{\partial s}\right) \tag{6.7}$$

which are stacked together in a global Jacobian matrix $\mathbf{J}_{ccd}$.

The final state is then computed using the Gauss Newton update to object level:

$$s = s + \Delta s \tag{6.8}$$

[1]As in [148], the dependence of R_{id} on p is neglected while computing the Jacobian matrices.

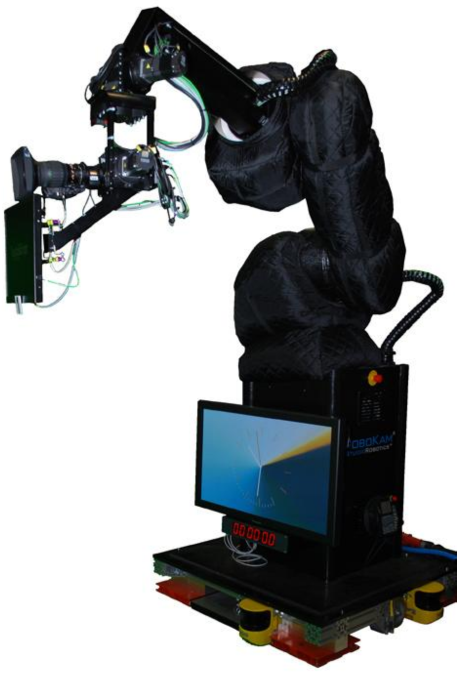


Figure 6.5: The robot system installed at the ZDF Studios in Mainz, Germany(*image courtesy: ZDF Television Studio [6] Mainz, Germany*)

$$\Delta s = J^{+}_{ccd} E_{ccd} \tag{6.9}$$

Finally, when the termination criteria is satisfied ($\Delta s \approx 0$)

$$s^{*} = Z_{ccd} \tag{6.10}$$

6.2.3 The Robot System

The robot system replaces traditional pedestal driven camera in normal and virtual studios with industrial robots capable of manipulation with up to 8 degrees of freedom in addition to zoom and focus control. The robot itself is a 6 axis Stäubli RX100L industrial arm [150] modified along joint four with a unique tilt-pan-tilt to suit dynamic camera movements and precise positioning. The optionally available pan-tilt or tilt-pan-tilt heads offer additional flexibility [146]. An additionally mounted portable and self-propelled air cushion platform allowing quiet and quick positioning in the newsroom [146].

The robot ensures maximum flexibility, speed, precision and efficiency in the Studio Automation. It is optimized for use in VR environments and live scenarios. It offers highest security through integrated anti-collision system. Fig 6.5 shows the robot system installed at the ZDF Studios [6] in Mainz.

6.2.4 Self Steering Studio

Combining the 2D face tracker and the 3D multiple human tracker an automatic control of the TV camera using industrial robots is achieved. With this functionality the presenter can control the camera himself through an user interface taking the cameraman and producer out of the loop of decision making. This self-steering capability results in less errors during live production. In a TV Studio set-up, production is done on a scene-by-scene basis using a run-down. The robot system can react intuitively to the switch from one scene to another. This can be achieved automatically using the aid of the visual trackers with almost no need of human intervention. Different scenes require the moderator to appear in different regions and with different zoom and focus settings. This information can be combined with the run-down information in order to enable a complete automatic switch of the camera position and zoom/focus in order to hold the moderator within the region of interest of the current scene.

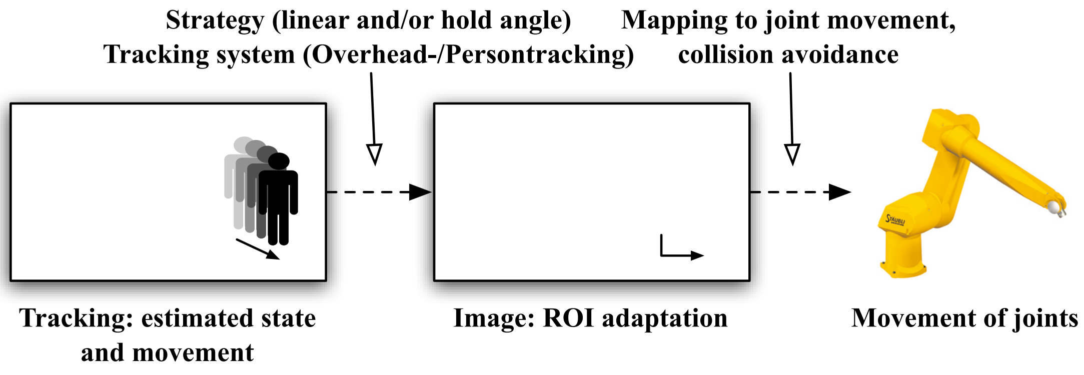

Strategy (linear and/or hold angle)
Tracking system (Overhead-/Persontracking)
Mapping to joint movement, collision avoidance
Tracking: estimated state and movement
Image: ROI adaptation
Movement of joints

Figure 6.6: Robot control methodology

In order to keep the target in a predefined ROI it is necessary to generate the relative motion parameters for the corresponding robot system. This is achieved by using this pixel-level information from the tracking system and converting them to 3D movement commands in world space. For this conversion different additional parameters have to be considered namely:

- Region of interest: It is the desired area in the image space of each camera system, where the target should be held, e.g. in weather broadcasts the target appears usually on the right hand side of the scene.
- Balance speed: The speed of the robot is specified for X, Y, pan and tilt as an absolute percentage ranging from $[-100\%; 0\%; 100\%]$. This speed is used by the camera system in order to get the target back into the ROI (region of interest). The actual speed used also depends on the distance between the target and the ROI. The effective speed is proportional to the calculated distance providing smoother motion properties.

As shown in Fig. 6.6, the 3D Cartesian movement is computed out of the X, Y, pan, tilt speeds. Two different operation modes for the joint control are proposed:

- Normal mode: The movement of the robot is limited to a linear motion in the X and Y direction, and angular motion for Pan and Tilt in order to get the target back into the desired ROI.
- Hold-angle mode: The movement of the robot is done in 2 phases: In the 1st phase, the robot uses only Pan and Tilt to bring the target back into the ROI, and in the 2nd phase it uses linear X and Y motion to compensate to hold a predefined viewing angle of the camera system.

In the laboratory experimental set-up a Stäubli RX-90 [150] was used. The robot has been illustrated earlier in Fig. 6.1. It is equipped with the CS-8 Stäubli controller. The kinematics computation provided by the controller are bypassed and the kinematics and motion trajectories are computed externally. This information is sent to the low level controller thorough a low level interface [151]. The robot controller runs a Real Time Extension of Linux. For the PID control we rely on the CS-8 controller shipped by the robot manufacturer.

6.2.5 Demonstration and Results

The system was evaluated in real existing TV studios used for virtual reality TV productions. For this purpose standard Desktop PCs with $2.4GHz$ Intel Pentium IV and standard graphics hardware have been used to realize the tracking for each camera system and the multiple human tracking system, all running on Linux operating system.

Fig. 6.7 show some experimental results of the 2D person tracker. The tracker keeps good track of the person during this sequence. It also illustrates the 3D multiple human tracker.

The use of CCD in the face tracker, assisted by the multiple human tracker has made possible a robust localization of the target in terms of stable scale needed for zoom control which is illustrated in the top-right of Fig. 6.7.

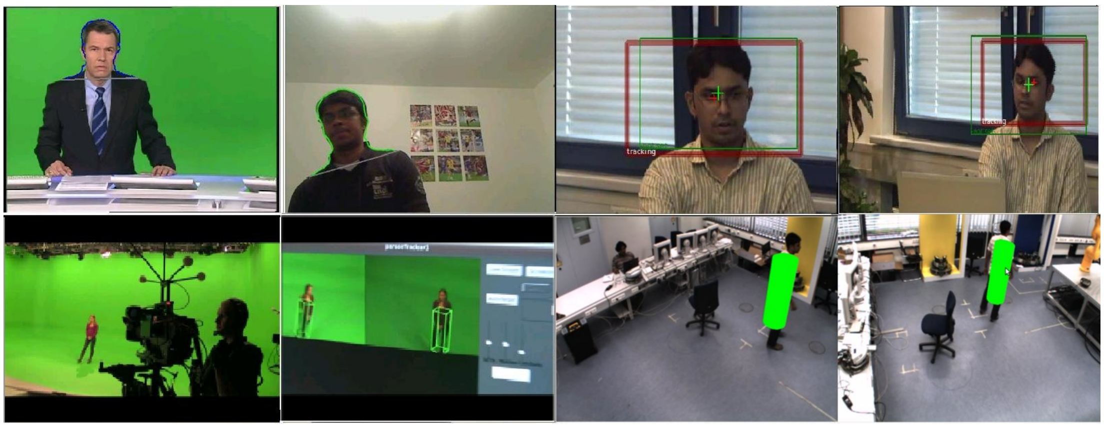


Figure 6.7: Experimental results Upper row: Real-world news studio using a green wall background and constant light conditions along with cluttered testing conditions. Bottom row: 3D multiple human tracker tracking the moderator using stereo cameras.

Figs. 6.8, 6.9, 6.10 and 6.11 demonstrate the self steering capability of the system. In such a system the moderator can control the camera motions according to the run down sequence. In studios this is done together by the producer and the cameraman during different parts of the show where the moderator needs to be in different regions of the scene. In manual control the chances of human errors are higher due to co-ordination errors. With a vision based robotic control system described in this section, the different positions of the moderator during the run down can be programmed offline and the moderator can direct the camera system to react in order to obtain the required scene with the moderator in the desired position by simple a push of a button. This reduces the chances of error to a great extent since the moderator can himself control the show. In order to demonstrate this, a news scenario was created where the news reader controls the camera positions in different run down steps and the camera system react automatically using the information from the visual trackers.

Fig. 6.8 illustrates a typical newsroom scenario. The TV camera is controlled by RoboKam and the on screen talent has access to the system GUI. The moderator can control the run down sequence himself using the GUI thereby eliminating the need for communication and co-ordination with the camera man and producer.

Fig. 6.9 show a scenario during the run down where the position of the moderator is required

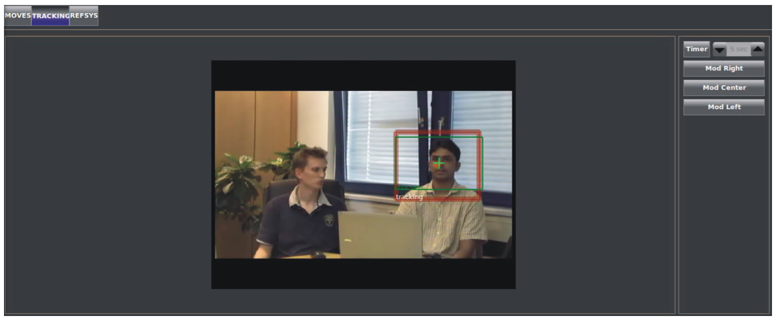


Figure 6.8: Self steering camera system with tracking results shown by red rectangle.

to be in the centre of the screen as shown by the green rectangle. The red rectangle represents the tracking results. It can be observed that as soon as the moderator chooses the predefined configuration for the current scene in the run down the robotic camera system reacts using the tracking results and moves the camera smoothly until the moderator occupies the desired position in the scene. This scenario where the moderator is in the centre of the screen is common during news shows when the content has to be read out without additional visual aid.

Figs. 6.10 and 6.11 demonstrate other run down scenarios where the moderator is required to be present in the right or left parts of the screen. This is typically seen during news shows when the moderator is reading out the news along with some visual information on his or her side. This information is rendered into virtual screens. It can be observed how the robotic camera system reacts to the run down change such that the moderator is filmed in the desired region of interest.

The trackers perform in real-time with approximately $15 - 20\ fps$, which is sufficient for the robot controller which requires an update every $200ms$. The image resolution is 640×480 pixels. Fig. 6.12 illustrates a lab robot with a pan-tilt unit and a TV camera. The robot tracks the target while the target is moving and interacting with other people in the scene.

The tracker performs in real-time. It communicates with the robot controller through a common middle ware using TCP Sockets. The robot controller send commands to the robot with a cycle time of 4 msecs and requests data from the tracker every 100 msecs. The tracker generates data at 15-25 fps thereby fulfilling the data request from the robot controller. There is indeed some delay (approx 200 msecs) when the robot reacts to fast movements of the moderator, but in TV Studio environments and especially in virtual sets for news production, this situation is very rare and usually avoided by the producer. Most of the motion control takes place during switching of scenes, where a small delay can be tolerated during production.

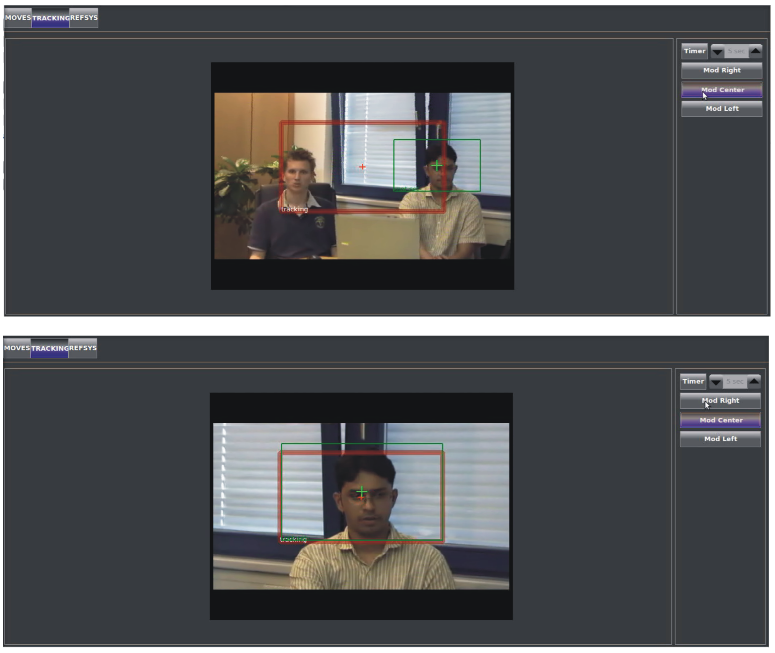


Figure 6.9: Self steering camera system with target position to be achieved is centre of the screen shown by green rectangle.

A complete video demonstration is available under [152, 153] or `http://www.youtube.com/watch?v=VDtIAByFJog` and `http://www.youtube.com/watch?v=xpJWHYinyrE`.

6.3 Human Tracking for Interaction with 3D objects in Virtual-augmented Reality TV Studios

6.3.1 Overview

Virtual reality TV studios have advanced to a level where, inclusion of augmented reality within virtual studio environments has become a possibility. The modern day studio is not only capable of rendering images of the virtual studio environment in real-time, but also able to accommodate 3D objects within the virtual world and render them with the camera perspective. Such virtual objects are used as a visual aid for the viewer and the moderator.

Although virtual objects enhance the aesthetics of the show, there are limitation on the possibilities of interaction between the on screen talent and virtual objects. A virtual object has a $3D$ position in the virtual scene but the position of the on screen talent in the studio environment is not know. During keying of the TV camera image with the rendered virtual

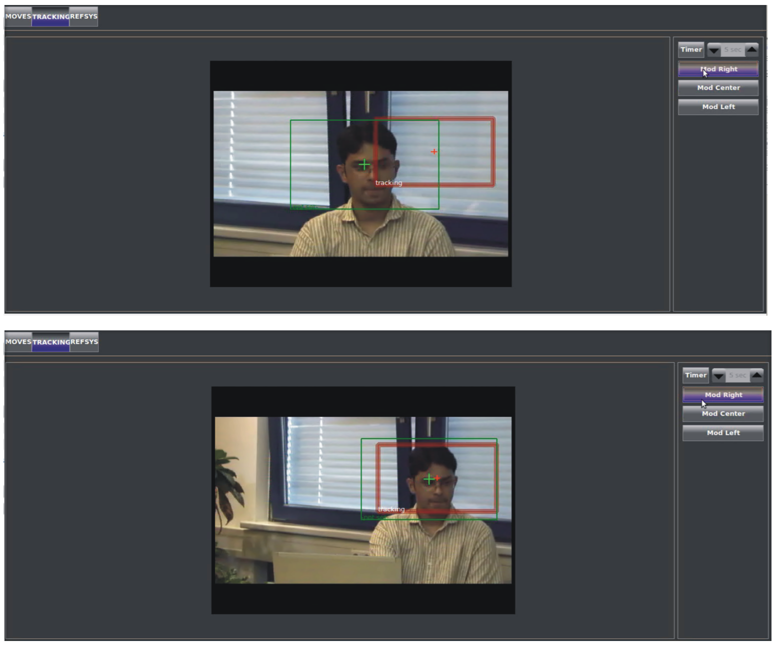


Figure 6.10: Self steering camera system with target position to be achieved is to the right of the screen shown by green rectangle.

environment, the foreground pixels from the TV camera image overwrite the rendered scene. In such circumstances, even if the on screen talent is behind the virtual object, he or she will always be rendered over the virtual object producing an ambiguous image. Therefore, in the current day studios the interaction of the on screen talent with virtual objects is restricted and overlaps are not permitted. The multiple human tracking system can be used to overcome this limitation and allow a more intuitive way of interaction between the on screen talent and the virtual object. This is possible by estimating the 3D position of the on screen talent and using it to decide if the virtual object should be rendered as foreground or background by the rendering engine. In this manner the on screen talent and the virtual object are rendered in the correct order without limiting the interaction between them, resulting in a impressive 3D experience.

6.3.2 System Architecture

Fig. 6.13 describes the system architecture. It consists of different modules communicating with each other in order to produce the final broadcast content. The different modules are described hereunder:

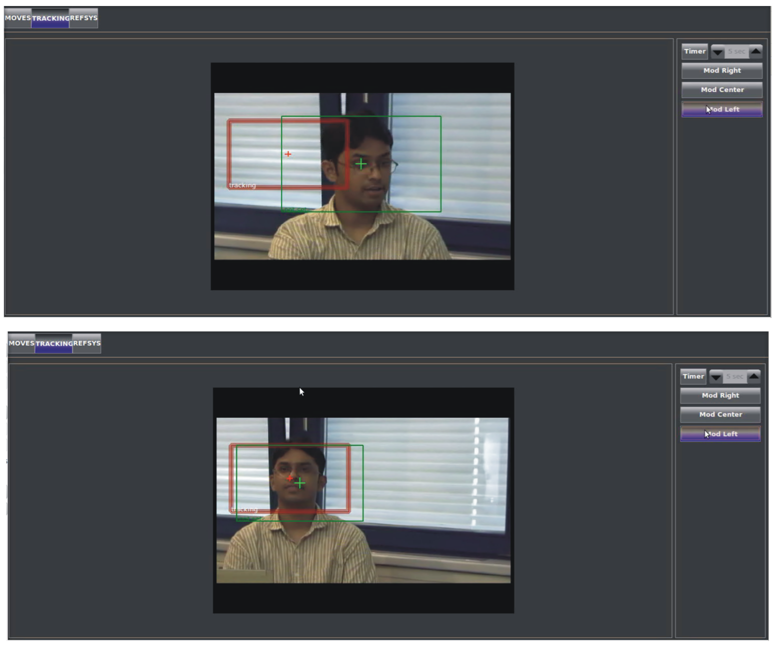


Figure 6.11: Self steering camera system with target position to be achieved is to the left of the screen shown by green rectangle.

- **VizEngine:** This is the 3D engine [154], which provides the virtual environment of the studio and serves as the core module of the virtual studio in terms of management, control and monitoring of all the rendering modules.

- **3D Objects:** This is the module where the 3D objects designed are stored in terms of their shape and appearance [154]. It contains a detailed description of the geometry to scale, texture information, calibration data and pose data.

- **Foreground/Background Context Switch:** It uses the pose data of a 3D object and the position data of each human in the studio environment in order to determine if it should be rendered as foreground or background. This process is performed for each 3D object.

- **3D Multiple Human Tracker:** This system as described in the previous chapters tracks the on screen talent in real-time returning the position of the target in 3D. This information is used to generate the stereo images from the image obtained from the TV camera.

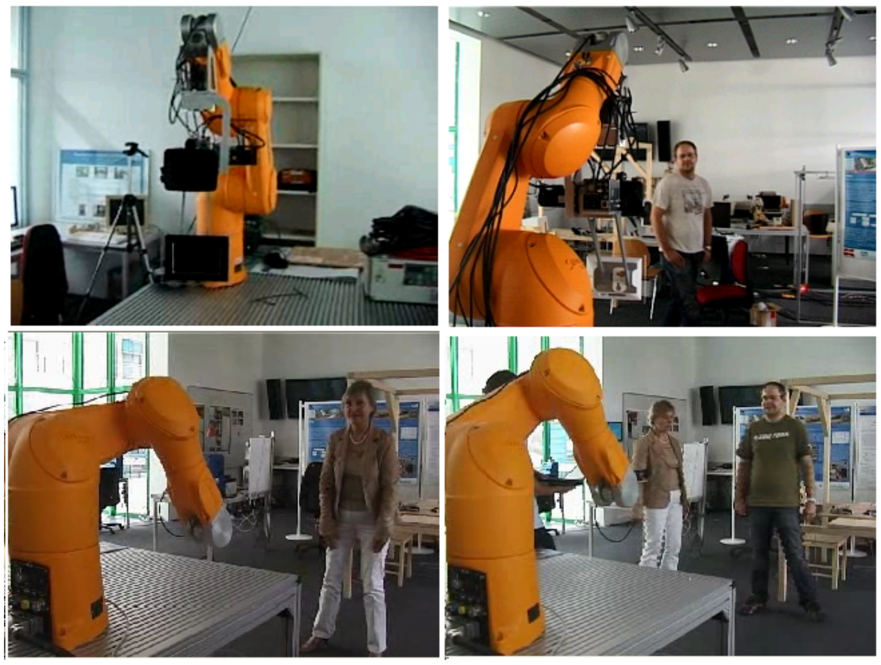


Figure 6.12: In-house testbed: Example sequence with the robot controller in action. Here the control of the robot arm as well as the pan tilt unit have to form a joint control to provide smooth jitter-free trajectories.

- **VizRenderer:** This module uses the virtual world design of the studio and produces real-time images of the virtual world with respect to the perspective of the TV camera. In order to generate such images, the pose of the TV camera is required in 6 DOF, which is obtained from the camera pose tracker. After rendering the virtual studio environment it performs the rendering of the 3D objects according to their context information.
- **Camera Pose Tracker:** This module tracks the TV camera pose in real-time with a very high degree of precision. This information is used by the renderers to produce the correct virtual images of the studio in terms of the camera perspective. The most commonly used trackers are infra red camera based or Odometry based. In case of robotic system this module is highly simplified since the camera pose can be easily computed through the robot kinematics.
- **TV Camera:** It is the camera used to film the on screen talent in the green box. The camera can either be pedestal mounted, steady cam or a robotic camera system.
- **Keyer:** The function of the Keyer is to mix the images from the TV camera and the

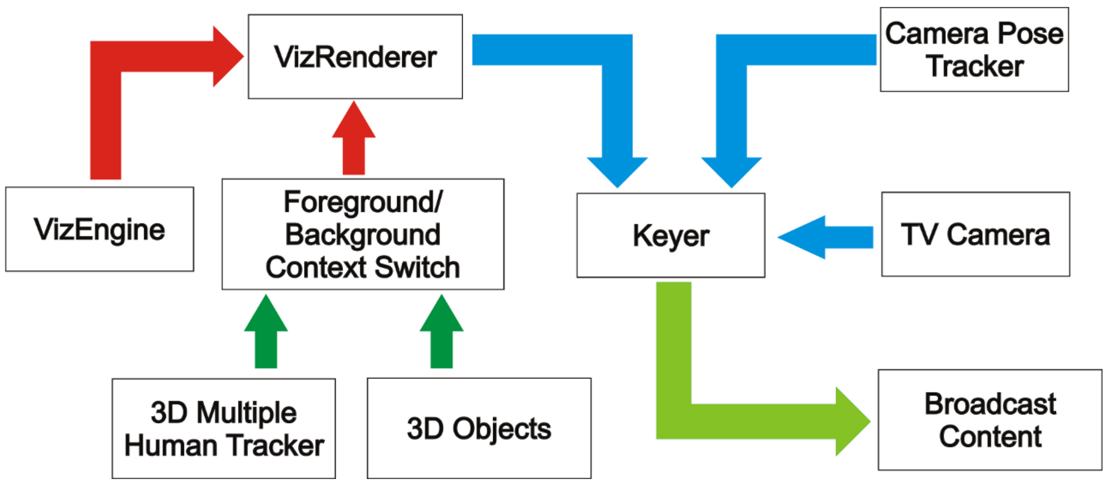


Figure 6.13: System Architecture

render. The mixing is performed in a way that each pixel representing the background in the camera image is replaced by the corresponding pixel of the image generated by the renderers. The background pixels are represented by the colour of the actual studio environment which is green when a green box is used.

- **3D content:** The output from the Keyer consists of the final broadcast content, in this case the moderator freely interacting with virtual objects in a virtual environment.

6.3.3 Demonstration and Results

Fig. 6.14 shows a scenario in which the system was tested. The test was performed in a lab environment using the rendering engine [154]. A simple green box environment was created and the virtual environment representing a soccer field was created using Viz Artist [154]. The camera system used was a RoboKam pan-tilt, housing the TV camera. The virtual object was modelled as a soccer ball and was rendered in the middle of the soccer field calibrated to the lab room. Two Fire-wire cameras were mounted on the ceiling to aid the human tracking system. These cameras were calibrated to the global origin in the lab room.

The human tracking system tracks the human in the room in real time. This information is used to decide if the human occludes the soccer ball or vice versa. It can be observed that when the human is closer to the camera as compared to the virtual object, the human occludes the soccer ball. On the other hand, if the soccer ball is closer then the human is occluded by the soccer ball.

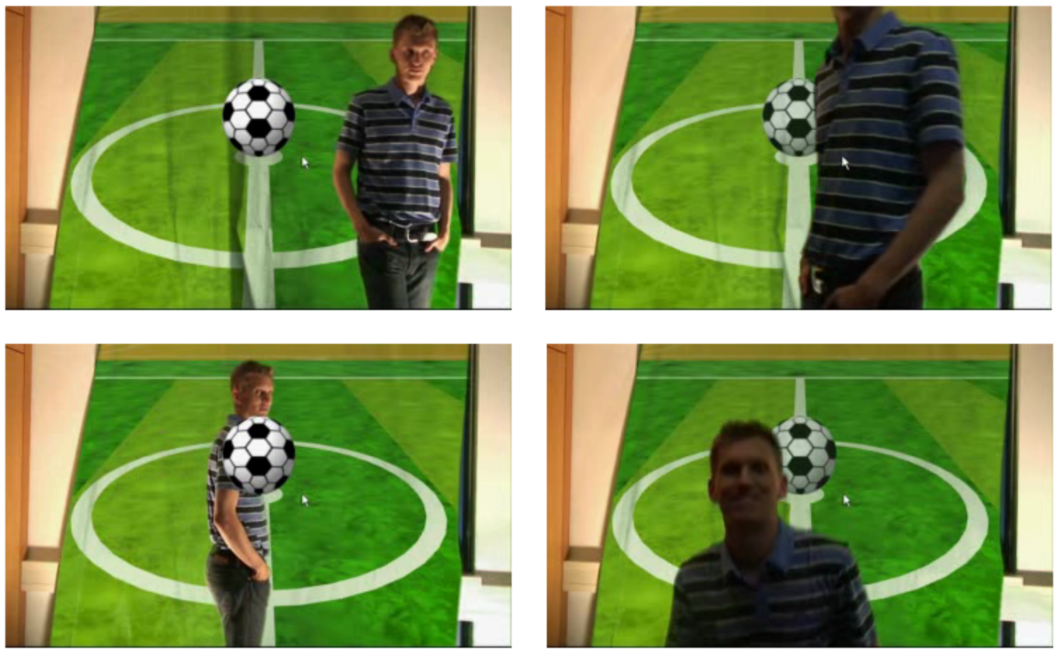


Figure 6.14: Interaction of human with virtual object maintaining the correctness of the rendered scene.

It is evident from the test performed that the multiple human tracking system makes the interaction of the on screen talent with the virtual object as it would be in reality. This results in an enhanced $3D$ experience. A complete video demonstration is available under [155] or `http://www.youtube.com/watch?v=-VfOPVnjHRs`.

6.4 Human Tracking for Stereo-scopic Rendering in HDTV Studios

6.4.1 Overview

In this section, a unique application is described in the area of computer graphics and virtual reality which requires the use of the multiple human tracking system. *HDTV* technology reaching new highs has seen vast improvement in the quality of content. In addition, there has been new developments in the hardware area. 3D content is not any-more restricted to cinema halls but has reached homes with the arrival of 3D televisions. This has motivated broadcasters to explore the possibilities of filming and broadcasting the current day contents in 3D. One unique application which has emerged is *Stereoscopic Rendering in Virtual HDTV Studios*. Virtual studios are used in production of daily news, talk shows etc. They use the green box technology to key a synthetically generated 3D environment over a moving TV camera image which films the show conducted in the studio. Although the studio design is in 3D, when rendered it appears in 2D on the screen. Stereoscopic rendering on the other hand, provides a true 3D experience. Instead of viewing the usual flat TV picture of the 3D environment, two real-time renderers create stereoscopic images that are used to create an illusion of depth. This technology already exists in cinema, where it is produced during post-processing. However, in virtual TV studios the content needs to be produced in real-time and in many cases for live broadcast. The real-time renderers are capable of handling this. Using special 3D glasses, the viewers can see moving 3D content on a typical TV screen. The content can also be transmitted in 3D and can be viewed on any 3D TV screen.

Stereoscopic rendering provides a 3D illusion on a flat screen, however the actual TV camera which films the green box with the moderator does not generate stereo images and is moving constantly. Therefore, to produce the same 3D illusion of the moderator together with the rendered scene, generating stereo images from the TV camera is essential. In order to achieve this in synchronisation with the rendered scene and maintaining zero parallax, the 3D position of the moderator has to be known. This is where the multiple human tracking system is used. Although the tracker can track multiple humans, in this demonstration only one moderator is present in the scene.

This technology does not require 3D cameras normally needed for 3D demonstrations. The stereoscopic rendering and the multiple human tracker ensures that the on-camera talent remains properly situated within the 3D scene for an impressive 3D illusion.

Some broadcasters such as Plasamedia [7], ESPN [156], NFL [157] and BSkyB [158] have already been on air with Stereo 3D content, showing a great amount of potential in this area. The following sections shows the importance of the multiple human tracking system in realizing this technology.

6.4.2 System Architecture

The multiple human tracking system plays a pivotal role in the larger system. The two renderers produce 3D content of the virtual world to be keyed over the green box at a perspective determined by the pose of the TV camera used to film the real-on screen talent. With two renderers positioned in the right manner, 3D illusion is created on a simple flat screen and can be experienced using special 3D glasses. However, the real content filmed by the TV camera which involves the on screen talent is still in 2D. Therefore, in order to provide a complete 3D illusion, where also the on screen talent can be viewed in 3D, it is necessary to modify the 2D content obtained from the TV camera. This can be achieved by generating stereo images out of the single image obtained from the TV camera in order to satisfy the stereo configuration of the renderers and compensate for zero parallax such that the on screen talent can be viewed with the right depth. This task is not trivial. In order to realise this, the 3D position of the moderator is required to be known in real-time. This task is performed by the multiple human tracker.

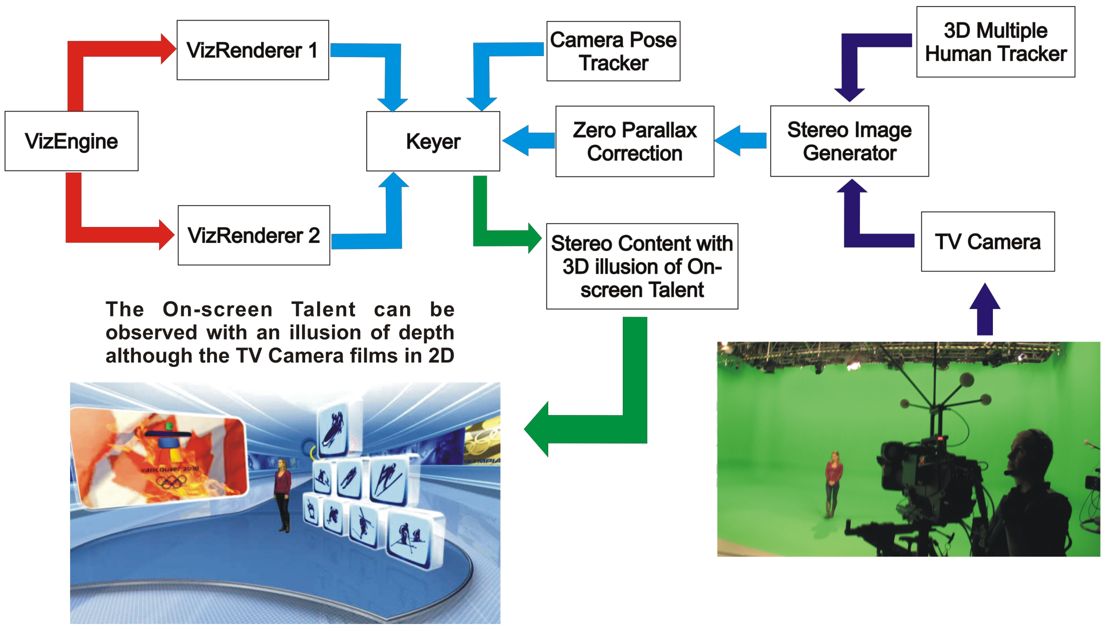

VizRenderer 1
Camera Pose Tracker
3D Multiple Human Tracker
VizEngine
Keyer
Zero Parallax Correction
Stereo Image Generator
VizRenderer 2
Stereo Content with 3D illusion of On-screen Talent
TV Camera
The On-screen Talent can be observed with an illusion of depth although the TV Camera films in 2D

Figure 6.15: System architecture of the Stereoscopic Rendering System for HDTV Studios

The multiple human tracker tracks the on screen talent in real-time returning his or her 3D pose with respect to a global origin in the studio environment. Using this information, the distance of the moderator from the TV camera is computed in real-time. This information is used in order to compute the horizontal shift required in each of the two stereo images to be generated such that there is zero parallax.

Fig. 6.15 illustrates the main building blocks of the entire system. To summarise the system architecture, the functions of each sub module is as follows:

- **VizEngine:** Same as described in Section. 6.3.2.

- **VizRenderer:** Same as described in Section. 6.3.2.
- **Camera pose tracker:** Same as described in Section. 6.3.2.
- **TV camera:** Same as described in Section. 6.3.2.
- **3D multiple human tracker:** This system as described in the previous chapters tracks the on screen talent in real-time and returns the target's position in 3D. This information is used to generate the zero parallax stereo images.
- **Stereo image generator and zero parallax compensator:** These sub modules together produce the stereo images from the TV camera stream in a way to produce a 3D illusion of the on screen talent. They use the target's pose from the multiple human tracker, the pose of the TV camera from the camera tracker and the stereo configuration of the renderers in order to generate the stereo images. The distance of the target is computed from the TV camera using the pose obtained from the multiple human tracker. Using this information the stereo images are modified through a horizontal shift in order to compensate for zero parallax [159, 129, 160].
- **Keyer:** Same as described in Section. 6.3.2.
- 3D content: The output from the keyer is a 3D illusion of the virtual studio and the on screen talent. The 3D content can be experienced on any flat screen using 3D glasses [161]. The content can also be directly produced for 3D screens [162, 163, 164, 165].

6.4.3 Demonstration and Results

Fig. 6.16 illustrates the results from the tests conduced at a test studio [154] using a blue box studio set-up. The target in the blue box is tracked by two stereo cameras in real-time providing the 3D position of the targets with respect to a global origin. All devices are calibrated with respect to this global origin.

Fig. 6.17 shows the system being demonstrated live at an exhibition organized by Plasamedia [7], [8] in Munich, Germany. The first row illustrates the on screen talent being filmed by the TV camera. The TV cameras used was the Technodolly [166] system and a steady cam. Although there were two TV cameras, the stream from only one camera was used at a time. The second row shows the tracking results from the human tracker. The third row illustrates the 3D content produced for flat screens which clearly shows the stereo illusion of the virtual

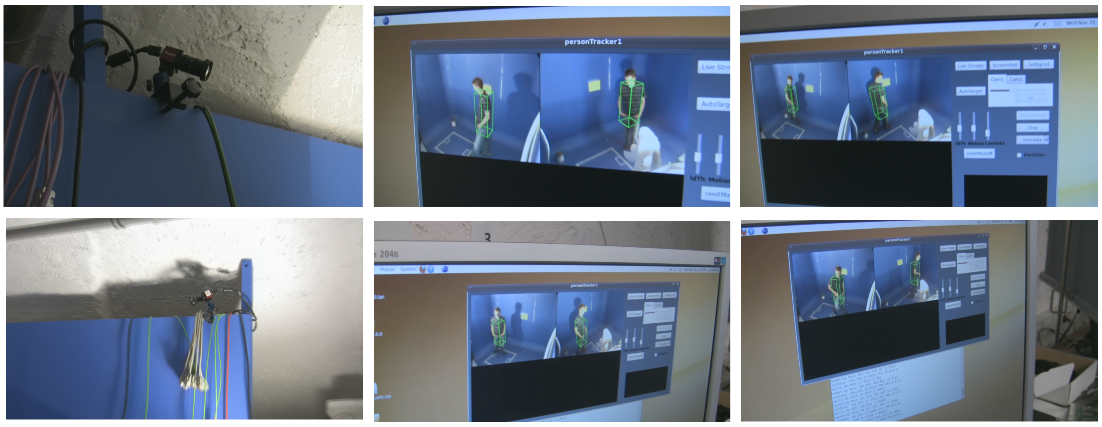


Figure 6.16: Demonstration of the multiple human tracking at the *Vizrt* test studios

studio and the on screen talent. Finally, the fourth row shows 3D content which can be directly viewed on any 3D screen for a 3D illusion of the entire scene without the need of 3D glasses. A complete video demonstration is available under [167, 168] or `http://www6.in.tum.de/~nair/stereoscopy.mp4` and `http://www.youtube.com/watch?v=GrEWGlWZCnE`.

6.5 3D Position based Visual Servoing of Multiple Humans

6.5.1 Overview

In this section, a position based visual servoing application is introduced using the 3D multiple human tracking system [108] which was implemented in a vision driven robot system for human robot interaction. The distributed system comprises of 4 subsystems: a) Multiple Human Tracking System, b) Robot Control System, c) 3D Visualization System and d) Remote Interface System. The Visual Tracking System performs real-time detection and tracking of humans in 3D within a large workspace. The Robot System uses the 3D position data of the targets obtained from the vision system to interact with the humans. The visual information is also used to monitor safe interaction within humans and robot. The Robot System is a 6DOF Stäubli TX90 industrial arm [150], controlled in real-time through a low-level interface. A real-time representation of the actual environment is rendered in 3D by the 3D Visualization System. The individual subsystems communicate with each other over a common communication engine based on TCP/IP. The complete system can be controlled and monitored through a wireless device. The distributed system will be discussed further in detail in the following subsection.

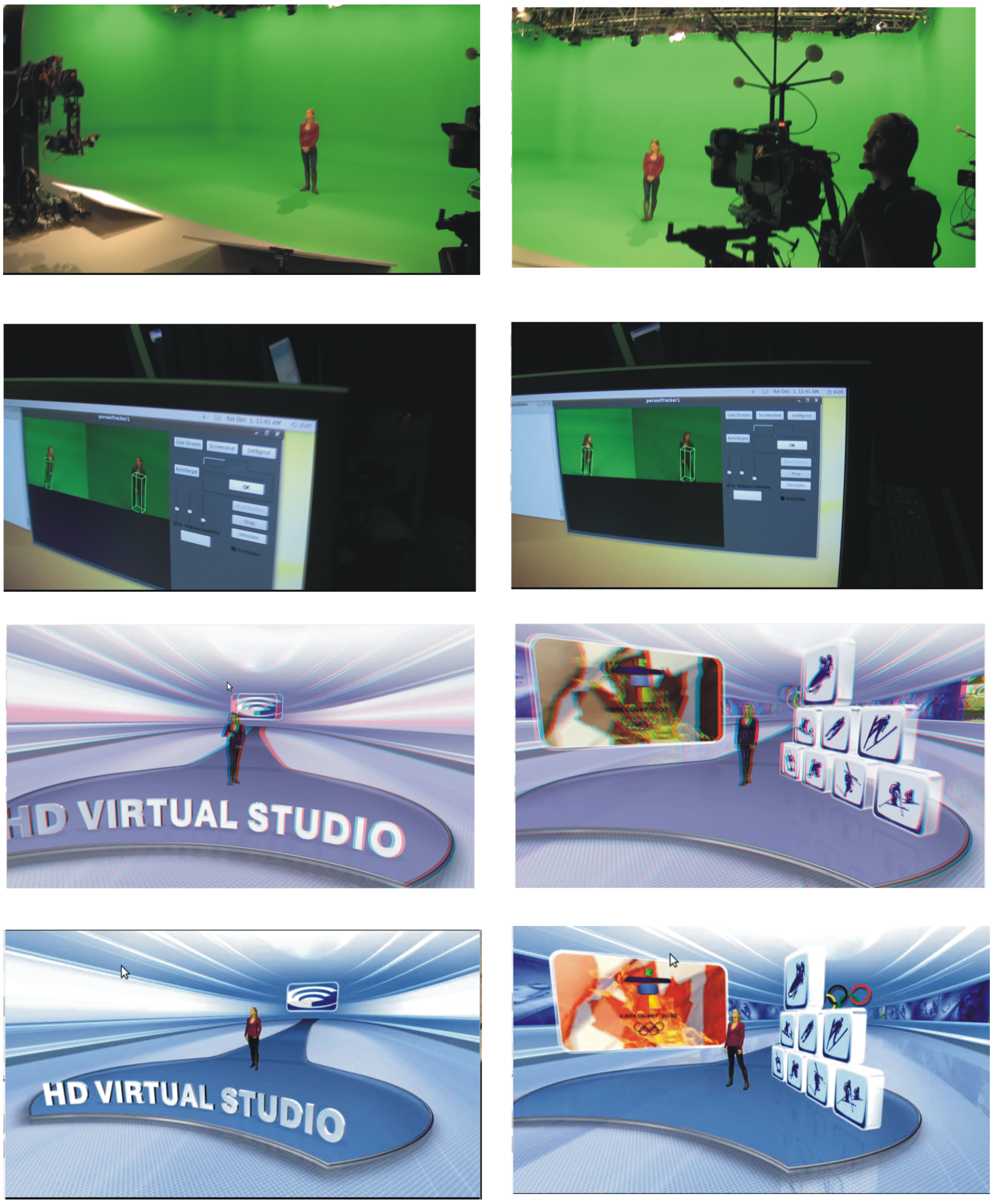


Figure 6.17: Demonstration of the stereoscopic rendering at the Plazamedia exhibition [7, 8].

6.5.2 System Architecture

The recent years have seen a rapid progress in computer vision and motion control technology. As a result many applications have evolved in this domain. Visual servoing is one such application which finds many use cases. It combines computer vision to visually track an object or target and robot motion control. The main idea behind visual servoing tasks is to control a robot system using visual information feedback [169, 170]. The target can be an object or a human, while the servoing device can be a simple pan-tilt unit or a complete industrial robot arm [170]. The reliability of the complete systems depends on both, the accuracy of the vision system and the robustness of the control approach. Special attention has been paid to *Human Robot Interaction*, where the vision system must provide the pose of the target with high accuracy. The robot uses this data to achieve a specific task [171]. Depending on the properties of the target pose returned by the visual tracker, there are two main categories within visual servoing namely, *image based* and *position based* visual servoing [170]. In image based visual servoing the typical configuration is a camera mounted on a industrial robot arm. The target, in this case a human, is tracked using visual tracking approaches through the single camera image, providing the pose mostly in 2D. This information is used by the robot controller to manipulate the camera position/orientation in order to hold the target within the field-of-view of the camera at a defined perspective. This task is simple when there is only one target, but in real scenarios the robot system might need to track more than one human at the same time. A single robot mounted camera cannot achieve this within a large workspace due to its limited field of view. The vision system must also handle object identification, labelling, and occlusions between targets in real-time, which is difficult using a single robot mounted camera.

In such circumstances a vision system capable of performing tracking of multiple human targets in real-time over a large workspace in 3D at all times is required. This information can be used by the robot system to servo all the targets within a large workspace. This approach is called position-based visual servoing since the targets position is computed in the Cartesian space.

This application presents a novel position-based robotic visual servoing system for multiple human targets over a large workspace. It uses a vision based 3D multiple human tracking system using externally mounted cameras and a 6 *DOF* industrial robot arm in order to visually servo the targets such that they are all visible in the field-of-view of the camera mounted on the robot end-effector. The vision based system uses ceiling mounted cameras in a stereo set-up to track each human target in 3D. The robot system uses information position of each target

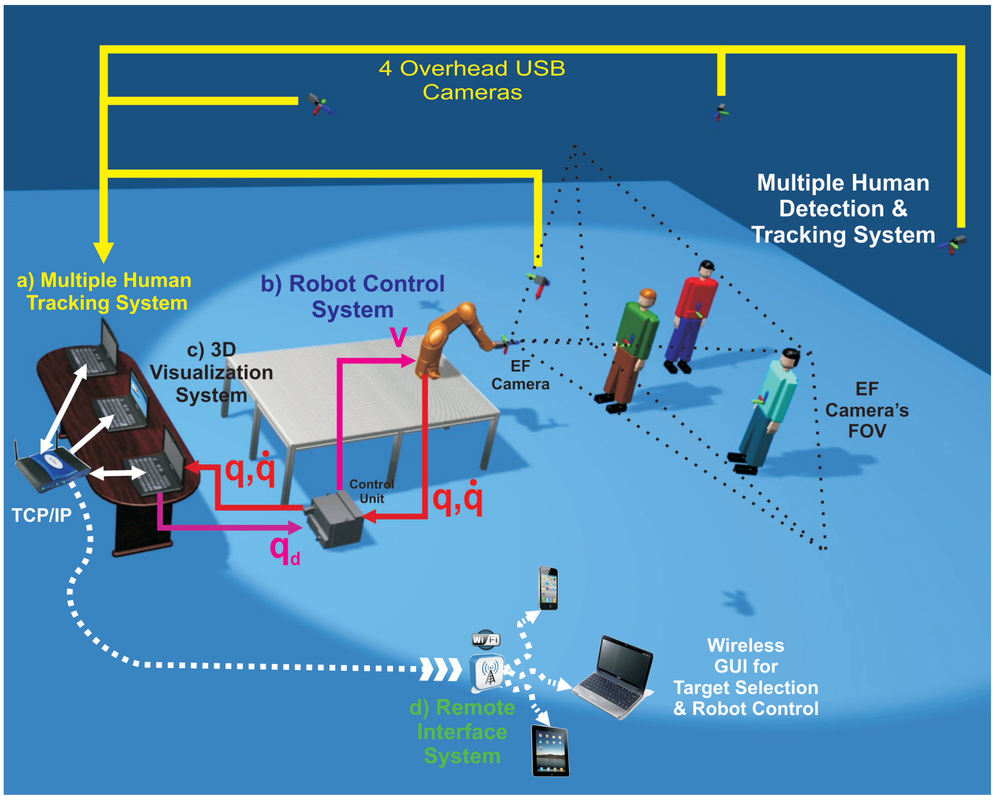

4 Overhead USB Cameras
Multiple Human Detection & Tracking System
a) Multiple Human Tracking System
b) Robot Control System
c) 3D Visualization System
v
EF Camera
EF Camera's FOV
q,q̇
Control Unit
q,q̇
q_d
TCP/IP
Wireless GUI for Target Selection & Robot Control
d) Remote Interface System

Figure 6.18: Complete Robotic Setup. The figure shows the different systems involved in the robotic set-up. a) the Multiple Human Tracking System: 4 USB cameras connected to a GNU/Linux OS PC, b) Robot Control System: An industrial robot StäubliTX90 and a CS8C control unit are connected to a GNU/Linux RT OS PC, c) 3D Visualization System: OpenGL-based virtual world visualization running on a GNU/Linux PC and d) Remote Interface System: it allows the user to select the targets.

in order to control the robot mounted camera such as all targets are visible in its field-of-view with the desired perspective. The robot mounted camera is not used by the vision system. The visual tracking system which functions independently, reacts to new targets even if they are not in the field-of-view of the robot mounted camera. The system can also servo specific selected targets through the remote interface system, see Fig. 6.18. The 3D multiple human tracking system has been discussed in detail in the earlier chapters. Therefore, only the other modules are emphasised in this section.

In order to validate the multiple human tracker, a complete Human-Robot-Interaction scenario has been implemented, where the position of each target is required in order to achieve specific tasks. The control flow of the vision-based robotic system is illustrated in Fig. 6.19, where the *Vision Tracking System* obtains the position of each target and sends them to the *Robot Control System*. This module computes the joint position reference vector and sends it to the control unit, which in turn feeds back the current joint positions of the robot. The *Robot Control Unit* updates the *3D Visualization* environment using the real joint positions. Each system is explained in the following sub-sections.

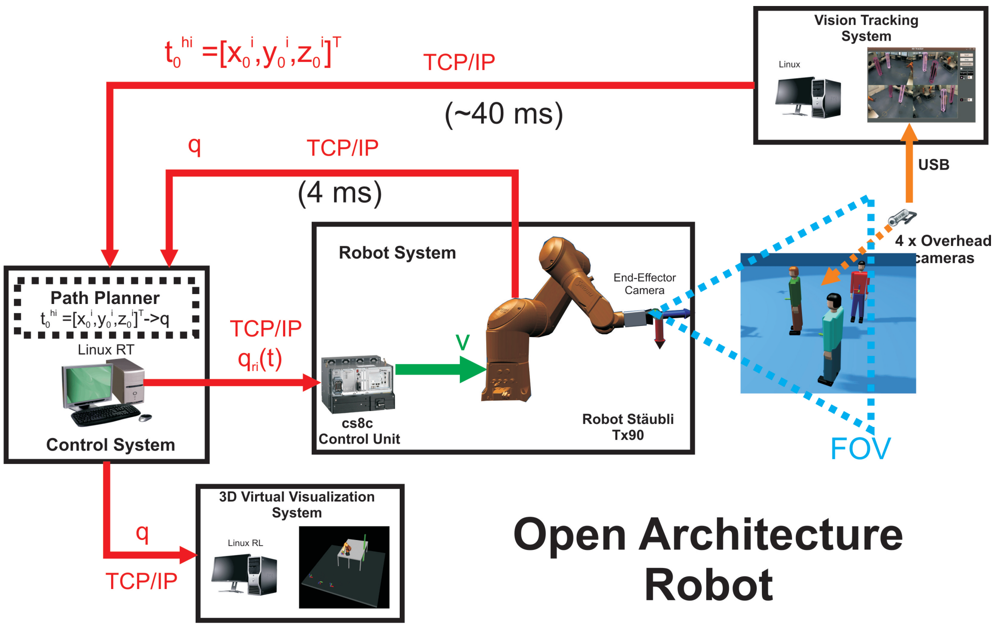


Figure 6.19: Human Tracking System Block Diagram.

6.5.3 Robot Control System

The robotic system comprises of a Stäubli TX90 [150] industrial robot arm, a CS8C [150] control unit and a Workstation running on GNU/Linux OS with real-time extension, see Fig. 6.19. The data communication between the PC and the control unit is through a local network based on TCP/IP. In order to open the architecture of the industrial robot a C++ library based on the Stäubli LLI was developed at our group [151]. This library allows the user to command the robot joint positions or the torque values for each motor drive. In this application, the joint positions were used to command the robot. This joint positions $q_r(t) \in \Re^n$ were obtained using the next trajectory planning.

6.5.3.1 Trajectory Planning

The multiple human tracking system provides the position of each target $t_0^{h_i} = [x_0^i, y_0^i, z_0^i]^T$ with $i = 1, 2, ..., m$ where m is the total number of targets, see Fig. 6.19. The trajectory planner uses this data to generate the desired joint positions $q_d \in \Re^n$. Given the kinematic decoupling properties of the Stäubli robot, the task can be divided in two phases, the first being the *Pan and Tilt* control of the eye in hand camera, and the second is to set the camera *Field of View* (**FOV**).

6.5.3.2 Pan and Tilt

Using the Denavith-Hartenberg convention, the *pose* of the robot's wrist is given by,

$$T_0^3 = \begin{bmatrix} R_0^3 & t_0^3 \\ 0^{1\times 3} & 1 \end{bmatrix}, \tag{6.11}$$

where,

$$t_0^3 = \left[x_0^3, y_0^3, z_0^3\right]^T \in \Re^{3\times 1} \tag{6.12}$$

and

$$R_0^3 = \left[X_0^3, Y_0^3, Z_0^3\right] \in SO\left(3\right) \tag{6.13}$$

represent the position and orientation of the wrist with respect to the world coordinate frame. Similarly, T_0^6, T_0^c and $T_0^{h_i}$ are the *pose* of the end-effector of the robot, the eye in hand camera and the target i, respectively.

In order to calculate the orientation of the camera T_0^c, the average target position is needed,

$$\widetilde{P} = \frac{1}{m}\sum_{i=1}^{m} t_0^{h_i}. \tag{6.14}$$

Then, the position error vector between t_0^3 and $\widetilde{P}$ is computed

$$\Delta P = t_0^3 - \widetilde{P}. \tag{6.15}$$

The vector in eq.(6.15) defines the orientation of the camera using its direction cosines in a general rotation matrix.

$$R_0^c = R_x\left(\alpha\right) R_y\left(\beta\right) R_z\left(\gamma\right) \in SO\left(3\right). \tag{6.16}$$

Where, $R_k\left(\theta\right)$ is the basic rotation matrix around the k axis through an angle θ.

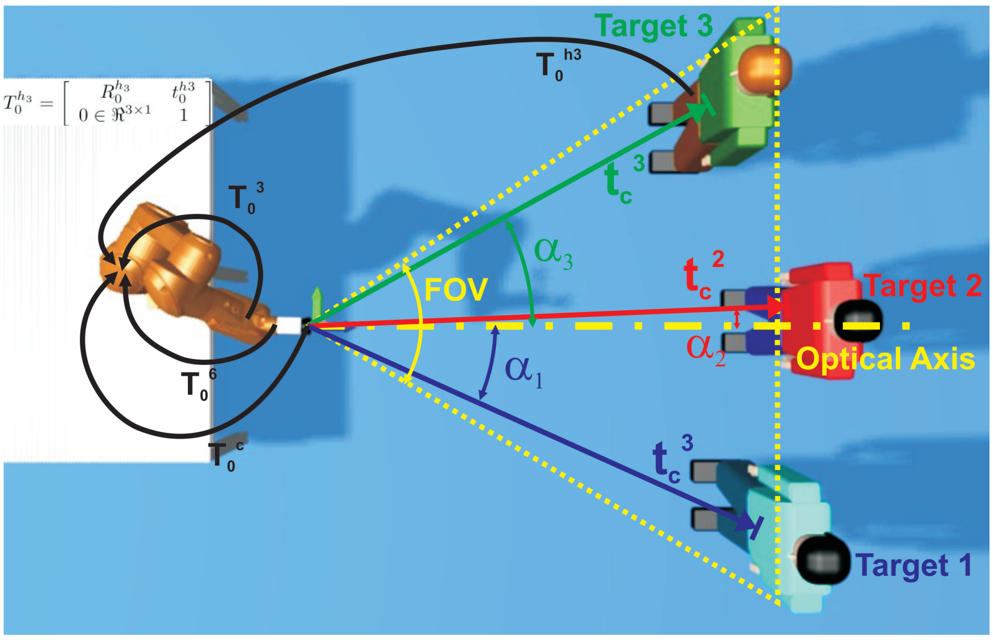


Figure 6.20: Camera Field of View and the Eye in Hand camera.

The robot direct kinematics can be used to define the camera orientation in terms of the *Wrist Orientation* R_0^3 and the *End Effector Orientation* R_3^6,

$$\begin{aligned} R_0^c &= R_0^3\,(q_1,q_2,q_3)\,R_3^6\,(q_{d_4},q_{d_5},q_{d_6}) \\ R_3^6 &= R_0^3\,(q_1,q_2,q_3)^T\,R_0^c \end{aligned} \tag{6.17}$$

The solution of eq.(6.17) generates the desired $q_{d_4}, q_{d_5}, q_{d_6}$, and depends on q_1, q_2, q_3. Now, to compute these first joints, the desired wrist position t_0^3 must be provided, which depends on the camera field of view.

6.5.3.3 Camera FOV

For this task, two facts affect the position of the wrist: a) the optical axis normal to the targets[1] and b) the camera's FOV. For the first part, the solution of $q_{d_1}, q_{d_2}, q_{d_3}$ is derived from eq.(6.16) with $R_0^3 = R_0^c$. This motion is implemented only when $q_{d_5} > q_{5_{max}}$, where $q_{5_{max}}$ is defined by the user.
In the second part, the camera's FOV must be fixed. Fig. 6.20 shows the relation of the camera's FOV and the angle of each target, where $t_c^i = t_0^c - t_0^{h_i} = [x_i, y_i, z_i]^T$ represents the position

[1]The idea is to keep the eye in hand camera in front of the targets.

vector of the target i with respect to the camera frame. α_i is the angle between t_c^i and the *optical axis* given by Z_0^c.

Thereafter, the optimal wrist position $\left\{t_0^3 \in \Re^3 | \alpha_i < \frac{FOV}{2}, \forall i = 1, 2, ..m\right\}$ must be computed. This solution is similar to calculating $\alpha_{max} < \frac{FOV}{2}$ with $\alpha_{max} = max\,(\alpha_i)$.
Then, to find the solution the next steps must be followed:

1. Compute $\alpha_i = a\cos(|z_0^c| \cdot \left|t_c^i\right|)\ \forall i = 1, 2, ..., m,$.

2. if $\alpha_i > \frac{FOV}{2}$, then:

   (a) compute distance from camera to target i,

   $$d_i = \left\|t_c^i\right\|_2^2,$$

   (b) compute minimum distance,

   $$d_{\min} = d_i \cos\left(\frac{FOV}{2}\right), \tag{6.18}$$

   (c) compute the projection of target i over the *optical axis* Z_0^c,

   $$d_{z_i} = d_i \cos\left(\alpha_i\right), \tag{6.19}$$

   (d) compute error distance,

   $$\Delta d = d_{\min} - d_{z_i}, \tag{6.20}$$

   (e) then,

   $$t_0^3 = t_0^3 - \Delta d Z_0^c, \tag{6.21}$$

3. If $t_0^3 > t_{\min}$, $t_0^3 = t_{\min}$, where $t_{\min}$ is a safety threshold to avoid collisions with the robot body. This safety threshold is selected such as it guarantees that the human is not in contact with the robot even with wide open arms.

4. Finally, use t_0^3 to obtain $q_{d_1}, q_{d_2}, q_{d_3}$.

6.5.3.4 Path Planning

Once the desired $q_{d_i}, i = 1, 2, ..., n$ have been set, the trajectory from the current position $q_0 \in \Re^n$ to the desired position q_d must be established. In this case, a 5th order polynomial function has been used,

$$q_{r_i}(t) = (f_{5_i}(t)(q_i - q_{0_i})) + q_{0_i} \tag{6.22}$$

$$f_{5_i}(t) = a_1 \left(\frac{t - t_0}{t_{fi} - t_0}\right)^3 + a_2 \left(\frac{t - t_0}{t_{f_i} - t_0}\right)^4 + a_3 \left(\frac{t - t_0}{t_{f_i} - t_0}\right)^5 \tag{6.23}$$

where $t \in \Re$ is the current time, $t_0 \in \Re$ is the initial time and the final time is $\{t_{f_i} \in \Re | \ddot{q}_{d_i}(t) < a_{\max}, \forall i = 1, 2, ..., n\}$ with a_{max} as the maximum joint acceleration defined by the user. This q_{r_i} is transmitted to the control unit in real time, see Fig. 6.19.

6.5.4 3D Visualization System

This module performs *OpenGL* based real-time rendering of the workspace in 3D. It uses *Qt* and *Coin*3*D* as a backbone. The scene is constructed using 3D models of different objects occupying the scene such as the robot, controller box, table and the human target locations. The robot kinematics and configurations are specified using XML files. The system updates the configuration of the robot arm and the positions of the humans in real-time. This system is connected to the Robot Control System and the Human Tracking System in order to obtain the joint positions of the robot and the 3D positions of the human targets. This is achieved by means of TCP/IP communication using sockets. Fig. 6.21 illustrates this module.

6.5.5 Remote Interface system

Each individual system can be controlled through a remote device supporting $WiFi$ interface. Therefore, devices such as iPhones, Tablet PCs, Net-books, etc. can be used to remotely control and monitor the entire system. The remote interface exchanges data with each subsystem through TCP/IP sockets.

We use a protocol, such as VNC, that allows a desktop to be viewed and controlled remotely. This is a common protocol that can be implemented in any OS.

6.5.6 Demonstration and Results

Fig. 6.22 demonstrates the results obtained in the real-world scenario. A complete video demonstration is available under [172] or `http://www.youtube.com/watch?feature=player_`

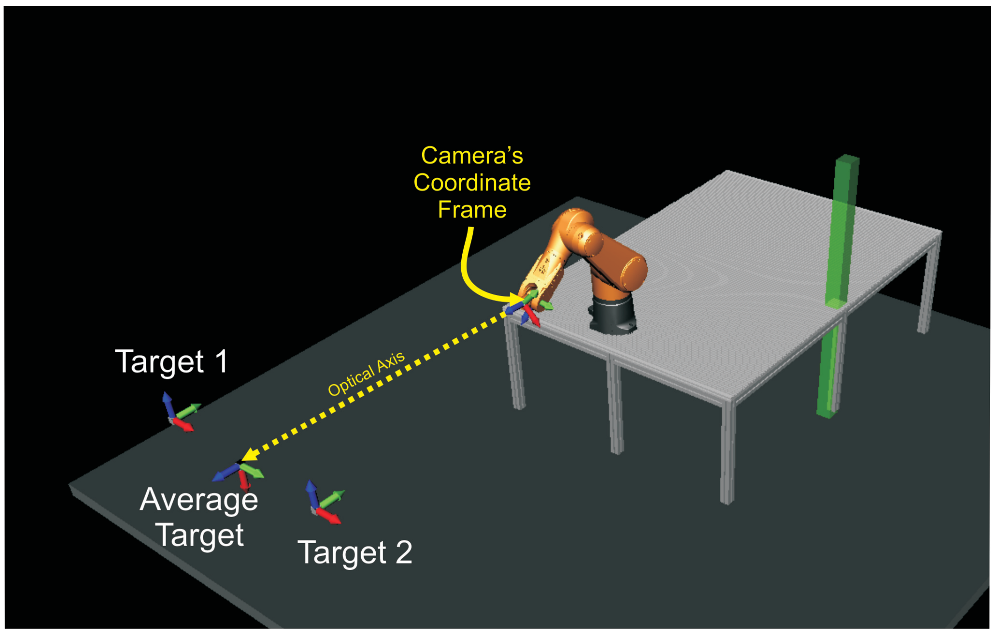


Figure 6.21: *OpenGL* based virtual visualization

`embedded&v=15JMRGa_qE4`. The tracker tracks 2 targets simultaneously in real-time and the robot arm servos both targets. Later, the operator disables target 2 such that the robot arms servoyes only target 1 followed by target 1 being disabled and target 2 being enabled. Later, both targets are enabled and it can be observed that target 1 gets closer than the safety limit of the robot which is detected by the tracking system and a signal is sent to the robot controller such that it moves to a safe parking position and thereafter all systems are shut down.

Fig. 6.23 shows how the system handles occlusion between targets. It can be observed that in camera 1 (top right), target 1 is occluded by target 2. Hence, during the likelihood computation for the filter associated with target 1, camera 1 is not considered. Similarly in camera 3 (bottom right), target 2 is occluded by target 1 and therefore camera 3 is not considered in the likelihood computation for the filter associated with target 2. The visual tracking system runs at approximately 15 fps on a Intel Core i7 desktop PC.

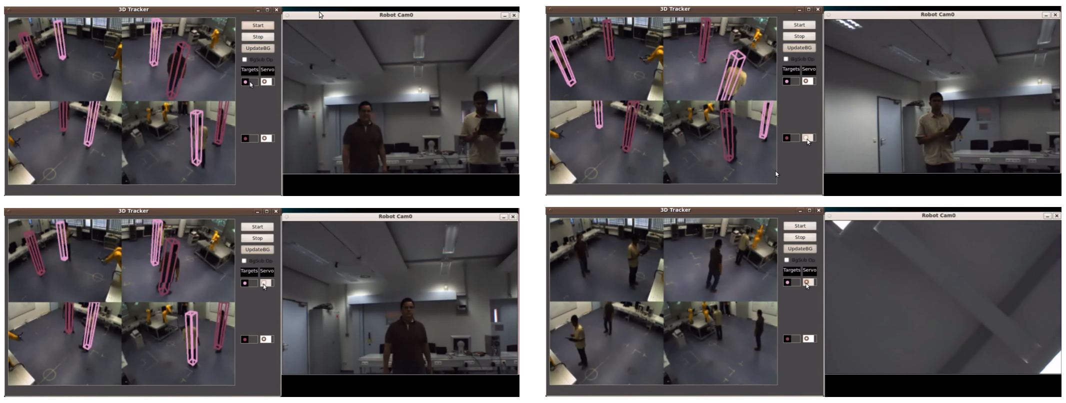


Figure 6.22: The figure illustrates the test results. The 4 clustered images represent the tracking system results along with an additional robot mounted camera output. **Top Left:** Two targets are tracked and servoyed by the robot. **Top Right:** Only target 1 is enabled to be servoyed by the robot. **Bottom Left:** Only target 2 is enabled to be servoyed by the robot. **Bottom Right:** Target 1 gets closer than the safety limit of the robot and robot goes to park position and all systems are shut down.

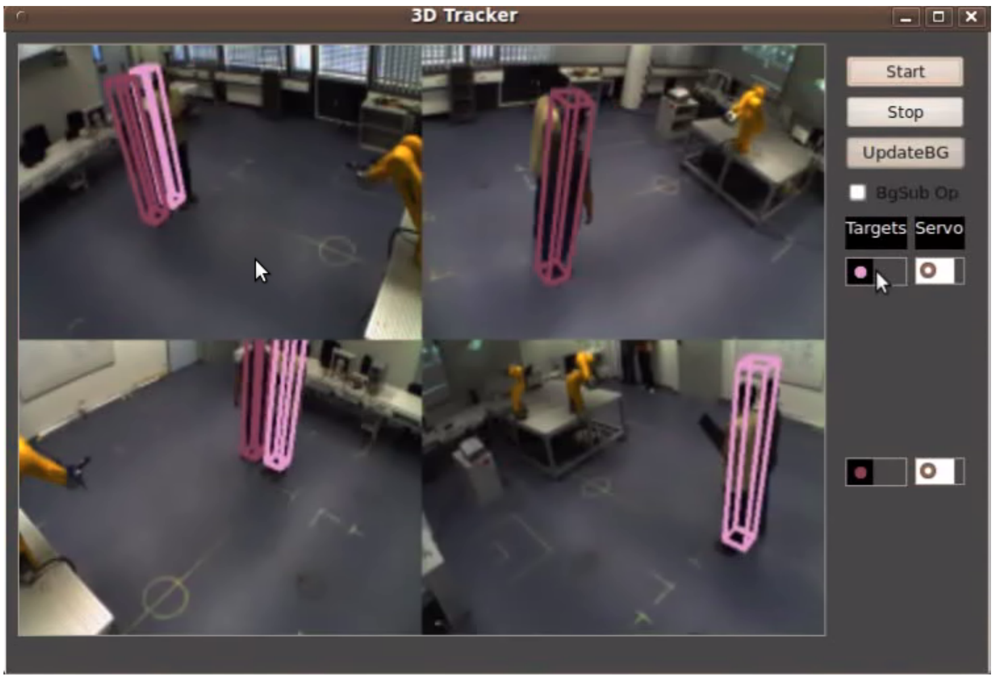


Figure 6.23: The figure illustrates how the system detects occlusion between targets.

6.6 Close Range Human Robot Interaction

6.6.1 Overview

This demonstration scenario validates the close range human robot interaction achieved using visual feedback. Fig. 6.24 shows the building blocks of the complete setup. The scenario consists of two visual tracking systems consisting of the multiple human tracking system and a object detection system. In this scenario there exist certain objects of different colours on the working table and each human in the workspace works with an object of a specific colour. The task of the robot system is to serve as a interaction medium between the humans and the objects by identifying the right object to be handed over to a human. The robot should also decide when and where to handover this particular object taking into account its own constraints. In order to make these decisions, the robot system relies on the two visual tracking systems. The following sections provide a detailed description of the system.

6.6.2 System Architecture

The two visual tracking system form the core of decision making capabilities of the robot system. The multiple human tracking system has been discussed in detail in the earlier chapters. In this section a detailed insight into the demonstration scenario is provided and the vision system used to detect the objects on the working table is discussed.

As mentioned earlier, there are many similar object lying on the working table. The industrial robot arm is mounted to this table. The object are large Lego Duplo [173] blocks with dimensions of 100 × 100 × 100 mm. Each block has the same dimensions but can have one of the three colours which are blue, green or red. The operational workspace is divided into three zones namely, the working table containing the objects reachable by the robot arm, the human worker zone where the human targets can freely move and the handover zone where the robot arm is capable of handing over the objects from the working table to the humans. Each working zone is selected considering all safety aspects such that safe human robot interaction is possible.

Fig. 6.25 describes the system work flow. The system consists of multiple modules working independently. These modules share a common communication interface through which they send or receive appropriate data. The 3D object tracker and the 3D multiple human tracker form the two visual tracking modules responsible of providing real-time position information of the objects and humans respectively. The object-human association module receives position

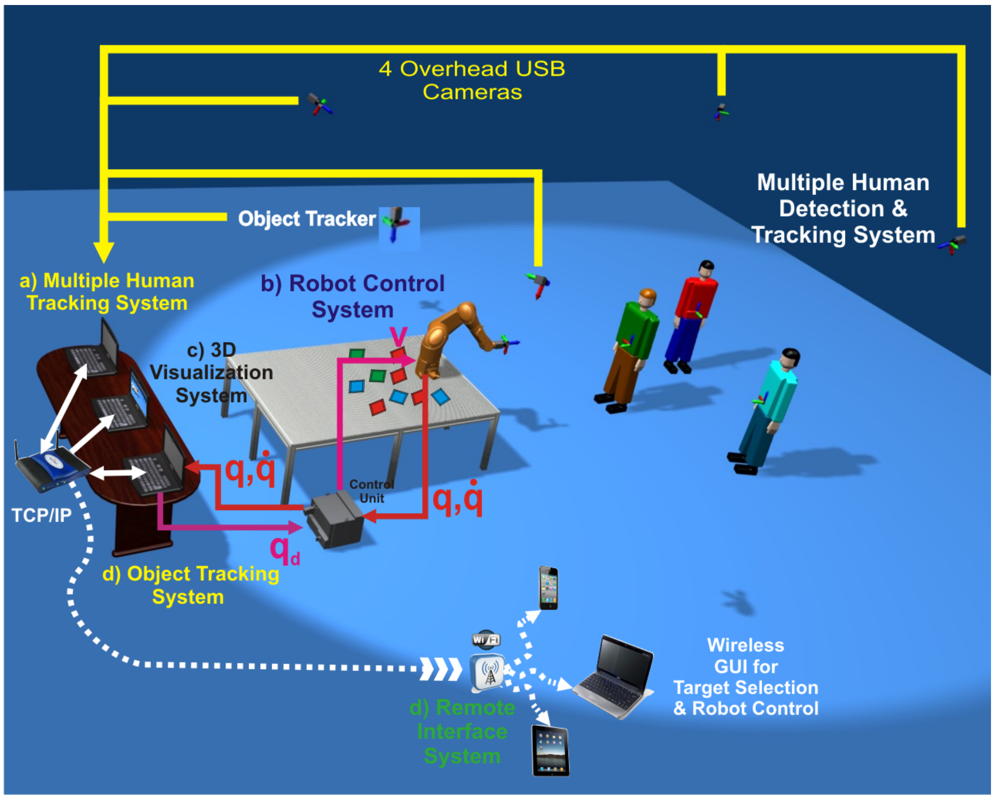

4 Overhead USB Cameras
Object Tracker
Multiple Human Detection & Tracking System
a) Multiple Human Tracking System
b) Robot Control System
c) 3D Visualization System
V
q,q̇
q,q̇
Control Unit
q_d
TCP/IP
d) Object Tracking System
d) Remote Interface System
Wireless GUI for Target Selection & Robot Control

Figure 6.24: Complete Robotic Setup. The figure shows the different systems involved in the robotic set-up. a) the Multiple Human Tracking System: 4 USB cameras connected to a GNU/Linux OS PC, b) Robot Control System: An industrial robot StäubliTX90 and a CS8C control unit are connected to a GNU/Linux RT OS PC, c) 3D Visualization System: OpenGL-based virtual world visualization running on a GNU/Linux PC, d) Object Tracking System and e) Remote Interface System: it allows the user to select the targets.

information from the two visual tracking systems. It uses this information along with the handover zone information and associates an object of a particular type to a unique human target when he or she is within the handover zone. The human can also reject or request for a different coloured object using physical gestures which are detected through a force sensor mounted on the robot arm. Once the association is done, the particular object remains assigned to the particular human target until he or she is within the workspace. If the human target to which the object was assigned leaves the workspace, that particular object-human association is freed and the object can be associated to another human target whenever possible. The following subsections provide functional descriptions of each module.

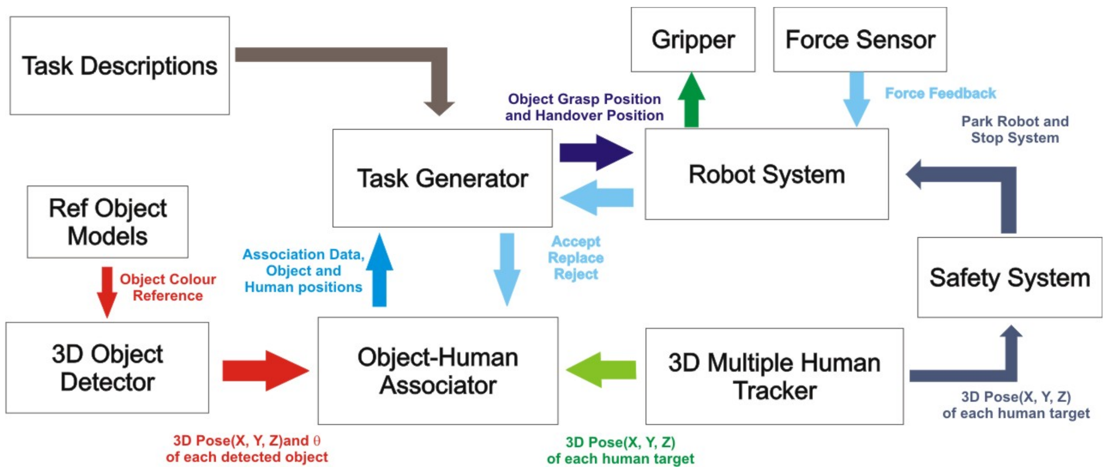

Task Descriptions
Gripper
Force Sensor
Object Grasp Position and Handover Position
Force Feedback
Park Robot and Stop System
Task Generator
Robot System
Ref Object Models
Association Data, Object and Human positions
Accept Replace Reject
Safety System
Object Colour Reference
3D Object Detector
Object-Human Associator
3D Multiple Human Tracker
3D Pose(X, Y, Z) of each human target
3D Pose(X, Y, Z)and θ of each detected object
3D Pose(X, Y, Z) of each human target

Figure 6.25: System Architecture of the Human Robot Object Interaction System.

6.6.2.1 3D Object Detection

The function of the object detection system is to detect all Lego blocks present on the working table in real-time and compute their 3D positions with respect to the global origin and also their rotation along the Z axis. This system uses one camera mounted on the ceiling and looking down towards the working table. Each Lego Duplo block on the working table is detected and classified into one of the three colours, following which the 3D pose is computed. The block diagram of the object detection system is illustrated in fig. 6.26 and each module is described below.

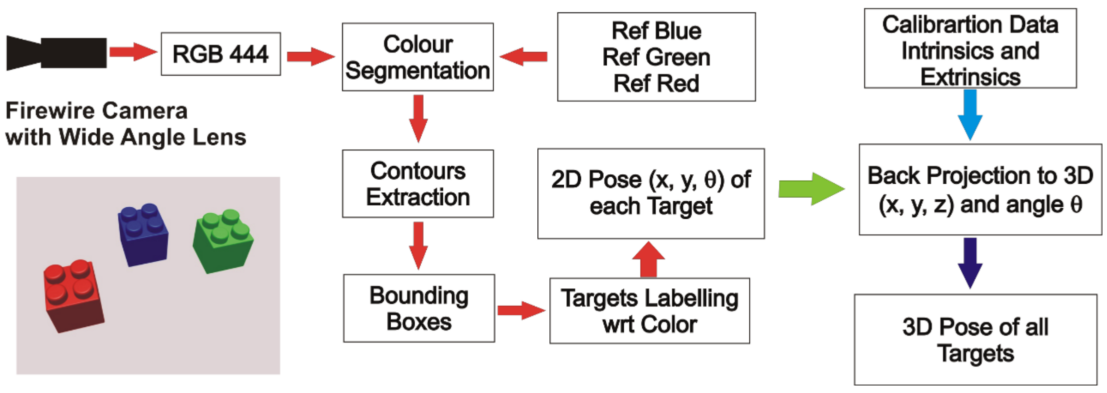

RGB 444
Colour Segmentation
Ref Blue Ref Green Ref Red
Calibrartion Data Intrinsics and Extrinsics
Firewire Camera with Wide Angle Lens
Contours Extraction
2D Pose (x, y, θ) of each Target
Back Projection to 3D (x, y, z) and angle θ
Bounding Boxes
Targets Labelling wrt Color
3D Pose of all Targets

Figure 6.26: Block diagram of the 3D Object Detector.

- **Camera:** The input sensor used is a Fire-wire camera with a wide angle objective such that the complete working table can be observed in the camera field of view. The camera is capable of streaming images at a rate of 35 fps which are de-bayered in order to be available in the desired RGB 444 format. The camera is calibrated for its intrinsic and extrinsic parameters. After the calibration the position of the camera is know with respect to the global origin.

- **Colour Segmentation:** The de-bayered image from the camera is processed for colour segmentation. Three reference colours are used to perform the colour segmentation. Each reference colour represents one of objects being used. In order to obtain the colour segmented image, the RGB image is converted to HSV space in order to isolate the illumination information from the colour. The hue and saturation channels are analysed for each pixel to decide if it belongs to one of the reference colours.

- **Contour Extraction:** During this stage a binary image is produced from the colour segmented image where every pixel belonging to a reference colour is set to a high value. A contour extraction step [174] if performed on the binary image and a set of contours are obtained which represent the contours of the objects to be detected.

- **Bounding Box Computation:** For every object whose contour is known, a compact and rotated bounding box is computed around that counter. This bounding box is represented by its position in pixels and orientation which is (x_p, y_p, θ).

- **Target Labelling:** Every bounding box which now represents an object position in the image space is labelled into a class in order to identify it among the three possible colours. In order to achieve this the colour of the pixels within the bounding box are analysed to identify which reference colour they belong to. After this step the position of the object in image space and its colour are known.

- **2D Pose of the Target:** In this step all the labelled targets are accumulated into a single container representing the complete data concerning each object. The pose at this level is 2D and in the image space.

- **3D Pose Data:** In this stage the pixel positions (x_p, y_p), for each target is back projected to the 3D space using the intrinsic and extrinsic parameters of the Firewire camera. This is possible due to the fact that the translation of the objects in the Z direction is constant and can be computed. The required transformations is described below.

  $P_{3D} = KEP_{screen}$

The matrix K represents the intrinsics matrix, while E represents the extrinsic matrix. P_{screen} are the positions of the object in image space while P_{3D} is the position is 3D. In this set-up Z belonging to P_{3D} is known. The resulting transformation allows the computation of the 3D position of the object from their image space positions provided Z is known.

6.6.2.2 3D Human Tracking System

This module has been discussed in detail in the earlier chapters. It automatically detects and tracks each human target within the robotic workspace. It maintains the position data concerning each target and uses the common communication interface to transmit this data to other modules on request. The target data sent is in the form (id, x, y, z), where id represents the target id which is unique for each human target. When a target leaves the workspace, he or she is deleted from the target data container and the target id which was associated to it is freed so that it can be assigned to new targets. The data requests come from the object-human association module and the safety modules.

6.6.2.3 Object-Human Associator

This module has the objective of associating an object of a specific colour on the work table to a human target and inform the association made to the task generation module. It uses the information of the workspace configuration to achieve this task. As discussed earlier, the workspace is divided into three zones namely, the work table, the handover zone and the tracking area. These zones are configured by the user before starting the system. The zones are selected such that safe close range human robot interaction can be achieved which depends on the safety considerations and the robot's working range. The object-human associator requests position data from the object detector and the 3D multiple human tracker every 100 ms. It checks if any human is within the handover zone and if found it performs a check if that particular target has already been associated with an object. If the target has not yet been associated then then the next free object is assigned to it until he or she is in the tracking area. When a target leaves the tracking area and if it had an association active, then this particular association is removed.

6.6.2.4 Robot System

The robot system is a Stäubli TX90 industrial arm controlled through a low level interface. A *Schunk* [175] gripper is mounted on the end effector to facilitate the grasping and handover. Between the gripper and the robot arm, a $JR3$ $6DOF$ [176] force sensor is installed. The force sensor provides the necessary indications to the robot controller during the handover in order to determine the intention of the human towards accepting or rejecting the handover. The robot controller receives start and end positions of task from the task generator. Using the robot kinematics and the task information, the robot controller generates position based trajectory from the start to the end position such that the robot can grasp the object and hand it over to the human. The low level details of the controller have been discussed in the earlier sections. During the handover the robot controller analyses the forces recorded by the force sensor in order to either open the gripper when the human pulls the object or to return it to the work table otherwise, It also receives information from the safety system when there is any safety breach. Under such circumstances the robot retreats to a safe parking position and the entire system is shut down.

6.6.2.5 Task Generator

This modules is responsible to generate tasks for the robot in order to serve the human targets. It receives information regarding the object-human association and positions of all the objects and human targets from the associator. Using this information the tasks are generated for the robot system. The task assigned to the robot consists of a start position and a end position. The start position defines the position of the object to be grasped and the end position defines the position where the handover should be performed.

When a human target is in the handover zone and has an object associated to it, the task generator assigns an object which should be grasped by the robot and handed over to the human. Before making this decision, the task generator performs certain checks. It checks if the human target is within the handover zone since the robot has a limited working range. It checks if the object assigned to the target is on the table and within the reach of the robot. Finally, it checks if the human target is stationary within the handover zone for the handover position to be assigned.

The decisions taken during handover depend on the forces recorder by the force sensors. The intention of the human are mapped to unique force patterns on the gripper which are measured by the force sensor. These intentions are divided into three types namely, 1. Accept the object, 2. Reject the object and 3. Replace with another object of the same colour.

The task information is sent to the robot system. When the robot executes the task there is an additional check performed. When the robot reaches the handover position, the task generator checks if the human target is still located at the handover position since there is a possibility that the human either moves to another position within the handover zone or leaves. The task generator check this and if the human is still at the same position it directs the robot system to perform an handover. If the human has moved to another position within the handover zone, the robot system is directed to move to the new position of the human and the same check is performed until a successful handover or the maximum number of attempts have exhausted. If the human has left the handover zone the robot system is directed to place the object back to its original position.

6.6.2.6 Safety System:

This module is very important to ensure safety off the humans and robot during the task executions. The safety system maintains a configuration defining the safe close range working distance permitted between the human and the robot. If any human breaches this distance, the safety system overrides all tasks allocated to the robot systems and directs it to park itself safely and thereby shuts down the whole system. In order to achieve this, it requests position information of the humans targets from the 3D multiple human tracking system.

6.6.3 Demonstration and Results

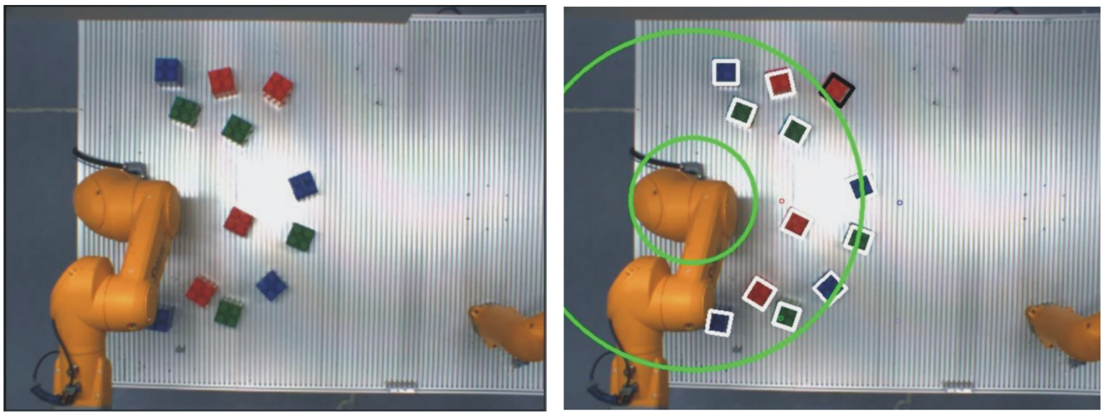


Figure 6.27: Tracking results of the 3D object tracker

Fig. 6.27 illustrates the results obtained from the 3D object tracker. The objects on the work table are tracked successfully with a 3D pose. Although the image shows a simple rectangle around each tracked object, the pose returned is 3D translation and rotation around the Z axis due to the calibrated camera. The two green circular boundaries represent the inner and outer working limits with the region between them being the working area. The objects within the working area are considered by the robot system and are shown as marked in white by the tracker. On the other hand, any object outside the tracking area is tracked but not considered by the robot system. These object can be seen to be marked in black by the tracker. A complete video demonstration is available under [177] or `http://www.youtube.com/watch?v=4XBn5UzgHp4`.

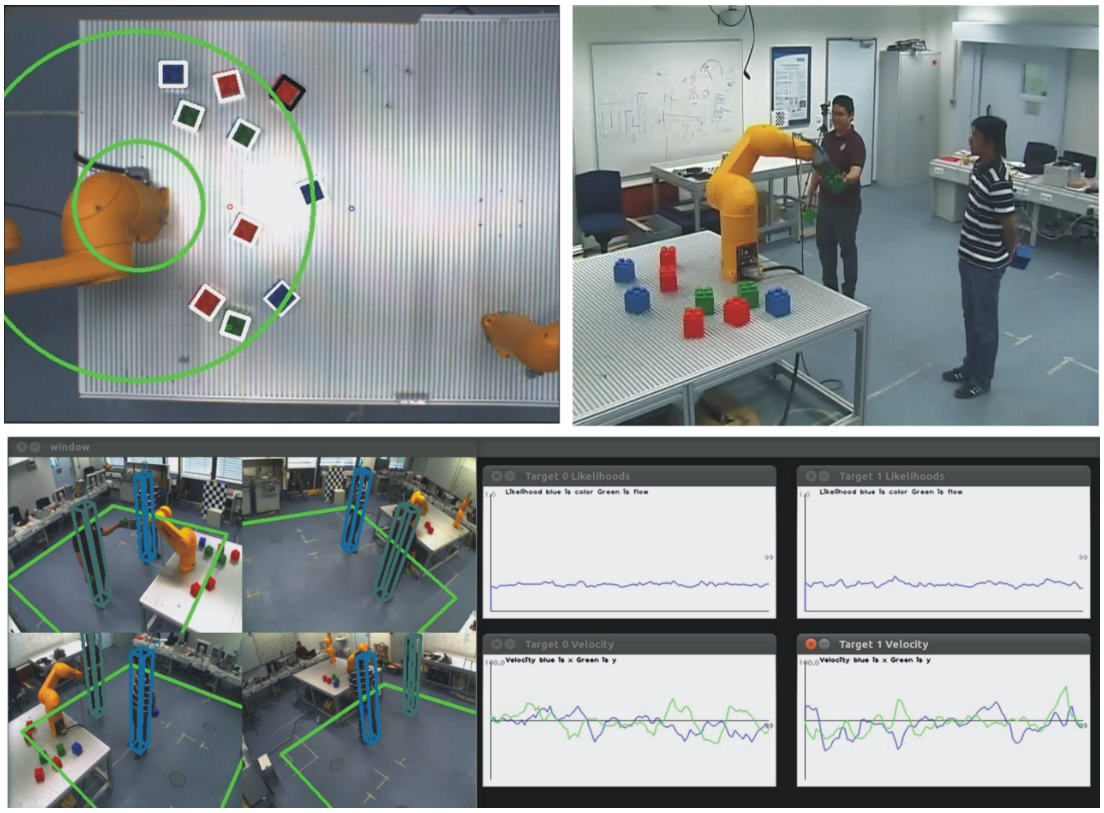

window
Target 0 Likelihoods
Likelihood blue is color Green is flow
Target 1 Likelihoods
Likelihood blue is color Green is flow
Target 0 Velocity
Velocity blue is x Green is y
Target 1 Velocity
Velocity blue is x Green is y

Figure 6.28: Demonstration scenario of close range human robot interaction

Fig. 6.28 shows the different components on this application working together. The top left images represents the 3D object tracker. The top right image shows the scenario where two humans are in close contact with the robot. The bottom image shows the results from the multiple human tracker along with the plots of the target velocities which is an important input in the task allocation stage. A complete video demonstration is available under [178, 179] or `http://www.youtube.com/watch?v=6OGJiRKB2zM` and `http://www.youtube.com/watch?v=eHHg-l3MOBO`.

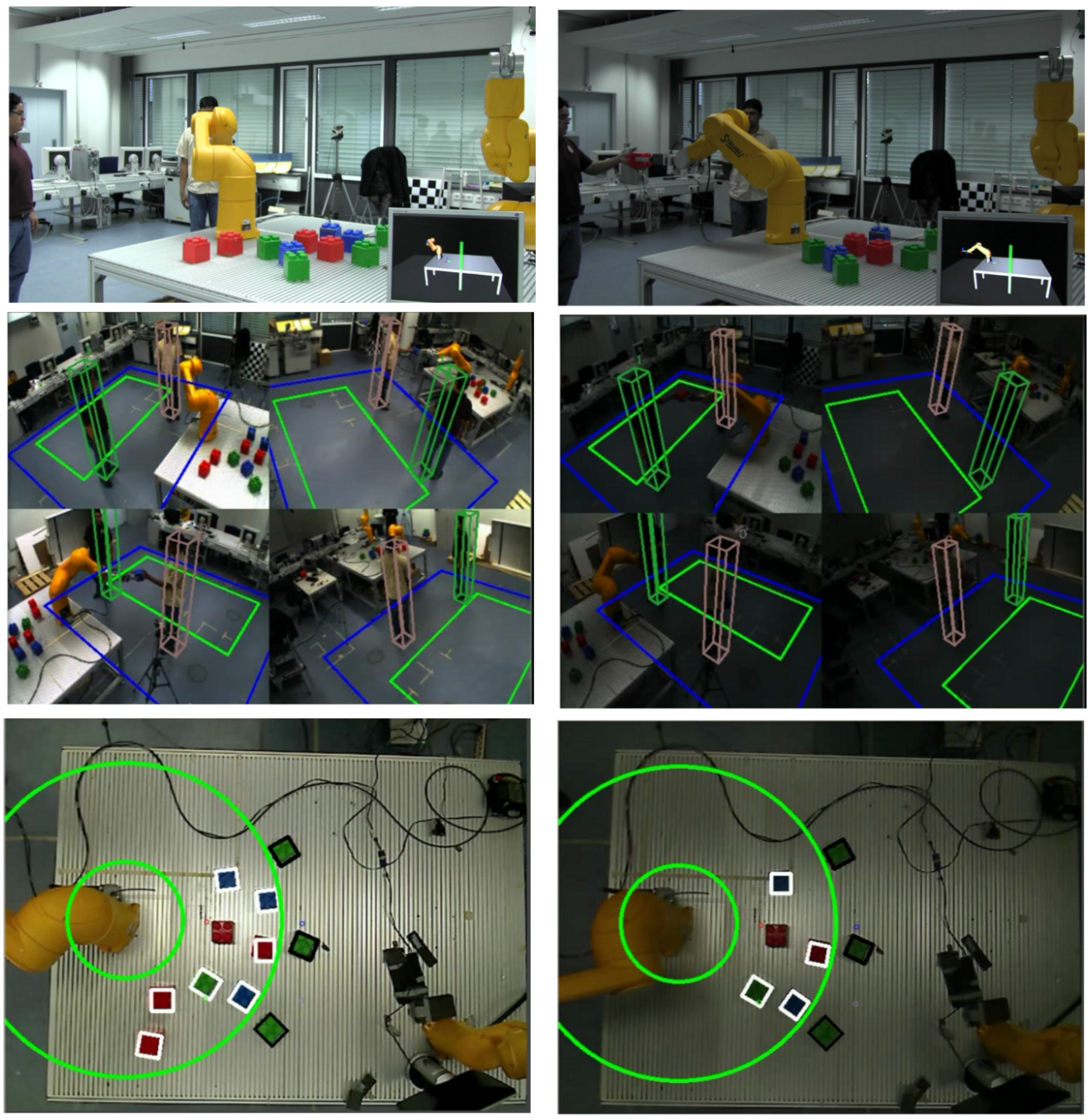


Figure 6.29: Demonstration scenario of close range human robot interaction under drastic changes in lighting conditions

Fig. 6.28 shows a similar demonstration but under drastically changing lighting conditions. The multiple human tracker and the object tracker perform robust tracking even when the lighting conditions change while the humans are interacting with the robot. In such a dynamic environment the humans continue to interact safely with the robot. The two column images show the state of the environment, multiple human tracker and object tracker in good lighting and bad lighting. It can be observed that all system continue to function in a robust manner even under such dynamic conditions. A complete demonstration is available under [180] or `http://www.youtube.com/watch?v=Pn7kyhlEkEc`.

Chapter 7

Conclusion

The individual chapters in this thesis provided a detailed insight into the research and development of a vision based multiple human tracker with primary focus on robustness enhancement. Novel techniques in the direction of modular construction and robustness enhancement were introduced and validated. Furthermore, the possibilities of easy integration with a variety of real-world robotic systems was demonstrated.

This chapter summarizes the primary contributions of the thesis and thereby the conclusion drawn from the thesis work. These conclusions also include the observed shortcomings in the system. With respect to these observations, the future work in order to overcome the current shortcomings are discussed.

7.1 Primary Contributions of the Thesis

The primary contribution of this thesis as discussed in the earlier chapters can be formulated as follows:

- 3D Multiple Human Tracking System: A vision based real-time 3D multiple human tracking system based on a modular building blocks approach. It is capable of automatically detecting and tracking multiple humans in real-time and with a 3D pose within a desired area of interest.
- Occlusion Handling System: It is a module which detects and handles multiple occlusions between human targets while they are being tracked. It resolves the occlusions in real-time by providing the individual trackers with information regarding which camera views are good to track for their individual targets. It plays an important role under ambiguous

tracking scenarios such as multiple targets with similar appearance moving close to each other.

- Machine Learning for Quality Analysis of Lighting Conditions: It is a model trained to classify the lighting conditions into one of its pre defined classes. Depending on the classification results the quality of the lighting conditions is determined. The model is trained using a large dataset of lighting conditions representing the desired classes.
- Background Model update under Drastically Changing Lighting Conditions: This module uses the machine learning based lighting conditions classifier in order to detect drastic changes in lighting conditions. It further performs the non trivial task of updating the background model in the presence of foreground targets being tracked.
- Intelligent Multi-Modal Fusion of Colour and Optical Flow Modalities: This modules uses the machine learning based lighting classifier along with the target motion analyser to generate the optimal weights for the visual modalities. The two visual modalities are fused in order to guarantee robustness under dynamically changing tracking environments.
- Modelling and Simulation of the Workspace for Ground Truth Generation: It is a novel approach through which zero error ground truth data for evaluation and validation of the tracker is obtained. This required modelling of the complete workspace to great detail and simulation of human motion in order to extract the trajectories.
- Experiments to Validate the Different Aspects of the Tracker: Unlike existing systems, the multiple human tracking system was validated through multiple experiments. Each experiment was focussed on a certain aspect of the system. The experiments were conducted both in simulated and real environments.
- Integration into Real-world Robotic Applications: In this phase, the system was completely integrated into a variety of diverse real-world robotic application scenarios. Its easy integration into these larger system confirms its flexibility in terms of usability.

7.2 Shortcomings of the System

Although the multiple human tracker performs robustly in dynamic environments, there are few aspects which need further optimization. This section introduces these shortcomings on the basis of which the research plan for the future will be formulated.

- Camera Configuration: In the current set-up the cameras are mounted on the ceiling opposite to each other. Each camera has a specific tilt to obtain the required field of view of the tracking area. This configuration, although well suited in most cases, can cause ambiguity for the detection module when a target is located at the central intersection zone of all the four cameras. This could sometimes lead to ghost targets due to ambiguity in the occupancy map. Furthermore, due to camera mounting constraints it is not trivial to obtain a configuration where all the cameras share exactly the same viewing area. Under such situation there could be zones where targets are out of the field of view of certain cameras while still being inside the designated tracking area. This can cause tracking ambiguity under multiple occlusion where targets are not visible in some view and out of the field of view of some cameras due to coverage issues. Therefore, optimising the camera configuration to improve the common area viewed by cameras is an important area for future investigation.

- Light Change Detection Stability: The light change detection modules performs with a good degree of robustness. However, it has been observed that in some rare cases multiple light changes are detected in a short interval if the time required by the light source to achieve a stable illumination is large. This situation can occur when multiple light sources are present in the tracking environment and their behaviour varies from each other.

- Optical Flow Processing Speed: Another shortcoming of the system is the performance speed of the optical flow module. Although the implementation uses the GPU, it still needs to be optimized in terms of speed.

- Target Model: Although the tests conducted do not show any false positives, it is possible that any cylindrical object representing human dimensions could be detected as a human target. This is because the detector solely depends on the occupancy map within the detection area. In order to handle this situation the target model can be classified as a human or not using a trained dataset such as the pedestrian detection datasets [96]. This test was already performed in the thesis and it was found that since the training set consists of only frontal images of pedestrians the classification does not works well for tilted camera configurations.

7.3 Proposed Future Work

From the conclusions drawn and the observed shortcomings, the future work in order to improve the performance of the system in certain aspects where it is currently lagging will be discussed in this section. The following are the planned areas for future research:

- Camera Configuration: The most optimal camera placement will be researched in order to obtain the best area coverage and viewing angles. The minimum number of cameras required to obtain optimal tracking results in all possible scenarios will also be investigated.
- Training Data: In order to improve the performance of the lighting classifier the training set will be extended with samples from more diverse environments in terms of lighting conditions. The classification module will be extended to perform a more extensive analysis of the behaviour of the light in order to determine more robustly the instance when the light changes and when the light becomes stable.
- Optical Flow: The optical flow processing module will be further optimized in order to utilize the full power of the latest GPUs in order to obtain performance boost.
- Target Detection: This module will be researched further in order to detect targets even in very constrained environments.
- Target Model: The target model will be extended from the current rigid model to a more detailed and articulated in order to include the hands. The appearance model will be made more detailed by modelling different parts such as head, torso and legs individually.
- Target Pose: The pose which is currently 3D translation will be extended to estimate also the orientation along the z axis. Techniques to track this extra degree of freedom will be researched.
- Training Set for Humans in Different Camera Configurations: A large training data set will be produced consisting of manually tagged images of humans in different camera configurations. Its difficult to handle all possible camera configuration schemes due to the large scale of data. Therefore, common configurations for multiple human tracking will be considered. This training set along with a false positive dataset will be used to train a classifier such as SVM. The classifier will be used to validate the detection process

and improve the robustness of the occlusion handling module in order to handle occlusion with objects other than humans.

Appendix A

Appendix

A.1 Software Architecture

Fig. A.1 illustrates the software architecture of the 3D multiple human tracking system. It consists of the tracker management module which serves as the central co-ordination unit among all the other components. It handles the individual trackers associated with every target, the camera inputs, communication engine and the graphical user interface. Each sub-module has its own dependencies as illustrated in the fig. A.1.

The tracker management manages each tracker present in the system. Every individual tracker is associated with its tracker data. The implementation requires the functionality of the *OpenTL* [36], *OpenCV* [34], *OpenGL* [181] and *Qt* [113] libraries. The camera module uses the API provided by the vendor. In the current set-up the *uEyeUSB* cameras [182] are used and along with the *libueyeusb* library.

The communication module which is mainly used as an interface to robotic systems, uses the low level libraries *librobutilstaubli* and *libsystemfunctions* [151]. The graphical user interface uses the *Qt* library [113]. The *QtGui*, *QDesigner* and *QGlWidget* are the main sub modules used.

There are many other low level dependencies for each software module. The implementation was done in C++ under the LINUX operating system with real time extension. Currently the software is supported only under LINUX but can be ported to other operating systems. The modules also make use of low level libraries such as *SSE* for parallel computing under the new generation Intel processors architectures. Threading at the tracker level is achieved using *QThreads* from the *Qt* library.

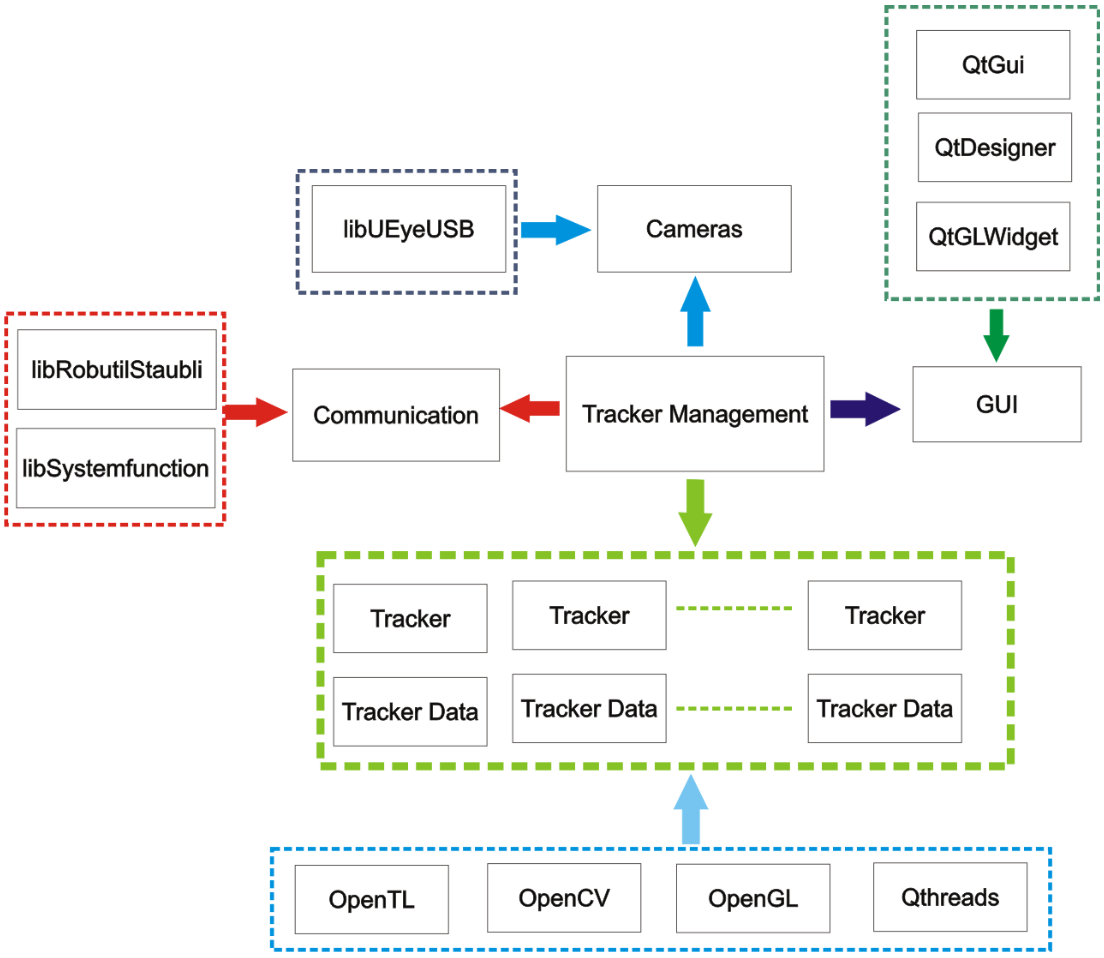


Figure A.1: Software Architecture

References

[1] M. Wojtczyk, R. Heidemann, K. Joeris, C. Zhang, M. Burnett, A. Knoll, and K. Konstantinov. **The Use of a Mobile Robot for Complete Sample Management in a Cell Culture Pilot Plant**. *Cell Technology for Cell Products*, pages 543–547, 2007. ix, 3

[2] G. Panin, C. Lenz, S. Nair, E. Roth, M. Wojtczyk, T. Friedlhuber, and A. Knoll. **A Unifying Software Architecture for Model-based Visual Tracking**. In *IS&T/SPIE 20th Annual Symposium of Electronic Imaging*, San Jose, CA, January 2008. ix, 2, 8, 19, 20, 21, 25, 26, 32, 88

[3] George H. Joblove and Donald Greenberg. **Color spaces for computer graphics**. In *Proceedings of the 5th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '78, pages 20–25, New York, NY, USA, 1978. ACM. ix, 25, 47

[4] Corinna Cortes and Vladimir Vapnik. **Support-vector networks**. *Machine Learning*, **20**(3):273–297, September 1995. x, 44, 45, 46

[5] COSROBE. **COSROBE**. www.cosrobe.com. xiii, 87, 88

[6] ZDF. **ZDF**. http://www.zdf.de/. xiv, 92, 93

[7] Plazamedia. **Plazamedia**. http://www.plazamedia.de. xiv, 103, 105, 107

[8] Vizrt and Plazamedia. **Vizrt Plasamedia Exhibition**. Plazamedia, http://www.vizrt.com/news/article5692.ece, December 2009. xiv, 105, 107

[9] C. Lenz, S. Nair, M. Rickert, A. Knoll, W. Rosel, J. Gast, A. Bannat, and F. Wallhoff. **Joint-action for humans and industrial robots for assembly tasks**. In *Robot and Human Interactive Communication, 2008. RO-MAN 2008. The 17th IEEE International Symposium on*, pages 130–135. IEEE, 2008. 1

[10] Suraj Nair, Giorgio Panin, Martin Wojtczyk, Claus Lenz, Thomas Friedelhuber, and Alois Knoll. **A Multi-Camera Person Tracking System for Robotic Applications in Virtual Reality TV Studio**. In *Proceedings of the 17th IEEE/RSJ International Conference on Intelligent Robots and Systems 2008*. IEEE, September 2008. 1, 32, 40, 85, 87

[11] S. Nair, T. Roder, G. Panin, and A. Knoll. **Visual servoing of presenters in augmented virtual reality TV studios**. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 3771–3777. IEEE. 1, 87

[12] J. Fleck and H. Scarbrough. **Labour-Management Relations and the Introduction of Industrial Robots in the Car Industry**. In *Proceedings of robots in the automotive industry: an international conference, Birmingham, UK, 20-22 April 1982*, page 23. IFS (Publications), 1982. 1

[13] C.H.A. Dassbach. **Industrial robots in the American automobile industry**. *Critical Sociology*, **13**(4):53, 1986. 1

[14] A. Camuffo and G. Volpato. **Dynamic capabilities and manufacturing automation: organizational learning in the Italian automobile industry**. *Industrial and Corporate Change*, **5**(3):813, 1996. 1

[15] P. Ingrassia and J.B. White. *Comeback: The fall and rise of the American automobile industry.* Simon and Schuster, 1995. 1

[16] D.T. Jones and J.P. Womack. **Developing countries and the future of the automobile industry**. *World Development*, **13**(3):393–407, 1985. 1

[17] K. Shimokawa. *The Japanese automobile industry: a business history.* Athlone Pr, 1994. 1

[18] W. Streeck. **Industrial relations and industrial change: the restructuring of the world automobile industry in the 1970s and 1980s**. *Economic and Industrial Democracy*, **8**(4):437, 1987. 1

[19] S. Watanabe. *Microelectronics, automation, and employment in the automobile industry.* John Wiley & Sons, 1987. 1

[20] V. Honovar and K. Balakrishnan. **On sensor evolution in robotics**. In *Genetic Programming: Proceedings of the First Annual Conference*, pages 455–460. 1

[21] BR Duffy, RW Collier, GMP O'Hare, CFB Rooney, and RPS O'Donoghue. **SOCIAL ROBOTICS: Reality and Virtuality in Agent-Based Robotics**. In *Bar-Ilan Symposium on the Foundations of Artificial Intelligence: Bridging Theory and Practice (BISFAI)*, 1999. 1

[22] F. Michaud, P. Pirjanian, J. Audet, et al. **Artificial emotion and social robotics**. 2000. 1

[23] J.J. Cabibihan, S. Pattofatto, M. Jomaa, A. Benallal, and M.C. Carrozza. **Towards humanlike social touch for sociable robotics and prosthetics: Comparisons on the compliance, conformance and hysteresis of synthetic and human fingertip skins**. *International Journal of Social Robotics*, **1**(1):29–40, 2009. 1

[24] Á. Castro-González, M. Malfaz, and M. Salichs. **Selection of actions for an autonomous social robot**. *Social Robotics*, pages 110–119, 2010. 1

[25] S.S. Ge. **Social Robotics: Integrating advances in engineering and computer science**. In *Proceedings of Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology International Conference*, pages 9–12, 2007. 1

[26] R.J. Schalkoff. *Digital image processing and computer vision.* Wiley New York:, 1989. 1

[27] K. Balakrishnan and V. Honavar. **On sensor evolution in robotics**. In *Proceedings of the First Annual Conference on Genetic Programming*, pages 455–460. MIT Press, 1996. 1

[28] J.X. Liu. *Computer vision and robotics.* Nova Science Pub Inc, 2006. 2

[29] M. Kam, X. Zhu, and P. Kalata. **Sensor fusion for mobile robot navigation**. *Proceedings of the IEEE*, **85**(1):108–119, 1997. 2

[30] Steffen Knoop, Stefan Vacek, and Ruediger Dillmann. **Fusion of 2d and 3d sensor data for articulated body tracking**. *Robotics and Autonomous Systems*, **57**(3):321–329, 2009. 2, 15

[31] H.P. Moravec. **Sensor fusion in certainty grids for mobile robots**. *AI magazine*, **9**(2):61, 1988. 2

[32] S. Shafer, A. Stentz, and C. Thorpe. **An architecture for sensor fusion in a mobile robot**. In *Robotics and Automation. Proceedings. 1986 IEEE International Conference on*, **3**, pages 2002–2011. IEEE, 1986. 2

[33] A.W. Stroupe, M.C. Martin, and T. Balch. **Distributed sensor fusion for object position estimation by multi-robot systems**. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, **2**, pages 1092–1098. IEEE, 2001. 2

[34] Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly, Cambridge, MA, November 2011. 2, 26, 133

[35] W. Eckstein and C. Steger. **The Halcon vision system: An example for flexible software architecture**. In *Proceedings of the 3. Meeting of Practical Applications on Real-Time Image Processing (organized by JSPE)*. Citeseer. 2

[36] G. Panin. *Model-based Visual Tracking: The OpenTL Framework*. John Wiley & Sons, 2011. 3, 8, 19, 25, 33, 40, 133

[37] Mathieu Salzmann, Julien Pilet, Slobodan Ilic, and Pascal Fua. **Surface Deformation Models for Nonrigid 3D Shape Recovery**. *IEEE Trans. Pattern Anal. Mach. Intell.*, **29**(8):1481–1487, 2007. 3

[38] Bodo Rosenhahn, Lei He, and Reinhard Klette. **Automatic human model generation**. In *in Computer Analysis of Images and Patterns (CAIP*, pages 41–48, 2005. 3

[39] S. Haddadin, A. Albu-Schaeffer, and G. Hirzinger. **Safety evaluation of physical human-robot interaction via crash-testing**. In *Robotics: science and systems conference (RSS2007)*, pages 217–224. Citeseer, 2007. 4

[40] J. Heinzmann and A. Zelinsky. **Quantitative safety guarantees for physical human-robot interaction**. *The International Journal of Robotics Research*, **22**(7-8):479, 2003. 4

[41] D. Kulic and E.A. Croft. **Real-time safety for human-robot interaction**. *Robotics and Autonomous Systems*, **54**(1):1–12, 2006. 4

[42] Y. Yamada, Y. Hirasawa, SY Huang, and Y. Umetani. **Fail-safe human/robot contact in the safety space**. In *Robot and Human Communication, 1996., 5th IEEE International Workshop on*, pages 59–64. IEEE, 1996. 4

[43] Jerome Berclaz Francois, Jérôme Berclaz, François Fleuret, and Pascal Fua. **Robust People Tracking with Global Trajectory Optimization**. In *In Conference on Computer Vision and Pattern Recognition*, pages 744–750, 2006. 5, 13, 68

[44] Dariu M. Gavrila. **Vision-based 3-D tracking of humans in action**. Technical report, 1996. 5

[45] Thomas B. Moeslund, Adrian Hilton, and Volker Kr&#252;ger. **A survey of advances in vision-based human motion capture and analysis**. *Computer Vision and Image Understanding*, **104**(2-3):90–126, November 2006. 9, 15

[46] Liang Wang, Weiming Hu, and Tieniu Tan. **Recent Developments in Human Motion Analysis**. 9

[47] D. M. Gavrila. **The Visual Analysis of Human Movement: A Survey**. *Computer Vision and Image Understanding*, pages 82–98, 1999. 9

[48] Thiago T. Santos and Carlos Hitoshi Morimoto. **Multiple camera people detection and tracking using support integration**. *Pattern Recognition Letters*, **32**(1):47–55, 2011. 9, 11

[49] Christopher Richard Wren, Ali Azarbayejani, Trevor Darrell, and Alex Pentland. **Pfinder: Real-Time Tracking of the Human Body**. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **19**(7):780–785, 1997. 10

[50] Katja Nummiaro, Esther Koller-Meier, and Luc J. Van Gool. **An adaptive color-based particle filter**. *Image Vision Comput.*, **21**(1):99–110, 2003. 10

[51] Patrick Pérez, Carine Hue, Jaco Vermaak, and Michel Gangnet. **Color-Based Probabilistic Tracking**. In *ECCV '02: Proceedings of the 7th European Conference on Computer Vision-Part I*, pages 661–675, London, UK, 2002. Springer-Verlag. 10, 33

[52] Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. **Kernel-Based Object Tracking**. *IEEE Trans. Pattern Anal. Mach. Intell.*, **25**(5):564–575, 2003. 10

[53] Bernt Schiele, Mykhaylo Andriluka, Nikodem Majer, Stefan Roth, and Christian Wojek. **Visual People Detection: Different Models, Comparison and Discussion**. In *Proceedings of the IEEE ICRA 2009 Workshop on People Detection and Tracking*, pages 1–8, 2009. 10, 11

[54] Christian Wojek, Gyuri Dorkó, André Schulz, and Bernt Schiele. **Sliding-Windows for Rapid Object Class Localization: A Parallel Technique**. In *DAGM-Symposium*, pages 71–81, 2008. 10

[55] Constantine Papageorgiou and Tomaso Poggio. **A Trainable System for Object Detection**. *Int. J. Comput. Vision*, **38**:15–33, June 2000. 10

[56] Navneet Dalal and Bill Triggs. **Histograms of Oriented Gradients for Human Detection**. In Cordelia Schmid, Stefano Soatto, and Carlo Tomasi, editors, *International Conference on Computer Vision & Pattern Recognition*, **2**, pages 886–893, INRIA Rhône-Alpes, ZIRST-655, av. de l'Europe, Montbonnot-38334, June 2005. 10

[57] Bastian Leibe, Edgar Seemann, and Bernt Schiele. **Pedestrian detection in crowded scenes**. In *In CVPR*, pages 878–885, 2005. 11

[58] **Multi-Aspect Detection of Articulated Objects**. 11

[59] F. Huo and E.A. Hendriks. **Real time multiple people tracking and pose estimation**. In *Proceedings of the 1st ACM international workshop on Multimodal pervasive video analysis*, pages 5–10. ACM, 2010. 11

[60] R. Eshel and Y. Moses. **Homography based multiple camera detection and tracking of people in a dense crowd**. 2008. 11

[61] F. Fleuret, J. Berclaz, R. Lengagne, and P. Fua. **Multicamera people tracking with a probabilistic occupancy map. Pattern Analysis and Machine Intelligence**. *IEEE Transactions on*, **30**(2):267–282, 2008. 11, 16, 61

[62] W. Hu, M. Hu, X. Zhou, T. Tan, J. Lou, and S. Maybank. **Principal axis-based correspondence between multiple cameras for people tracking. Pattern Analysis and Machine Intelligence**. *IEEE Transactions on*, **28**(4):663–671, 2006. 11

[63] K. Kim and L. Davis. **Multi-camera tracking and segmentation of occluded people on ground plane using search-guided particle filtering**. *Computer Vision–ECCV 2006*, pages 98–109, 2006. 11, 16

[64] T.T. Santos and C.H. Morimoto. **People detection under occlusion in multiple camera views**. In *Computer Graphics and Image Processing, 2008. SIBGRAPI'08. XXI Brazilian Symposium on*, pages 53–60. IEEE, 2008. 11

[65] B. Han, S.W. Joo, and L.S. Davis. **Multi-Camera Tracking with Adaptive Resource Allocation**. *International Journal of Computer Vision*, pages 1–14, 2011. 12

[66] C. Soto, B. Song, and A.K. Roy-Chowdhury. **Distributed multi-target tracking in a self-configuring camera network**. 2009. 12

[67] R. Olfati-Saber. **Kalman-consensus filter: Optimality, stability, and performance**. In *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*, pages 7036–7042. IEEE, 2009. 12

[68] G. Englebienne, T. Van Oosterhout, and B. Krose. **Tracking in sparse multi-camera setups using stereo vision**. In *Distributed Smart Cameras, 2009. ICDSC 2009. Third ACM/IEEE International Conference on*, pages 1–6. IEEE, 2009. 12

[69] A. Mittal and L.S. Davis. **M 2 Tracker: a multi-view approach to segmenting and tracking people in a cluttered scene**. *International Journal of Computer Vision*, **51**(3):189–203, 2003. 12

[70] J. Berclaz, F. Fleuret, and P. Fua. **Multi-camera tracking and atypical motion detection with behavioral maps**. *Computer Vision–ECCV 2008*, pages 112–125, 2008. 12, 17

[71] S.M. Khan and M. Shah. **Tracking multiple occluding people by localizing on multiple scene planes**. *IEEE transactions on pattern analysis and machine intelligence*, pages 505–519, 2008. 12, 13

[72] MR Weathersby and DE Schmieder. **An experiment quantifying the effect ofclutter on target detection**. In *Proceedings of the International Society for Optical Engineering (SPIE)*, pages 26–33, 1984. 13

[73] D.E. Schmieder and M.R. Weathersby. **Detection performance in clutter with variable resolution**. *Aerospace and Electronic Systems, IEEE Transactions on*, (4):622–630, 1983. 13

[74] GR Loefer, DE Schmieder, WM Finlay, and MR Weathersby. **An Infrared Background Clutter Model Using 3-D Computer Graphics**. *IEEE Computer Graphics and Applications*, pages 55–66, 1983. 13

[75] T.J. Doll and D.E. Schmieder. **Infrared target detection in structured urban scenes**. In *Human Factors and Ergonomics Society Annual Meeting Proceedings*, **32**, pages 1324–1328. Human Factors and Ergonomics Society, 1988. 13

[76] Rafael Muñoz-Salinas, Eugenio Aguirre, and Miguel García-Silvente. **People detection and tracking using stereo vision and color**. *Image Vision Comput.*, **25**:995–1007, June 2007. 13

[77] I. Haritaoglu, D. Harwood, and L. S. Davis. **W4: A Real Time System for Detecting and Tracking People**. In *CVPR '98: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, page 962, Washington, DC, USA, 1998. IEEE Computer Society. 13

[78] Nils T. Siebel and Stephen J. Maybank. **Fusion of Multiple Tracking Algorithms for Robust People Tracking**. In *ECCV '02: Proceedings of the 7th European Conference on Computer Vision-Part IV*, pages 373–387, London, UK, 2002. Springer-Verlag. 13

[79] Michael Isard and John MacCormick. **BraMBLe: A Bayesian Multiple-Blob Tracker**. In *ICCV*, pages 34–41, 2001. 13, 26

[80] D. M. Gavrila. **Pedestrian detection from a moving vehicle**. In *Proc. of European Conference on Computer Vision*, pages 37–49, Dublin, Ireland, 2000. 13

[81] Sohaib Khan, Omar Javed, Zeeshan Rasheed, and Mubarak Shah. **Human Tracking in Multiple Cameras**. In *In International Conference on Computer Vision*, pages 331–336, 2001. 13

[82] D. Focken and R. Stiefelhagen. **Towards vision-based 3-d people tracking in a smart room**. 2002. 13

[83] Q. Cai and JK Aggarwal. **Tracking human motion using multiple cameras**. In *Pattern Recognition, 1996., Proceedings of the 13th International Conference on*, **3**, pages 68–72. IEEE, 1996. 14

[84] S.L. Dockstader and A.M. Tekalp. **Multiple camera tracking of interacting and occluded human motion**. *Proceedings of the IEEE*, **89**(10):1441–1455, 2001. 14

[85] T.H. Chang and S. Gong. **Tracking multiple people with a multi-camera system**. *womot*, page 0019, 2001. 14

[86] K. Toyama and E. Horvitz. **Bayesian modality fusion: Probabilistic integration of multiple vision algorithms for head tracking**. In *Proceedings of ACCV 2000, Fourth Asian Conference on Computer Vision*. Citeseer, 2000. 14

[87] K. Hayashi, M. Hashimoto, K. Sumi, and K. Sasakawa. **Multiple-person tracker with a fixed slanting stereo camera**. In *Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference on*, pages 681–686. IEEE, 2004. 14

[88] T. Zhao, M. Aggarwal, R. Kumar, and H. Sawhney. **Real-time wide area multi-camera stereo tracking**. 2005. 14

[89] Daniel Grest, Jan Woetzel, and Reinhard Koch. **Nonlinear Body Pose Estimation from Depth Images**. In Walter Kropatsch, Robert Sablatnig, and Allan Hanbury, editors, *Pattern Recognition*, **3663** of *Lecture Notes in Computer Science*, pages 285–292. Springer Berlin / Heidelberg, 2005. 10.1007/11550518_36. 15

[90] Christian Plagemann, Varun Ganapathi, Daphne Koller, and Sebastian Thrun. **Real-time identification and localization of body parts from depth images.** In *ICRA'10*, pages 3108–3113, 2010. 15

[91] M. Siddiqui and G. Medioni. **Human pose estimation from a single view point, real-time range sensor**. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, pages 1–8, june 2010. 15

[92] Jamie Shotton, Andrew Fitzgibbon, Mat Cook, Toby Sharp, Mark Finocchio, Richard Moore, Alex Kipman, and Andrew Blake. **Real-Time Human Pose Recognition in Parts from a Single Depth Image**. In *CVPR*, 2011. 15

[93] C. Schlegel, J. Illmann, H. Jaberg, M. Schuster, and R. Worz. **Vision based person tracking with a mobile robot**. In *British Machine Vision Conference*, pages 418–427. Citeseer, 1998. 15

[94] M. Weser, D. Westhoff, M. Huser, and J. Zhang. **Multimodal people tracking and trajectory prediction based on learned generalized motion patterns**. In *Multisensor Fusion and Integration for Intelligent Systems, 2006 IEEE International Conference on*, pages 541–546. IEEE, 2006. 15, 16

[95] Marin Kobilarov and Gaurav Sukhatme. **People tracking and following with mobile robot using an omnidirectional camera and a laser**. pages 557–562, 2006. 16

[96] *MIT Pedestrian Database MITP.* Online, 2000. 17, 61, 129

[97] S. Bahadori, G. Grisetti, L. Iocchi, G. R. Leone, and D. Nardi. **Real-Time Tracking of Multiple People through Stereo Vision**. In *In Proc. Of IEE International Workshop on Intelligent Environments*, 2005. 17, 61

[98] Giorgio Panin, Erwin Roth, Thorsten Röder, Suraj Nair, Claus Lenz, Martin Wojtczyk, Thomas Friedlhuber, and Alois Knoll. **ITrackU: An Integrated Framework for Image-based Tracking and Understanding**. In *Proceedings of the International Workshop on Cognition for Technical Systems*, Munich, Germany, October 2008. 19

[99] Fatih Kurugollu, Bülent Sankur, and A. Emre Harmancl. **Color image segmentation using histogram multithresholding and fusion**. *Image Vision Comput.*, **19**(13):915–928, 2001. 25

[100] Yaakov Bar-Shalom and Xiao-Rong Li. *Estimation with Applications to Tracking and Navigation.* John Wiley & Sons, Inc., New York, NY, USA, 2001. 25

[101] I. Gallo and E. Binaghi. *Advances in Computer Graphics and Computer Vision,* **4**, chapter Dense Stereo Matching With Growing Aggregation and Neural Learning, pages 343–353. 2007. 26

[102] Andreas Wedel, Thomas Pock, Christopher Zach, Horst Bischof, and Daniel Cremers. *An Improved Algorithm for TV-L1 Optical Flow.* Springer-Verlag, Berlin, Heidelberg, 2009. 26

[103] M. Isard and A. Blake. **Condensation – conditional density propagation for visual tracking**. *International Journal of Computer Vision (IJCV)*, **29**(1):5–28, 1998. 26, 32

[104] Andrew Blake and M. Isard. *Active Contours: The Application of Techniques from Graphics, Vision, Control Theory and Statistics to Visual Tracking of Shapes in Motion*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1998. 26

[105] Michael Isard and Andrew Blake. **ICONDENSATION: Unifying Low-Level and High-Level Tracking in a Stochastic Framework**. In *ECCV '98: Proceedings of the 5th European Conference on Computer Vision-Volume I*, pages 893–908, London, UK, 1998. Springer-Verlag. 26

[106] Greg Welch and Gary Bishop. **An Introduction to the Kalman Filter**. Technical report, 2004. 26, 87

[107] Zia Khan. **MCMC-Based Particle Filtering for Tracking a Variable Number of Interacting Targets**. *IEEE Trans. Pattern Anal. Mach. Intell.*, **27**(11):1805–1918, 2005. 26

[108] Suraj Nair, Emmanual Dean, and Alois Knoll. **3D Position based Multiple Human Servoing by Low-Level-Control of 6 DOF Industrial Robot**. In *Proceedings of the 2011 IEEE International Conference on Robotics and Biomimetics*. IEEE, December 2011. 27, 106

[109] P. KaewTraKulPong and R. Bowden. **An improved adaptive background mixture model for real-time tracking with shadow detection**. In *Proc. European Workshop Advanced Video Based Surveillance Systems*, **1**. Citeseer, 2001. 27

[110] Darren E. Butler, Jr. V. Michael Bove, and Sridha Sridharan. **Real-time adaptive foreground/background segmentation**. *EURASIP J. Appl. Signal Process.*, **2005**:2292–2304, January 2005. 27, 41

[111] D.M. Titterington, A.F.M. Smith, and U.E. Makov. *Statistical analysis of finite mixture distributions*. Wiley series in probability and mathematical statistics. Applied probability and statistics. Wiley, 1985. 27

[112] A. Bhattacharyya. **On a measure of divergence between two statistical populations defined by their probability distributions**. *Bulletin of the Calcutta Mathematical Society*, **35**:99–109, 1943. 33

[113] Qt. **Qt**. http://qt.nokia.com/. 36, 133

[114] W. Richard Stevens. *UNIX network programming, volume 2 (2nd ed.): interprocess communications*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1999. 37

[115] **The ethernet: a local area network: data link layer and physical layer specifications**. *SIGCOMM Comput. Commun. Rev.*, **11**:20–66, July 1981. 38

[116] Suraj Nair, Emmanual Dean-Leon, and Alois Knoll. **REAL-TIME 3D MULTIPLE HUMAN TRACKING WITH ROBUSTNESS ENHANCEMENT THROUGH MACHINE LEARNING**. 39

[117] K.S. Fu and JK Mui. **A survey on image segmentation**. *Pattern Recognition*, **13**(1):3–16, 1981. 41

[118] N.R. Pal and S.K. Pal. **A review on image segmentation techniques**. *Pattern recognition*, **26**(9):1277–1294, 1993. 41

[119] C. Kamath and SS Cheung. **Robust techniques for background subtraction in urban traffic video**. Technical report, Lawrence Livermore National Laboratory (LLNL), Livermore, CA, 2003. 41

[120] Julien Pilet, Christoph Strecha, and Pascal Fua. **Making Background Subtraction Robust to Sudden Illumination Changes**. In *In Proc. European Conf. on Computer Vision*, 2008. 41

[121] P. Carr. **Gpu accelerated multimodal background subtraction**. In *Computing: Techniques and Applications, 2008. DICTA'08. Digital Image*, pages 279–286. IEEE. 41

[122] A. Griesser, SD Roeck, A. Neubeck, and LV Gool. **GPU-based foreground-background segmentation using an extended colinearity criterion**. In *Vision, Modeling, and Visualization (VMV)*, 2005. 41

[123] N. Cristianini and J. Shawe-Taylor. *An introduction to support Vector Machines: and other kernel-based learning methods*. Cambridge Univ Pr, 2000. 44

[124] N. Cristianini and J. Shawe-Taylor. **Support vector machines**, 2000. 44

[125] M.A. Hearst, ST Dumais, E. Osman, J. Platt, and B. Scholkopf. **Support vector machines**. *Intelligent Systems and their Applications, IEEE*, **13**(4):18–28, 1998. 44

[126] T. Joachims, B. Schoulkopf, C. Burges, and A. Smola. **Making Large-Scale SVM Learning Practical. Advances in Kernel Methods - Support Vector Learning**. In *MIT Press*, 1999. 44

[127] J.C. Platt. **12 Fast Training of Support Vector Machines using Sequential Minimal Optimization**. 1998. 44

[128] J. Shawe-Taylor and N. Cristianini. **Support Vector Machines**, 2000. 44

[129] I. Steinwart and A. Christmann. *Support vector machines.* Springer Verlag, 2008. 44, 105

[130] T. D'Orazio, M. Leo, N. Mosca, P. Spagnolo, and PL Mazzeo. **A semi-automatic system for ground truth generation of soccer video sequences**. In *Advanced Video and Signal Based Surveillance, 2009. AVSS'09. Sixth IEEE International Conference on*, pages 559–564. IEEE, 2009. 61

[131] N. Dalal. **INRIA Person Dataset**. *Online: http://pascal. inrialpes. fr/data/human*, 2005. 61

[132] P. Dollár, C. Wojek, B. Schiele, and P. Perona. **Pedestrian Detection: A Benchmark**. In *CVPR*, June 2009. 61

[133] Ton Roosendaal and Stefano Selleri. *The Official Blender 2.3 Guide: Free 3D Creation Suite for Modeling, Animation, and Rendering.* No Starch Press, June 2004. 62

[134] G. Klein. **SPORTS AND LEISURE-30 Years of Playmobil: Investing in the Future**. *Kunststoffe-Plast Europe*, **94**(10):330, 2004. 62

[135] Suraj Nair, Emmanual Dean-Leon, and Alois Knoll. **Video of 3D Multiple Human Tracking with 4 Targets in simulated environment**. http://www.youtube.com/watch?v=cJ8sDGKAcac, August 2011. 68

[136] Suraj Nair, Emmanual Dean-Leon, and Alois Knoll. **Video of 3D Multiple Human Tracking with 5 Targets in simulated environment where 2 Targets have Similar Appearance**. http://www.youtube.com/watch?v=g0HbIYap-hw, August 2011. 69

[137] Suraj Nair, Emmanual Dean-Leon, and Alois Knoll. **Video of 3D Multiple Human Tracking with 3 Targets in Constrained Real Environment**. http://www.youtube.com/watch?v=wePVQ7cXB9c, October 2011. 71

[138] Suraj Nair. **3D Multiple Human Tracking in simulated environment under drastic changes in Lighting Conditions**. http://www.youtube.com/watch?v=Jm2D1jUsuFM. 72

[139] Suraj Nair, Emmanual Dean-Leon, and Alois Knoll. **Video of 3D Multiple Human Tracking in Real Environment with Drastically Changing Lighting Conditions**. http://www.youtube.com/watch?v=yZHCXgdDf14, October 2011. 73

[140] F.A. Haight and Operations Research Society of America. **Handbook of the Poisson distribution**. 1967. 75

[141] P.C. Consul and GC Jain. **A generalization of the Poisson distribution**. *Technometrics*, pages 791–799, 1973. 75

[142] R.A. Boie and I.J. Cox. **An Analysis of Camera Noise**. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **14**:671–674, 1992. 75

[143] Suraj Nair, Emmanual Dean-Leon, and Alois Knoll. **3D Multiple Human Tracking in simulated environment with similar targets moving very close to each other**. http://www.youtube.com/watch?v=0vi7vtXIBxU. 76

[144] Suraj Nair, Emmanual Dean-Leon, and Alois Knoll. **Video of 3D Multiple Human Tracking with 3 Targets in Real Environment Moving Very Close to each other and in Black Clothing**. http://www.youtube.com/watch?v=3ChU0W1QHoc, October 2011. 77

[145] Suraj Nair, Giorgio Panin, Thorsten Röder, Thomas Friedelhuber, and Alois Knoll. **A distributed and scalable person tracking system for robotic**

visual servoing with 8 dof in virtual reality tv studio automation. In *Proceedings of the 6th International Symposium on Mechatronics and its Applications (ISMA09)*. IEEE, March 2009. 87

[146] Robotics Technology Leaders, GmbH. **www.rtleaders.com**. 87, 92

[147] Robert Hanek and Michael Beetz. **The Contracting Curve Density Algorithm: Fitting Parametric Curve Models to Images Using Local Self-Adapting Separation Criteria**. *Int. J. Comput. Vision*, **59**(3):233–258, 2004. 87, 89, 90

[148] Giorgio Panin, Alexander Ladikos, and Alois Knoll. **An Efficient and Robust Real-Time Contour Tracking System**. In *ICVS*, page 44, 2006. 87, 91

[149] G.E. Uhlenbeck and L.S. Ornstein. **On the theory of the Brownian motion**. *Physical Review*, **36**(5):823, 1930. 89

[150] Staubli. **Staubli**. www.staubli.com. 92, 94, 106, 110

[151] Thomas Friedlhuber, Kai Klimke, Markus Rickert, and Alois Knoll. **Echtzeitsteuerung eines Stäubli Industrieroboters über TCP/IP**. Technical report, Technische Universitaet Muenchen, March 2007. 94, 110, 133

[152] Suraj Nair, Sebastian Fruehling, and Alois Knoll. **Video of Tracking Video with Zoom Control**. http://www.youtube.com/watch?v=VDtIAByFJog, October 2010. 97

[153] Suraj Nair and Alois Knoll. **Video of Person Tracking Video for HDTV Studios**. http://www.youtube.com/watch?v=xpJWHYinyrE, August 2011. 97

[154] VIZRT. **www.vizrt.com**. 99, 101, 105

[155] Suraj Nair, Sebastian Fruehling, and Alois Knoll. **Video of 3D Multiple Human Tracking for interaction with Virtual Objects in a HDTV Studio**. http://www.youtube.com/watch?v=-VfOPVnjHRs, October 2010. 102

[156] ESPN. **ESPN**. http://espn.go.com. 103

[157] NFL. **NFL**. http://www.nfl.com/. 103

[158] sky sky sky sky. **SKY**. http://www.sky.com. 103

[159] D. Runde. **How to realize a natural image reproduction using stereoscopic displays with motion parallax**. *Circuits and Systems for Video Technology, IEEE Transactions on*, **10**(3):376–386, 2000. 105

[160] C. Fehn. **3D-TV using depth-image-based rendering (DIBR)**. In *Proc. Picture Coding Symp*, page 2004. 105

[161] 3D Glasses. **3D Glasses**. http://www.3dglasses.net. 105

[162] K. Perlin, S. Paxia, and J.S. Kollin. **An autostereoscopic display**. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 319–326. ACM Press/Addison-Wesley Publishing Co., 2000. 105

[163] N.A. Dodgson. **Autostereoscopic 3D displays**. *Computer*, **38**(8):31–36, 2005. 105

[164] W. Matusik and H. Pfister. **3D TV: a scalable system for real-time acquisition, transmission, and autostereoscopic display of dynamic scenes**. In *ACM Transactions on Graphics (TOG)*, **23**, pages 814–824. ACM, 2004. 105

[165] L. Zhang and W.J. Tam. **Stereoscopic image generation based on depth images for 3D TV**. *Broadcasting, IEEE Transactions on*, **51**(2):191–199, 2005. 105

[166] TECHNODOLLY. **TECHNODOLLY**. http://www.supertechno.com/. 105

[167] Suraj Nair, Sebastian Fruehling, and Alois Knoll. **Video of 3D Multiple Human Tracking for Stereoscopic rendering in HDTV Studios**. http://www6.in.tum.de/ nair/stereoscopy.mp4, August 2011. 106

[168] Suraj Nair, Sebastian Fruehling, and Alois Knoll. **Video of 3D Multiple Human Tracking for Stereoscopic rendering in HDTV Studios**. http://www.youtube.com/watch?v=GrEWGlWZCnE, August 2011. 106

[169] F. Chaumette and S. Hutchinson. **Visual servo control. I. Basic approaches**. *Robotics Automation Magazine, IEEE*, **13**(4):82–90, 2006. 108

[170] S. Hutchinson, G.D. Hager, and P.I. Corke. **A tutorial on visual servo control**. *Robotics and Automation, IEEE Transactions on*, **12**(5):651–670, October 1996. 108

[171] Michael Goodrich and Alan Schultz. **Human–Robot Interaction: A Survey**. *Foundations and Trends in Human–Computer Interaction*, **1**(3):203–275, 2007. 108

[172] Suraj Nair, Emmanual Dean-Leon, and Alois Knoll. **Video: Position Based Visual Sevoing of Multiple Humans**. September 2010. 114

[173] Lego Duplo. **Lego Duplo**. http://duplo.lego.com. 117

[174] Satoshi Suzuki and Keiichi Abe. **Topological structural analysis of digitized binary images by border following**. *Computer Vision, Graphics, and Image Processing*, **30**(1):32–46, April 1985. 120

[175] Schunk. **Schunk GmbH**. http://www.de.schunk.com. 122

[176] JR3. **JR3. Inc**. http://www.jr3.com. 122

[177] Suraj Nair, Emmanual Dean-Leon, and Alois Knoll. **Video of 3D Object Tracker**. http://www.youtube.com/watch?v=4XBn5UzgHp4, May 2011. 124

[178] Suraj Nair, Emmanual Dean-Leon, and Alois Knoll. **Video of Close Range Interaction with Humans, Robot and Objects**. http://www.youtube.com/watch?v=6OGJiRKB2zM, October 2011. 125

[179] Suraj Nair, Emmanual Dean-Leon, and Alois Knoll. **Video of 3D Multiple Human Tracker for Close Range Interaction with Robot and Objects**. http://www.youtube.com/watch?v=eHHg-l3MOB0, May 2011. 125

[180] Suraj Nair, Emmanual Dean-Leon, and Alois Knoll. **Video of Close Range Interaction with Humans, Robot and Objects under Drastically Changing Lighting Conditions**. http://www.youtube.com/watch?v=Pn7kyhlEkEc, October 2011. 126

[181] ARB OpenGL, M. Woo, J. Neider, and T. Davis. **OpenGL programming guide**, 1999. 133

[182] IDS Imaging. *IDS Imaging*. http://www.ids-imaging.de/. 133