Active Vision for Humanoid Robots

Xin Wang

因为有你，我和这世界息息相关—— To my daughter, Emilie

Active Vision for Humanoid Robots

PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus Prof. ir. K.C.A.M. Luyben,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op vrijdag 25 september 2015 om 10:00
uur
door

Xin Wang

Master of Science in Signal and Information Processing Engineering
Northwestern Polytechnical University
geboren te Shaanxi, China

Dit proefschrift is goedgekeurd door de promotor:

Prof. dr. ir. P.P. Jonker

Samenstelling promotiecommissie:

Rector Magnificus	
voorzitter	
Prof. dr. ir. P.P. Jonker	Technische Universiteit Delft, promotor
Onafhankelijke leden	
Prof. dr. ir. M.J.T. Reinders	Technische Universiteit Delft
Prof. dr. R.C. Veltkamp	Universiteit Utrecht
Dr. Çağatay Soyer	NATO Communications and Information Agency, The Hague
Prof. dr. ir. Peter Veelaert	Universiteit Gent
Prof. dr. F.C.T van der Helm	Technische Universiteit Delft
Overige leden	
Dr. B.A.J. Lenseigne	Technische Universiteit Delft
Prof. dr. ir. TR. Babuška	Technische Universiteit Delft

Cover drawing by Stephan Timmers, 2014

ISBN 978-94-6203-877-6

Author email: wangxin0913@gmail.com

Acknowledgements

The road was more difficult than I expected, however, reaching the end brings much more than what I expected. It is a great opportunity to express my sincere thanks to those who were so generous to spend their time to offer me professional and personal help and were always there to encourage me to continue one step further, until I reached this terminal point.

I am especially indebted to my supervisor Prof. Pieter Jonker, who introduced me from China to here, and allowed me to grow as an independent researcher. Your visionary thoughts in the field of robotics always spark my interests and propel me to seek innovative ideas. Besides, you gave me enormous help in my life, especially during my pregnancy. You were also very patient to teach me how to express myself and discuss problems using a Netherlands way instead of a Chinese way. These 5 years, I learned a lot by working with you.

I would like to give my special thanks to all the members of the Delft Biorobotics Lab. This Ph.D thesis could not be finished without your valuable support. Martijn, thanks for letting me be a coach of Minor robotics projects. I learned a lot by supervising students and I was so glad to see that robots can do so many things. Boris thank you for leading me to get to the right track of my research and sharing interesting ideas during the lunch time. Maja thanks for helping me from every perspectives, especially writing together with me my first paper. I still remembered we were always the last two persons that came out of the office. Eelko, you were so willing to give valuable advice to me not only for the technical part but also for personal life. Jan and Guus, you helped me so much for building up my lovely robot head, from which I gained a lot of mechanical and electronic knowledge. Tim and Wouter, when I worked downstairs, you were so generous to spend your effort to teach me how to solve the control problems of my setup. I learned from you how to become a good researcher. Toby, thanks for all the discussion on vision related algorithms. You were such an easy going person. My old colleagues Erik and Oytun, you were like my big brothers, offering me help whenever there was

a need. Kimberly, your positive attitude towards research influences me a lot and you also helped me to translate the summary of my thesis. I also own thanks to Aswin, Berk, Daniel, Floris, Lei Qujiang, Jeff, Machiel, Michiel, Mukunda, Rob, Shiqian, Susana, Tomas, Wietse, Zhan jun (in alphabetical order).

I have spent a great time in Delft with my good friends Liangyue, Hu yu, Qi gao, Zeng yuan, Zhengzhong, Yangyang, Cui hao, Alberto, Zhu tian, Tiago, Qiaole, Chunman, Steven, Xiong liang, Cong zhe, Chunyan, Cuiting, Zhang lu, Huajie, Huaizhou, He yuan, Qu chao, Ling yun, Kang ni, Junchao, Wang chang, Changyun, Tao ke, Ke qian, Congli, Claire, Xuexue, Li ying, Kimberley, Rolf, Peter, Melanie, Claudia, Layla, Marta, Andres, Milene, Huijun, Mini, Jiaojiao, Panpan. Here I specially want to thank Linlin and Lingyan, you helped me so much during my pregnancy.

Nobody has been more important to me in the pursuit of this Ph.D project than my parents. Although we were so apart from each other, you gave me infinitive love, support, trust, understanding throughout the years in whatever I pursed and encountered.

No words is heavy enough for expressing my thanks to my dear husband Éric. I was such a lucky woman to meet you and marry you. You support me and provide unending inspiration to my life and work. Most important of all, even during struggling days, you are always holding my hands no matter what happened. I wish we will continue our life journey together and explore this unknown world the same as my robot head.

My dearest daughter Émilie, you are my endless power and energy. You angel smile can always bring me happiness after a day of work. You taught me how to love and being loved. I am so glad to have you and being dependant on you. This thesis is my gift to you.

Contents

1 Introduction

Computer vision seeks to develop algorithms that replicate one of the most amazing capabilities of the human brain - inferring properties of the external world purely by means of the light reflected from various objects into the eyes. From a technical point of view, computer vision is a set of methods that covers acquiring, processing, analyzing and understanding images. Computer vision offers solutions that are cheap, practical, non-invasive and most important of all - it mimics a natural way of sensing the world similar to human vision. Based on these advantages, it is widely applied to numerous fields, such as robotics, video surveillance, automatic driving, automatic inspection, medical imaging, object modeling, human-computer interaction, augmented reality and so on.

As humans, it seems that we are able to perceive the 3D world around us and make decisions inside it without too much difficulties. In order to do that, our vision systems combine low level algorithms together with high level cognitive reasonings to be able to anticipate what we are going to see and select attention based on learning patterns. Nowadays, this high level part is still out of the reach for machines and artificial intelligence. However, we are not going to deal with the high level part in this thesis; we are more interested in building up a system that integrates mechanism, control of eye movements (Chapter 2) and low level functions such as visual primitives representation (Chapter 3), object tracking and segmentation (Chapter 4) and 3D perception (Chapter 5). So that, the high level functions can be built upon the low level functions, making such a cognitive reasoning humanoid robot possible.

1.1 Active vision

Most past and present research in machine perception has involved analysis of passively sampled data (images). Human perception, however, is not passive. It is active [1]. The basic of perceptual activity includes exploration and searching. "An active vision system is one that is able to interact with

its environment by altering its viewpoint rather than passively observing it, and by operating on sequences of images rather than on a single frame" [2]. Moreover, since a human's fovea [1] can scan over the scene, the range of the visual scene is not restricted to that of the static view. The ability to physically follow a target to maintain it in fovea increases the target resolution for higher level tasks such as classification. Besides, different eye movements are combined together to ensure a more effective way to perceive the 3D world. For instance, vergent eye movements can help to perceive objects and perform tasks within short distances.

For a mobile robot application, it is accustomed and crucial to have active vision. Active vision ensures the robot to cover a wide range of views, coordinating with other components such as arm and gripper, moving wheels to accomplish object manipulating tasks, for instance, to allow for exploration of scenes and interesting objects from a higher perceptional point of view. Most existing active vision systems either use one camera or a fixed stereo pair. Nowadays, with the advent of the Microsoft Kinect [3] that can provide real-time 3D map and gesture recognition, a great many researchers favor and use the Kinect to develop vision algorithms. However, human perception is a combination of eyes and neck movement which includes two eyes that move in a way different from the fixed stereo set-up or/and Kinect. Thus, a more complicated device design together with advanced vision algorithms is required. Our research focuses on developing a human-like vision system for mobile robots. It investigates the perception ability of humans and provides insight into mobile robot applications.

1.2 Active vision in humans

The brain is an immensely complicated structure, in which the cerebral cortex is a 3-4mm thick surface layer on top of the cerebral hemispheres. It plays a key role in memory, attention, perceptual awareness, thought, language, and consciousness. The brain contains about 100 billion neurons and it has been estimated that about 40 percent of the primate brain is involved in seeing [4]. From this we could conclude that vision plays a crucial part in information processing in the human brain. How is vision connected with the brain and how does the brain process visual information input?

[1] The fovea centralis is a small, central pit composed of closely packed cones in the eye. It is located in the center of the macula lutea of the retina. The fovea is responsible for sharp central vision (also called foveal vision), which is necessary in humans for activities where visual detail is of primary importance, such as reading and driving. Source: Wikipedia, `https://en.wikipedia.org/wiki/Fovea_centralis`

The Human visual system is shown in Figure 1.1. Vision is generated by photoreceptors in the retina, a layer of light-sensitive cells at the back of the eye. The images are transferred using the optic nerve, through the crossing at the optic chiasm, where there are partially crossed axons and partially uncrossed axons. It means that some fibers within each optic nerve cross over at this point and therefore send their information to the cerebral hemisphere on the other side of the brain and others stay on the same side of the brain. This is to ensure that the visual information from both retinas can be integrated for 3D perception. Then, through left and right optic radiation, the visual information is carried to the visual cortex (also called striate cortex), which is highly specialized for processing information about static and moving objects and is excellent in pattern recognition. In the meantime, optical nerves also provide visual information to the left and right halves of the superior colliculus, which is in concern of visual attention. For example, if an object of interest appears in the field of view, a mechanism within the superior colliculus detects its presence and guides eye movements so that the novel object can be observed directly with the full visual processing power of the central vision. Humans make on average 3 to 5 eye movements every second, which sums up to something in the order of 4.5 billion eye movements in a lifetime [5]. Therefore the perception is an active process to explore and perceive the visual environment.

What visual needs must the eye movements satisfy? Clear vision of an object requires that its image is held fairly steadily on the central, foveal region of the retina [6]. If we had no eye movements, images of the visual world would "slip" on the retina with every head movement. This would cause our vision to become blurred and our ability to recognize and localize objects to be impaired whenever we move through the environment. And when a new object of interest appears in the visual periphery, we need to point the central portion of the retina so that the object can be seen best. This requires eye movements to change the angle of gaze. *Thus, eye movements are of two main types: those that stabilize gaze and keep images steady on the retina, and those that shift gaze and redirect the line of sight to a new object of interest.* Just tracking an object to maintain it in the center of view improves stability. However, heavy vision computation will cause a significant delay. *To compensate for this effect, the human body (including the neck) uses the vestibular organ to compensate its eye movements. It has a faster processing time, therefore is able to handle disturbances at a higher frequency.*

With continuous eye movements, the human brain is able to actively explore unknown environments and learn from it. By performing a "sense-

think-act" learning pattern, perception leads to action, and in another way, through the interaction with the environment, action leads to new perceptions and learning samples as well. This basic dynamic cycle of learning can also be applied to the robotics field to gain more insight into the human visual system.

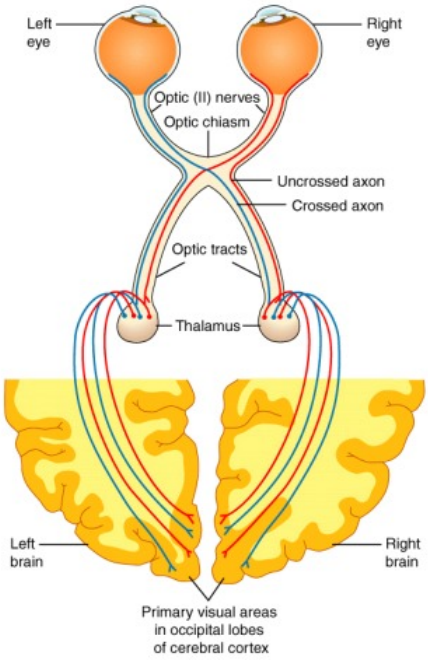


Figure 1.1: Wiley Human Visual System (Source: Wikimedia)

1.3 Active vision and robotics

This section gives an overview of active vision systems. We make a distinction between general vision systems and human-like vision systems.

1.3.1 Active vision systems

As stated in [7], there are about 2000 research papers published during 1986–2010 that are closely related to the topic of active vision perception in robotics. All the literature covers a large range of active vision research fields in robotics: humanoid vision systems, interactive robots, surveillance, attentive vision

Table 1.1: Advances in active vision systems

Humanoid attentive vision system	Harvard Binocular Head [8], MIT Binocular, Foveated Active Vision [9], MAVERic Humanoid Robot Head [10], MERTZ [11], ISAC Humanoid Robot [12], The Robot-Cub (iCub) [13], Two cameras per eye foveated vision system [14], KTH two cameras per eye active vision system [15], The Karlsruhe Humanoid Head [16], ASIMO
Vision surveillance	Pan-tilt-zoom (PTZ) cameras for video surveillance [17], Attentive vision [18]
Localization and mapping	Kalman Filtering and Extended Kalman Filtering [19], Particle Filtering [20], Sequential Monte-Carlo [21], Parallel Tracking And Mapping [22], Feature based [23], Patch based [24]
Manipulation	Model-based grasping [25] and unknown object grasping [26]
Tracking	Intensity-based [27], Motion-based [28], Template matching [29], Active contour [30] Feature-based [31], Tracking by detection [32]
Intelligent vehicle system [33]	Knowledge based methods, Stereo vision based methods, Motion based methods, Template based methods, Appearance based methods, Integrating tracking with detection (Google driverless car)
Facial interactive robots	Feelix [34], Nao robot [35], Minerva [36], Infanoid [37], Philips iCat Robot [38], Albert Einstein Hubo [39], Actroid Robot, Flobi [40]
Others	Industrial inspection, Augmented Reality, Online object modeling (SLAM based methods), Online object recognition, Various robotics platform and service robots

mechanism, object and site modeling, robot localization and mapping, navigation, path planning, exploration, tracking, search, recognition, inspection, robotic manipulation, automatic car driving, assembly and disassembly, and other purposes. We will first give an overview of the advances in each of the topics, shown in Table 1.1.

Various Simultaneous Localization and Mapping (SLAM) algorithms and systems [41] were brought up not only for mobile robotics applications but also for Augmented Reality (AR) applications, on-site object modeling, etc. In such applications, active vision is applied to map scenes and reconstruct objects from different viewpoints, in which the camera pose needs to be estimated and a 3D map needs to be reconstructed. [42, 43] gives a review about

object tracking in mobile robotics applications, and the new trend focuses on using active vision to track objects of interest inside dynamic scenes. Thus robust real-time trackers that are able to cope with dynamic environments, illumination change, and motion blur, while still keep tracking from different viewpoints, are highly required. [44] offers a survey on socially interactive Robots, in which active vision works in a way to show vivid facial expression and convey emotion. [33] is a review paper about intelligent vehicles on the road and the best known is the Google driverless car. Normally, multiple sensors together with active vision are deployed to detect obstacles and vehicles to ensure driving safety. Beside autonomous driving cars, this technique is also widely used in driver assistance.

From all above, we can conclude that there is a vast field in which active vision can be applied and it is becoming more and more popular in the robotics domain. And why?

- Mobile robots need active vision to perceive the world in a natural explorative way. By using active vision, a robot will interact with the world and perform tasks actively. For instance, in robocup@home service robot applications, active vision is used to complete tasks such as follow me, fetching an object, etc.

- Active vision can provide an effective approach for extracting useful information from a complex scene. Inspired by human vision, an active vision system usually consists of two or more cameras that can adjust its attention to the most important areas of the scene. Such a system can be useful in many applications such as active learning in an unknown environment with gaze shifting strategy, extending the field of view for autonomous vehicles or smoothly following objects.

- 3D sensors have a limited field of view and can only see a portion of a scene from a single viewpoint. A global description of objects can be obtained using active vision.

- Many active vision algorithms benefit from an ever increasing computational power, making it possible to be applied in more and more application areas.

- Active vision encompasses many computer vision techniques from low-level tasks such as feature detection, feature matching, to high-level tasks such as object detection, and 3D geometry estimation.

1.3.2 Human-like active vision systems

There are many active vision research topics and it is nearly impossible to cover all of them. Just talking about the sensor inputs, there are intensity cameras (one moving camera, two moving cameras with fixed stereo, two cameras moving separately), range sensors - among which the most popular one is Kinect - and the combinations. In this thesis we mainly focus on studying an active vision system that works in a similar way as humans, while in the mean time still having a practical use in mobile robots. Now we will first look into recent literature on humanoid vision systems.

Humanoid robots have a very long history and the first complete robot was built in 1984, called Wabot-1. However, the humanoid vision system was developed years later. In 1988, [45] proposed an "Agile camera system " with 11 degrees of freedom, which was among the first prototypes of a humanoid vision system. It presented two test cases: one is to obtain depth maps using range from focus and vergence/stereo; the other is 2-D image segmentation. However, it did not give implementation details and the test cases were rather simple. In 1992, the Harvard Binocular Head [8] with 7 degrees of freedom was presented. Three degrees of freedom were for positioning and the other degrees of freedom were for controlling of focus and the aperture of the lens. They provided examples on blob-based tracking to show saccade and smooth pursuit tracking performance. For retinal position greater than a threshold, a saccade was triggered. Saccades used position control to direct the eye to move to an absolute position, while smooth pursuit used velocity control to move to a certain displacement in a given direction. They also provided a very simple attentive model to fixate the attention using a saliency map. It also showed a depth map which was only a calculation on a static scene without any relation to eye movements. In 1994, Theimer [46] proposed a unified theory for binocular vergence control and depth recovery using phase-based techniques on their active vision setup. This disparity-evoked vergence, which was different from target-evoked vergence, was quite innovative. The MIT Cog project designed a 6 degrees of freedom, binocular, foveated active vision system. In their paper, they gave design specifications and example tests on saccades, using a saccade map generated by a simple image correlation algorithm as well as smooth pursuit, but how the control system worked was not detailed. Klarquist and Bovik [47] actively directed a pair of vergent stereo cameras to fixate on surfaces in a scene, performing multi-resolution surface depth recovery at each fixation point. However, the computation load was quite high, and depth was computed approximately in 3 – 5 min at each

fixation. Aryananda and Weber [11] created a social robot that learns to recognize a set of individuals during human-robot interaction. It had a humanoid face with a pair of eyes. However, there were no eye movements and no 3D perception was involved. The KTH active perception lab did much research on humanoid vision systems. Mårten Björkman [48] presented a real-time solution on epipolar geometry estimation for active stereo heads. The camera system in [49] consisted of two sets of cameras, a wide field pair and a foveal one for visual attention, foveating and recognizing. The former was employed to search objects of interest in a larger field of view and the latter is to attend and foveat on details. Mårten Björkman [50] presented an integrated real-time vision system that performed tasks such as object recognition, tracking and pose estimation. Rasolzadeh [51] extended previous work to perception and action. Its disparity map provided cues for figure-ground segmentation and object grasping. However, the gripper and the vision system were not integrated. Dingrui Wan [52] used a dual Pan-Tilt-Zoom camera, and proposed a novel stereo rectification method. Asfour [16] proposed an advanced humanoid vision system for studying various visual perception tasks. It had two cameras per eye and was able to do tracking and saccadic motions towards salient regions. iCub is one of the most advanced humanoid robots with a humanoid vision system [13]. It provids an open platform for cognitive and neuroscience research. Its head design had 5 degrees of freedom. The control for object tracking only used image positions of the object as feedback for visual servoing. The balancing used an inertial sensor to keep the head always in an upright position. There was also a separate sound localization function. The Karlsruhe Humanoid Head [16] was also a two cameras per eye vision system. It had a similar size as human eyes and a mechanical design as iCub, two degrees of freedom for each eye and three degrees of freedom for the neck. Open-loop and closed-loop controllers were implemented for saccade and foveation, in which the accuracy of the open-loop controller was improved by solving the inverse kinematics problem.

More and more advanced humanoid robot heads research springs up. To summarize, most state of the art humanoid robot vision systems developed so far have various research purposes. There are following the categories.

1. Control schemes of eye movements with multiple degrees of freedom
2. Attentive vision with high resolution fovea either using two cameras per eye or spatial-variant sampling
3. 3D reconstruction based on stereo and vergence control

4. High level computer vision tasks such as face recognition and object recognition
5. Saliency detection and saliency based gaze shifting strategies
6. Calibration of such an active vision system including calibration of two cameras per eye and extrinsic calibration of two moving eyes

As stated, active vision involves moving cameras that work in interaction with surrounding environments. The human vision system is one of the most advanced active vision systems because it has the feature to explore the surrounding world and "gaze" at interesting objects. By using this active observing nature, instead of just passively receiving input from the surrounding, we are able to direct our vision towards "the potentially need-to-be-learned" objects and environment and perceive useful and important information. Mobile robots that autonomously perform tasks in unknown dynamic environments also need to use active vision to search useful information. Based on an intentive vision strategy to self-explore unknown environments, mobile robots are able to develop intelligent cognitive learning skills. For example, one of the typical tasks for service robots at the robocup@home competition involves fetch-and-carry operations, in which a robot needs to find interesting objects by exploring an unknown environment, and track objects from different perspectives while approaching, fetching, and carrying them towards a user. During the whole processes, the mobile robot explores the unstructured environment and navigates around the interesting objects in which active vision plays an important role. Thereupon, we proposed an advanced active vision system that has the mechanism and controllers to achieve eye movements in a similar way to humans'. Besides, low level functions such as visual primitives representation, object tracking and segmentation as well as 3D perception were researched and implemented. The extensive experimental results prove that the proposed active vision system provides the possibility for future high level cognitive research.

1.4 Thesis outline

To fully illustrate our active vision system and its related algorithms, we divided this thesis into the following chapters. This chapter presents an overview of research on active vision systems. It covers the reasons why this research is important as well as recent developments and progress in this field.

- Chapter 2 presents the set-up and overview architecture of our active system including hardware and software design. The control mechanism of different eye movements is also shown in this chapter. Experimental results prove that our vision system is able to mimic different kinds of eye movements in a similar way as humans.
- Chapter 3 starts from a low level visual primitives representation of objects and world, and proposes a novel adaptive tracking selection mechanism based on the properties of objects. It treats different objects with different tracking algorithms in order to avoid a universal solution, which is impossible under real world constraints.
- Chapter 4 describes our robust online segmentation algorithm, by which not only the position of the object is known, but also the precise contour and shape information is provided. Besides, it can cope with viewpoint changing, occlusion, clutterd background, illumination variance, and so on.
- Chapter 5 provides a kinematic model of the whole setup and explains the process of extrinsic calibration of such a low-cost system. It also introduces a multimodal depth perception method which is inspired by the human visual system for depth estimation.
- Chapter 6 concludes and describes the future work. It also lists many applications to which active vision can be applied.
- Appendices summarize the mathematical background of the thesis, which ranges from 2D image formation, camera model, to 3D visual odometry. It provides detailed explanations of the algorithms used in the implementation of our active vision system.

2 Design and control of our active vision system

[1]For design of an advanced humanoid active vision system, eye movements similar to human eyes should be taken into consideration. More degrees of freedom will bring advanced features for active perception. However, it will also bring more difficulties to control as well as computer vision related tasks. Due to this concern, we opt for a simplified mechanical design which works for most perception tasks, while simplifying all the tasks involved. We also put emphasis on mechanical designs for different eye movements and the vision tasks that drive these movements. Many state of the art algorithms did not explore enough the importance of multiple cues that contribute to depth perception, for instance motion parallax, optical flow, and so on. Especially, the two eyes are working together to obtain 3D perception, for instance convergence and stereopsis. There is also other vision related computing involved, such as object tracking and segmentation during smooth tracking. Therefore we propose our vision system (see Figure 2.1), which is composed of the following parts.

1. The mechanical design

   The goal of our humanoid active vision system is to gain insight into biological inspired vision systems. It is desirable to have an independent vergence angle control for two cameras. In human vision there is a limited ability to perform independent tilt of the eyes. In general the use of separate tilt for each eye will complicate the stereo reconstruction, thus this will not be treated further.

2. Control scheme and eye movements

   a) The lower level control uses PID controller to drive motors. It outputs the actuator state including position and velocity information.

[1]Chapter modified from article: Xin Wang; Joris van de Weem; Pieter Jonker, "An advanced active vision system imitating human eye movements, "2013 16th International Conference on Advanced Robotics (ICAR), pp.1-6, November, 2013

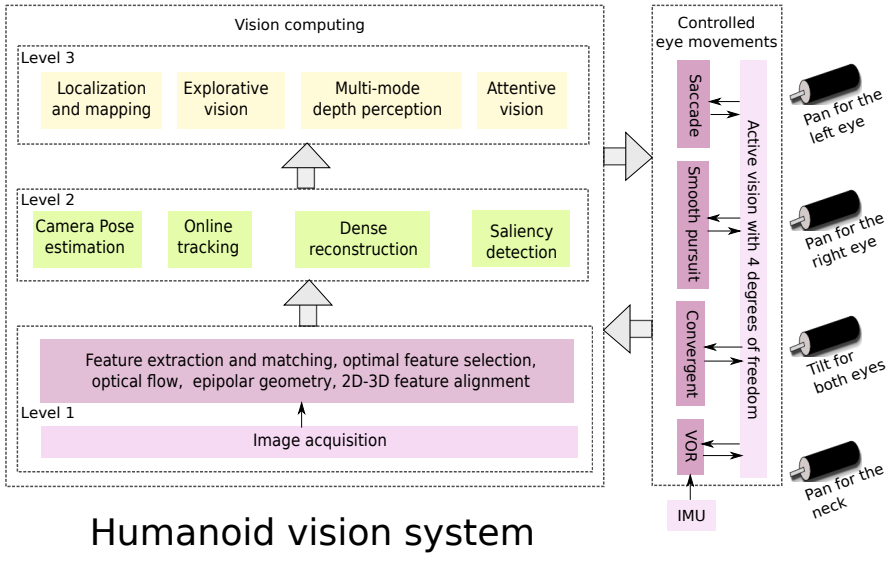


Figure 2.1: Architecture of proposed active vision system

b) The higher level control of saccades are the movements of the eyes when they jump from one fixation point in space to another.

c) The higher level control of smooth-pursuit maintains a fixation point of a target moving at moderate speed on the center of the view.

d) The higher level control of convergence adjusts the both eyes so that the optical axes keep intersecting on the same target while depth varies. It ensures that both eyes fixate on the same point on the target.

e) The higher level control of the vestibulo-ocular reflex (VOR) is the mechanisms to stabilize the image of the target during head movements. An Inertia Measurement Unit (IMU) is used to input the neck pose for stabilization.

3. Lower level image processing

   As soon as an image is acquired, feature extraction and matching, optimal feature selection, optical flow, epipolar geometry, and 2D-3D feature alignment are conducted and prepared for higher level processing.

4. Higher level vision computing

    a) Owing to vergent eyes moving at different angles, intrinsic calibration as well as extrinsic calibration is needed to ensure the accuracy of 3D perception. The camera pose needs to be updated frame by frame.

    b) In order to smoothly pursue an object and learn to recognize the object from different perspectives, three different trackers are used: a color based tracker, an AR marker based tracker for testing, and our proposed robust online tracker.

    c) Dense reconstruction based on stereo matching is used for 3D perception.

    d) Saliency detection is used as input for active vision to fixate on objects of interest.

5. Lower cognitive level vision computing (To be developed)

    a) Location and mapping are required to enable a mobile robot to navigate around different places, still "remembering" where it is.

    b) During navigation, explorative vision is helping the robot to learn from the unknown environment and gain better understanding of its senses, thus building up its long term memory for more complicated tasks.

    c) The attentive active system is for a robot to shift its gaze to the most interesting objects, or most interesting parts on objects. By this pattern, the robot is able to combine bottom-up and top-down information for learning.

    d) Humans utilize multiple cues for depth perception. Depth perception is strongly related to eye movements. Multi-mode depth perception is required to perceive environments and objects in 3D for further vision tasks such as object recognition and object grasping.

2.1 Related work on control of active vision system

As discussed in the previous chapter, active vision is a broad concept and covers a wide application area. In our case, we explicitly specify an "active vision" system as the ability to move an image acquisition system in a controlled manner. Active vision systems usually consist of one or more cameras

mounted in such a way that their orientation and imaging parameters (focus, zoom, aperture) can be controlled and adjusted.

The development of active vision platforms has rapidly evolved over the last decades [53, 54, 55, 45]. We divide existing active vision systems into two main categories: one is focusing on the design of a system that explores the cognitive aspect of the human vision system and imitates a human's eye movement; the other is more task orientated and designed for a specific application.

With respect to the first category, advances in hardware for active vision have given rise to high performance systems, in some respects comparable with the human oculomotor system. A Pan-Tilt-Zoom (PTZ) camera is a typical and the simplest active camera, whose foveation can be achieved by zooming [56]. [52] extended previous work to a stereo set-up. However, the changing of focal length will bring difficulties for precise calibration. Many researches used a log-polar map to achieve the similar effect of foveation. [57] gave a review of log-polar imaging for visual perception in robotics, which is not our main concern, since log-polar imaging is very related to foveation while from a hardware point of view, the active perspective involving eye movements does not fully appear. There are also several systems using two cameras per eye [9, 58, 59], i.e. a narrow-angle foveal camera for foveal vision and a wide-angle camera for peripheral vision to mimic the foveated structure of biological vision systems. However, they paid more attention to the vision part and the algorithm design while ignoring the importance of the eye movement together with the head movement. In the late eighties, [8] studied the control of the Harvard Binocular Head. Its control is based on the model of the oculomotor control described by Robinson, with separate subsystems for smooth pursuit and saccadic motion. [60] focused on control of an active vision system which combined foveal vision, smooth tracking and saccades and was also concerned about non-uniform resolution. [61] developed an oculomotor model based on the human eye's anatomical structure and physiological mechanism. However, the experimental results are based on simulation and on a single eye. [62] extended the work of [61] to a binocular control model that integrates smooth pursuit, saccade, vestibulo-ocular reflex (VOR) and optokinetic response (OKR). However, all these methods did not take vergence eye movements into consideration, therefore the binocular aspect is not fully explored. [63] gave a comprehensive comparison study on stereo, vergence, and focus as depth cues for active vision, which was limited to the mathematical models.

Besides the research on humanoid vision systems, other active vision based applications are booming. Many popular applications for active vision are mobile robot applications for various tasks such as object tracking, object recognition, grasping as well as localization. [64] combined foveal and peripheral vision for object recognition and pose estimation. [15] utilized top-down and bottom-up attention to facilitate manipulation, however, its eye-head system is separated from its grasping system. Active vision is also widely used in video surveillance for tracking, especially on PTZ camera systems [65, 56].

The design of active vision systems brings along many difficulties. First of all, for the design of such an active system with a lot of factors need to be taken into consideration e.g. blur and vibration caused by fast motion, illumination changing as well as hardware instability. Besides, the control implementation of a comprehensive humanoid robot eye movement is very difficult to achieve. The more complex a system, the more complicated its control mechanism. Most existing systems do not have real-time performance, which is very crucial to robotics applications. Furthermore, until now, a large part of the human visual system is yet unknown, therefore existing active vision systems are not able to perform as well as a humans' active vision system.

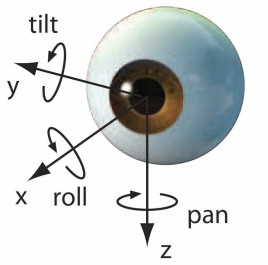
2.2 Requirements for controllers

An active vision system that mimics a human being's visual system while still brings in robustness for mobile robot applications is mandatory for our design. For the design of a human-like eye-head setup that detects and directs visual attention, the understanding of the eye movements of human beings is very important.

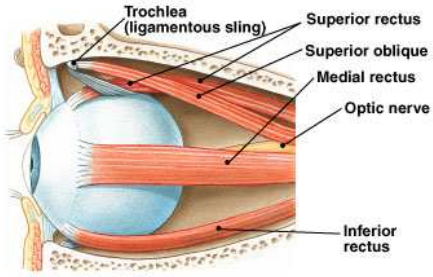
The human eye has three degrees of freedom, which are the rotations around the x-axis, y-axis and z-axis and we call them roll, tilt and pan rotations in analogy of PTZ cameras (Figure 2.2). An oculomotor system consists mainly of the following eye movements [66].

1. Saccade eye movements

Saccades are accurate, high-velocity eye movements used to foveate objects of interest in the field of the fovea, which is the spot of the retina that is responsible for sharp central vision, occupying only 2° of the visual field. The visual stimulus for a saccade is the displacement of the target object. Typically saccades occur with a latency of 200 to 250 msec after an instantaneous displacement of the target [69]. Although most naturally occurring saccades



(a) Degrees of freedom of the eye [67]



(b) Oculomotor muscles [68]

Figure 2.2: Mechanism of the human eye

(~85%) are less then 15° in amplitude, they show a remarkably dynamic behavior [70].

2. Pursuit eye movements
The smooth pursuit is evoked by the slow movement of a fixated target and has a latency of about 125 ms, which enables us to smoothly track discrete objects of interest moving in our surrounds. The sustained periods of foveal pursuit allow maximal resolution, information gathering, and processing of fine details of a moving object. One of of most typical functions of pursuit eye movements is object tracking.

3. Vestibulo-ocular reflex (VOR) eye movements
Activities such as jogging, walking, playing basketball,... produce perturbations of the head that will lead to blurred retinal images or oscillopsias[2], or both. In order to prevent such disturbances in visual perception and maintain a steady sight, the vestibular-occular eye movement, occurs as a compensatory response to a head movement, and is elicited by the vestibular system. The latency can be up to 100 msec and the peak eye velocity can be as fast as 300°/sec. In general, the eyes counter rotate with respect to the head movement and take place as a smooth movement under continuous feedback control, interrupted by intermittent saccades that recenter the eyes [71, 72].

4. Vergence eye movements
Vergence ensures that both the left and right eyes fixate on the same target;

[2]Oscillopsia is a visual disturbance in which objects in the visual field appear to oscillate. The severity of the effect may range from a mild blurring to rapid and periodic jumping. Source: Wikipedia, http://en.wikipedia.org/wiki/Oscillopsia

in other words, it is to coordinate the images of a target to fall on the fovea of both eyes. To look at an object closer by, the eyes rotate towards each other (convergence), while for an object farther away they rotate away from each other (divergence). The latency is approximately 160 ms and the maximum velocity is about 20°/sec [69] as opposed to the 500°/sec velocity of saccade movements.

Convergence is the simultaneous inward movement of both eyes toward each other, usually in an effort to maintain single binocular vision when viewing an object [73]. Convergence is the process that an eye does to properly focus on an image on the retina.

For humans, active vision is the combination of eye and head movements. Figure 2.3 typically represents the evolution of eye, head and gaze rotation: the gaze is directed towards the visual target as fast as possible by a saccadic eye movement. Subsequently, the head follows the eye direction and the gaze is kept stable by counter rotation of the eye. The counter rotation of the eye is vestibular driven and is such that the gaze does not affected much by the head movement [74, 68].

As a result, our robot vision system design is driven by the following three main parts:

Firstly, visual attention plays an important role when we interact with the environment, allowing us to deal with the complexity of everyday scenes. Similarly, a design of a robot vision system that mimics the human vision system and has the ability to autonomously acquire new knowledge through interaction with the environment is one of our main concerns.

Secondly, the requirements on artificial “seeing" systems are highly dependent on the task and have historically been developed with this in mind [15]. For robocup@home and other service robot applications, tasks such as “follow me”, “fetch me an object” in an unstructured environment are challenging for robots. All these tasks require the robot vision system to explore unknown environments. Attentive vision is very essential to search objects of interest and perception-actions needs to be taken into consideration when manipulating objects.

Thirdly, the goal is to understand how humans sense objects and environments. The paradox we face searching for this understanding is that although we still do not understand perception, perceiving is something that occurs almost effortlessly [76]. By designing a humanoid robot vision system we expect to push forward our knowledge in understanding our own visual system. However, for requirements of designing a mobile robot vision system, just focusing on behaviors of the human visual system is impractical. A robot vision

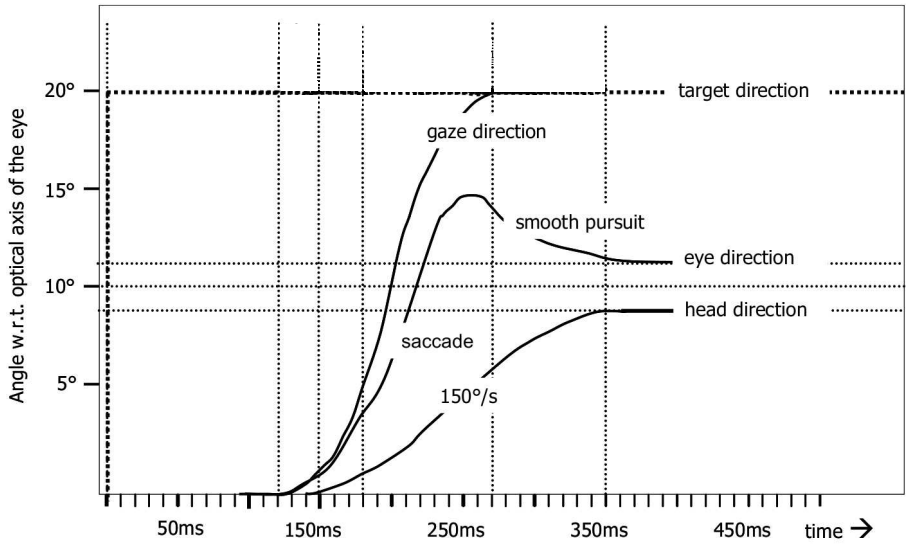


Figure 2.3: Fixing - by both head and eye rotation - the fovea on a virtual target that shifts instantaneously 20° from the optical axis of the eye. (derived from [75])

system is closely connected with other components and is not an isolate one. The performance as well as the design is also restricted by the tasks the robot has to perform and the environment in which the robot resides. One of the most important attributes is robustness, which means that the more complicated a system is, the more complex mechanism needs to be controlled, and the more unreliable performance will result. For mobile robot requirements, we will opt for a simplified design that still preserves backbone functions. Besides, computational speed might sometimes be preferable over accuracy or vice versa, based on different factors.

2.3 Controller design

A system that suits for every kind of tasks and performs well in every kind of conditions is infeasible in reality. As discussed in the previous section, a system that can maximally simulate the humans' eye movements including saccade eye movements, pursuit eye movements, VOR eye movements as well as vergence eye movements is preferred. Besides, taking the mobile robot requirements into consideration, the system should be designed based on tasks

and real world constraints which has properties such as robustness, real-time, and so on. Finally, most existing research employ very precise mechanisms as well as electronics with very high cost, which is unaffordable for daily use or industrial mass production. We prefer household webcams and affordable motors instead, which will lead to more challenges for algorithms and software.

2.3.1 Hardware design

Our hardware system is a combination of actuators and sensors that mimic the human head, eyes and vestibular system and it is composed of the following parts:

- The head can move separately on their pan and tilt axes, each degree of freedom is actuated by a Maxon DC motor Amax-22, in combination with actuator Maxon MR-M, which is controlled by a home made controller board (3Mxl), jointly referred to as "3Mxl Amax-22".
- Each eye can move separately on their pan axes using a Maxon DC motor RE-16 which is controlled by a home made controller board (3Mxl) and jointly referred to as "3Mxl RE-16"(Table 2.2).
- Logitech C905 webcams are used to serve as robot eyes. They have an image resolution of 640×480. Another advantage of this selection is their small size making integration very easy.
- The Xsens MTi inertia measurement unit (IMU) helps to measure the angular velocity of the head to achieve the Vestibulo-ocular reflex (VOR) stabilization.
- A PC with an Intel(R) Core(TM) 2 Duo CPU running on Linux Ubuntu is used to connect the hardware (cameras, IMU and actuators) via USB connections to support the control algorithms.

The mechanical design of our system had three different versions. Figure 2.4(a) is the first version, using only Dynamixel RX-28 with specifications in Table 2.1. Later on we improved this design by changing the Dynamixel to the "3Mxl" design shown in Figure 2.4(b). Eventually, all the motor components were replaced with the "3Mxl" design as shown in Figure 2.5. So, as one can see from Figure 2.5, the eyes are driven by in-house developed motor controllers with higher resolution than the old design using the commercially available Dynamixel. Therefore we can achieve a more precise 3D

Table 2.1: Specifications of the Dynamixel RX-28

Property	Value
Dimension (mm)	35.6x50.6x35.5
Weight (g)	72
Resolution (deg)	0.29
Max Speed (RPM)	59.9 (at 12v)

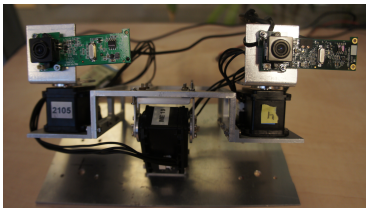
Table 2.2: Speicifications of the 3Mxl RE-16

Property	Value
Dimension (mm)	length:≤ 40.5 diameter:16
Weight (g)	38
Resolution (deg)	0.009
Max Speed (RPM)	264.2 (at 12v)

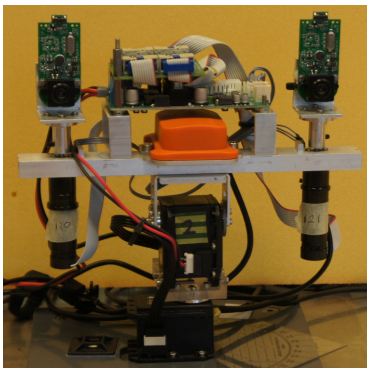
depth perception. Besides, the maximum speed of the new design is higher, which means it can control movements very fast to locate the cameras to a specific position. It also generates a more smooth trajectory than Dynamixels. Furthermore, it weights less and has a smaller size, which is very flexible for eye control. The advantage of the Dynamixel lies in its interface design of connection to other Dynamixel components. It is possible to daisy chain them on a serial line, address them, and provide them with specific commands for its internal motion control processor. This connection protocol was taken over by our own “3Mxl" board. Finally, we added springs to reduce backlash.

2.3.2 Control loop

For humans, a fast eye movement is performed when a salient object appears in the field of view (FOV), followed by a slower head movement to track the salient object until new salient objects appear. Another task is to smoothly pursue an object until tasks such as grasping, object recognition are completed. During these process, if the object is very close to the head, the eyes will converge to achieve a better perception of the object. In Figure 2.6, a gaze control model that describes the combination of eye and head movements due to visual stimuli is illustrated. We have implemented this on a human-like eye-head setup with 1 degree of freedom for the head and 3 degrees of freedom for the eyes (pan for each, tilt for both) and an inertia measurement unit (IMU) that imitates the human vestibular system. In Figure 2.6, the x-y-z



(a) The first version



(b) The second version

Figure 2.4: The previous designs of our active vision system

axes of the world coordinate is depicted. It will be used as convention in our system description.

Most state-of-the-art humanoid robots have 4 or 5 degrees of freedom, which adds one more degree of freedom on pan of the neck. It is worth noting here that we made some simplifications in our design. Based on the common sense that left eye and right eye of humans move up and down together, we choose the design that the tilt rotation for both eyes are coupled with head tilt movement. We do not need any raw rotation of the neck because such rotations will not change the visual data, only its orientation. The anticipatory roll head movements during turning are likely to be utilized to overcome inertial forces that would destabilize balance during turning [77].

Eye servo control

As soon as the image coordinates (x, y) of a target are given to the visual servo control, the eye will be actuated according to the target position information such that the target is kept in the center of the field of view. Saccades have an open loop, or "ballistic", mechanism: the gaze is shifted towards the visual target with high speed and can not be changed during this movement (i.e. no feedback). Smooth pursuit movements are slower and use a feedback loop to constantly adjust the eye velocity and direction to the movements of the object. Since we have no direct knowledge about the 3D postion of the object and as it is an accumulated process, it can be described as a velocity controlled movement with the property that the further the object appears from the center of the FOV, the faster the eyes move in the target direction. The eye

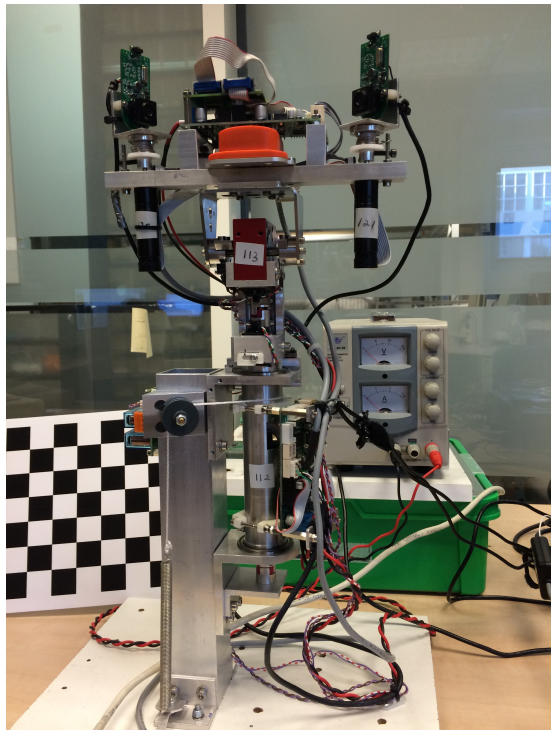


Figure 2.5: The latest design of our active vision system

slows down when the target image gets closer to the center of the FOV. Since there exists differences between the control of saccades and smooth pursuit, we combined position controller for saccades together with velocity controller for smooth pursuit with the maximum angular velocity adjusted to be well above the limit of smooth pursuit (50 deg/s).

Here we use the pinhole camera model to achieve saccade eye movements. Assuming the camera is calibrated, then we have

$$x/f_x = X/Z$$

$$y/f_y = Y/Z$$

Thus we obtain the pan and tilt rotation angles as

$$\omega_X = \arctan(X/Z) = \arctan(x/f_x) \tag{2.1}$$

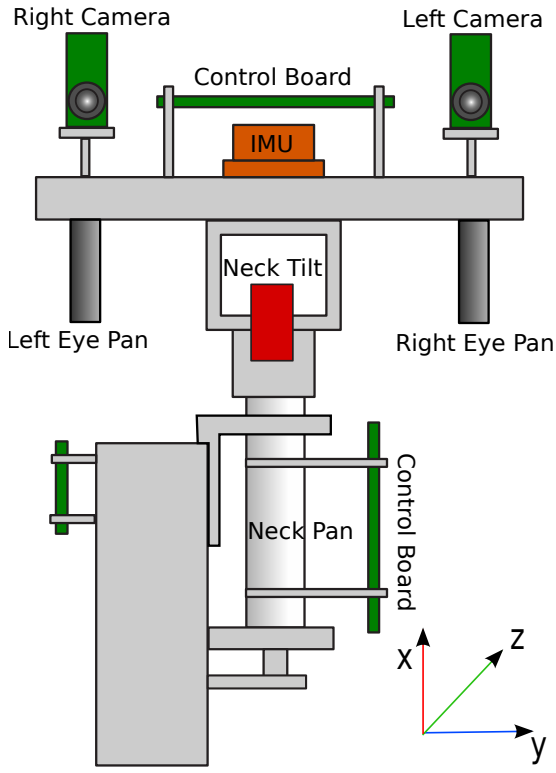


Figure 2.6: Eye-head mechanical model

$$\omega_Y = \arctan(Y/Z) = \arctan(y/f_y) \tag{2.2}$$

where f_x and f_y are the focal length in pixel unit in x and y direction, respectively. (X, Y, Z) are the 3D coordinates of the object of interest. It is worth noting that we have no direct knowledge about the 3D position of the object, and only knowing the focal length and image information can not guarantee precise foveation of the object to be tracked since the rotation center and optical center are not aligned. Besides, velocity control is more smooth than position control, which is more suitable for smooth pursuit. Thus we will use a velocity controller instead of position controller to pursue the object, which is a feedback-loop to constantly adjust the eye velocity and direction to the movement of the object.

For smooth pursuit, image-based visual servoing is adopted. A visual servo controller is needed to actuate the actuators such that the target is centered in the left and right images and the error is defined as the off-center pixel displacement (x_e, y_e), which is

$$\begin{pmatrix} x_e \\ y_e \end{pmatrix} = \begin{pmatrix} x - \frac{FOV_{width}}{2} \\ y - \frac{FOV_{height}}{2} \end{pmatrix}$$

We use a proportional-integral-derivative (PID) controller which is very robust and flexible for deducing the two rotational velocities ω_X and ω_Y in our pan-tilt setup. As such the motors will drive the cameras to move towards the direction that minimizes (x_e, y_e). The further the target is away from the image center, the faster the speed will drive the cameras, and vice versa.

$$\omega_{X,eye} = K_P x_e(t) + K_I \int_0^t x_e(\tau)\, d\tau + K_D \frac{d}{dt} x_e(t) \tag{2.3}$$

$$\omega_{Y,eye} = K_P y_e(t) + K_I \int_0^t y_e(\tau)\, d\tau + K_D \frac{d}{dt} y_e(t) \tag{2.4}$$

Head servo control

Walking, jogging, playing tennis... all these activities produce perturbations of the head that will lead to blurred retinal images and oscillopsias. In order to prevent disturbances in visual perception and maintain a steady view, the vestibular-occular eye movements, occurs as a compensatory response to a head movement, and is provoked by the vestibular system. The latency can be up to 100 msec and the peak eye velocity can be as fast as 300°/sec. In general, the eyes counter rotate with respect to the head movement and take place as a smooth movement under continuous feedback control, interrupted by intermittent saccades that recenter the eyes [71, 72].

When the eyes move towards a visual target, the head follows the eye movements to ensure the same angles of left and right eyes. The head velocity $\omega_{X,head}$ is determined by the angles of both eyes, and is defined as

$$\omega_{X,head} = K_P \varphi_e(t) + K_I \int_0^t \varphi_e(\tau)\, d\tau + K_D \frac{d}{dt} \varphi_e(t) \tag{2.5}$$

$\varphi_e = \varphi_{left} - \varphi_{right}$ is the difference between the current left and right angles for the pan direction.

The vestibulo-ocular reflex (VOR) stabilizes vision in many vertebrates. It integrates inertial and visual information to drive the eyes in the opposite

direction of the head movement and thereby stabilizes the image on the retina [78]. Inertia trackers, such as the Xsens inertial measurement unit (IMU) can measure linear accelerations, the magnetic field and angular velocities. This last property can imitate the vestibulo ocular reflex if the IMU is placed at the rotational axis of the head. These measurements can be combined with the visual servo controller as

$$\begin{pmatrix} \omega^*_{X,eye} \\ \omega^*_{Y,eye} \end{pmatrix} = \begin{pmatrix} \omega_{X,eye} \\ \omega_{Y,eye} \end{pmatrix} - K \begin{pmatrix} \omega_{X,VOR} \\ \omega_{Y,VOR} \end{pmatrix} \tag{2.6}$$

Vestibulo ocular reflex control

Convergence is the simultaneous inward movement of both eyes toward each other, usually in an effort to maintain single binocular vision when viewing an object [73]. We ensure convergence based on visual information and the information of the current angles

$$\omega_{X,eye} = \begin{cases} PID(x_e(t)) & \text{if } |x_e| > thd \\ 0 & \text{if } |x_e| < thd \end{cases} \tag{2.7}$$

$$\omega_{Y,eye} = \begin{cases} PID(y_e(t)) & \text{if } |y_e| > thd \\ 0 & \text{if } |y_e| < thd \end{cases} \tag{2.8}$$

$$\omega_{X,head} = \begin{cases} PID(\varphi_e(t)) & \text{if } \varphi_{left} \neq \varphi_{right} \\ 0 & \text{if } \varphi_{left} = \varphi_{right} \end{cases} \tag{2.9}$$

$PID(u_e(t))$ and $PID(v_e(t))$ refer to Equation 2.3, Equation 2.4 and Equation 2.6. $PID(\varphi_e)$ refers to Equation 2.5.

The convergence can be realized by adjusting the motion of the neck to make sure the left and right eye have the same angle, and both eyes are foveating the object of interest in the center of both fields of views. The whole process is a simultaneous process with eyes and neck working together. thd is a threshold to stop the movements, which is achieve by setting the speed to 0 under velocity control.

As discussed above, the whole control mechanism is depicted in Figure 2.3.2. The eye movements of the system are realized by the teamwork of eyes and neck, which ensures that the robot explores and exploits the unknown environment in a similar way humans do. Besides higher level controllers, there are lower level controllers to drive the motors to reach specified position. The reason why we have two levels of PID controllers is because they run

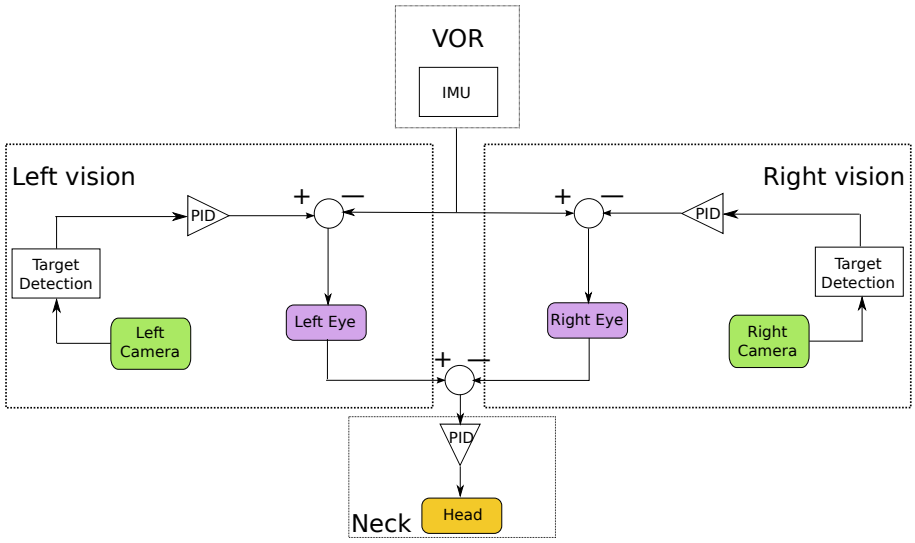


Figure 2.7: Eye-head kinematics model

at different frequency: the PID controllers inside motors run at $1K$ hz and the PID controllers described in Figure run at 25 hz. Running at $1K$ hz is to read and set the speed and position of the motors at a fast rate; running at 25 hz is the longer computational time that is required to process an image.

2.4 Experiments and results

With respect to real time requirements, the software is written in C++ and integrated into the Robotics Operation System (ROS), making the design easily integrable into other robotics developments.

The attended direction depends on the task or purpose of the system. For example, a saliency algorithm can be used to attend object of interest. In other cases, a pre-defined object model can be memorized or manually selected to direct the attention. Most saliency detection algorithms are computational heavy and still experimental. In order to generally suit other tasks, we used top-down visual attention to search for a particular object. Provisionally we use a marker since it can also provide us with precise 3D position and rotation information as well as 2D central image coordinates. This is easy and precise for testing purpose. The marker we used is shown in Figure 2.8.

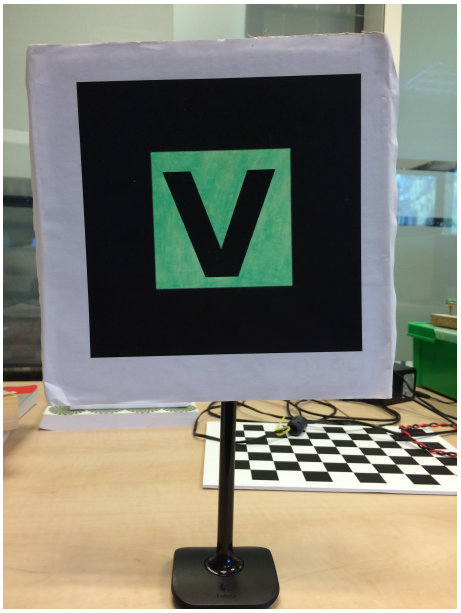


Figure 2.8: Marker used for testing proposed controllers

Table 2.3: PID parameters setting

	left eye	right eye	eye tilt	neck pan
P	0.0024	0.00165	0.007	0.78
I	0.0001	0.0001	0.000	0.000
D	0.0001	0.0001	0.008	0.002

The optimal adjustment of proportional gain (P), integral gain (I) and derivative gain (D) is very crucial for achieving optimal performance of the whole system. After carrying out a number of experiments with different PID parameters to track the marker in a predefined position. Comparing the position curves, we set the PID parameters as in Table 2.3 with no overshoot, less vibration and fast response time. It is worth noting that for different motors, the PID setting is different. It should be tuned carefully based on experiments.

2.4.1 Saccade eye movements with open-loop controller

Figure 2.9 shows the saccade eye movements of the left eye. As shown in this figure, compared with a closed-loop controller that constantly needs im-

age coordinates information as input, it can immediately direct the view towards the object of interest without any feedback. This property ensures a fast saccade eye movement with a very sharp curve reaching the target position within 200*msec* rather than the 1100*msec* in feedback mode. As stated, typically saccades occur with a latency of 200*msec* to 250*msec* after an instantaneous displacement of the target. Our system shows a performance that is comparable with the human vision system for saccade eye movements.

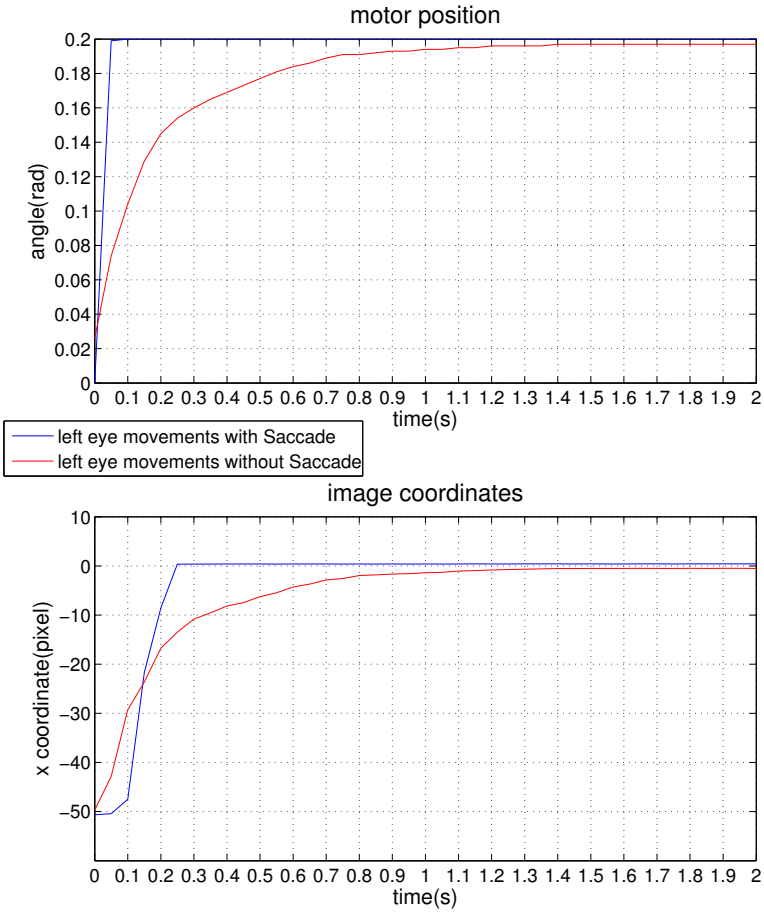


Figure 2.9: Saccade eye movements with open-loop controller

2.4.2 Smooth pursuit eye movements with closed-loop

Figure 2.10 shows the right eye smoothly tracking an object of interest using a velocity based closed-loop controller that uses image information. The basic function is to adjust the velocity according to the image coordinates with respect to the image center. When it is far away from the image center, it will change its velocity to a higher value; when it is close by the image center, it will change its velocity to a lower value. Figure 2.10 shows the behavior that the image coordinate influence the velocity to make sure that the object can be maintained in the center of the view. All movements are very smooth and the eye can constantly track objects of interest without loosing them.

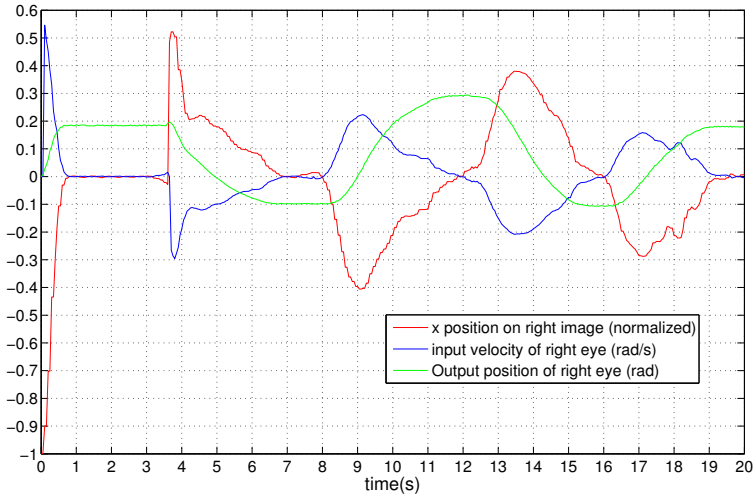


Figure 2.10: Smooth pursuit eye movements

2.4.3 Vergence eye movements

Figure 2.11 shows that during vergent eye movements, the eyes keep on moving until the object of interest is in the center of both views. The two curves in the figure, which represent the object in the left image coordinates and the right image coordinates, converged to 0 position; $(x_e, y_e) = 0$. In other words, after convergence, the object of interest is foveated in the center of both images. As seen in Figures 2.12 and 2.13, with the movement of the neck, the

left and right eye position angles finally adjust to the same angle. This ensures that the perspective distortion for both eyes will be minimal.

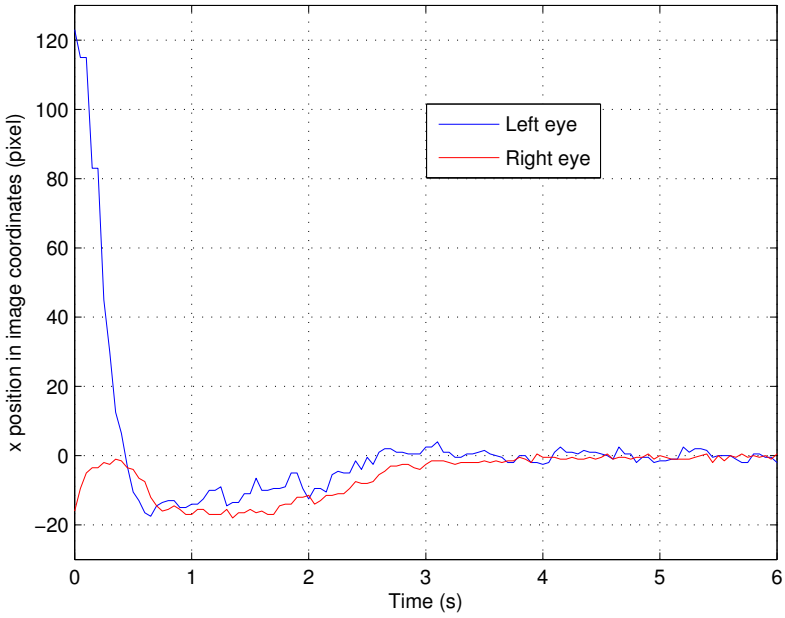


Figure 2.11: Vergence eye movements from image perspective

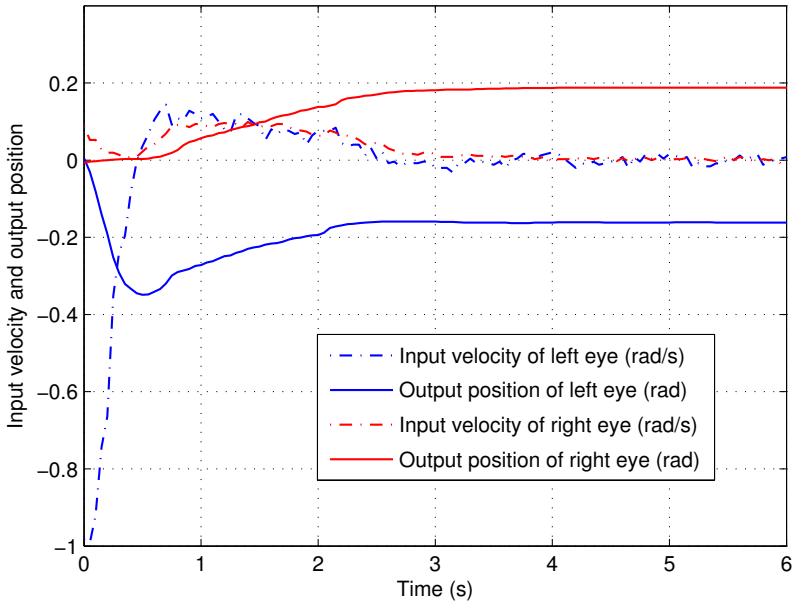


Figure 2.12: Vergence eye movements from velocity and angle perspectives

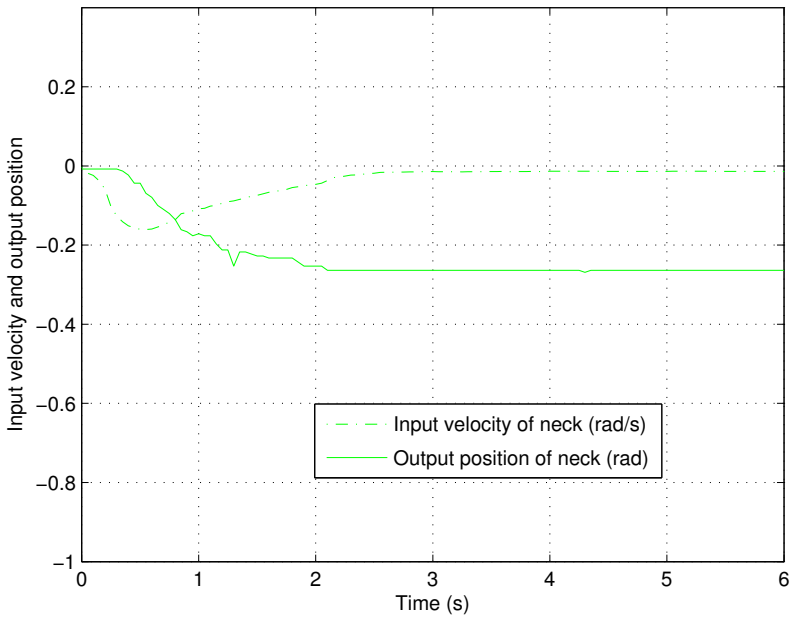


Figure 2.13: Neck movements in vergence eye movements

2.4.4 VOR eye movements

We first move the target to a predefined location, then the eyes direct to the target together with the neck movements. We tested the performance both with VOR and without VOR eye movements. Without VOR eye movements, the eyes first direct to the object of interest and then the neck moves towards this object. In this case, the two eyes will move together with the neck and shift away from the object. VOR eye movements will shift the eyes back during the movement of the neck. As seen in Figures 2.14 and 2.15, the period of reaching the target without VOR eye movements takes about 5s, while for VOR eye movements this is 2.75s. The reason for this is that the eyes are counter rotating in the direction of the neck movement, which makes the total movement towards the object. From image perspective, the VOR eye movement has the ability to stabilize the image, as shown in 2.16 and 2.17. With VOR eye movements, the maximum and minimum x position in image coordinates will decrease, which means that the counter rotation of the eyes will eventually foveate the object in a faster and more stable way.

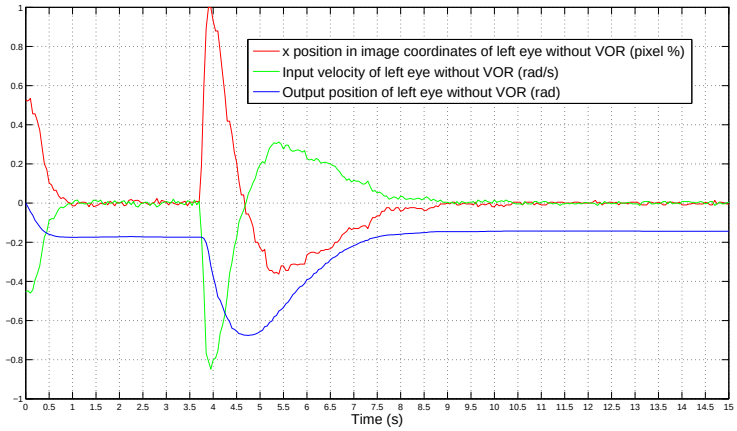


Figure 2.14: Eye movements without VOR

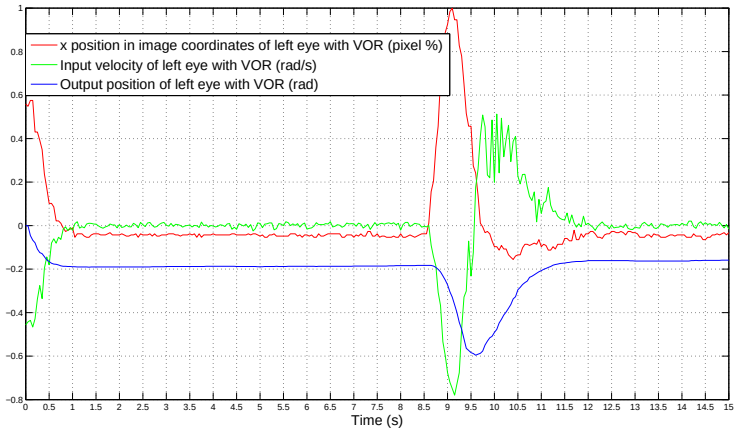


Figure 2.15: Eye movements with VOR

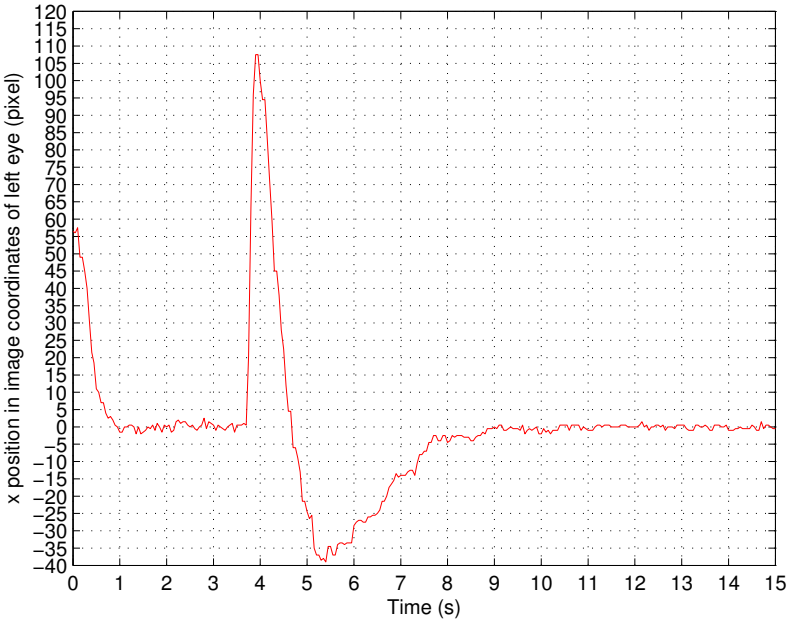


Figure 2.16: x position in image coordinates of the left eye without VOR

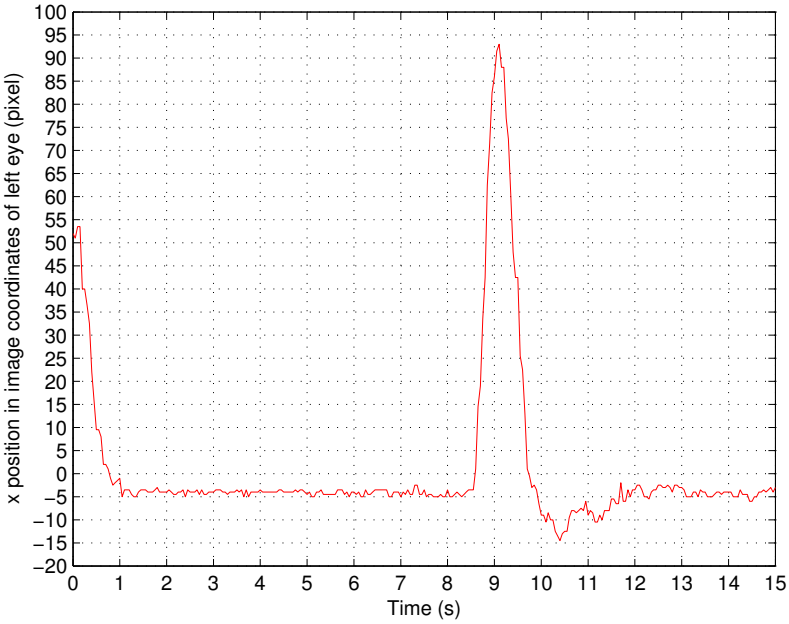


Figure 2.17: x position in image coordinates of the left eye with VOR

2.5 Conclusion and discussion

In this chapter, we showed our design of an advanced vision system which is inspired by the human visual system. It implements different types of eye movements such as saccade eye movements, pursuit eye movements, vestibulo-ocular reflex (VOR) movements, as well as vergent eye movements. By combining each of those movements, it is possible for a humanoid robot to imitate vision-based exploration.

As investigating the entire cognitive learning visual system of humans is our long-term goal, there is still a long way to go with many issues remaining. As opposed to the human visual system, most cameras commercially available provide a uniform resolution, raising the question of whether it is beneficial to implement a fovea in an active vision system [60]. From a biological point of view, foveation can bring the most important information under focus, which is a very interesting subject. Most industrial cameras for industrial inspection have a programmable region of interest (ROI), which can be considered as a special fovea. Usually, the ROI can be read out considerable faster than the entire image, which might have a larger field of view. Finally, the benefits of the collaborative aspect of two eyes should be investigated more in future.

3 Visual primitives representation

[1]Visual perception aims at gathering information about an agent's surrounding, allowing the agent to plan, navigate, and interact with its environment [79]. In real world constraints, we do not have any prior information about the input images and videos. Merely based on pixel value information, visual primitives such as color, shape, features, textures are formed to perceive the surrounding world and objects within it. They provide a bottom-up solution for various applications such as visual tracking [42, 80], simultaneous localization and mapping(SLAM) [81], image stitching [82, 83], stereo correspondences [84], 3D reconstruction [85, 86], object recognition [87], image retrieval [88], etc.

3.1 Related work

A feature is an image pattern which differs from its immediate neighborhood. Within the scope of visual primitives, feature detection and matching are an essential component in many computer vision applications [89]. It is not just a method to select interesting locations in an image, but it is also a powerful image representation tool, allowing for description of objects without the need for image segmentation.

One significant group of visual primitives is the point feature, which receives great attention owing to its distinctiveness. [90] is one of the first publications that emphasizes on the importance of corners and junctions in visual recognition. A wide variety of interest point and corner detectors exists in the literature such as [91, 92, 93]. The majority of point feature detection algorithms works in a way by computing a corner response function

[1]Chapter modified from articles: Xin Wang; Maja Rudinac; Pieter Jonker, "A robust real-time tracking system based on an adaptive selection mechanism for mobile robots, "2012 12th International Conference on Control Automation Robotics & Vision (ICARCV), pp.1065-1070, 5-7 December, 2012

across images which is explained in 3.2. Moravec [94] computed the sum-of-squared-differences (SSD) between a patch around a corner in a template image and patches shifted a small distance in a candidate image. The classic Harris detector [95] was built on this by computing an approximation to the second derivative of the SSD with respect to the shift. Shi and Tomasi [96] proposed a new feature selection criterion called "dissimilarity" and adopted the smallest eigenvalue of a corner response function to select good features. [97] listed various matrix forms of the corner response function. The corner detectors used in these approaches have a major failing, which is that they examine an image at only a single scale. [98] proposed a local Scale Invariant Feature Transform (SIFT), which is formed by computing the gradient at each pixel in a 16×16 window around a detected keypoint. The success of SIFT in object recognition lead to further research such as PCA-SIFT [99], which simplified the SIFT descriptor by utilizing principal component analysis(PCA) to normalize Gradient patches to achieve fast matching and invariant to image deformations. Gradient Location-Orientation Histogram (GLOH) [93] extended SIFT by changing the location grid and using PCA to reduce the size. It outperformed SIFT by a small margin. [100] introduced an affine invariant shape descriptor for Maximally Stable Extremal Regions (MSER). It outperformed SIFT in non-planar scenes inspite of illumination changes. However, above approaches all suffered from a high computation load. From the perspective of real-time requirement, Speeded Up Robust Features (SURF) [101], which was inspired by SIFT, exploited integral images for fast speed. Another high speed corner feature detector, Accelerated Segment Test Feature (FAST) detector [102] adopted a machine learning approach, therefore can achieve real-time performance, with the AGAST detector [103] extending this work for improved performance in both indoor and outdoor environments. Studies have been continuing on feature detection and more and more feature detectors sprung up, such as [104, 105]. Besides, a great amount of research was carried on to improve local point feature detection and matching. Adaptive Non-Maximal Suppression (ANMS) [106] was used to ensure a more uniform spatial distribution among point features. [107] can achieve sub-pixel resolution for detected feature points.

Point features have been widely used for finding correspondences across images, and edge features can provide plentiful semantic information, which is useful for detecting boundary and shape [108]. Several authors [109, 110] reviewed edge detection techniques and compared their performance, in which the Canny detector [111] is the most well known edge detector, which fulfills three performance criteria: good detection, good localization as well as

response to a single edge. Edges and contours are used to describe natural objects, while straight lines are a strong symbol of human influence such as buildings, corridors, etc. Line detection is an indispensable compensation for the representation of the semantic world, in which the Hough Transform [112] is the most used technique for having edges to 'vote' for plausible line parameters. Line detection is also used to estimate vanishing points to reconstruct the geometry of the 3D world [113]. In order to deal with insufficient local information and occlusion, another group of visual primitives, the image segment, is used to segment the image based on color and texture and applied to uniform objects and objects with texture pattern, respectively. [114] presented a comparative study of texture features, with particular emphasis on the applicability to unsupervised image segmentation. The uprising texture feature: local binary pattern (LBP) feature [115, 116], with its discriminative power and computational simplicity, becomes very popular in solving classification problems.

Given a large number of visual primitives approaches, the need for independent performance evaluations also increases. [117] stated out 6 properties to evaluate good features: repeatability, distinctiveness, locality, quantity, accuracy and efficiency. Feature matching is also an important step to evaluate the performance of different approaches by measuring the similarity or dissimilarity among features across images. [92] gave a comprehensive evaluation and compared different approaches. It is worth noting that it is unfair to drive the conclusion that one specific visual primitive is better than the other ones to describe a given object. The importance is the awareness of advantages and disadvantages of different visual primitives. Therefore the suitable visual primitive according to applications can be carefully chosen as well as properties of objects.

3.2 Visual primitives in active vision

The main task of our active vision system is to represent objects, 3D geometry as well as visual odometry. Feature detection and matching is the main technique employed all over the system. One of the most important requirements for a feature point is that it can be differentiated from its neighboring image points. Therefore, by computing how an image patch around a point is different from its neighborhood patches using auto-correlation, a feature point can be detected.

$$\begin{aligned} F(\Delta u) &= \sum_i w(x_i)[I(x_i+\Delta u)-I(x_i)]^2 \\ &\approx \sum_i w(x_i)[I(x_i)+\nabla I(x_i)\Delta u - I(x_i)]^2 \\ &= \Delta u^T A \Delta u \end{aligned} \tag{3.1}$$

where

$$A = w * \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

is called the Harris matrix.

The corner response function that encodes the eigenvalue information has the form

$$R(\lambda_0, \lambda_1) = det(A) - \alpha trace(A) = \lambda_0\lambda_1 - \alpha(\lambda_0+\lambda_1)^2 \tag{3.2}$$

The two 'large' eigenvalues of the corner response function indicate a feature point [95]. Various other ways are also used to find a feature point based on eigenvalues [96, 118].

There are two main approaches to find feature points and their correspondences. The first is to detect features in one image and track them in another image using a local search technique. The other is to independently detect features in both images then match features based on their local appearances. The former is more suitable when images are taken by nearby viewpoints. And the latter is often used in case of large motion and appearance change. For implementation of object detection and tracking in our active vision system, we will apply a motion based tracker for short term tracking and a model based tracker for long term tracking to ensure a robust, precise, yet fast performance. With respect to motion based tracking, we will adopt the first approach and use Shi's corner criteria [96] to detect feature points. Here, if the smallest singular eigenvalue λ_{min} is bigger than the prefixed threshold τ, then the pixel is marked as a feature point.

For long term model based tracking, the texture feature LBP which has the property of being invariant to any monotonic gray level change and is computationally simple is widely used [119]. Given a feature point p, and q its neighborhood point, the LBP is calculated as

$$LBP_{p,q} = \sum_{q=0}^{Q-1} f(I_q - I_p)2^q \tag{3.3}$$

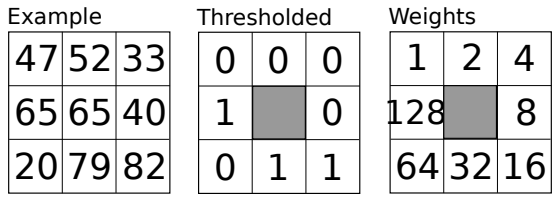


Figure 3.1: An example of LBP feature computation

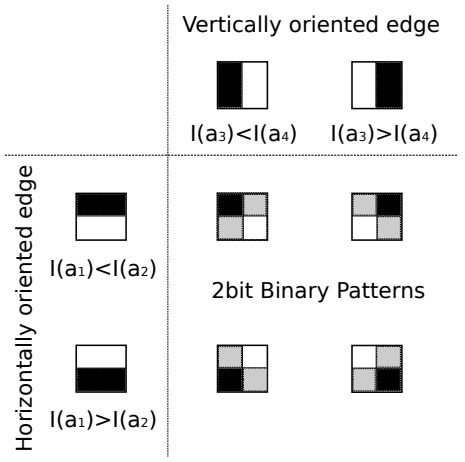


Figure 3.2: Local 2bit Binary Patterns encode local gradient orientation

An example of LBP calculation is shown in Figure 3.1, and its LBP feature value is

$$LBP = 0 + 0 + 0 + 0 + 16 + 32 + 0 + 128 = 176$$

Here we use a simplified LBP version called 2bit Binary Patterns (2bitBP), which differs from standard LBP that encodes 3×3 pixel surrounding and represents a certain area by a distribution of the codes. It encodes the area by a single code and is similar to Haar-like features [120]. The 2bitBP is illustrated in Figure 3.2 and outputs 4 codes in contrast to 256 for standard LBP, which increases resistance to overfitting [121]. The other widely used feature is Haar-like feature, which has become almost standard in tracking by detection research.

Considering the properties of objects to be detected, for objects with texture patterns, the above representations work very well. However, for uniform objects it is a difficult task. Numerous uniform object tracking algorithms

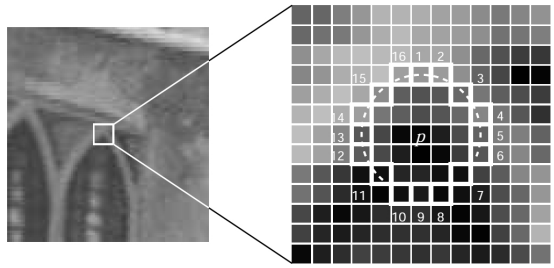


Figure 3.3: FAST and AGAST corner detection criterion

were proposed [122, 123, 124], among which color information is a strong cue. [125] gave a thorough survey about color descriptors in object and scene recognition. Color based visual primitives is preferred in our thesis mainly because of its fast computation time. We opt for using the HSV color space instead of the RGB color space because Hue and Saturation components are able to better cope with illumination variance.

With respect to 3D geometry, the camera pose needs to be estimated. The angles read from encoders can only provide a rough position estimation. For precise estimation we have to consult image information, which feature matching is frequently used to estimate epipolar geometry. We use the low computation cost FAST feature, which outperforms many feature detectors (20 times faster than the Harris detector). Moreover, it has high levels of repeatability under various transformations and different environments. Although it is sensitive to high noise compared to features such as SIFT, its high levels of repeatability and speed makes it a good selection for our purpose. A 12 point segment test corner detection in an image patch as used by Rosten and Drummond [126] is shown in Figure 3.3. The pixels on a discretized circle of 16 pixels surrounding the center pixel are compared to the nucleus p. All points much brighter than $p+\tau$ or much darker than $p-\tau$ is considered to be a corner point.

In order for stereo vision to generate a dense disparity map to fully represent the 3D world and objects within it, each point in the left image needs to find its matching point in the right image. Since after rectification, a 2D searching problem reduces to a 1D searching problem, and the correspondent points in the right image are lined on scan line with respect to points on the left image. A straight forward way is to employ block matching to achieve a fast performance.

3.3 Optimal feature selection algorithm

Extensive research has been conducted in the domain of object tracking. Most of the existing tracking methods focus on using a variation of cues such as color, texture, contour, features, motion as well as depth information to achieve a robust tracking performance. The tracking methods themselves are highly emphasized while the properties of the objects to be tracked are usually not exploited enough. However, there is no universal method that can manage diverse objects. Some trackers work effectively for textured objects because they use texture and feature information, while others based on color information to track uniform objects, often fail textured objects. Therefore, a new trend sprung up in visual primitives by combining several different features to fully describe objects. The state of the art trackers [127, 128, 129] use distinctive features to cope with illumination changes, occlusions as well as cluttered background and do not specifically target uniform objects. [130] utilized multiple cues to overcome disadvantage of using a single feature. However, the advantages of each feature were averaged. [131] combined texture and color information while the complexity of the algorithm made the computation load too high for realtime robotics applications. [132] used an online appearance learning and adaptive algorithm to attain a robust tracking result. However, prior knowledge about the properties of the objects is ignored. Instead of finding a universal tracking algorithm that works for every single object, we employed an adaptive tracking selection mechanism which is driven by the properties of the objects. In order to improve the robustness of the system, we make use of the most distinctive attributes: textureness for textured objects and color for uniform objects, respectively. Thereupon, we propose a novel tracking system that treats different objects with different tracking methods. In this thesis, we first propose a novel adaptive tracking selection mechanism dependent on the properties of the objects. The system will automatically choose the optimal tracking algorithm after examining the textureness of the object. In addition, we propose a robust tracking algorithm for uniform objects based on color information which can cope with real world constraints. In the mean time, we deployed a textured object tracking algorithm which combines the Lucas-Kanade tracker and a model based tracker using the Random Forests classifier. The details for tracking will be explained in Chapter 4.

The initial input is a bounding box x_0 around the object, which encodes the location information. In order to precisely investigate the property of the object, we need to segment the object from the bounding box. Thus we

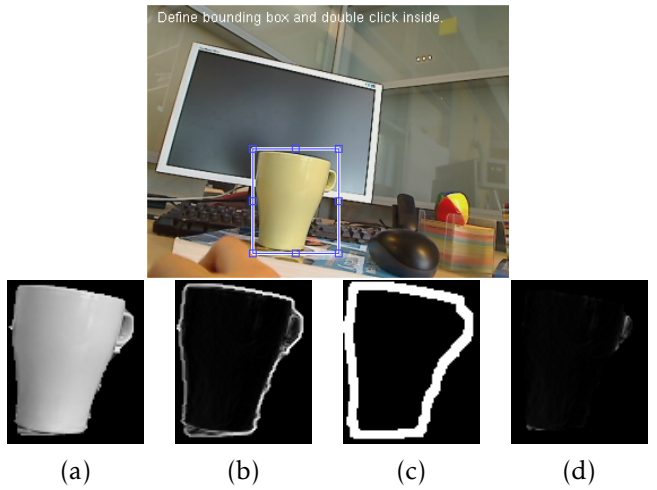


Figure 3.4: Textureness information extraction for a uniform object

opt for the interactive segmentation algorithm GrabCut [133]. The pixels inside the bounding box are treated to be object and the other pixels to be background, therefore foreground and background Gaussian Mixture Models (GMMs) [134] are constructed. The GMMs is a linear combination of Gaussians providing good performance even when the object has complex texture and color. According to the GMMs models, we can label each pixel in the image. The final segmentation can be obtained as a global minimization using graph cuts.

Therefore we segment the object from the bounding box. Both textured objects and uniform objects have a contour, hence the contour is not an essential factor for measuring textureness. For better criteria to determine the properties of the objects, we first exclude the contour information.

Histogram of Oriented Gradients (HOG) features [135] are widely used for pedestrian detection and achieve high detection precision. The HOG features are competent to represent the amount of texture, 'textureness'. We generate HOG features within the object using a cell size of 8x8, after which the properties of objects are deduced by the amount of HOG features. For textureness estimation and computation efficiency, we do not need normalization. Two examples of texture information extraction for a uniform object and a textured object are shown in Figure 3.4 and Figure 3.5, respectively.

By selecting an initial bounding box, we can calculate (a) the segmented object, (b) the encoded texture information, (c) the contour of the object and

Figure 3.5: Textureness information extraction for a textured object

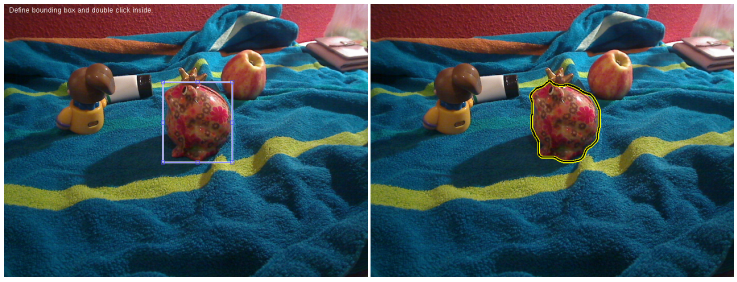
(d) the texture information without contour inference. As seen in Figure 3.4 and Figure 3.5, we can deduce that uniform objects have a fairly uniform distribution of textureness compared to textured objects.

Textureness in our algorithm is expressed by the amount of HOG features. Within a cell image, the HOG features distribution of textured objects and uniform objects is different. We first calculate the number of HOG features with magnitudes above an experience based threshold. Then we divide this number by the total number of HOG features. Based on this we can decide if the object is textured or uniform. Afterwards, we switch to either textured object tracking or uniform object tracking.

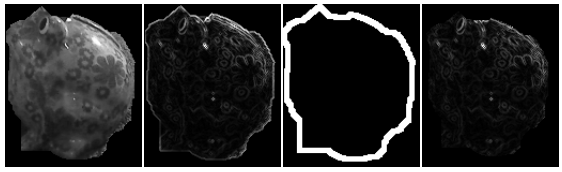
3.4 Experiments and results

We used 25 objects in 1500 frames for determining the optimal threshold. Two examples of system performing on a textured object and a uniform object are shown in Figure 3.6 and Figure 3.7, respectively. (a) is the original image with object of interest selected by a bounding box, (b) and (c) are the segmented object, (d) encodes texture information, (e) is the extracted contour

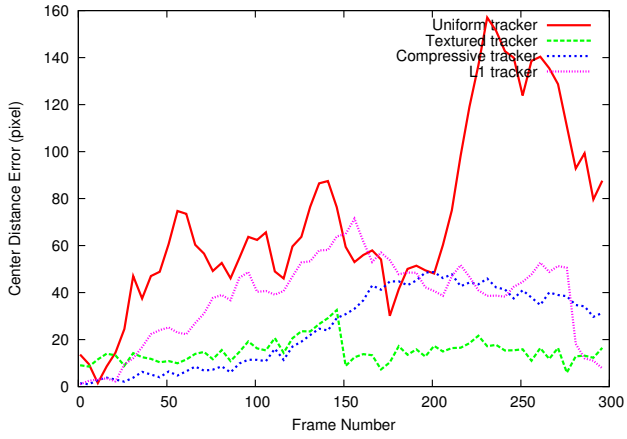
and (f) is the texture of the object without contour inference. The two (f) images show that the uniform object has a low textureness (textureness: 0.1256) compared to the textured object (textureness: 0.4269). The two graphs compare the performance of 4 trackers: uniform tracker, textured tracker [128] (TLD), an improved realtime L1 tracker [129] (L1), and compressive tracker [136] (CT). For the textured object, the uniform tracker performs the worst since it mainly relies on color information, thus can not cope with very textured objects with complex color distribution. The textured tracker achieves very promising results with respect to textured objects. CT fails when the scale changes. For the uniform object, the uniform tracker outperforms all the other trackers with a center distance error always below 15 pixels and a very high stable score. L1 has a growing distance error as more frames are processed. A textured tracker experiences a performance decrease for uniform objects. Consequently, we can deduce that combining a uniform tracker and a textured tracker dependent on the properties of the objects is an effective method to achieve a robust performance.



(a) (b)



(c) (d) (e) (f)



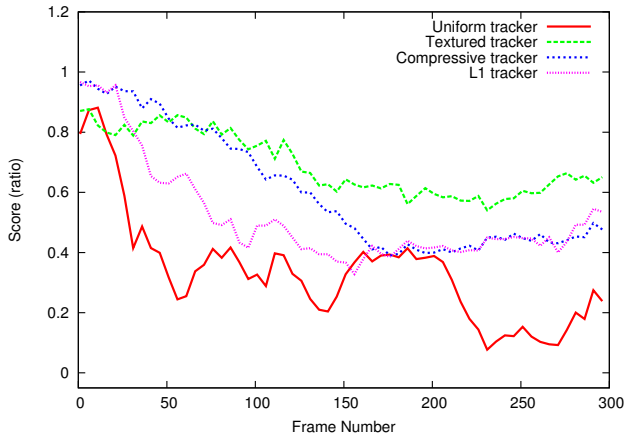


Figure 3.6: System performance on a textured object

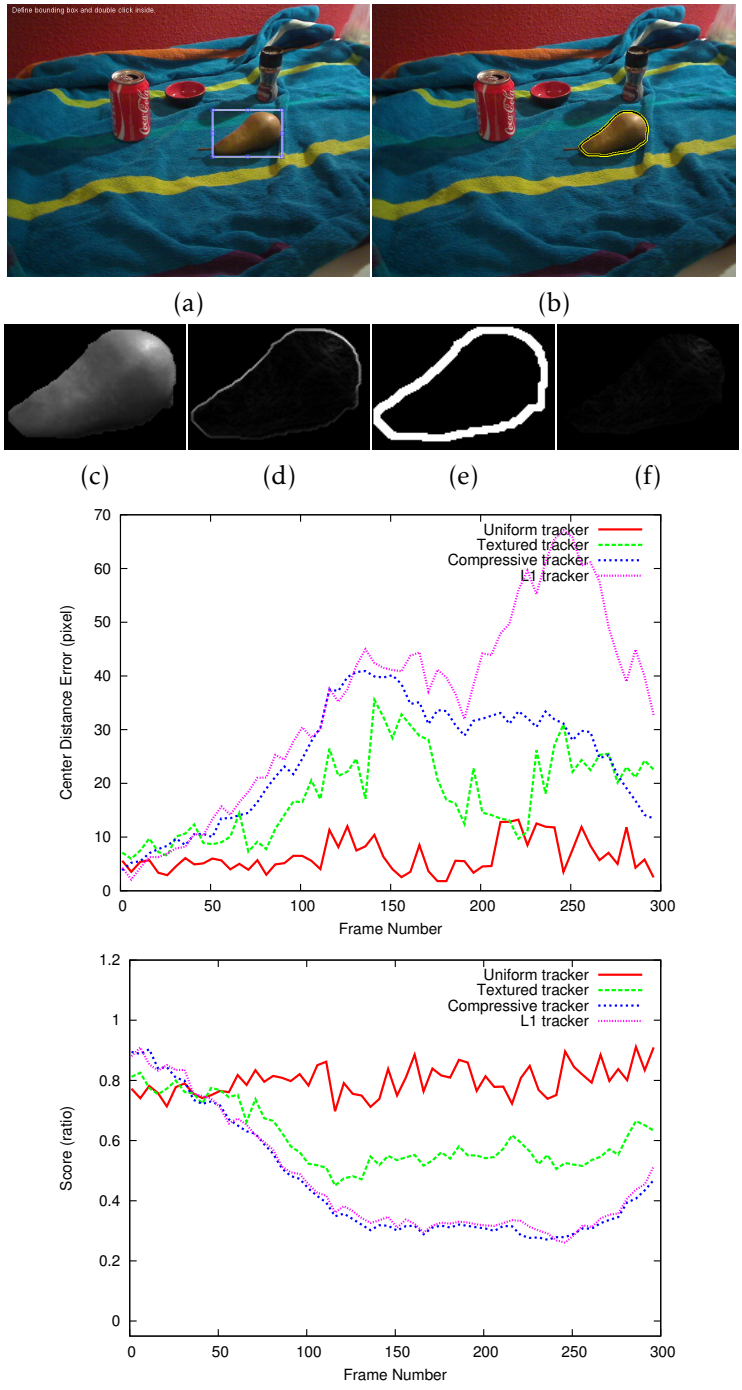


Figure 3.7: System performance on a uniform object

For optimal textureness threshold determination, we used a selection error equation as

$$L(t) = \frac{\sum_{i=1}^{N} e(x_i \leq t) + \sum_{i=1}^{N} e(x_i > t)}{N} \tag{3.4}$$

where

$$e(x_i \leq t) = \begin{cases} 1 & \text{if } x_i \leq t \wedge S_u(i) < S_t(i) \\ 0 & \text{otherwise} \end{cases} \tag{3.5}$$

$$e(x_i > t) = \begin{cases} 1 & \text{if } x_i > t \wedge S_u(i) > S_t(i) \\ 0 & \text{otherwise} \end{cases} \tag{3.6}$$

For each object with its estimated textureness x_i, $S_u(i)$ and $S_t(i)$ are the average scores obtained respectively by the uniform and textured tracker. N is the total number of tested objects. For a given textureness threshold t, an object with textureness below t leads to uniform object tracking, while above t textured object tracking is activated. By comparing the ground truth of the training dataset with the tracking algorithms performance, the selection error can be calculated. The textureness threshold with the lowest selection error value is the optimal threshold. As we can see from Figure 3.8, the optimal textureness threshold is chosen to be 0.2.

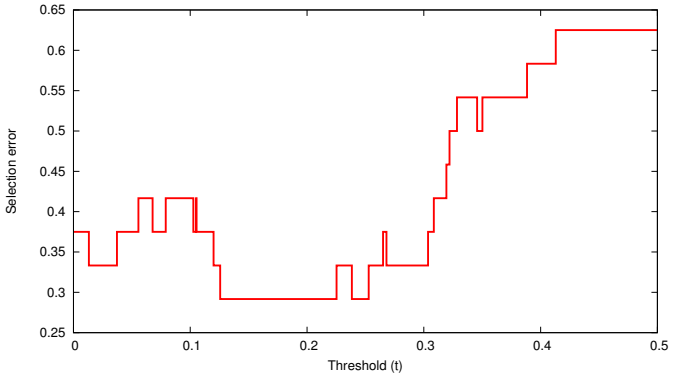


Figure 3.8: Selection error plot

Uniform Samples

Textured Samples

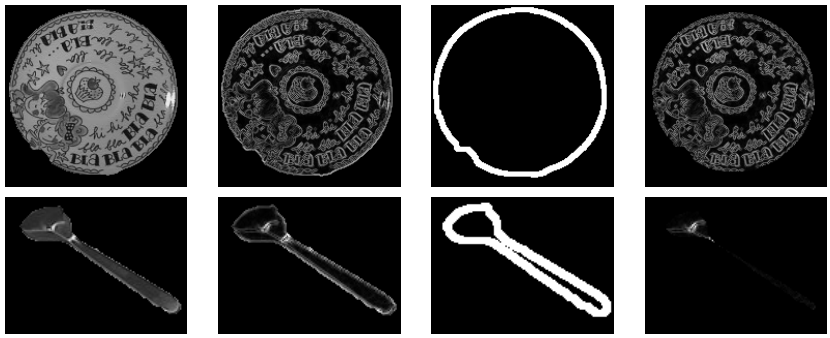
Figure 3.9: Samples for testing tracking selection mechanism

Table 3.1: Performance of the tracking selection mechanism

object vs performance	maximum textureness	minimum textureness	average textureness	precision%
textured	0.5790	0.2291	0.3702	100
uniform	0.1981	0.0307	0.1022	90

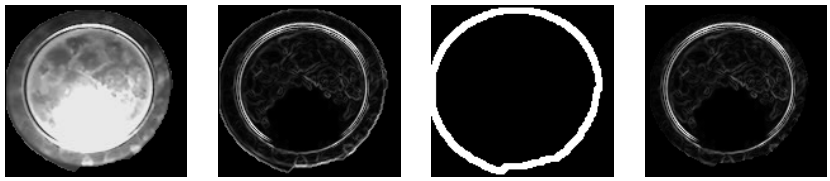
For testing the performance of the tracking selection mechanism, we randomly pick up 40 objects with different properties. 20 of them are textured and 20 of them are uniform. Some samples are shown in Figure 3.9. The performance results are shown in Table 3.1. It shows that the textureness measurement results are very adaptive to different objects carrying different properties. The overall selection precision is 95%, which shows the promising performance of the proposed tracking selection mechanism. The maximum textureness and minimum textureness examples are shown in Figure 3.10. The description of (a)(b)(c)(d) is the same as in Figure 3.4.

We noticed that the quality of the results depends on some properties of the objects we choose. Reflective objects normally give a bad performance because of the reflection of the their surrounding. Therefore often a reflective uniform object is mistakenly considered to be a textured object. A failure case is shown in Figure 3.11.



(a) (b) (c) (d)

Figure 3.10: Maximum (top) and minimum (bottom) textureness



(a) (b) (c) (d)

Figure 3.11: Failure case

3.5 Conclusion and discussion

We proposed a novel adaptive tracking selection mechanism which automatically selects the tracking method dependent on the visual properties of the object. To automatically select the optimal tracking method, we first deploy a GrabCut based algorithm to segment the object and eliminate the effect of background and contour on object property estimation. Then we measure the amount of textureness within the object. Dependent on the measurement result, either the textured object tracking or uniform object tracking method is deployed.

In our future work, we will further exploit this method in service robot applications such as autonomous grasping and object class learning. We plan to add more cues for object property measurement to improve the automatic feature selection method. If computation load allows, the online feature selection method can be applied to achieve robust performance in case of changing of appearance. We will also explore an online color learning method to improve the tracking performance in variable lighting conditions.

4 Object tracking and segmentation

[1]Object tracking is an important task within the field of computer vision [137] and is pertinent in tasks such as motion-based recognition, automated surveillance, video indexing, human-computer interaction, traffic monitoring, vehicle navigation, etc. For any active vision system, object tracking plays an essential role to ensure a robust and high performance system as well. In an unknown environment, a natural ability of humans is to use "active vision" to explore this environment and gain knowledge about this. To efficiently explore an unknown environment a mobile robot needs to be able to track objects of interest and observe them from different perspectives. Another learning ability is to perceive spatial relations among objects and their environment, in which active vision, especially stereo vision, is employed to estimate the geometry. In order to make sure that a stable and robust performance is achieved, long-term video tracking is of great importance for many applications in real world scenarios. A key component for achieving this is the tracker's capability of updating its internal representation of targets (the appearance model) to changing environmental conditions [138]. This is also one of the main concerns for designing a tracking algorithm. In this chapter, we present our work on building up a task-driven humanoid robot that mimics a human's vision system.

Tracker will indicate the location of objects, detailed information such as contour, silhouette, shape will provide us with more comprehensive information about the objects, which will be further used in object recognition, object grasping and so on.

[1]Chapter modified from article: Xin Wang; Maja Rudinac; Pieter Jonker, "Robust Online Segmentation of Unknown Objects for Mobile Robots," VISAPP 2012 - Proceedings of the International Conference on Computer Vision Theory and Applications, Volume 1, page 365-374, Rome, Italy, 24-26 February, 2012

Xin Wang; Pieter Jonker, "An Object-driven Online Segmentation System for Mobile Robots", "MVA2013 IAPR International Conference on Machine Vision Applications", pp.379-382, May, 2013

Still so far, even advanced tracking and segmentation algorithms can not cope with all kind of real-world constraints because of the inherent difficulties of the tasks and its prerequisites.

4.1 Related work

Visual tracking essentially deals with non-stationary data, both the target object and the background, that change over time [139]. In this process, the object or the camera is moving, or both are moving separately.

There is a broad range of applications of object tracking that motivate the interests of researchers worldwide, in which template tracking is the most straightforward approach for tracking. The object is described by a target template (an image patch, a color histogram) and the motion is defined as a transformation that minimizes the mismatch between the target template and the candidate patch. There are two main categories in template based tracking, including feature-based approaches using local features like points, line segments, edges, or regions, and global approaches taking the template as a whole and using the Sum of Squared Differences (SSD) to minimize the difference between a reference template and a region of the image [140]. [141] employed SIFT for object tracking, but this suffers from a high computational load and sensitivity to noise. [142] adopted SURF points for tracking, which provided a better performance than SIFT and allowed faster calculation. The Lucas-Kanade (LK) algorithm [143] was widely used in object tracking by first sampling a grid of pixels on an image patch, and then tracking the motion of pixels in the next frame, thus tracking the image patch as well as object. However, if there are multiple moving objects appearing in the scene, or the object to be tracked disappeared, the motion based tracker will lose its target. [27] extended the meanshift method to the video tracking domain using a template of histogram. It is very efficient for objects with uniform color, while for textured objects, its performance will degenerate. An efficient second-order minimization tracker which is also known the Efficient Second-order Minimization (ESM) tracker, based on minimizing the SSD between a given template and the current image. Trying to avoid Hessian computation in the Newton method, this tracker is able to achieve real-time performance [144].

However, these algorithms usually fail to observe the object motion or have significant drift after some period of time, due to a drastic change in the object's appearance or large lighting variation in its surroundings. Adaptive appearance modeling tracking is a modern tracking method which can model as objects' appearance online, in which adaptive tracking-by-detection ap-

proaches have become particularly popular. This kind of methods maintain a classifier trained online to distinguish the target object from its surrounding background and treats the tracking problem as a detection task applied over time. [145] integrated the Support Vector Machine (SVM) classifier into an optical flow based tracker. [146] used an online boosting method to classify pixels belonging to foreground and background. [147] adopted multiple instance learning (MIL) to handle ambiguously labeled positive and negative data obtained online to reduce visual drift caused by classifier update. [148] proposed a novel learning method (P-N learning) to estimate and correct the errors and achieve real-time performance, known for the tracking-learning-detection (TLD) tracker. [149] used a kernelized structured output SVM (STRUCK) for adaptive tracking and a budgeting mechanism to prevent unbounded growth in the number of support vectors. There are also many other tracking algorithms appearing and showing superior results such as sparse coding based tracking methods [150], real-time compressive sensing tracking (RTCST) [151], etc.

Most tracking by detection algorithm can be seen as semi-supervised algorithms, since the prior knowledge needs to be provided either by offline models in the dataset, or by an initial model that is manually selected. For the localization of unknown objects in a scene, no top-down knowledge can be used. Object detection methods based on point clouds calculated from stereo images [152] provide good results in case of textured objects. However, they fail in the case of objects with uniform color which are also widely present in environments. As a solution to this challenging problem, we therefore consider bottom-up visual-attention methods. The saliency method presented in [153] was used, for instance, in [51] to guide the attention of a robot. An attention method based on local symmetry in the image was proposed in [154] to fixate on objects in the scene. Finally, the method [155] provided fast segmentation of objects based on their saliency. Since it assumes no prior information about the scene and only requires input from a single camera, we will further exploit it in the initial step. Once the initial position of an object is calculated, the robots should be able to navigate around the objects to inspect them from multiple viewpoints. Therefore, very fast and robust object detection and object tracking methods must be applied. We are interested in a mobile robot system that can autonomously explore unknown environments. Therefore, an online detection method that allows automatic segmentation of unknown objects is indispensable. Most of the state of the art methods require user defined object model, which is unusable in our case. The robot has the task to navigate around unknown objects to inspect them from different viewpoints.

For this online segmentation task, existing background subtraction methods [134] fail due to a constant change of the background. Motion based online segmentation [156] is not an option since the objects in the environment are static without any motion information. Thus a model based tracker that can update online is needed. However, histogram based online segmentation such as Camshift [157] can not handle textured objects. Therefore we require an object-driven segmentation method which is able to work in case of complex scenes and objects.

4.2 Major issues in object tracking and segmentation

There are various challenges that object tracking methods need to cope with in real-world constraints.

1. Loss of information caused by the projection of a 3D world onto a 2D image. Most of the visual tracking algorithms are based on 2D knowledge while 3D knowledge can provide more rich information of objects as well as their spatial relations.
2. Noise in images will pose difficulties for object detection using intensity values and features as elements.
3. Complex or multiple object motion will lead to the failure of motion based trackers. Besides, the relative motion between objects and the camera will make the estimation of ego-motion of camera and motion of objects very difficult.
4. Nonrigid or articulated nature of objects do not obey the rigid transformation rule, which gives rise to more challenges for trackers to adapt to their rapid appearance change.
5. A robust tracker needs to cope with partial and full object occlusions and needs to resumes the tracker afterwards.
6. Scene illumination changes will make that most of the color based trackers lose objects, since the color components will change their values very drastically.
7. Cluttered backgrounds will pose challenges for most trackers since the distinctivity between objects and their surrounding is weakened.
8. For objects with complex shapes, the segmentation will be a problem. It is very difficult to segment complete objects from their surroundings.

9. Most applications acquire real-time processing, especially mobile robots, and in constrast, most of the tracking algorithms are very time consuming.

[138] gave a comprehensive survey and performance evaluation of adaptive appearance modeling for video tracking. Four trackers exhibit good performance overall: the discriminative tracker Structured Output Tracking with Kernels (STRUCK) tracker outperformed other approaches, followed by the Tracking-Learning-Detection (TLD) tracker, the Incremental Visual Tracking (IVT) tracker and the Multiple Instance Learning (MIL) tracker. However, TLD achieves 18 frames per second, much faster than 8 frames per second for STRUCK. Since TLD is an adaptive appearance tracking method and has a very high performance, we use it as the baseline for textured object tracking in our system. However, its performance will decrease in case of uniform objects. Therefore we proposed a switch mechanism as described in Chapter 3 to select the proper trackers based on the properties of objects. Moreover, we implemented a vision system that can autonomously perceive objects in unknown environments without any prior knowledge. Furthermore, we proposed a robust online segmentation method which provides refined information about the objects such as shapes and contours instead of only locations. It should be highlighted, that in our setup the camera moves around the static objects, which is in contrast with to other tracking applications where static cameras track or segment moving objects. Therefore our method not only works in general cases such as object tracking with input from a bounding box, but also provides a cognitive approach for robots to develop a self-learning ability.

4.3 System scheme

Figure 4.1 shows a schematic overview of the proposed system. In our previous work [158], a mobile robot maneuvered around the object to track and learn it from different perspectives. Our system was used in the Robocup@home application.

We assume that no initial knowledge on a scene or object is given. In the initial step it is necessary to detect the approximate positions of unknown objects. For initial segmentation, we propose a bottom-up segmentation based on the salient information in the static scene. After the saliency map of the scene is calculated, saliency points in the map are detected and clustered into salient regions, where every region represents a potential unknown object. A

cluster with the most salient points is assumed to be the most dominant object in the scene and its initial model is extracted. Details can be found in Section 4.4.1. In order to examine the property of the object, the system segments the salient object and discards the contour. Then HOG features are generated to estimate the textureness of the object. The amount of HOG features will determine if the object is textured or uniform. If the textureness of the object is below a threshold, the system will switch to a uniform object tracking algorithm. Firstly, a Hue-Saturation joint histogram is used to label the foreground object (background), then a smooth constraint is added to enforce similarity between neighborhood regions. Finally, blob tracking is employed to locate the objects in the frame. In textured object tracking, the dominant object is tracked by a motion based tracker, and the model of the object is rebuilt and constantly updated using Random Forests based classification. By combining the detection results of the motion tracking and the model tracking the location of the object in the new frame is derived. More detailed information is given in Section 4.4.2. In the final step, for every viewpoint and updated object model we do refined object segmentation. The Gaussian Mixture Models (GMMs) are used to create the object model and the background model. Finally, graph cuts is used to obtain the optimal segmentation as is described in Section 4.5. As a result, detailed contour information of the dominant object is extracted.

4.4 Online tracking and segmentation

4.4.1 Saliency detection

In order to be able to learn novel objects in unstructured environments, an initial step is to correctly segment the objects without any prior knowledge about the objects or their background. In our previous research [155], we proposed a method for fast object segmentation based on the salient information in the scene. In the original method [159], saliency was detected using a spectral residual approach on three different color channels, red-green, yellow-blue, and the illumination channel. The saliency map was further calculated as the inverse Fourier transform of each spectral residual, and the results were combined to obtain a more robust saliency map. The bright spots in the saliency map represent points of interest. In order to detect those peaks, we applied the MSER blob detector [160] directly on the saliency map. Once the interesting points were detected, close points were clustered together using a Parzen window estimation, leading to the segmentation of objects in the scene.

The described method was designed for still images and here we propose an extension to process video. Given that the spectral residual process rep-

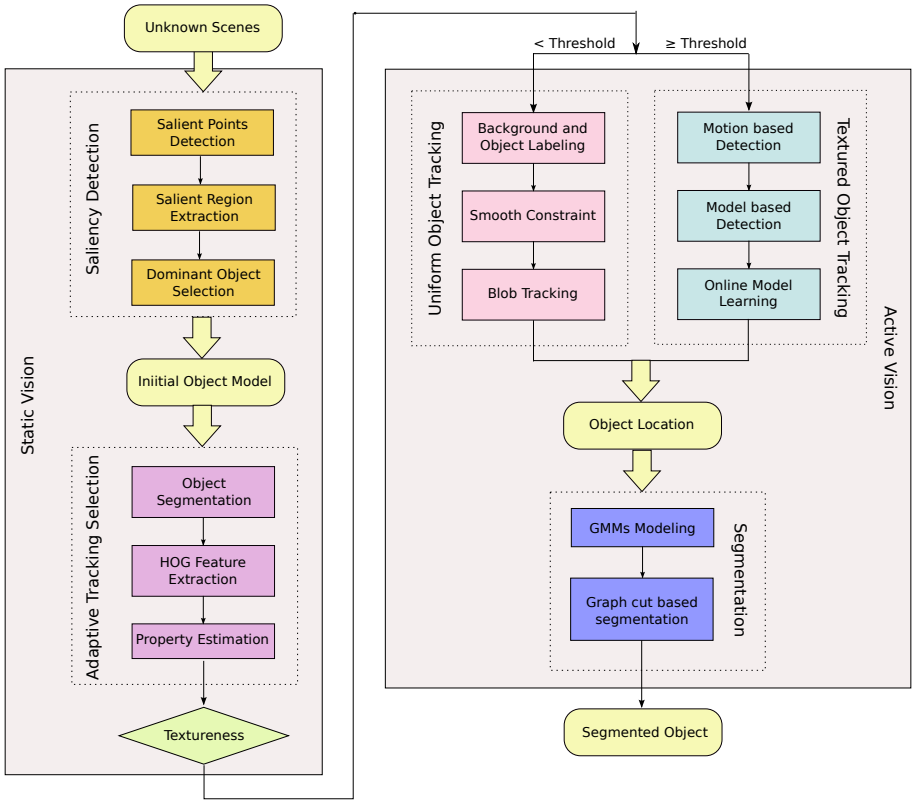
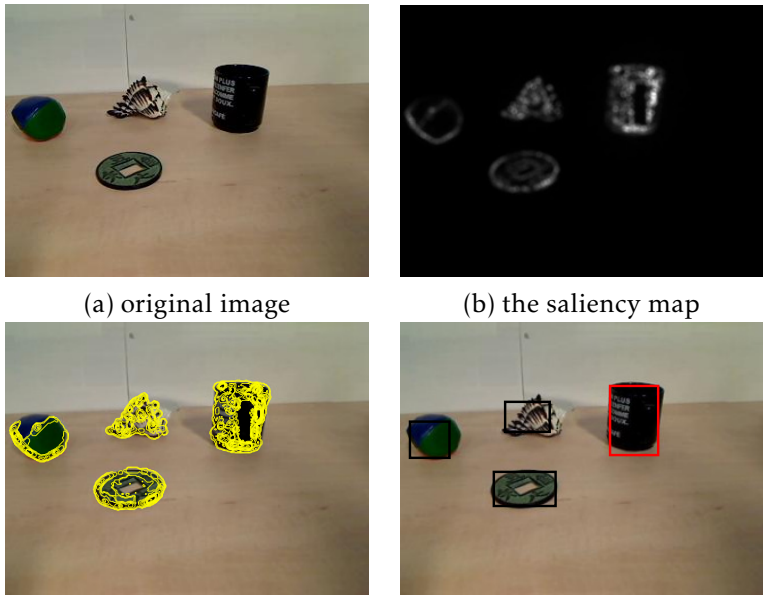


Figure 4.1: Schematic overview of the system

resents the difference between the original scene in Figure 4.2(a) and the scene average, acquiring information about the scene average from successive frames will improve the saliency map. The saliency map is displayed in Figure 4.2(b). Therefore, for each frame we detect MSER points on the saliency map in the standard way and merge the result with those from previous frames to obtain more stable salient points. In our setup we used 5 successive frames. The number of merging frames must be carefully chosen, since too many frames could lead to segmentation larger than the object. To solve this problem, we use an active segmentation method in addition to the initial segmentation.

Once we obtained stable salient points from successive frames, for each detected point the contour describing the MSER region is calculated [160]. The resulting contours can be seen as yellow points in Figure 4.2(c). These con-



(a) original image (b) the saliency map

(c) salient points (d) the dominant object

Figure 4.2: Initial localization of objects using saliency

tours are then clustered leading to the segmentation of objects in the scene. For clustering, we use an adapted Parzen window estimation [161], which automatically fits a probability density function to the contour centers. For each point we calculate the probability $P(x)$ defined by Equation 4.1 where x_i and σ represent the Gaussian kernel center and the kernel size, while S is the number of contour centers and $m = 2$, since every contour center has a two-dimensional coordinate. Subsequently, outlier points that have low probability values and belong to isolated clusters are removed, as defined in Equation 4.2. Finally, the positions of the contour centers and their probability values are clustered using the Mean-shift method [122]. As a result, we find the regions of interest around each object in the scene, see Figure 4.2(d). The cluster with the most salient points represents the dominant region in the scene, the red bounded object in Figure 4.2(d) which will further be segmented.

$$P(x) = \frac{1}{S}\sum_{i=1}^{S}\frac{1}{\sqrt{2\pi^m\sigma^m}}\exp-\frac{\|x-x_i\|^2}{2\sigma^2} \tag{4.1}$$

$$\log(P(x)) < \log(\tfrac{1}{S}\textstyle\sum_{i=1}^{S}P(x_i)) - 3var(\log(\tfrac{1}{S}\textstyle\sum_{i=1}^{S}P(x_i)) \tag{4.2}$$

Table 4.1: Saliency detection algorithm

(1)	Generate a saliency map using spectral residuals: minimize the redundant visual information, calculate residual on three color channels and perform the inverse Fourier transform on every residual, sum up the results from different channels.
(2)	Detecting MSER regions in the saliency map: use Maximally Stable Extremal Regions to find high saliency and represent every region with a calculated contour around all pixels belonging to that region.
(3)	Clustering detected points: calculate the probability using Parzen window esitimation and reject outliers, cluster keypoints and estimated Parzen probabilities using mean shift clustering.
(4)	Recognizing objects in interesting regions: locate the salient objects and select the most salient object as dominant object.

4.4.2 Online tracking

In the Sec. 4.4.1, we proposed a method which segments the unknown objects in the scenes and selects the most dominant one that will further be inspected by a robot from multiple viewpoints. Based on the location of the initial model, the robot should develop a self-learning system by observing objects from different perspectives and perceive its environment without any prior knowledge. One of the necessary steps towards such a system is an object-driven and on-line learning segmentation method. In our application, objects are static while the robot navigates around objects to explore them from different viewpoints. Pure motion based and background modeling based online segmentation methods will fail in this situation. A robust online object segmentation method is proposed to cope with this situation. From the initial position located by saliency, we build up the object model using texture features for textured object and color information for uniform object and update the model frame by frame to efficiently track the object. Then we segment the interested object inferred from the model using GMMs and graph cuts. We will now explain the two steps for tracking and segmentation.

It is worth mentioning that the input can be the model detected by our

saliency method or can be more general, ie. is manually selected. Similarly, our tracking method can be also used in more general applications which is not only limited to cognitive vision research.

Tracking for uniform objects

For uniform object tracking, color is a strong cue. Standard Camshift and blob tracking algorithms are commonly used. For real-time robotics applications, the other benefit of using color information lies in its low computational cost. However, these methods have two strong weaknesses. First, the methods do not work well in different lighting conditions. Second, they just consider the probability distribution of color while ignoring the smoothness between neighborhoods. Therefore, we propose an algorithm to overcome these weaknesses, which is described in Figure 4.3.

Require: Initial frame I_0, Select x_0
  for $t = 1$ to ∞ **do**
    $I_t \rightarrow I_{hsv}$ %convert RGB to HSV
    $calcHist(I_{hsv})$ %calculate the histogram of Hue-Saturation
    for $\forall p \in I_{hsv}$ **do**
      $[l_t(p), c_t(p)] = label(p)$ %label the pixel
    end for
    for $\forall p \in I_{hsv}$ **do**
      $l_{t+1} = l_t$
      $c_{t+1} = c_t$
      for $\forall q \in N(p)$ **do**
        if $calcSimilarity(p,q) \cdot c_t(q) > c_t(p)$ **then**
          $l_{t+1}(p) = l_t(q)$
          $c_{t+1}(p) = calcSimilarity(p,q) \cdot c_t(q)$ %calculate similarity between center pixel and neighborhood
        end if
      end for
    end for
    $x_t = blobtracking(I_{seg})$ %blob tracking
    $t = t + 1$
  end for

Figure 4.3: Uniform objects tracking framework

The HSV color space corresponds closely to the human perception of color [162] and the hue component in HSV color space is insensitive to illumination changing, thus we first convert RGB images to HSV images.

$$H = \begin{cases} cos^{-1}\{\frac{0.5[(R-G)+(R-B)]}{\sqrt{(R-G)^2+(R-B)+(G-B)}}\} & B \leq G \\ 360^\circ - cos^{-1}\{\frac{0.5[(R-G)+(R-B)]}{\sqrt{(R-G)^2+(R-B)+(G-B)}}\} & B > G \end{cases} \tag{4.3}$$

$$S = \frac{max(R,G,B) - min(R,G,B)}{max(R,G,B)} \tag{4.4}$$

$$V = max(R,G,B)/255 \tag{4.5}$$

Instead of using the RGB color space, we compare the back projection based on the hue histogram with the hue-saturation joint histogram, which is shown in Figure 4.5. As one can see, the hue-saturation histogram achieves better segmentation result than only using the hue histogram. After obtaining the probability distribution of the color, we can label the pixel to be either object, background, or undefined. According to this probability value, we can also assign a label confidence to each pixel. The labeled image can only tell the region property, thus we used smoothness constraint to enforce the similarity and refine the segmentation. We are inspired by the interactive segmentation method [163]. For each pixel p, we calculate the similarity between p with its neighborhood pixel q. If they are very similar to each other and the label confidence of q is very high, the neighborhood pixel q will affect the pixel p and p will have the same label as q. The label confidence of q will also change accordingly which is shown in Figure 4.4.

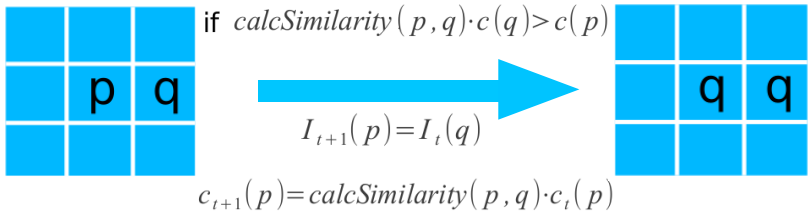
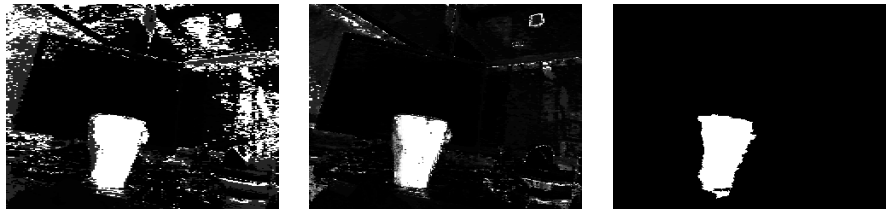


Figure 4.4: Smooth constraint

After iterations until no change occurs, we obtain the segmented image of the object and background. According to Figure 4.5 we conclude that the segmentation performance after smooth constraint is better than both the



Hue Hue-Saturation Smooth Constraint

Figure 4.5: Segmented image comparison

hue histogram based segmented image and the hue-saturation joint histogram based segmentation image, with less noise in the background and the object information enforced. It proves that by using a smooth constraint, the inter connections between pixels from the object are refined while the exterior connections between object pixels and background pixels are weakened. Since we have the refined segmented image, we use blob tracking to obtain the bounding box in the new frame.

Tracking for textured objects

With respect to the task of observing objects from different viewpoints, we need to online build up a training data set to model the object from initial object information and update the model so that it can adapt to the constant change in object appearance. Both methods [128] and [164] for adaptive online tracking use Local Binary Pattern (LBP) variants to represent the texture of the object. The LBP features are randomly distributed on an image patch, thus the spatial information among the features is kept. Then the image patches are used to train a Random Forests classifier. Therefore the object tracking problem turns into a foreground and background classification problem. The drawback of [164] lies in that it needs to offline generate an affine transformation training data set from the original image to build up the tracking model. [128] goes a step further and just requires a user defined bounding box around the object and further updates the model online. However, they do not provide any detail on the object shape. In our system we propose a fully automatic system that utilizes a uniform tracking method (4.4.2), a baseline method [128] for textured object and a refined online segmentation method which also provides shape information.

Assuming that we have an object model M that contains a variety of model elements $(m_1, m_2, \ldots, m_N)$, each m_i uses a group of features $(f_{i1}, f_{i2}, \ldots, f_{iK})$ to

encode the different appearances of the object. The combination of model elements can provide a more comprehensive and robust description of the object than a single model element. Using probability theory we deduce the probability of features based on a given object model element $P(f_{i1},\ldots,f_{ik}|o_i), i = 0,1,\ldots,N$.

Given a potential candidate C, we use

$$P(C) = \prod_{i}^{N} P(c_i|f_{i1},\ldots,f_{iK}) \tag{4.6}$$

to denote the classification of C based on features.

According to the Bayes' Theorem

$$P(c_i|f_{i1},\ldots,f_{iK}) = \frac{P(f_{i1},\ldots,f_{iK}|c_i)P(c_i)}{P(f_{i1},\ldots,f_{iK})} \tag{4.7}$$

We assume the uniform prior $P(c_i)$ and the denominator to be the normalization constant to ensure that the sum of probabilities is one.

Then Equation 4.6 transforms into

$$P(C) \propto \prod_{i}^{N} P(f_{i1},\ldots,f_{iK}|c_i) \tag{4.8}$$

Since we have the criterion to denote the object

$$P(O) = \prod_{i}^{N} P(f_{i1},\ldots,f_{iK}|o_i) \tag{4.9}$$

We can assign C to the class of object or background. Random Forests have the structure of fast and generalized classification, thus we use it to build and update the model. Here, the model elements are represented by trees and the features are nodes of the trees.

First we cover the input salient region with an image patch $x_0 \in X$, where $X = \{x_t, t = 0,1,\ldots,T\}$ depicts the trajectory of the object, in which t is the frame number increased by time. We use LBP as local texture feature descriptor and randomly generate the features on the image patch to maintain the spatial information, therefore we have the first object model and features distribution $P(f_{i1},\ldots,f_{ik}|o_i), i = 0,1,\ldots,N$. We can initialize the construction of the Random Forests which has N trees. Here it is worth noting that the more trees, the more distinctive a group of features can appear. The side effect is that it will lead to an overfitting problem, as well as a heavy computation load.

Second, a grid is generated on the image patch. For each pixel on this grid, its motion in the consequent frame is tracked using the Lucas-Kanade tracker. Thus, the whole displacement $(median(dx), median(dy))$ is calculated and the new location of the image patch and the scale of the object in the new frame are known.

Every new frame is scanned from left to right, from up to down using an image patch with different scales. Within every image patch we use the generated features to compare it with the model. From the viewpoint of Random Forests, the search is carried out for each tree and if the search reaches the leaf the image patch is considered to be a potential object according to the given model element. Finally, we use majority votes from all the trees to decide if it is a confident object. Among all confident objects in the frame, we select the most confident ones and cluster them by distance measurement using normalized cross-correlation.

$$f(x,y) = \frac{\sum_{(i,j)\in W} I_1(i,j) I_2(x+i, y+j)}{\sqrt{\sum_{(i,j)\in W} I_1^2(i,j) \sum_{(i,j)\in W} I_2^2(x+i, y+j)}}$$

Then by combining the image patch location and scale obtained by Lucas-Kanade tracking and the image patch location obtained by detector we derive the image patch of object x_t in the new frame.

Updating the model is an online learning procedure to cope with viewpoint changes. If the image patches detected by the detector are close to the object, they are considered to be a positive data set and add to the branch of the trees, otherwise they will be treated as a negative data set and pruned from the trees. In this way, a robust and "memorized" model is updated.

4.5 Online segmentation

Although the position of the object is known, the information about its contour, edge or shape is still unknown. In our application, the object segmentation will be a cue for further tasks such as the object recognition, scene understanding, object grasping as well as convergent vision, and therefore a detailed contour of the object is necessary. For these reasons we need to further refine the object model and perform detailed segmentation. Most existing segmentation methods need interaction from users [133] and [163]. In order to automatize this process we use the object model from the previous part and in order to decrease the computation time the segmentation is not carried out frame by frame. Object segmentation is performed only in key frames while

for other frames, we use the confident segmentation from the previous frame. The key frame is determined by comparison of the current image patch x_t with the previous image patch x_{t-1}. If the displacement and the scale difference are larger than a specified threshold, the frame t is considered to be a key frame.

In the object modeling part, we combine both texture information of an intensity image and color information. We first apply the hard constraints to label the image and then use soft constraints to optimize the segmentation.

The task of the hard segmentation is to split the scene into an object and a background and we adopt the GMMs for a construction of the object and background models. The GMMs is a linear combination of Gaussians that gives complex densities and better characterization than histogram based methods, thus it provides good performance even when the object has complicated texture and color. For a known image patch x_t calculated by previous steps, we assume that within the image patch the properties of the object are preserved, while all pixels outside the patch have the attributes of background. Based on this, we derive the object GMMs and background GMMs in a following way.

With regards to a pixel $x_p, p = 1, 2, \ldots, P$, the GMMs are defined as

$$P(x_p) = \sum_{k=1}^{K} \pi_k N(x_p|\mu_k, \Sigma_k) \tag{4.10}$$

where the Gaussian density $N(x|\mu_k, \Sigma_k)$ is called one component with mean vector u_k and covariance matrix Σ_k. π_k is the weight. Here the mean vector u_k is composed of three values R, G and B while K is the number of components. K needs to be adapted to the scene, and more textured scenes require higher values of K. Typically $K = 5$.

Since we have the initial model, we can assign each pixel to each component in object GMMs and background GMMs. Therefore we have the label for all the pixels in the image.

After hard segmentation, we use energy minimization to optimize the segmentation. The energy minimization equation is

$$\begin{aligned} E(L) &= \lambda R(L) + B(L) \\ &= \lambda \sum_{p \in P} R_p(l_p) + \sum_{(p,q) \in N} B_{(p,q)} \cdot \delta(l_p, l_q) \end{aligned} \tag{4.11}$$

where $L = (l_1, \ldots, l_p, \ldots, l_P)$ is the label set for each pixel. $l_p = 1$ represents that p is assigned to object and $l_p = 0$ represents that p is assigned to the background. q is one of neighboring elements of p and $\delta(l_p, l_q)$ is defined as

$$\delta(l_p, l_q) = \begin{cases} 1 & \text{if } l_p \neq l_q \\ 0 & \text{otherwise} \end{cases} \tag{4.12}$$

where

$$R_p(l_p) = -\log P(x_p) \tag{4.13}$$

describes the region property based on GMMs models.

$$B_{(p,q)} = \exp(-\beta \|I_p - I_q\|^2) \tag{4.14}$$

describes the coherence of similarity within a region according to a distance between two pixel. Where I_p is the RGB value for a given pixel. λ is a parameter that relatively balance the region property based on GMMs versus the region property based on similarity.

Segmentation can now be estimated as a global minimization using graph cuts [165]

$$c = \arg\min_{L} E(L) \tag{4.15}$$

Then we have the foreground object and background. It is worth noting that the computation cost of segmentation using graph cuts will be a challenge for online applications. In our case, we confine background to be a region surrounding the image patch instead of using the region of the whole image. By doing this, we lower the computation cost. We also use the output of the segmentation result as a refined input of the online model for more precise tracking.

4.6 Experiments and results

4.6.1 Experimental Setup

In order to test the whole system, we made ground truth data from 50 objects in 88 different scenes with 4400 image frames in 4 different test scenarios. They are the following: a single object placed in the scene with uniform background, multiple objects placed in the scene with uniform background, a single object placed in the scene with textured background and multiple objects placed in the scene with texture background. We used different objects which varied in shape (simple vs complex) and in appearance (uniform color vs textured). It is also worth noticing that all of the experiments were carried out in different illumination conditions with natural light as well as artificial

light. Moreover, we tested our system in difficult cases such as the objects with occlusion, as well as similar objects appearing in the same scene.

Here we also need to emphasize that in most state of the art online segmentation methods, the cameras are fixed to capture the motion of the objects in the scene. In contrast, in our experiments the objects are static and the camera moves around the object, which is a more challenging case. There are two types of such active vision setups, one where the camera moves around the objects to "see" them from different viewpoints, and the other where the camera moves to keep the objects in the center of the view, so called foveated vision system. We performed experiments using both setups.

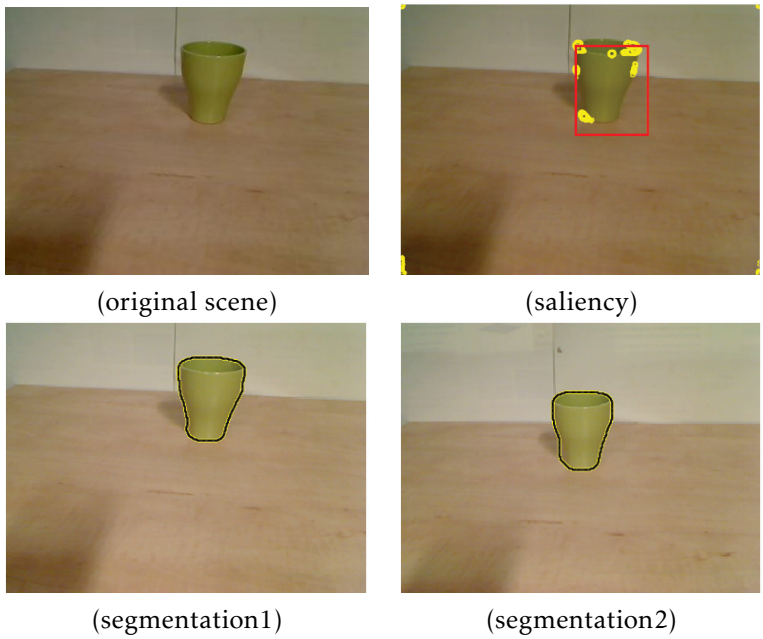
The input from saliency detection will influence how the object model is built up and updated and on the other hand, the input from the object model will affect the GMMs and the graph-cut based segmentation performance. The three parts are strongly interrelated, and for that reason we present total segmentation results.

4.6.2 Saliency Detection and Online Segmentation Results

Our saliency detection selected 30 dominant objects from different scenes. In order to clearly demonstrate the performed tests, with regard to the types of objects and scenes, and to show the saliency detection and segmentation results, we show a number of figures with both single and multiple objects in the scenes. In each figure, we show the original image, the image after saliency detection, the image after object segmentation and one more example of the object segmentation from a different viewpoint. Figure 4.6 shows a single object with uniform color and simple shape in textureless scene, while Figure 4.7 depicts a single object with uniform color and complex shape in textured scene.

For the same reason, we also show a number of figures of the multiple objects scenes. Figure 4.8 shows the textureless scene with multiple objects and the dominant object with texture and complex shape, while Figure 4.9 shows the textured scene with multiple objects and the dominant object with uniform color and complex shape.

Table 4.2 presents the segmentation performance of a single object placed in an textureless or textured background. The rows represent the different types of objects and the columns the types of scenes. Table 4.3 shows the segmentation performance of multiple objects placed in textureless and texture environment. Rows and columns are defined in the same manner as in Table 4.2. Both tables give the overall performance from all test frames. As



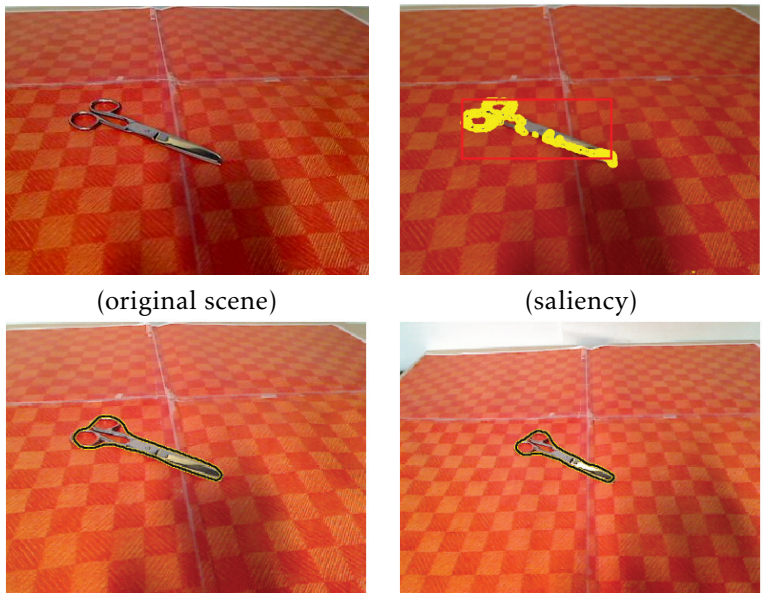
(original scene) (saliency)

(segmentation1) (segmentation2)

Figure 4.6: A single object with uniform color and simple shape in textureless scene

can be seen from the very high precision rates, above 90%, the proposed algorithm gives a very robust segmentation of various types of objects in different scenes. We also come to the conclusion that in most cases, it is easier to segment the objects from textureless than from textured scenes and it is easier to segment the dominant object within single object background than multiple objects background. We can also notice that the multiple object cases show only a slight drop in precision rates. From the perspective of different types of objects, the uniform and simple shape objects make the task of saliency detection nontrivial. On the other hand, the objects with uniform color and complex shape increase the segmentation difficulty. Regarding very textured objects, saliency detection provides good results, but in modeling an over-segmentation can occur, since the number of GMMs components might be low. The case of multiple objects with textured and complex shape is the most difficult one. However, our method gives a very good performance in all aforementioned situations, even in case of large viewpoint changes.

Besides testing the active vision of moving the camera around the objects, we also tested the foveated vision setup. We carried out experiments in 8



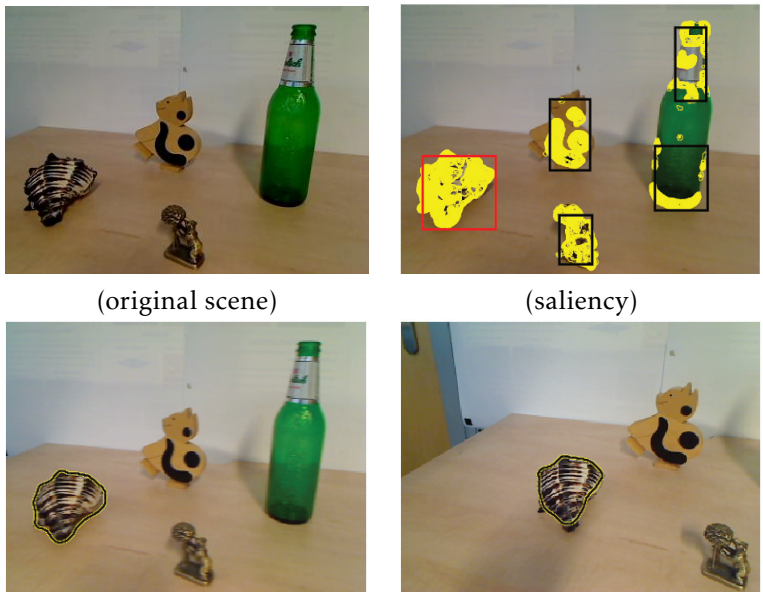
(original scene) (saliency)

(segmentation1) (segmentation2)

Figure 4.7: A single object with uniform color and complex shape in textured scene

Table 4.2: Segmentation results of a single object placed in a textureless and textured scene

objects vs scene	textureless %	textured %
uniform color and simple shape	98.7	96.4
uniform color and complex shape	98.5	94.8
texture and simple shape	98.4	96.4
texture and complex shape	98	92.8
total	98.4	95.1



(original scene) (saliency)

(segmentation1) (segmentation2)

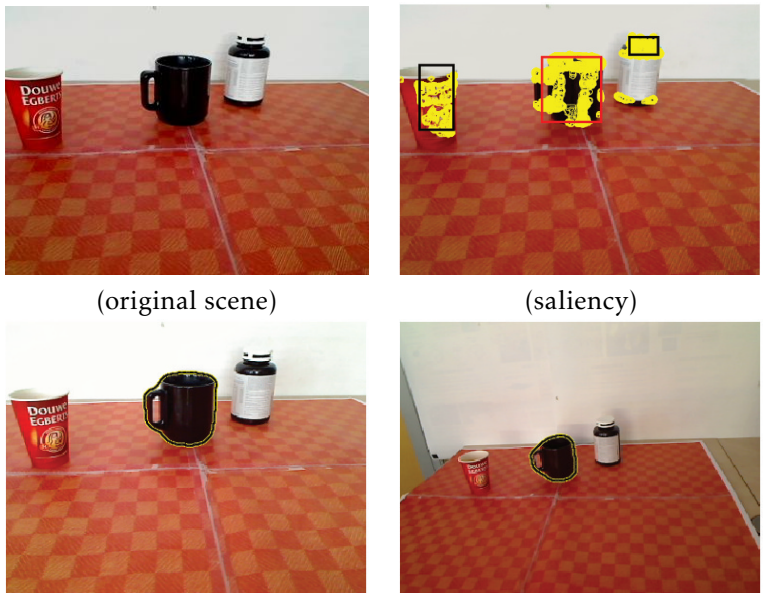
Figure 4.8: Textureless scene with multiple objects and a dominant object with texture and complex shape

Table 4.3: Segmentation results of multiple objects placed in the textureless and textured scene

object vs scene	textureless %	textured %
uniform color and simple shape	94.4	97.3
uniform color and complex shape	98.6	93.4
texture and simple shape	95.6	90.4
texture and complex shape	90.8	86.4
total	94.85	91.875

different scenes with various objects and in total 400 images. The test results show an overall precision rate of 95.5%, which proves the effectiveness of the method on foveated active vision setups as well. One example is shown in Figure 4.10.

To test the robustness of segmentation in more challenging conditions, we performed tests on similar objects appearing in the same scene, occluded objects as well as the motion of objects themselves. The testing result of per-



(original scene) (saliency)

(segmentation1) (segmentation2)

Figure 4.9: Textured scene with multiple objects and a dominant object with uniform color and complex shape

ceiving objects from different viewpoints is shown in Figure 4.11. As we can see from this figure, the algorithm has a good segmentation performance despite the viewpoint changes. In Figure 4.12, regardless of occlusion, the algorithm can correctly extract the dominant object. Even with a similar object occluded in front of the dominant object which is shown in Figure 4.13, the segmentation result is still good. And Figure 4.14 proves that the motion of the dominant object does not affect the performance.

4.6.3 Failed cases

During testing, we observed different situations that were difficult to cope with and those reduced the overall performance rate. We noticed that the segmentation results depend on the property of the object we choose. Transparent and reflective objects normally give a bad performance, as shown in Figure 4.15(a) and 4.15(b). The saliency detection will also affect the online segmentation results if the selected salient region only detects a part of the object, which can happen in the case of multiple object scenarios containing both

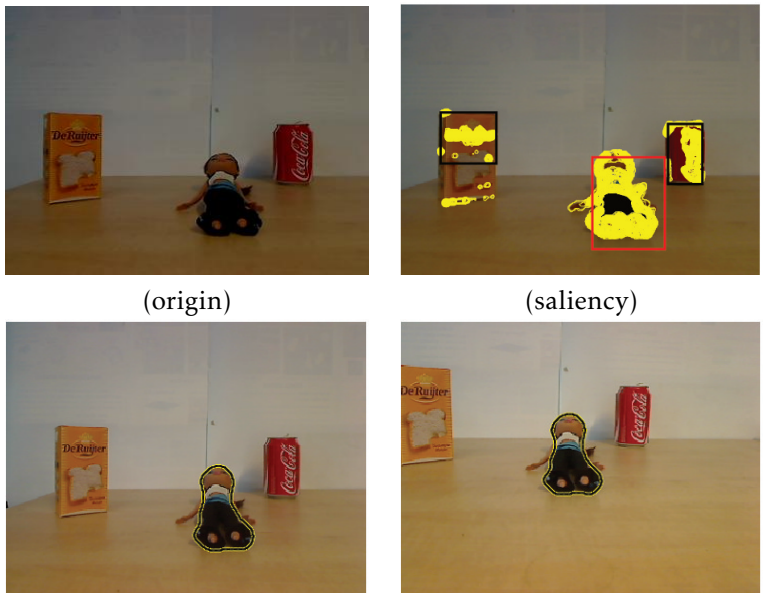


Figure 4.10: Online segmentation results on foveated vision setup

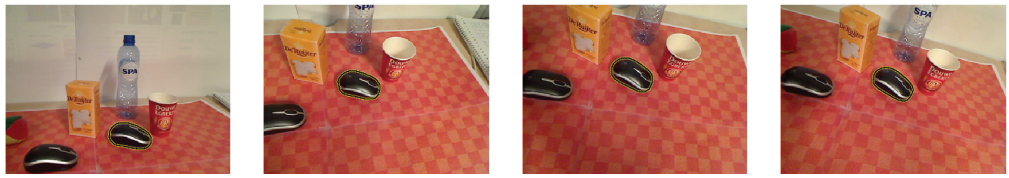


Figure 4.11: Online segmentation results with viewpoint changes

uniform color and textured objects or if the objects are too close to each other. Another problem that rises, is in the case of very textured objects, the selected number of GMMs components might not be sufficient to efficiently segment the object. The failed case is shown in Figure 4.15(c). One way to solve this problem is to adaptively select the number of GMMs components according to this measure. We will investigate this solution in our future work. Finally, if the color or texture of the object is very similar to the background, it is difficult for the algorithm to extract it. Such example is shown in Figure 4.15(d). Also, sometimes the shadow might become a part of the object.

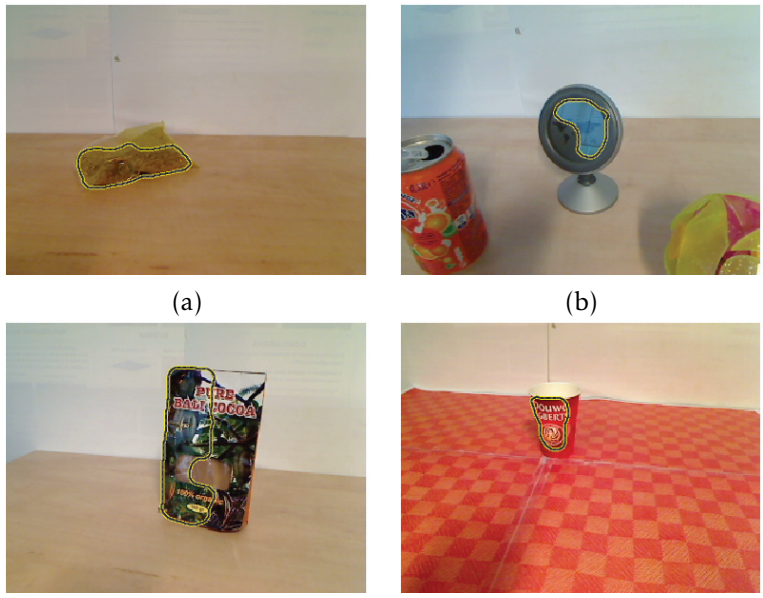
Figure 4.12: Online segmentation results under occlusion

Figure 4.13: Online segmentation results with similar objects appearing in the same scene and occlusion

Figure 4.14: Online segmentation results with motion of the dominant object

4.7 Conclusion and discussion

We introduced a novel method for robust online segmentation of unknown objects. Our method automatically detects unknown objects in the scene based on saliency information, selects the most salient object, tracks the salient object with a movable camera, and finally refines the object model using GMMs and graph cuts. The obtained outputs are the contours of the dominant object in different viewpoints. We tested our system in challenging conditions and the test results with a total segmentation precision above 90% in both textureless and textured scenes. It can efficiently segment both simple and complex shapes as well as objects with uniform color or texture. Our method performs well in spite of large viewpoint changes, illumination changes, occlusion as well as the case of similar object appearing in the same scene. The promising results inspire us to apply our system on mobile robot heads to



(a) (b)

(c) (d)

Figure 4.15: Failed cases

autonomously explore, track and segment unknown objects in unknown environments. The output of our system also provides a strong cue for further tasks such as object recognition, manipulation and learning.

5 Multimodal visual odometry perception for humanoid robot

[1]Perceiving and acting are intertwined to explore and search information while manipulating, listening to, and looking at objects. This multimodal activity involved in exploration is considered as contributing both to the specification of the properties of objects and the perceivers themselves [166]. Young infants display behaviors that are "exploratory" in nature because they appear to be primarily oriented toward bringing sense organs into various relations with objects in the environment. New borns show elements of reaching with arms and hands toward an object moving close to them [167]. By tracking objects moving in their field of view with both eyes and head [168], they are developing a self-learning ability about the environment and objects within it, in which 3D perception is essential to describe spacial relationships and support precise actions. Similarly, while exploring and navigating in 3D space, a mobile robot should be able to locate interesting objects and control the vergence angles of its eyes to observe nearby objects in an object-centered manner. Thus, the objects to be observed are fixated at fovea in the left and right images and depth is estimated for further actions such as object grasping, object manipulation, object recognition, etc. Besides, depth perception also provides information for obstacle avoidance and path planning. This integration of position control, image acquisition and depth perception inaugurate the performance of such a humanoid vision system in real world environments.

[1]Chapter modified from article: Xin Wang; Boris Lenseigne; Pieter Jonker, "Depth from Vergence and Active Calibration for Humanoid Robots, Advanced Concepts for Intelligent Vision Systems Lecture Notes in Computer Science, Springer, Volume 7517, pp.24-35, 2012

"An Advanced Active Vision System with Multimodal Visual Odometry Perception for Humanoid Robots" by Xin Wang and Pieter Jonker submitted to International Journal of Humanoid Robotics (IJHR)

5.1 Multimodal depth perception

5.1.1 Basic concept

Figure 5.1 lists all the sources that animals and humans use to estimate the distances of objects or a distance traveled. As one can see in the figure, there is a variety of sources using visual information as well as without using visual information. Depending on how many eyes are used, the visual information can be divided into two main categories: from monocular and from binocular sources. Figure 5.2 focuses on different depth information used, related to the average depth for humans. Here D_1 and D_2 are the distances of two objects; $2(D_1-D_2)/(D_1+D_2)$ is the ratio of the just-determinable difference in distance between them over their mean distance. $(D_1+D_2)/2$ is the mean distance from the observer. As we can see, the personal space and action space are the main functional spaces for robots to navigate and explore as well as conduct precise actions, in which convergence and binocular stereopsis play an important role.

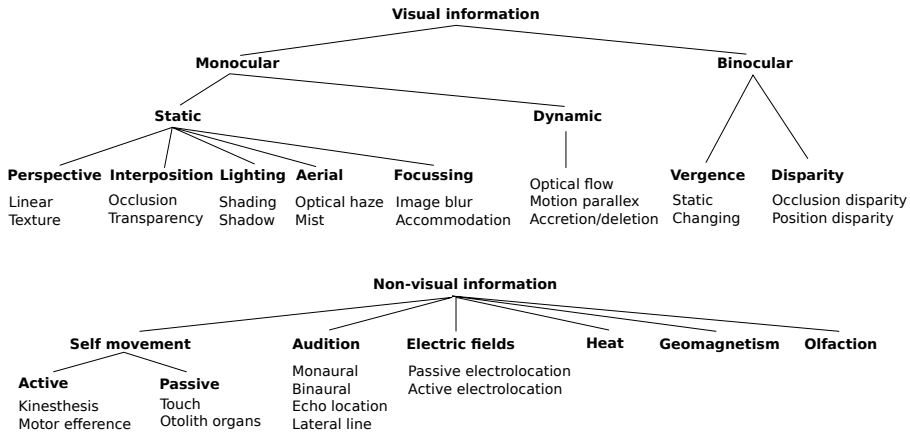


Figure 5.1: Depth perception [169]

With regard to development of a humanoid vision system, we mainly focus on visual information based depth perception methods. Considering action space and personal space, and depth perception for a robot to perform tasks in indoor environments, we concentrate on visual information based binocular cues for depth perception. Thus we employ multimodal depth perception: stereopsis and convergence. The stereopsis mainly works in long distance

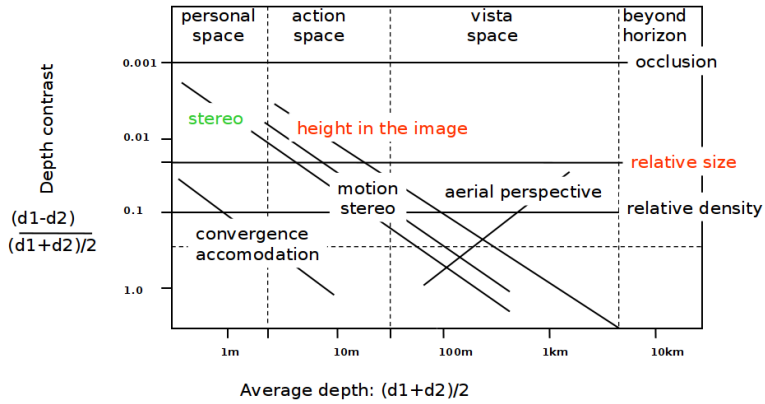


Figure 5.2: Just-discriminable depth thresholds as a function of the log of distance from the observer, from 0.5 to 5000 meters, for nine different sources of information about layout. Such plots were originated by Najata (1981) and are extensively modified and elaborated here; they are plotted with analogy to contrast sensitivity function. Our assumption is that more potent sources of information reflect supra-threshold utility. These functions, in turn, delimit three types of space around the moving observer - personal space, action space and vista space - each served by different sources and with different weights. This array of functions, however, is idealized [170].

conditions and can explore a whole scene and objects inside the scene; In contrast, the convergence functions mainly at nearby object tasks such as object tracking, object grasping as well as object recognition in case that the objects do not appear in the pair of views, because of the fixed baseline in most standard vision systems. Convergent vision has generally been clear for objects up to 10 centimeters from the nose, though this has recently been changing, with vergence accurate to even closer objects [171]. Another concern for choosing these cues is that depth perception is strongly connected with eye movements, where during object moving, camera tracking, depth information can be extracted using stereopsis and convergence. When both eyes are looking straight forward, stereopsis based depth perception is the dominant function; on the contrary, while both eyes are converging while objects are moving close, convergence based depth perception is gradually taking in charge.

5.1.2 Related work

Spatial perception is one of the cognitive skills to evaluate how things are arranged in space and how are their relations in the environment. For humans, spatial perception is mainly based on visual input. Similarly, vision is also one of the primary inputs for spatial perception of humanoid robots. However, the comparison can not go further as the performance of robot vision systems is far below the one of humans. The latter is more capable of adapting itself to a specific task and processing information according to a specific goal based on depth information. Depth estimation plays a crucial role in spatial perception and it has various ways of calculation regarding ranges of distance (personal space, action space, vista space, etc) [170]. Stereopsis is one of the most developed techniques to estimate depth and perceive 3D world.

Stereopsis using binocular disparity to calculate distance has been widely investigated for decades [172]. The regular procedure is stereo calibration, image rectification, stereo matching and depth calculation [173]. However, when it comes to a close distance, the object could drop off the view because of a fixed baseline. Convergence, in the way that it compensates for fixed baseline, directs two eyes towards the object to keep it in the center of both views, which can be seen as a "must-have" feature for bio-inspired vision systems. The first motivation of building a vision system with both cameras can simultaneously look at the same object by panning around, is to mimic the scene exploration mechanism that has been observed in primate or human vision systems. From this cognitive point of view, recent works such as [174] represents the state-of-the-art. This robot head performed scene exploration as well as object recognition and used convergence to build a 2.5D representation of the scene. In some other works [175, 176, 177], the design of space variant sensors and models to make a vision system with a higher resolution concentrated at foveation was presented; [178] generated a log-polar map to obtain foveated images; in [179], convergence was used as a part of a foveated system; [14] designed a system where a low resolution/wide field of view of images was combined with high resolution/narrow field of view of scene details. This kind of foveated systems combined two cameras per eye in order to simulate peripheral and foveated visions. [180] used PTZ camera to mimic the way of foveated vision system; More technical aspects of depth from convergence concerned positioning and reconstruction errors. Pioneer work on this topic can be found in [181]. Several studies [182, 183] investigated the system control with accuracy and smoothness constraints but are far from requirements of an accurate reconstruction system. [184] employed the eye-hand calibration method to calibrate an active stereo system which requires

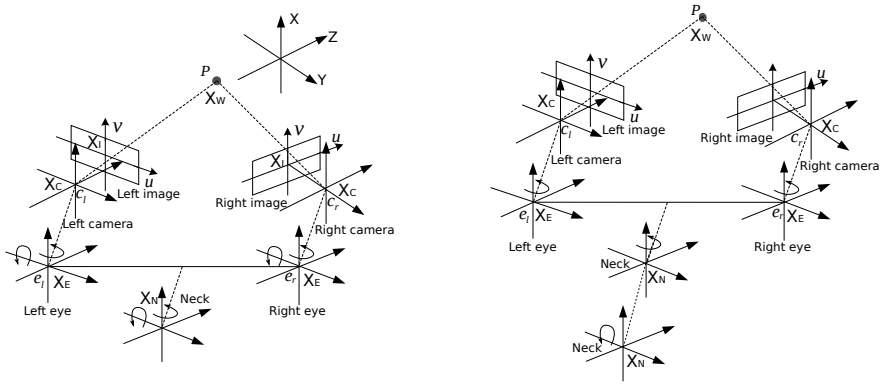
high precision motors. However, most existing robot vision systems merely rely on a single depth estimation method for most of the situations the robots encounter. Considering that a robot uses steropsis to navigate around the environment and convergence to manipulate nearby objects, the use of a specific depth estimation method is not sufficient. Therefore, a framework using 2D image properties, integrating with 3D depth perception based on real world constraints should be made.

We developed a complete system that performs both stereopsis based depth perception as well as convergence based depth perception. A novel multimodal depth mechanism based on encoder position information is proposed to allow for various tasks with different eye movements. Moreover, our system integrates a platform comprising an attention selection mechanism, tracking, motor control, stereopsis and convergence based depth estimation. As a matter of fact, not only it a mimics a humans' visual system, but it is also a valuable alternative to stereopsis when objects are too close to be seen, which is often the case in object manipulation.

5.2 Kinematics of an active head-eye system

In this section, we describe a kinematic model of a standard head-eye system, having 6-DOFs as well as a kinematic model of our head-eye system with 4-DOFs. This can be easily constructed and incorporated into a standard camera model of a stereo vision system. Moreover, the mathematical model based kinematics is also given for further explanation of calibration problems.

Figure 5.3 (a) shows a standard geometric configuration of such a head-eye system. The system has 6-DOFs: both eyes can pan and tilt around the eye axes and the neck can pan and tilt around the neck axis. The assumption here is that the pan and tilt axes intersect with and are orthogonal to each other. Comparably, our head-eye system is shown in Figure 5.3 (b). There are two main differences between this standard head-eye system and our head-eye system. First, as discussed in Chapter 2, based on the common sense that the left and the right eye of humans move up and down together, we choose to design that the tilt rotation for both eyes are coupled with a head tilt movement. In this case, each eye is driven separately by each motor with only pan rotation. Together the eyes are driven by neck motors. Second, the neck motors are composed of a motor for pan rotation and a motor for tilt rotation. For a standard head-eye system setup, the pan and tilt axes intersect with and are orthogonal to each other. However, this is very difficult to realize with regard to mechanics, thus we opt for the design that a pan motor is connected



(a) A standard head-eye system (b) Our head-eye system

Figure 5.3: Head eye kinematics

with a tilt motor through a transformation. It will bring difficulties for the offline calibration algorithms, which will be explained later. Now we derive the mathematical model of our head eye system.

First we introduce some mathematical notations:

$\mathbf{X_W} = \begin{bmatrix} x_w & y_w & z_w \end{bmatrix}^T$ is a point in our 3D world (reference) frame.

$\mathbf{X_N} = \begin{bmatrix} x_n & y_n & z_n \end{bmatrix}^T$ are the 3D coordinates of a neck.

$\mathbf{X_E} = \begin{bmatrix} x_e & y_e & z_e \end{bmatrix}^T$ are the 3D coordinates of an eye, in which we use e_l and e_r to indicate the rotation centers for left and right eye.

$\mathbf{X_C} = \begin{bmatrix} x_c & y_c & z_c \end{bmatrix}^T$ are the 3D coordinates of a camera, in which we use c_l and c_r to indicate the optic centers for left and right camera.

$\mathbf{X_I} = \begin{bmatrix} u & v \end{bmatrix}^T$ are the 2D coordinates of an image. We use the homogeneous transformation H to describe the relationship between two 3D coordinate systems. H has the form

$$H_{4\times4} = \begin{pmatrix} R_{3\times3} & T_{3\times1} \\ 0_{1\times3} & 1_{1\times1} \end{pmatrix} \tag{5.1}$$

where R is the rotation matrix and T is the translation vector. It is worth noting that here we use the homogeneous coordinates of a 3D point.

In contrast with a fixed stereo-vision set-up, each camera is mounted on each eye motor and the eye system together is mounted on a joint on a neck

motor in the head-eye system. Hence there is projection relationships between world frame and neck frame

$$\tilde{\mathbf{X}}_{\mathbf{N}} = H_W^N \tilde{\mathbf{X}}_{\mathbf{W}} \tag{5.2}$$

between neck frame and eye frame

$$\tilde{\mathbf{X}}_{\mathbf{E}} = H_N^E \tilde{\mathbf{X}}_{\mathbf{N}} \tag{5.3}$$

and between eye frame and camera frame

$$\tilde{\mathbf{X}}_{\mathbf{C}} = H_E^C \tilde{\mathbf{X}}_{\mathbf{E}} \tag{5.4}$$

Here the homogeneous coordinates are denoted by $\tilde{\mathbf{X}}$, H_i^j meaning the homogeneous transformation from coordinates i to coordinates j.

Define the initial neck coordinates at a start-up position as $\mathbf{X_N}(\mathbf{0})$. Let $H_{np}(t)$ being the pan rotation transformation of the neck at time t relatively to this start-up position, which is $\tilde{\mathbf{X}}_{\mathbf{NP}}(\mathbf{t}) = H_{np}(t)\tilde{\mathbf{X}}_{\mathbf{N}}(\mathbf{0})$, containing only a rotation around the y-axis.

$$H_{np}(t) = \begin{pmatrix} R_{np}(t) & 0 \\ 0 & 1 \end{pmatrix} \tag{5.5}$$

The tilt rotation transformation of neck at time t is defined as

$$H_{nt}(t) = \begin{pmatrix} R_{nt}(t) & 0 \\ 0 & 1 \end{pmatrix} \tag{5.6}$$

Accordingly, the pan rotation transformation of left eye and right eye are

$$H_{el}(t) = \begin{pmatrix} R_{el}(t) & 0 \\ 0 & 1 \end{pmatrix} \tag{5.7}$$

$$H_{er}(t) = \begin{pmatrix} R_{er}(t) & 0 \\ 0 & 1 \end{pmatrix} \tag{5.8}$$

respectively.

Here, the rotation matrices around the x-axis and y-axis have the form

$$R_x(\varphi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\varphi) & -\sin(\varphi) \\ 0 & \sin(\varphi) & \cos(\varphi) \end{bmatrix} \tag{5.9}$$

$$R_y(\theta) = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \tag{5.10}$$

We can derive the corresponding kinematic model of our head-eye vision system as follows. Equation 5.11 is the extrinsic transformation from world coordinates to left camera coordinates and Equation 5.12 is the extrinsic transformation from world coordinates to right camera coordinates.

$$\tilde{\mathbf{X}}_{\mathbf{Cl}}(\mathbf{t}) = H_{El}^{Cl} H_{el}(t) H_{Nt}^{El} H_{nt}(t) H_{Np}^{Nt} H_{np}(t) H_{W}^{N} \tilde{\mathbf{X}}_{\mathbf{w}} \tag{5.11}$$

$$\tilde{\mathbf{X}}_{\mathbf{Cr}}(\mathbf{t}) = H_{Er}^{Cr} H_{er}(t) H_{Nt}^{Er} H_{nt}(t) H_{Np}^{Nt} H_{np}(t) H_{W}^{N} \tilde{\mathbf{X}}_{\mathbf{w}} \tag{5.12}$$

Above two equations form the kinematic model of our head-eye vision system and describe the extrinsic transformation from world coordinates to camera coordinates known at any time t. While from camera coordinates to image coordinates the intrinsic parameters are needed.

Assuming that intrinsic parameters are measured using standard camera calibration tools, we have K_l and K_r for left and right intrinsic parameters. Therefore we obtain the projection matrices that map the 3D world coordinates to image points on the left and right image, respectively.

$$\lambda_l \tilde{\mathbf{X}}_{\mathbf{Il}}(\mathbf{t}) = K_l [I|0] H_{El}^{Cl} H_{el}(t) H_{Nt}^{El} H_{nt}(t) H_{Np}^{Nt} H_{np}(t) H_{W}^{N} \tilde{\mathbf{X}}_{\mathbf{w}} \tag{5.13}$$

$$\lambda_r \tilde{\mathbf{X}}_{\mathbf{Ir}}(\mathbf{t}) = K_r [I|0] H_{Er}^{Cr} H_{er}(t) H_{Nt}^{Er} H_{nt}(t) H_{Np}^{Nt} H_{np}(t) H_{W}^{N} \tilde{\mathbf{X}}_{\mathbf{w}} \tag{5.14}$$

5.3 Camera calibration

Intrinsic calibration can be done on each of the cameras using the standard MATLAB calibration toolbox or the openCV calibration functions. However, with two moving cameras, the extrinsic parameter can not be obtained directly. To control the camera motion for data acquisition, it is convenient to mount the camera on a positioning device such as a pan-tilt table or a robot. In computer vision, such a head-eye setup greatly facilitates motion stereo, continuous object tracking, and active perception [185].

A principal trade-off is the choice between a parallel baseline system, which provides a simple matching geometry but little overlap between views, and a vergent geometry, which better exploits a common field of view between the cameras at the cost of a more complex correspondence problem [47]. No matter for which configuration, calibration is very critical to ensure a precise pose and 3D estimation.

Traditionally, camera calibration is determined off-line by observing special, well-known reference patterns. This marker-based camera pose estimation has been studied over decades. However, it is restricted to pre-defined set-up, which is not able to work in unknown or unstructed environments. As natural feature based camera pose tracking becomes the new trend, pure image based solutions have achieved great progress. Moreover, with a humanoid concept, any odometry measurement sensor should not be adopted in our design. Thus, image information together with motor information by online processing will provide us with the extrinsic parameters estimation, therefore more accurate visual odometry can be achieved.

It is worthy noting that low-cost sensors are deployed in our system, whose use is unavoidable in most mass-market robotic domains, because of economic constraints: "extensive market analyses show that a complex sensing system for a mobile robot cannot cost more that 10,000 US$, for a consumer-level robot"[186]. As a result, this cheaper solution is going to bring more challenges for camera calibration.

5.3.1 The formulation of the offline calibration problem

Calibrations involves eye calibration and neck calibration. Eye calibration is to estimate the transformation from eye coordinates to camera coordinates and neck calibration is to estimate the transformation from neck coordinates to camera coordinates.

Eye calibration

Assume from time t to time $t+1$, we keep the neck still and only move one eye to drive the camera. Using camera calibration we can establish the relation between camera coordinates and world coordinates at given time t

$$\tilde{\mathbf{X}}_{\mathbf{C}}(\mathbf{t}) = H_E^C H_e(t)\tilde{\mathbf{X}}_{\mathbf{E}}(\mathbf{0}) \tag{5.15}$$

and at given time $t+1$

$$\tilde{\mathbf{X}}_{\mathbf{C}}(\mathbf{t+1}) = H_E^C H_e(t+1)\tilde{\mathbf{X}}_{\mathbf{E}}(\mathbf{0}) \tag{5.16}$$

Then we get the transformation of camera coordinates, which is

$$\tilde{\mathbf{X}}_{\mathbf{C}}(\mathbf{t+1}) = H_E^C H_t^{t+1}(e)[H_E^C]^{-1}\tilde{\mathbf{X}}_{\mathbf{C}}(\mathbf{t}) \tag{5.17}$$

where $H_t^{t+1}(e) = H_e(t{+}1)[H_e(t)]^{-1}$ is transformation of eye coordinates from t to $t+1$ that can be directly obtained from reading the encoder.

With the moving of the camera, we can have camera coordinates with respect to world coordinates from time t to $t+1$ using a standard extrinsic calibration method with a calibration pattern

$$\begin{aligned}\tilde{\mathbf{X}}_{\mathbf{C}}(\mathbf{t}) &= H_W^C(t)\tilde{\mathbf{X}}_{\mathbf{W}} \\ \tilde{\mathbf{X}}_{\mathbf{C}}(\mathbf{t+1}) &= H_W^C(t+1)\tilde{\mathbf{X}}_{\mathbf{W}}\end{aligned} \tag{5.18}$$

Therefore we have

$$\tilde{\mathbf{X}}_{\mathbf{C}}(\mathbf{t+1}) = H_t^{t+1}(c)\tilde{\mathbf{X}}_{\mathbf{C}}(\mathbf{t}) \tag{5.19}$$

where $H_t^{t+1}(c) = H_W^C(t+1)[H_W^C(t)]^{-1}$.
Combining Equation 5.17 and Equation 5.19, we can get

$$H_t^{t+1}(C)H_C^E = H_C^E H_t^{t+1}(e) \tag{5.20}$$

Since the camera is rigidly attached to the eye motor, the relation between eye coordinates and the camera coordinates H_E^C remains unchanged from time t to $t+1$ and this is what we want to estimate. $H_t^{t+1}(C)$ can be estimated by camera calibration and $H_t^{t+1}(e)$ can be obtained from reading motor encoders. Therefore the problem is defined as solving

$$AX = XB \tag{5.21}$$

Neck calibration

For our head-eye system, since the pan and tilt axes of the neck motors do not intersect with and are orthogonal to each other as in a standard setup, there exists a transformation from the pan neck motor to the tilt neck motor. Thus we keep the eye motor static and move the neck pan and neck tilt separately.

We have the following equations

$$\begin{aligned}\tilde{\mathbf{X}}_{\mathbf{Nt}}(\mathbf{t}) &= H_{Np}^{Nt}H_{np}(t)\tilde{\mathbf{X}}_{\mathbf{W}} \\ \tilde{\mathbf{X}}_{\mathbf{Nt}}(\mathbf{t+1}) &= H_{Np}^{Nt}H_{np}(t+1)\tilde{\mathbf{X}}_{\mathbf{W}} \\ \tilde{\mathbf{X}}_{\mathbf{Nt}}(\mathbf{t}) &= H_W^{Nt}(t)\tilde{\mathbf{X}}_{\mathbf{W}} \\ \tilde{\mathbf{X}}_{\mathbf{Nt}}(\mathbf{t+1}) &= H_W^{Nt}(t+1)\tilde{\mathbf{X}}_{\mathbf{W}}\end{aligned} \tag{5.22}$$

Therefore we obtain the same $AX = XB$ equation as

$$H_t^{t+1}(Nt)H_{Np}^{Nt} = H_{Np}^{Nt}H_t^{t+1}(np) \tag{5.23}$$

Similarly, we only move the tilt motor and keep the others static and we have

$$\begin{aligned}
&\tilde{\mathbf{X}}_{\mathbf{E}}(\mathbf{t}) = H_{Nt}^{E}H_{nt}(t)\tilde{\mathbf{X}}_{\mathbf{Np}}(\mathbf{t}) \\
&\tilde{\mathbf{X}}_{\mathbf{E}}(\mathbf{t+1}) = H_{Nt}^{E}H_{nt}(t+1)\tilde{\mathbf{X}}_{\mathbf{Np}}(\mathbf{t}) \\
&\tilde{\mathbf{X}}_{\mathbf{E}}(\mathbf{t}) = H_{W}^{E}(t)\tilde{\mathbf{X}}_{\mathbf{W}} \\
&\tilde{\mathbf{X}}_{\mathbf{E}}(\mathbf{t+1}) = H_{W}^{E}(t+1)\tilde{\mathbf{X}}_{\mathbf{W}}
\end{aligned} \tag{5.24}$$

So we have another $AX = XB$ equation as

$$H_t^{t+1}(E)H_{Nt}^{E} = H_{Nt}^{E}H_t^{t+1}(nt) \tag{5.25}$$

$H_t^{t+1}(Nt)$ and $H_t^{t+1}(E)$ being transformations of neck tilt and eye coordinates from t to $t+1$ with respect to world coordinates which can be calculated using extrinsic calibration. $H_t^{t+1}(np)$ and $H_t^{t+1}(nt)$ are the neck pan and neck tilt motion that can be obtained from reading the encoders. H_{Np}^{Nt} and H_{Nt}^{E} are what we need to know. It is worth noting that H_{Nt}^{E} is the general term used for left and right eyes calibration of H_{Nt}^{El} and H_{Nt}^{Er}. In this case, the extrinsic calibration will be conducted separately on each eye through left image and right image information.

Is AX =XB solvable with our kinematics setup

The head-eye problem is similar to the hand-eye problem in the way that they both have the form of $AX = XB$, where the latter has been researched for decades. There are various ways to solve the problem. Early solutions decoupled the rotational part from the translational one, yielding uncomplex, fast linear solutions for estimating both rotational and translational part, among which the most classic one is TSAI and LENZ [187] using a closed-form solution. Chou and Kamel [188] simplified the formulation introducing quaternions for the estimation of the rotational part using singular value decomposition (SVD). Wang [189] presented an early comparison work which showed that TSAI and LENZ achieves best performance with smallest standard derivation. [185] described the hand-eye geometry in a screw representations. Meanwhile, nonlinear methods were also proposed to increase the estimation accuracy. Since estimating translation based on rotation leads

to rotation estimation errors propagate to the translational part, a lot of approaches that simultaneously estimating rotational and translation part have been proposed [190, 191, 192].

As stated in [187, 185, 193], especially [185] did a thorough research on uniqueness requirements of such a hand-eye solution. All of them derived the same conclusion that for hand-eye problem specifically two movements with nonparallel rotation axes are required to have a unique solution.

[184] extended the hand-eye calibration solution to solve the head-eye problem using a non-linear optimization approach. Here it is worth noting that its kinematic model is very similar to the standard one shown in Figure 5.3 (a), which has tilt and pan movements for both eye axis and neck axis. However, in our kinematic model, this does not hold. For instance, with respect to H_E^C, the transformation from eye coordinates to camera coordinates, there are only pan movements. The underlying reason for this is that in the standard hand-eye calibration setup, H_{gij} [187] is known and H_{cij} can be obtained using extrinsic calibration. In our case H_gij is not known precisely because of off-the-shelf properties.

According to [187] Lemma VII: $skew(P_{gij} + P_{cij})$ is singular and has rank 2. We did tests on simulation and real data to prove the $skew(P_{gij} + P_{cij})$ is a nonsingular matrix and there exists no unique solution to our calibration problem.

5.3.2 Online calibration

[48] proposed a continuous external calibration by estimating epipolar geometry. In the first place, it proves the plausibility of our design of such an active system: we do not need any rotations around the optical axes, because such rotations will not change the visual data, only the rotations [48]. In the second place, a joint tilt of both cameras around the baseline, will not change the nature of the problem and can be ignored. As a result, the pan movements of both eyes will contribute to the 3D perception. In other words, neck pan and tilt movement can be ignored under the condition that we only want to know the 3D position of the object with respect to the robot eye (Here we use the left eye as origin for measuring the depth). However, the method did not take the advantage of known encoder information into consideration. Besides, it assumed that the optical centers are the rotation centers, which is not the case in our setup.

[194] and [195] used motor information. The former used motor information to update the length from rotation center to optical center while the latter

used homography based techniques to derive the relation between motor angle and real angle. However, both models assumed that the optical centers are aligned with the rotation centers and therefore do not work for our setup. With insufficient encoder resolution and backlash, it is necessary to consult image information for more accurate camera position estimation instead of only reading from the encoders. Here we propose an online calibration approach which mainly uses the motor encoder information, together with image processing to improve the calibration accuracy. Using motor encoder information increases the robustness of the fundamental matrix calculation and avoids choosing the wrong rotation matrix and translation vector after extraction of the essential matrix.

Compared with all existing methods that only did test on epipolar distance error and vergence angle, we also performed tests on accuracy of the estimated depth.

Building up stereo correspondences

Natural features are used in our system instead of markers. With regard to feature detection, abundant research has been done so far. The FAST feature is among the most efficient ones. For our real-time application, we adopt the FAST feature to speed up the process. For feature descriptors, we use FREAK features which is a novel keypoint descriptor inspired by the human visual system, more precisely the retina. They are in general faster to compute with lower memory load and also more robust than SIFT, SURF or BRISK.

We do not use a standard brute-force matcher and the FLANN matcher, instead we use a constraint feature matcher. Since the left eye and the right eye only move around the x-axis, the y image coordinates should be restricted to a search range. The comparison results are shown in Figure 5.4 and 5.5. From this we can see that the constraint matcher removes lots of false matching.

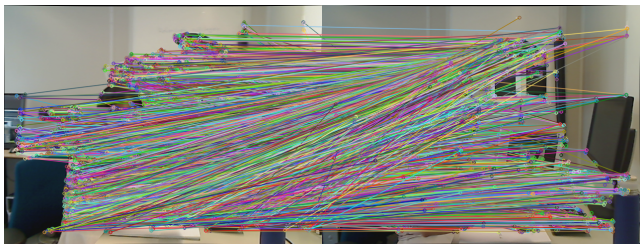


Figure 5.4: Matching result using the Brute-Force matcher

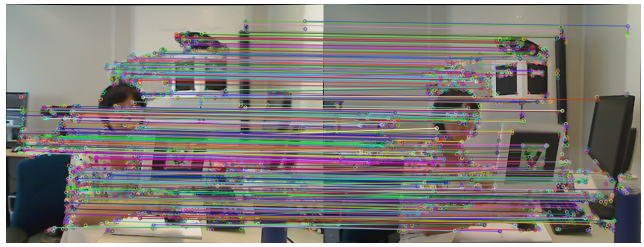


Figure 5.5: Matching result using the constraint matcher

Estimation of fundamental matrix

The equation of estimating the fundamental matrix is described in Section 6.3 and is an $AX = 0$ problem. There are normally more than 8 matching points, therefore the problem turns into a linear least-square problem. Since the error in localization for most points of interest is small (within one or two pixels), we can assume that the image points distribution follows a Gaussian behavior. However, the incorrectly localized few points (with more than three pixels) are very likely to severely degrade the accuracy of the estimation and we have to seek robust solutions. An M-Estimator in which a robust penalty function $\rho(r)$ is applied to residuals, can be used for such a case. Torr [196] gave a review about all the robust methods that can be used for solving a fundamental matrix problem.

First we revisit the fundamental matrix estimation problem as a residuals minimization problem.

Let $F = \begin{bmatrix} f_1 & f_2 & f_3 \\ f_4 & f_5 & f_6 \\ f_7 & f_8 & f_9 \end{bmatrix}$ be the fundamental matrix; $(x_i, y_i, 1)$ and $(x'_i, y'_i, 1)$ be the homogeneous correspondences, thus the residuals are

$$r_i = f_1 x'_i x_i + f_2 x'_i y_i + f_3 x'_i + f_4 y'_i x_i + f_5 y'_i y_i + f_6 y'_i + f_7 x_i + f_8 y_i + f_9 \tag{5.26}$$

and

$$f = \min_f \sum_{i=1}^{n} (r_i)^2 \tag{5.27}$$

In the least-square method, the residuals for any measurement can be arbitrarily large. However, with Equation 5.29 small error values correspond to

Gaussian noise and are included in the minimization process while the influence of large outliers errors are either bounded or totally eliminated. The cost function is defined as

$$f = \min_{f} \sum_{i=1}^{n} \rho(r_i) \tag{5.28}$$

A typical weighting scheme in the statistics literature is the one that is proposed by Huber [197]:

$$\rho(r_i) = \begin{cases} 1 & d_i < \sigma \\ \sigma/|d_i| & \sigma < d_i < 3\sigma \\ 0 & d_i > 3\sigma \end{cases} \tag{5.29}$$

The standard deviation of the error σ is either known a priori or is found as a maximum likelihood estimate using the median

$$\sigma = \frac{median(d_i)}{0.6475} \tag{5.30}$$

Equation 5.26 describes algebraic distance. Minimization of the algebraic distance was found to be sub-optimal. Sampson [198] proposed using a first order approximation to the distance. Therefore the optimal weighting is used in residuals minimization.

$$f = \min_{f} \sum_{i=1}^{n} \rho(w_i f^T Z_i) \tag{5.31}$$

Where $Z_i = (x_i, y_i, x_i', y_i')$ describes the image correspondences. The optimal weighting is

$$w = \frac{1}{\nabla r} \tag{5.32}$$

where gradient $\nabla r = (r_x^2 + r_y^2 + r_x'^2 + r_y'^2)^{\frac{1}{2}}$, and the partial derivatives

$$r_x = f_1 x' + f_4 y' + f_7 \tag{5.33}$$
$$r_y = f_2 x' + f_5 y' + f_8 \tag{5.34}$$
$$r_x' = f_1 x + f_2 y + f_3 \tag{5.35}$$
$$r_y' = f_4 x + f_5 y_i + f_6 \tag{5.36}$$

Besides, due to noise F will have full rank with non zero singular values. In order to force F to have rank 2, let $\Lambda^+ = diag(\lambda_1, \lambda_2, 0)$, and the constraint $F = V\Lambda^+ U^T$.

There are also other robust algorithms to estimate the fundamental matrix such as RANSAC [199], LMedS [200], MLESAC (Maximum LikElihood SAmple Consensus) [201] and MAPSAC (Maximum A Posteriori SAmple Consensus) [202]. We compared MestTorr [196] with before the metioned robust estimators and other estimators. The experimental results in Section 5.5.1 showed that MestTorr achieves a better performance with a lower computational cost.

It is said that in [203], the performance of the M-estimator will degenerate when there is a significant amount of outliers. In order to improve robustness of the M-estimator, we can utilize the motor encoders information to preprocess and remove outliers before estimation. The initial fundamental matrix F_0 derived from encoder information is

$$R_0 = R_m(\phi_r) R_s R_m(\phi_l)^{-1} \tag{5.37}$$

$$T_0 = R_m(\phi_r) T_s \tag{5.38}$$

$$E_0 = R_0 S_0 \tag{5.39}$$

$$F_0 = K_r^{-T} E_0 K_l^{-1} \tag{5.40}$$

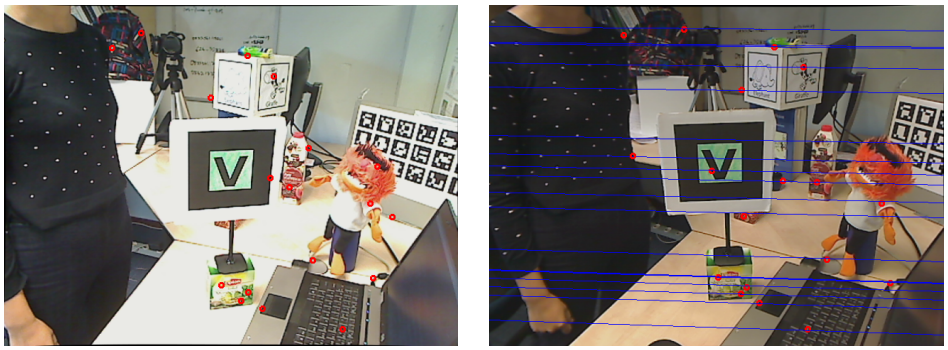
Where S_0 is the screw matrix of T_0, which is

$$S_0 = \begin{bmatrix} 0 & -T_0(3) & T_0(2) \\ T_0(3) & 0 & -T_0(1) \\ -T_0(2) & T_0(1) & 0 \end{bmatrix}.$$

$R_m(\phi_l)$ and $R_m(\phi_r)$ are the left and right rotation matrix calculated by the motor positions. And R_s and T_s are the offline calibrated rotation matrix and the translation vector when two eyes look straight forward.

If $r_i = f_0 Z_i$ is above a threshold, then the image pairs are considered to be outliers. The rest are used to compute the fundamental matrix.

Figure 5.6 shows epipolar lines in which the fundamental matrix is calculated by MestMotor. And in order to clarify the results, we only took several matched points. As we can see, even with outliers, the calculated fundamental matrix still works well. Most of the matched right correspondences are very well located on epipolar lines.



left image right image

Figure 5.6: Epipolar lines calculated using images

Extraction of the rotation matrix and the translation vector

The Essential matrix E has the form $[t]_X R$ with 5 degrees of freedom. Like the fundamental matrix, the essential matrix is a homogeneous matrix having a scale ambiguity (Here we use $[t]_X$ to denote S).

We use SVD to decompose E and we will use two matrices W and Z

$$W = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5.41}$$

$$Z = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{5.42}$$

Note that W is orthogonal and Z is screw-symmetric.

S maybe written as $S = kUZU^T$ where U is orthogonal. Noting that, up to sign, $Z = diag(1,1,0)W$, then up to scale, $S = Udiag(1,1,0)WU^T$, and

$$E = SR = Udiag(1,1,0)(WU^T R) \tag{5.43}$$

Equation 5.43 reveals two things: first it proves one property of an essential matrix: E_{3X3} is an essential matrix if and only if two of its singular values are equal, and the third is 0. Second it points out a way to decompose an essential matrix E into a rotation matrix R and a translation vector T since

E has the form of $E = Udiag(1,1,0)V^T$. If we define the first camera projective matrix as $P = [I|0]$, there are four possible choices for the second camera projective matrix P', namely

$$P' = [UWV^T| + u_3] \tag{5.44}$$

or

$$P' = [UWV^T| - u_3] \tag{5.45}$$

or

$$P' = [UW^TV^T| + u_3] \tag{5.46}$$

or

$$P' = [UW^TV^T| - u_3] \tag{5.47}$$

The four solutions are illustrated in Figure 5.7.

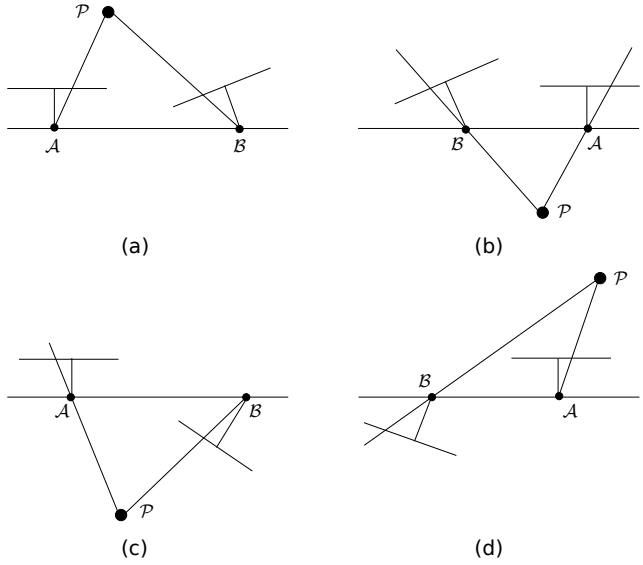


Figure 5.7: The four possible solutions for calibrated reconstruction from E ([204])

It is wise to pick up the solution that best describes the geometry of our setup. One way is to compute the 3D position of the image correspondence of the left eye P_l and the right eye Pr to see if the depth Z_l and Z_r are positive. However, false matching will corrupt this estimation. In a standard stereo correspondence problem, we can make use of the fact that X_l is always bigger

than X_r. However, for convergent vision, this does not hold. It is very difficult to pick up the right camera projective matrix out of four plausible solutions.

In this sense, we can use the encoder positions to find the right projective matrix by checking if the signs of the rotation matrices and translation vectors of the estimated 4 solutions and the one computed from the encoders are the same or not.

One way to ensure the robustness of the algorithm for the estimation of depth is to check the rotation angles and translation calculated from images. If it is very close comparing with the motor encoder information and at the meantime, if the reconstruction errors using images are less, we use image based depth calculation. Otherwise we will still use motor information to perceive depth.

5.4 Multiple cues for depth perception

5.4.1 Depth calculation of steropsis

Stereo calibration is performed offline when two cameras look straight forward. Assuming the camera pairs are calibrated [205], and the intrinsic parameters as well as extrinsic parameters are known, the 3D information about the object can be extracted using image rectification and stereo matching.

(x_l, y_l) and (x_r, y_r) are image correspondences in left and right images.

$$x_l = x_r - d, and y_l = y_r \tag{5.48}$$

here d is the disparity we need to calculate the 3D position of a world point using Equation 6.18.

Two broad classes of stereo matching algorithms are local-based algorithms and global-based algorithms. In local window-based algorithms, the disparity is calculated using only the intensity values within the support window. Although they are computationally cheap, it treates the stereo matching problem per pixel and does not take into account the regional smoothness constraints on the disparities. The global approach solves an optimization problem by minimizing an energy function that combines a data fitting term and a smoothness term.

$$E(d) = E_{data}(d) + \lambda E_{smooth}(d) \tag{5.49}$$

where the data term $E_{data}(d)$ measures how well the assigned disparities values minimize the global aggregated matching cost, similar to the minimal

aggregated value in local algorithms but for all pixels. The smoothness term $E_{smooth}(d)$ enforces a disparity smoothness assumption between the neighbor pixels.

For stereo matching we use 2 different algorithms: a traditional block-matching (BM) algorithm [173], and a modified Semi Global Matching (SGBM) algorithms [206].

The depth map generated provides the depth information of the whole scene, which does not specifically concern the object. Therefore we need to match the 3D information to the tracked object. By assigning a depth value to each point on the object we obtain the 3D object model.

5.4.2 Depth calculation from vergence

The human visual system obeys Listing's law, which means that the cyclorotation of the eyes can be predicted from the direction of a fixed point, which is also called vergence. The first to employ the principle of foveated vision is in [207]. Listing's law can be expressed in terms of suitable rotation matrices, which provides the foundation for the model construction of a convergent eyes system [208].

Why converge?

For humans, convergence is competent for nearby object manipulation compared to stereopsis. It can converge both eyes to focus on the object in order to maintain the object in the center of both views.

With a conventional stereo vision system, the object does not always appear in both views because of a fixed baseline, which limits the handling of the object within a short distance.

Foveas with vergence attend objects of interest with a higher resolution while for peripheral vision a lower resolution using less computational power. This non-uniform resolution property of the human visual system will lead to more advanced humanoid vision system research.

Besides providing a flexible working range and attentive mechanism, vergence has other advantages even for systems without foveas. By using vergence angles to estimate the depth, it brings mathematical simplification.

When the fixation point has zero, and points nearby have small disparities, it is possible to use stereo algorithms within only a limited range of disparities, providing fast computation capacity.

Disparity may be used to filter objects and scenes that are not currently of interest, in this case, disparity-based segmentation is achieved [209]. There-

fore it changes a world centered coordinate system into an object-centered coordinate system.

Two approaches of depth calculation

For a convergent stereo system, as soon as the external calibration is obtained, either a sparse stereo matching method or a dense stereo matching method is deployed to obtain depth perception. After which the convergent depth calculation method modifies the traditional stereo matching methods around the zero disparity, leading to a dense map around the fovea.

Vergence geometry is a special case of stereo geometry in the sense that it can provide 3D information about one particular point in the visual field for a given camera configuration; the point at which the optic axes of the two cameras intersect, which we will call it the fixation point. The depth calculation for such a fixation point differs from the computed stereopsis as it uses angles while computed stereopsis uses disparities.

It is worth noting that the baseline for the depth calculation should be chosen with care. In a stereopsis system the baseline is the line from the left to the right camera. In a convergent vision system there are two choices of a baseline; one is from the left to the right motor while the other is from the left to the right camera. The latter choice of baseline is not constant, resulting in a more complicated computational model for the reason that the lengths between rotation centers and optical centers are less easy to estimate.

We used a simplified model for the depth calculation of a fixation point as shown in Figure 5.8

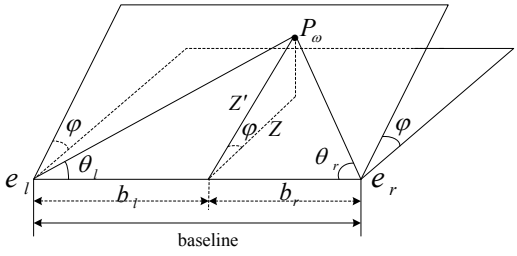


Figure 5.8: Depth calculation model

Assuming that the left and right sight line from the rotation center to the object is a straight line, i.e. eye, camera and object are aligned without rotation

and translation, we can easily derive the following equations:

$$\tan(\theta_l) = \frac{Z'}{b_l} \tag{5.50}$$

$$\tan(\theta_r) = \frac{Z'}{b_r} \tag{5.51}$$

$$b = b_l + b_r \tag{5.52}$$

From (5.50), (5.51) and (5.52), we have:

$$Z' = \frac{b}{1/\tan(\theta_l) + 1/\tan(\theta_r)} \tag{5.53}$$

where $b = e_r - e_l$ is the baseline, θ_l and θ_r are the left angle of the left eye axis and the right angle of the right eye axis, respectively. Then based on which coordinates are used to estimate depth, we have

$$Z = Z'cos(\varphi) \tag{5.54}$$

and

$$Z_l = \frac{Z'}{sin(\theta_l)} \tag{5.55}$$

According to Figure 5.8, Z is the depth from object to the robot and is affected by the tilt rotation angle φ. For now, the tilt rotation angle φ_l of the left eye equals the tilt rotation angle φ_r of the right eye. That is, $\varphi = \varphi_l = \varphi_r$.

However, we generally use the distance from the object to the left eye coordinates which is shown in Equation 5.55. We also used this visual odometry definition in the experimental tests.

5.4.3 Combination of depth cues based on eye movements

The human being's visual system is task orientated and range dependant. For scene exploration, it uses nearly parallel stereo vision while for nearby object manipulation, the convergence is a strong cue. For instance, reading is a vision task that requires both eyes to converge on characters. Here, we mainly use eye movements to switch between different types of vision. Initially, the two eyes will search for interesting objects with the position looking straight forward from left to right (This is also offline calibrated).

$$mode = \begin{cases} stereopsis & \text{if } \phi_{el} - \phi_{er} > 0.01 \\ convergence & \text{if } \phi_{el} > \phi_{er} \end{cases} \tag{5.56}$$

where ϕ_{el} is left motor encoder angle and ϕ_{er} is the right motor encoder angle. Looking straightforward is angle 0 and left is minus and right is plus. When two eyes are looking straight forward, which means $\phi_{el} - \phi_{er} > 0.01$, the stereopsis based pre-calibrated extrinsic parameters are used for estimating depth. While the two eyes are converging to track the object, the convergence based depth perception is working.

In [210], the sparse depth is generated on features instead of on one point, and in [211], two narrow angle cameras are used to implement foveation and two wide angle camera are used to generate the dense disparity map. We argue that, for real-time constraints, the depth on a fixation point is sufficiently enough for tasks such as 3D object tracking and object grasping.

5.5 Experiments and results

5.5.1 Simulation experiments

To evaluate the different methods, a number of simulations were performed. Series of 500 randomly generated points are spread over in front of the cameras. Each point is projected to the left and right image with a resolution of 640×480. For each image pairs, noise with a standard deviation of about one pixel is added, reflecting Gaussian noise. For testing the performance with outliers, we add different amounts of outliers, ranging from 20% up to 50%.

Several approaches were tested for comparison. They are: 1. Seven-point, 2. Least square using eigen analysis, 3. Newton-Raphson iterative method [212], 4. Gradient-based iterative method [204], 5. M-estimator using eigen analysis, 6. MestTorr [196], 8. LMedS [200], 9. RANSAC [199], 10. MLESAC [201], 11. MAPSAC [202].

Here it is worthy noting that in previous experiments [184, 48], they did not take the transformation from camera to motor into consideration, while in our mathematical simulation model, we took this into account. We use a total of 100 data elements for each test and we use the median value to show the performance results. Figure 5.9 compares the above mentioned algorithms when the outliers are 20% and Figure 5.10 when the outliers are 50%. We have three criteria to compare: the mean and variance of points to the epipolar lines distances, the angle error and the computation time. Since the main contribution of the rotation motion comes from the y-axis, the angle error is defined as the difference of the estimated rotation around the y-axis and the ground truth rotation around the y-axis.

It is very clear to see that in general the robust method performs better than least square or iterative methods in coping with outliers and Gaussian

noise. Among the robust methods, the M-estimators outperform the other methods, especially the M-estimator by Torr. It shows 0.9403 pix for the mean value of distance from points to epipolar lines and 0.6328 pix for the variance, 0.0024 rad for the angle error and 0.0835 s for the computation time when there are 20% outliers. When there are up to 50% percent outliers, the performance gets a bit worse, with 7.0821 pix the mean value of the distance from points to epipolar lines, 7.9502 pix for the variance, 0.0667 rad for the angle error and 0.0847 s for the computation time. The M-estimator using eigen analysis is the one having a comparable performance, however its computational cost is a bit higher. LMeds has a fairly higher computational load. MAPSAC is claimed to have the best performance in some papers, however in our setup, the M-estimator by Torr seems to be more robust and accurate. Another advantage is its low computational cost. In conclusion, the M-estimator by Torr outperforms other fundamental matrix estimators, and it can cope with pixel inaccurate locations as well as missed matching pairs. It is used as a benchmark method to develop our own method for estimating the epipolar geometry between two views.

Finally, we will run a different experimental test with all the algorithms mentioned above plus the improved M-estimator using motor information that we denote as "MestMotor". We will show the test performance with 50% outliers shown in Figure 5.11.

As we can see from the figure, MestMotor significantly improves the overall performance. It achieves 1.0577 pix for the mean value of the distance from points to epipolar lines and 0.7496 pix for the variance, 0.0031 rad for the angle error and 0.0590 s for the computation time when there are up to 50% outliers. Comparably, MestTorr has performance of 4.6413 pix for the mean value of the distance from points to epipolar lines and 5.2656 pix for the variance, 0.0617 rad for the angle error and 0.0918 s for the computation time. The underlying reason for this improvement is that by using motor position information, potential outliers are kicked out from the final computation. More inliers are kept and therefore the final accuracy of the M-estimator is improved. And it also taks less time to compute, which is very crucial for real-time applications.

In order to prove the robustness of the improved M-estimator, we carried out a series of experiments with motor vergence angles change. That means that an object is moved from far away to close by and the two eyes are converging during this process.

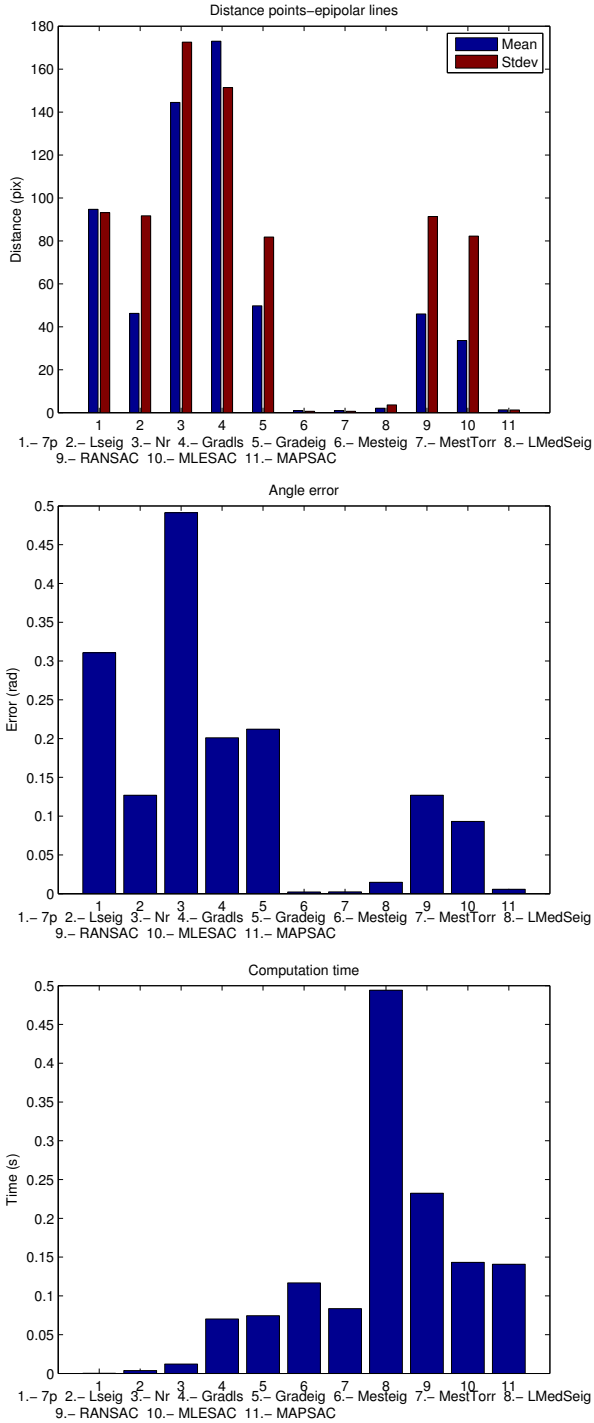


Figure 5.9: Performance comparsion with 20% outliers

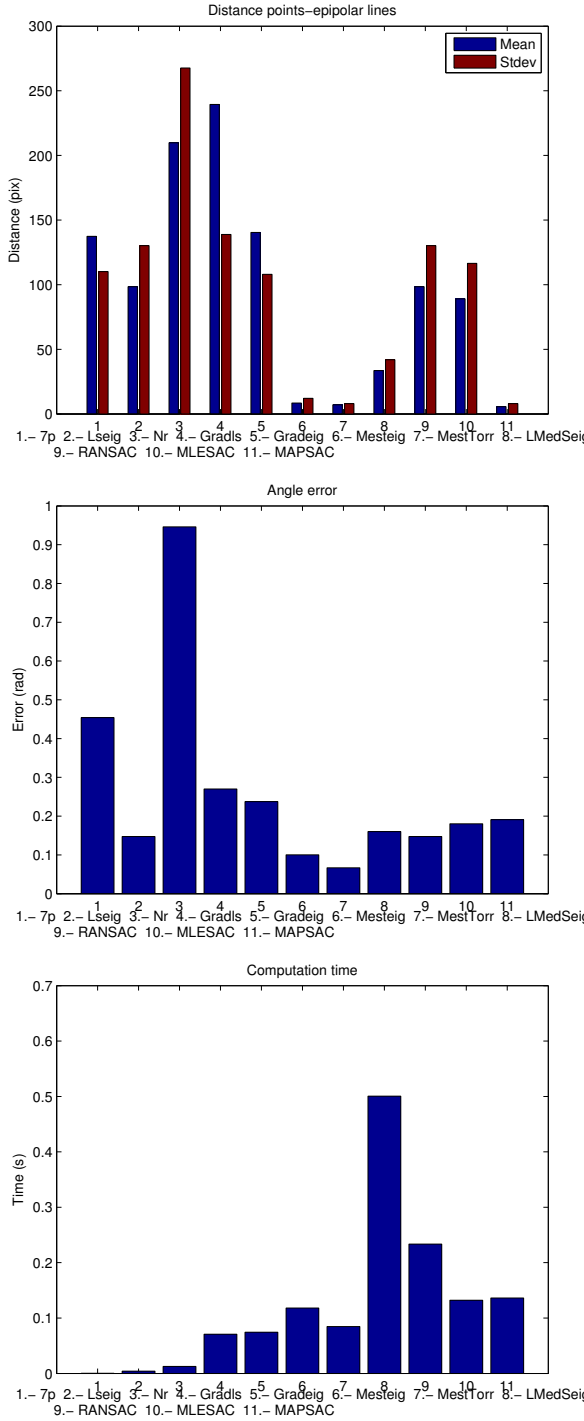


Figure 5.10: Performance comparsion with 50% outliers

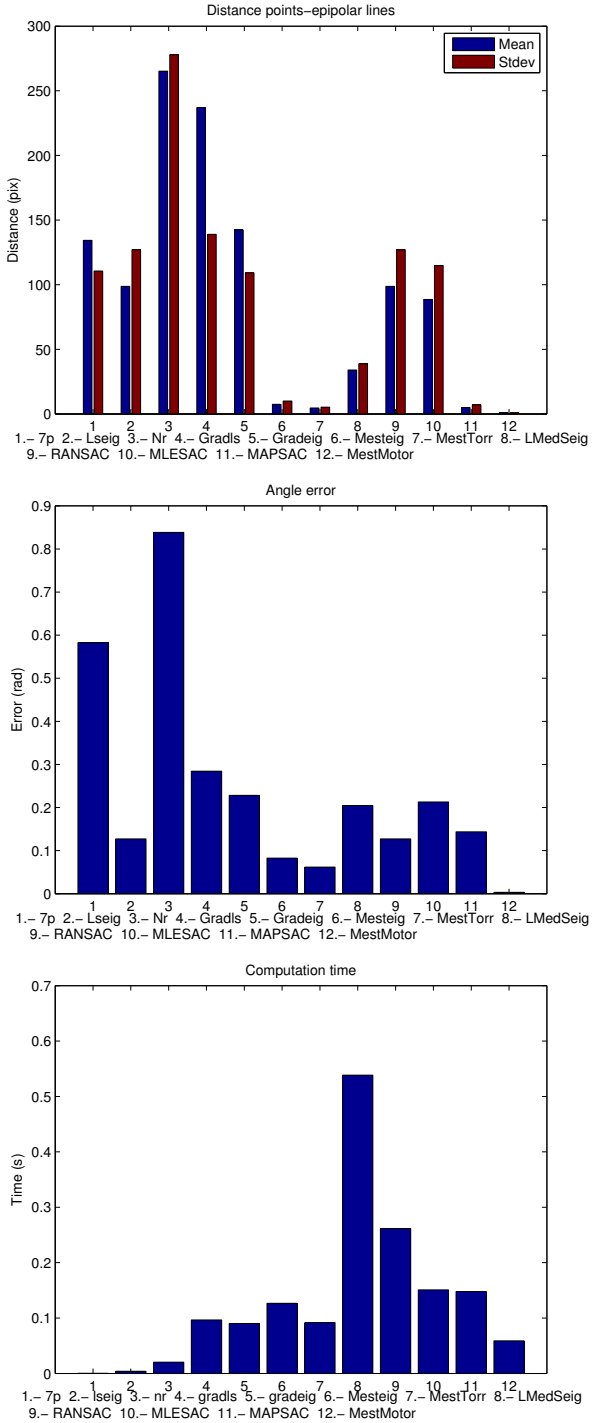
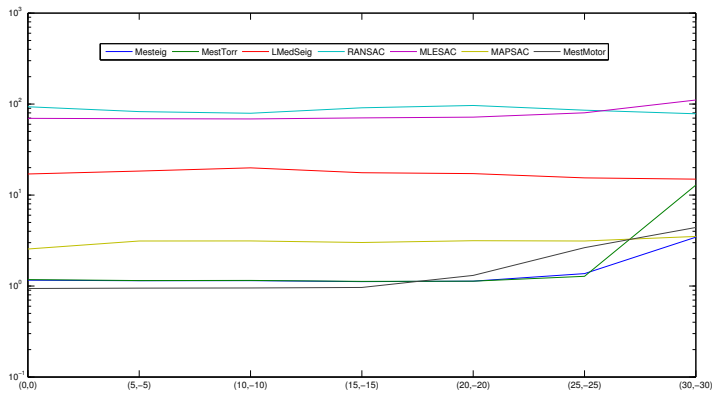
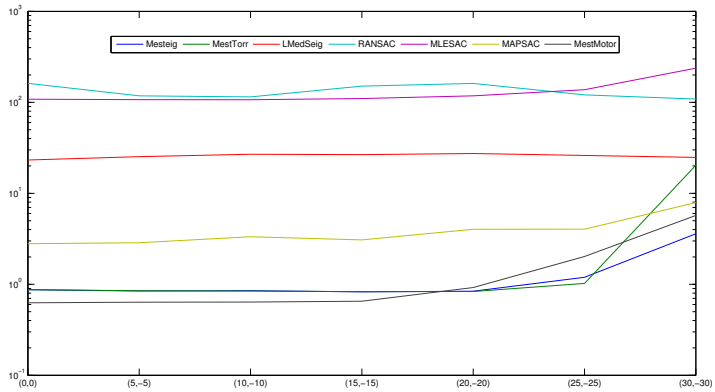


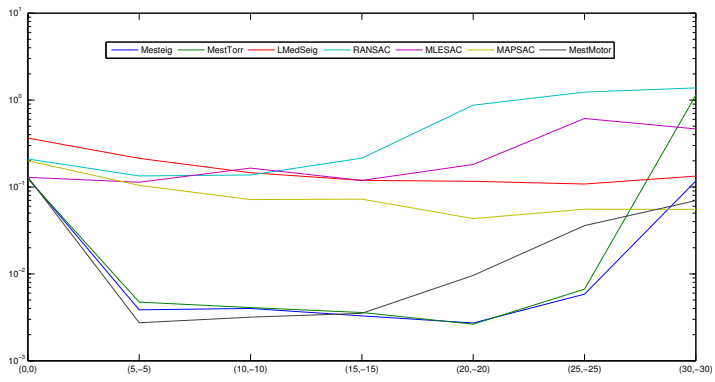
Figure 5.11: Performance comparsion with 50% outliers including MestMotor



(a) Mean value of distance from points to epipolar lines



(b) Variance value of distance from points to epipolar lines



(c) Angle error

Figure 5.12: Performance with different vergence angles with 40% outliers

The results are shown in Figure 5.12. (ϕ_l, ϕ_r) are the left and right eye angles. We can see that it moves from nearly parallel $(0°, 0°)$ when two cameras are looking straight forward to near by $(30°, -30°)$. MestMotor performs in general very well especially when the vergence angle are small. However, when the vergence angle becomes bigger, its performance is slightly worse than MestEig and MestTorr. The reason is that when the vergence angle becomes bigger, the transformation between camera and motor is affected more by the larger rotation angle. In this case, the initial guess using motor information is less accurate. Still, the M-estimator outperforms the other robust methods. With a baseline 14.8 cm and with $((30°, -30°))$ vergence, the object is approximately 25.63 cm in front. When the object is too close to the view, the large view points will cause matching problem while in the mean time, the focus will get blurred. We will not consider the situation when objects are too close. Therefore, we still opt for MestMotor to estimate the fundamental matrix together with motor information.

Real system setup

We use our head-eye system with the eye and neck moving together when tracking an object. With regard to the visual odometry performance estimation, we placed a marker in the scene as an interesting object to be tracked with its 3D position already known. This is used as a ground truth of depth to compare with the algorithm we used. Another advantage of using a marker for testing is that the fixation point in the left image and right image is very well defined. We tested in different scenes with more than 2643 frames in 6 different scenes. We moved the marker randomly with changing in x-axis, y-axis and z-axis. We also moved the object towards the robot head and move it away from it.

We showed the experimental results in one of the 6 scenes, which is in Figures 5.13, 5.14, 5.15, and 5.16, with performance with regard to estimation in x-y-z axes and reprojection error, respectively. The estimation error is defined as

$$X_e = \bar{X}_i - X_i \quad Y_e = \bar{Y}_i - Y_i \quad Z_e = \bar{Z}_i - Z_i \tag{5.57}$$

and the reprojection error is defined as

$$d(x_i, \hat{x}_i)^2 + d(x_i', \hat{x}_i')^2 \tag{5.58}$$

Where $\bar{x}$ represents the ground truth data, x being the estimated data and $\hat{x}$ are the projected image data points. As noticed from the figures, there are

some abrupt estimation errors and the cause of these errors may come from multiple sources: the error in the trackers, the noise that comes from the images induced by motion blur, false matches, the lack of rich image features, encoder position errors, etc. Figure 5.17 shows frame 617 where there is a tracker error which causes inaccurate depth estimation. However, after analysis, we still found out that the 80% of error in the x-axis is below 0.2872 cm and for 90% this is below 0.6136; 80% of the error in the y-axis is below 0.2739 cm and for 90% this is below 0.5275; 80% of the error in the z-axis is below 3.4535 cm and for 90% this is below 5.4436 ; 80% of the reprojection error is below 0.9327 cm and for 90% this is below 1.5906; As stated in [48], in many applications the depths are in fact approximately known and the question is whether this knowledge can be used to simplify the problem. In our case, the accuracy of the depth calculated should be sufficient enough for many real world applications such as autonomous navigation, obstacles avoidance, object grasping, etc. We argue that with accuracy in x-y axes less than 1 cm and if we e.g. add proximity sensors on the gripper, it is enough for object manipulation tasks and it could also perform well in navigation and exploration in 3D space. Table 5.1 and Table 5.2 show average estimation error, and reprojection error and the median estimation error and reprojection error in 6 scenes. It further proves that the algorithm is able to work in practice. Here it is worth noting that median estimation error is lower than the average estimation error in general. This is due to the inrobustness in some estimations which is caused by an abrupt estimation error. This can be reduced by using a filter. For instance, when the depth calculated changs abruptly, it will be discarded.

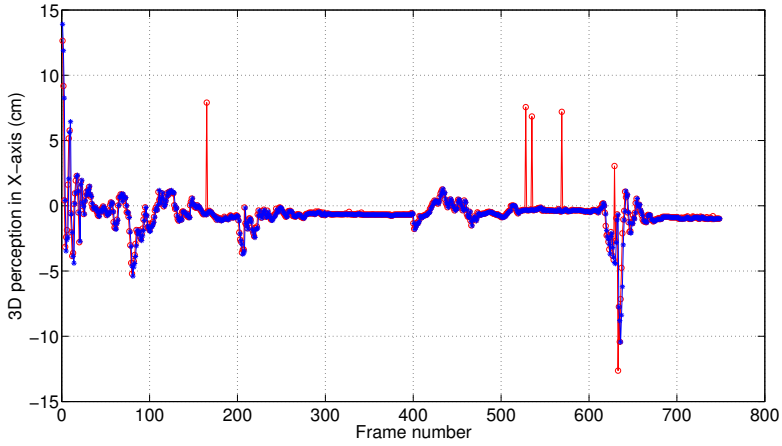


Figure 5.13: Estimation error in x-axis

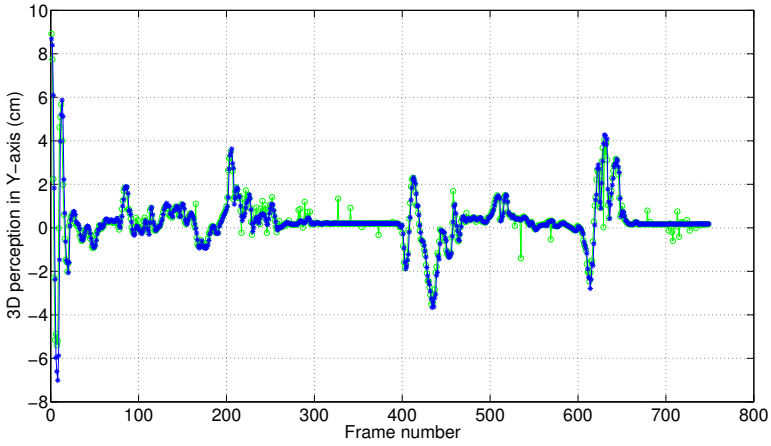


Figure 5.14: Estimation error in y-axis

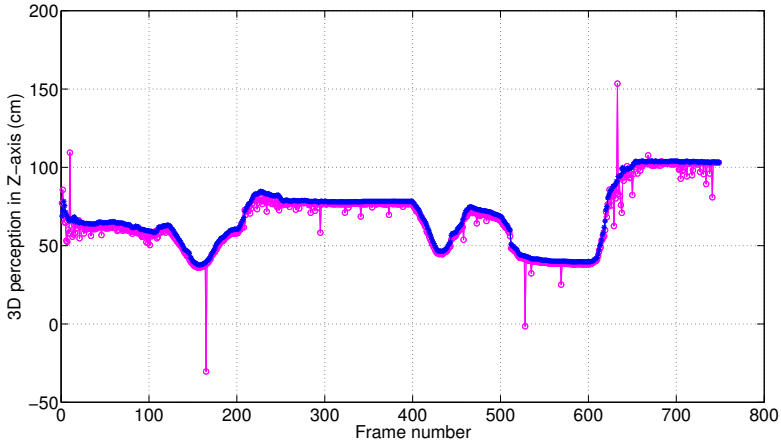


Figure 5.15: Estimation error in z-axis

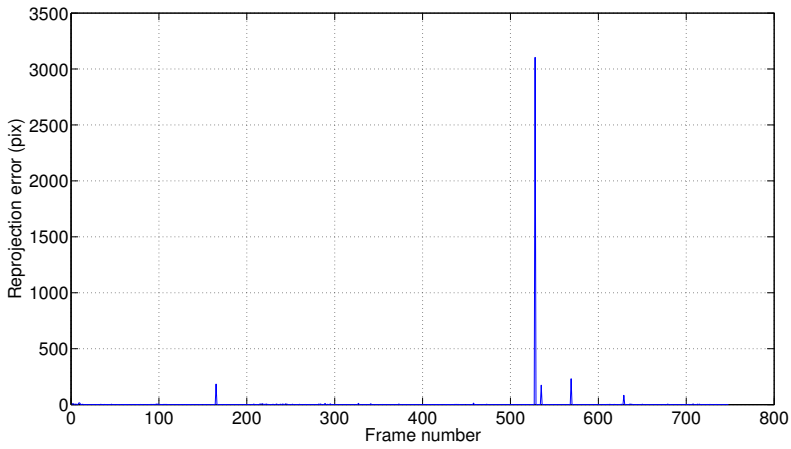
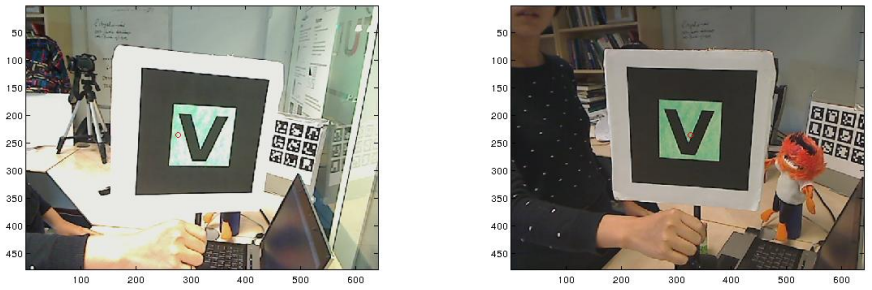


Figure 5.16: Reprojection error



left image right image

Figure 5.17: The effect of tracker performance on depth perception

Table 5.1: Average estimation error and reprojection error

scene	1	2	3	4	5	6
average x-axis error	0.8025	0.8401	0.3140	0.4252	0.5643	0.8142
average y-axis error	0.3066	0.2353	0.2138	0.2418	0.2006	0.5808
average z-axis error	4.3401	4.5882	3.0856	4.5453	5.0387	4.6247
average reprojection error	22.5189	31.0368	5.8402	35.4874	4.4402	6.5329

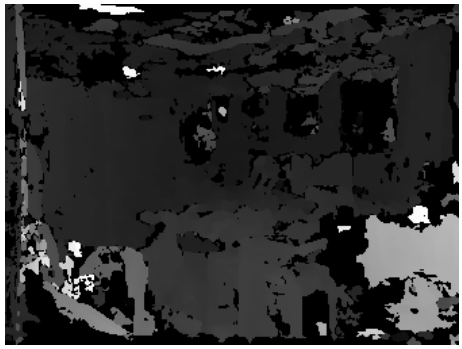
Table 5.2: Median estimation error and reprojection error

scene	1	2	3	4	5	6
median x-axis error	0.3511	0.2077	0.7010	0.1655	0.2595	0.2714
median y-axis error	0.1369	0.1116	0.0676	0.1353	0.1086	0.4051
median z-axis error	2.3210	1.7787	2.0393	1.9903	2.5240	2.3584
median reprojection error	0.5929	1.0010	0.3741	0.4461	1.0557	0.5365

Figure 5.18 gives several results when stereo matching is performed at various vergence angles. The left group of figures displays the original images and rectified images, and the right group of figures shows the disparity map. The disparity maps reflect the overall depth distribution in the scenes. Even with moving objects inside the scenes, the stereo matching method still works well. One advantage of our stereo vision is that moving objects in dynamic scenes will not affect the pose estimation between corresponding images. However, in a classic SLAM system with one camera, this is a big problem. We can also see that at frame 144, where the left angle moves at -9.5684 degree and right angle moves at 1.4897 degree, the disparity map stills gives good depth perception results. It also holds at frame 166 with a left angle 2.2345 degree and a right angle 6.7609 degree and frame 227 with

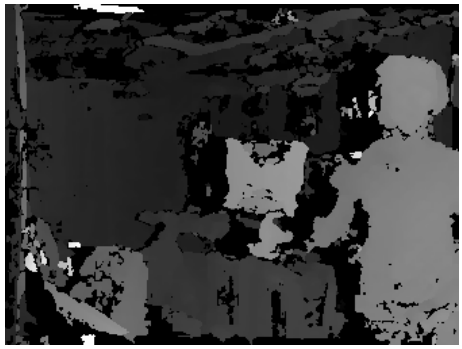
left angle at -12.6624 and right angle at -2.5210. When the vergence angle increases, which is shown in frame 333 in figure 5.19, the fundamental matrix still works and the rectified images shows that the correspondences are located at the scan line in the right images. However, the disparity map does not give a good result and the reason is very clearly illustrated in the figure. With big vergence angles, the overlap of close by objects increases and the overlap of far away objects decreases. However, the big vergence angles also bring a problem for stereo matching, since correspondences suffer from severe viewpoint changing and projective distortion. We can see how the hands in the two images are different from each other. This will cause the failure of the stereo matching methods and advanced stereo matching methods targeted to this situation should be developed.

(a) rectified image at frame1



(b) disparity map at frame 1

(c) rectified image at frame 20



(d) disparity map at frame 20

(e) rectified image at frame 75

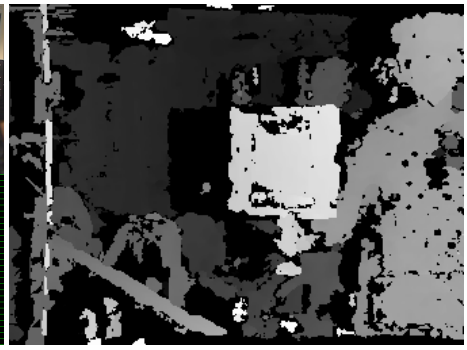
(f) disparity map at frame 75

Figure 5.18: Stereo matching results

(g) rectified image at 144

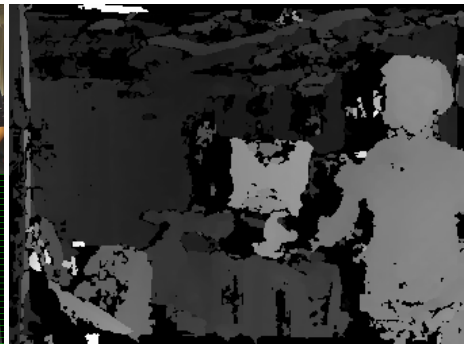
(h) diaparity map at frame 144

(i) rectified image at frame 166



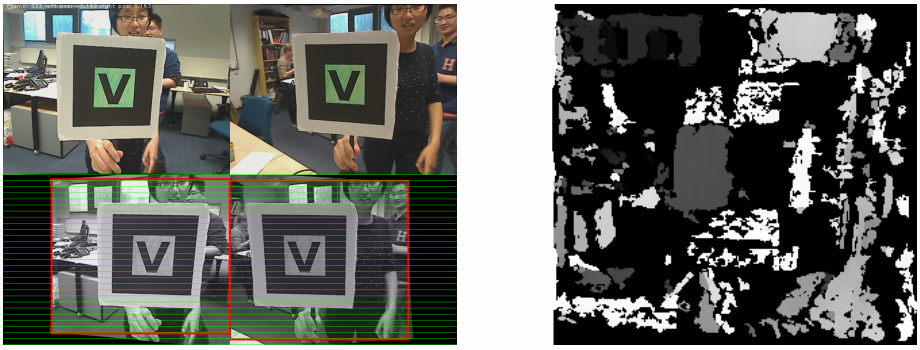
(j) diaprity map at frame 166

(k) rectified image at frame 227



(l) disparity map at frame 227

Figure 5.18: Stereo matching results



(a) rectified image at frame 333 (b) disparity map at frame 333

Figure 5.19: The effect of large vergence angles on stereo matching results

The stereo matching and point clouds generated are shown in Figure 5.20. Compare with inter-frame matching, one advantage of stereo matching is that it can cope with dynamic scenes. As one can see, with a person moving from left to right in the background it can still reconstruct the 3D scene without any problem. Dense stereo matching costs much more computation time than depth perception of a fixation point and feature based sparse matching. Dependent on different situations, different depth perception methods can be chosen. For object grasping as well as obstacles avoidance, fixation based depth estimation should be enough. Sparse feature based stereo matching can be used for 3D perception of the environment or even 3D object modeling. Dense stereo matching works better in case of low feature presence or the case that high 3D detail is needed. With the increase of computation power or using a GPU, the computation time can be reduced.

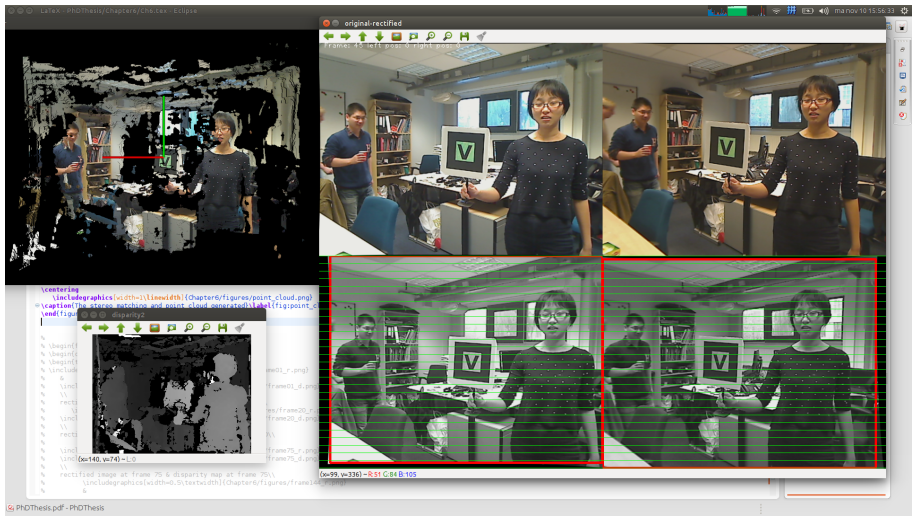


Figure 5.20: The stereo matching and point cloud generated

5.6 Conclusion and discussion

The experimental results proves the effectiveness of the multimodal depth perception concept, which is inspired by the human visual system. To realize such a system for a robot head, we first have built up a mathematical model, investigated several robust methods and then proposed our online calibration method, which takes advantage of known motor information. We also proposed to use various cues such as stereopsis, and convergence to calculate the depth. Then a multimodal depth selection mechanism based on eye movements was devised to ensure the robot to perceive the 3D depth in personal space as well as in action space. According to our knowledge, stereopsis is used to provide general 3D spatial relations for humans. In this case, the absolute depth value is not known precisely. In the case of personal space with an arm distance from hand to eye, humans use jointly vision information and action feedback to gain precise 3D information of the objects that being operated. Convergence can be used in this case to either assign 3D information to a fixation point or generate a high resolution 3D map around foveas. As a conclusion, in the future, the action should combined with vision to explore unknown environments.

Our objective is to push forward the understanding of the human visual system by carrying on research in the domain of humanoid robots, with multimodal depth perception for humanoid robots as an initial step. There are various ways of depth perception besides stereopsis and convergence, and we aim to proceed in this topic.

Besides, when converging, a more advanced stereo matching method that takes advantage of zero disparity should be developed. Meanwhile, it can cope with large view point change across stereo views. From this point of view, foveation should also be developed to generate a high-resolution depth map in the center of view.

6 Conclusion

6.1 Research goal

As already stated, most research in machine perception is involved in the analysis of passively sampled data (images). Human perception, however, is not passive. It is active [1]. Gradually, active vision research area gains more and more popularity. There are about 2000 research papers published during 1986 – 2010 that are closely related to the topic of active vision perception in robotics [7]. Among these, the study on humanoid vision systems is one of the branches that is inspired by the human visual system, which covers a wide range of research fields such as visual attention, control of eye movements, action and perception, 3D perception, SLAM, etc. Based on its active nature, researchers have to deal with several degrees of freedom, dynamic scenes, as well as ego-motion, making it a very challenging research field.

In this thesis, we design and build a robotic active vision system that can perceive unknown environments in a similar way as humans. We have to meet some requirements. First of all, our active vision system should have eye movements as well as neck movements. Secondly, two cameras are used as the main source to acquire visual information, while no other advanced sensors are employed. Thirdly, the proposed vision system is able to deal with tasks such as saliency detection, object detection and tracking, and 3D perception to prove that it can be used to carry out more complex tasks, such as object recognition and grasping. Fourthly, a real-time constraint is added to make the system to operate in real world constraints. Fifthly, most of the robotic systems are fairly expensive. We would like to have an affordable solution. With a low cost off-the-shelf product, robust algorithms should be applied to deal with hardware inaccuracy and instability. The eventual goal is to have a complete working setup that is able to mimic the eye movements of humans, and still have the robust functionalities of most mobile robot systems. Moreover, it can provide insight into human vision systems and a platform for other researchers to work on more advanced features for humanoid robot active vision systems.

6.2 Summary and applications

The mechanical part of the system is composed of 4 motors and has 4 degrees of freedom: the robot head can move separately on its pan and tilt axes and the left eye and right eye can move separately on their pan axes.

6.2.1 Hardware

The system underwent 3 major different mechanical developments. At the beginning, the 4 motors used Dynamixls, which are easy to be incorporated due to their daisy chain property, however, the resolution of the encoders is low and the movements have a lot of friction. In order to have higher resolution and reduce the friction, the eye motors were changed to two Maxon DC motors RE-16 which are controlled by home made controller boards (3Mxl) and jointly referred to as "3Mxl RE-16". Then the neck motors were also changed to two Maxon DC motors Amax-22, in combination with actuator Maxon MR-M, which are controlled by a home made controller board (3Mxl), jointly referred to as "3Mxl Amax-22". A Xsens MTi IMU was attached later to achieve VOR eye movements. Extra springs were added to both eyes to reduce backlash. This improved the precision when reading positions from encoders. After all these adaptions, a more precise 3D depth perception was attained. Besides, the maximum speed of the motors is higher, which means they can control all the movements faster to locate the cameras to specific positions. Besides, the neck tilt axis center and neck pan axis center are different in our design. In the future, it could be designed to be at the same location, making the calculation easier. The difficulty in kinematic calibration of such a system comes from the fact that optical center and rotation center are not aligned. If there is a way to make them aligned, the calibration will be straightforward. In our design, we prefer an affordable solution. The total cost of our system is fairly low, which makes it interesting for many applications. Further implement can be expected from cheaper and better electronics and mechanical design.

6.2.2 Control

Based on our mechanical design, the controllers are built. There are two main control components: a lower control component and a higher control component. The lower controller controls the 3Mxls and the higher one is to send position and velocity information to a lower controller to jointly accomplish all the eye and neck movements based on images input. The controllers achieve

different types of eye movements such as saccade eye movements, pursuit eye movements, vestibulo-ocular reflex (VOR) movements, as well as vergent eye movements. Saccade eye movements are controlled by an open-loop controller mainly using focal length and image information at a high speed. Pursuit eye movements are controlled by a closed-loop controller using off-center pixel displacement error at a low speed. During the tracking process, both eyes are moving to make target in the centers of both images and the neck is moving towards the target to ensure the same vergence angles for both eyes. Meanwhile, the eyes counter rotate with respect to the head movement to achieve vestibular-occular eye movements. These eye movements all work as a whole for a humanoid robot to imitate vision-based exploration. This is proved by our experiments.

We mainly use PID controller, however, we also tried to implement Kalman filters (KF). The KF helps to smooth the noise and to improve the accuracy. However, it causes time delay compared to the ones without filtering. For the KF with the driving function, it improves its prediction ability and decreases the time delay. Still, it does not improve the performance much while it adds computational load. In this case, the gain should be adjusted to be smaller in order to prevent overshoot. Since speed is one of our main concerns and the image noise is minor, we eventually did not adopt KF in our system.

With respect to saccade eye movements, it is achieved by using focal length and image information. However, since the optical center and rotation center are not aligned and the focal length has estimation error, the attention is not perfectly shift to the centers of the images during saccade eye movements. Newborns train the muscle to control the saccade eye movements, therefore obtain better attention performance. In a similar way, machine learning could be used here to train the saccade eye movements for humanoid robots.

There are also other eye movements, for instance Optokinetic reflex eye movements, which are not yet implemented in our system. It is a reflex when an object moves out of the field, the eye shifts back to the position when it first saw the object. This can also be implemented in the future, once the system is mounted on a mobile robot platform.

6.2.3 Visual primitives representation

Next, we proposed a novel adaptive tracking selection mechanism which automatically selects the tracking methods dependent on visual properties of objects. The reason why we are interested in this selection mechanism is based on our observations and experiments with many state of the art trackers. We

found that the trackers which use corner features, local binary features are good at tracking objects with lots of features. However, they perform not so well on more uniform objects. On the other hand, color based trackers work better on uniform objects and fail to perform well on textured objects. Is there a way to combine two advantages and avoid the weakness and develop a more robust tracker? If there is a way, how to choose in which case which one should be used? Based on these questions, we developed a mechanism that can automatically select the optimal tracking method. In order to do this, we first deployed a GrabCut based algorithm to segment the object and eliminated the effect of background and contour on object property estimation. Then we measured the amount of textureness within the object. Dependent on the measurement result, either textured object tracking or uniform object tracking method was used.

We tested our optimal feature selection mechanism and the test results proved the effectiveness of our method. From this research, we can conclude that a general purpose system should be able to select an optimal set of features for a given object and then adjust tracking methods accordingly. In our system we mainly used texture and color as cues, however, more cues for object property measurement should be added to improve the automatic feature selection method. If computation load allows, the online feature selection can be applied to achieve more robust performance in case of changing of appearance. How these cues are combined should be thoroughly studied as well.

6.2.4 Object tracking and segmentation

We developed in total three trackers. One is a marker-based tracker to test the control and 3D perception performance. Because the marker-based tracker encodes 2D image information together with 3D position information, it is a also optimal measurement tool for testing purpose. The other one is a color-based tracker mainly for demonstration purpose since it runs at real-time. The other one is our proposed tracker which is based on object properties. We tested the proposed tracker and it showed to have better performance than most state-of-the-art algorithms, since it adopts different trackers for different objects, depending on their properties.

Meanwhile, we proposed a novel tracker for uniformed objects. Compared with standard color-based tracker e.g., Camshift, it improves using a smoothness constraint, where the inter connections between pixels from the object are refined while the exterior connections between object pixels and background pixels are weakened.

Afterwards, online segmentation is applied to obtain more detailed information rather than just the location of the tracked object. GMMs are used to create the object and the background models. Then, graph cuts is used to obtain the segmentated object. This segmentated object can then be used for object recognition, object grasping, etc.

We tested on our proposed tracker and it achieved very promising results. However, its online computation cost still need to be improved to further realize real-time process speed. Based on this, we are able to integrate the whole system together.

Besides, the uniform tracker has already improved compared with existing color-based tracker. However, this tracker is still sensitive to varying conditions, e.g., illumination change. Further research is needed to provide a more robust object description.

When two eyes are converging, zero-disparity can be used to segment the salient object. This can combined with our segmentation method to achieve a better segmentation performance and also lead to less computation cost.

One last topic that needs to be investigated further is how tracking and segmentation works in case of fovation. The resolution is not uniform everywhere. It can greatly save computation cost and make human vision focus on dominant objects and information. However, in this case, how we are going to take this advantage and make tracking and segmentation more robust and less computation load.

6.2.5 Multimodal Visual Odometry Perception

Since in our set-up, the extrinsic parameters including rotation matrix and translation vector from left eye rotation center and right eye rotation center remain unknown, therefore before we attain 3D perception, calibration should be carried out. The main problem is the rotation center is not the optical center. In order to obtain 3D perception, we proposed an online calibration method which takes advantage of known encoder information to speed up and improve the precision of the calibration process.

The depth perception of humans utilizes multiple cues. In this thesis, we mainly focus on stereo-based depth perception which are mostly stereopsis and convergence based depth perception. For stereopsis, we adopt the standard methods. For convergence, we have implemented two types visual odometry perception methods. One works on the focus point only by bringing the attention onto principal point of each camera; while the other one will consider an area rather than a single point. The first one provides accurate short range depth estimation for object manipulation tasks, while the

latter is more suited for building up 3D maps of objects from a close distance for further exploration including learning, recognition, etc. Besides, the one that works on the focus point also simplifies the calculation mathematics and works in a very efficient way.

Finally we presented a multi-mode depth perception method using multiple cues to gain 3D spatial information. The stereopsis and convergence based 3D perception are combined using eye movements. When eyes are converging, the convergence based 3D perception takes effect and in other cases the stereopsis based 3D perception functions. The experimental results proves the effectiveness of the multimodal depth perception concept, which is inspired by the human visual system.

The better resolution of the encoders, the better calibration results. The way we rely more on motor encoders information rather than image information is because image information can be more precise sometimes, but is not robust enough. In featureless and blur scenarios which often occurred in indoor environments with moving eyes, the image based calibration performance is fairly degenerated. If precise kinematics model is already known, then calibration is a straightforward problem.

There are various cues to calculate depth (Chapter 5). Here we only explore two kinds of cues, and there are still lots of other cues left for us to explore. Other monocular cues should be combined such as motion parallax, accommodation, lighting and shading, perspective, etc. One of the most difficult part is how to combine all these cues, especially when there exist redundant even conflict and inconsistent information from all these various cues. How human system make use of all these cues and get a fairly good 3D perception performance? In human visual systems, high level knowledge of the scene contributes to solving the conflicts. More research about this is still needed.

Depth information is for a robot to know its relative distance with respect to its 3D environment and objects inside. Structure from motion is for robot to locate itself in the world and build up maps and memory. Stereo vision can be used to generate a scene flow map, which can be used to solve traditional structure from motion problems working in dynamic scenes.

Zero disparity around the fovea can be employed for object segmentation in action space, which can be further used in object manipulation. Besides, human visual systems are vision systems based on learning activities, which means that we keep our focus on a salient object near fovea. We can use disparity to segment the salient object and we can also use it to estimate the 3D

information. In our set-up, we do not have fovea, which should be researched in the future.

Moreover, there are also different types of depth: absolute depth, relative depth and so on. Relative depth plays an important role in our spatial perception. In a 3D space, we use relative depth to know where are things organized in 3D space and where do we locate in this space. With respect to a close distance, we know the relative distance between our arms and objects. Based on this, we are able to manipulate close-by objects. For humans, the precise absolute visual odometry information is not always necessary. In the contrary, a more relative depth information is more needed. The same holds true for humanoid robot. How precise do we need to make a robot that is able to navigate in a 3D space and learn from it? Using them to improve the capability of robots still needs to be done.

6.2.6 Applications

We have built up a humanoid robot head. It can be installed on a mobile robot platform. It has various usages.

For instance, it can be used on a service robot. The robot can take the order from a user to get a bottle of beer. It has the ability to store the object representations in memory which contains both 2D and 3D descriptions. As soon as the robot enters a room, the active vision system is used to search around and try to find the matched one. Once the robot finds the bottle, it is able to track and approach the object from a far distance to a close distance. Meanwhile, the robot will converge both eyes to obtain a more precise distance estimation until grasp the bottle and a refined 3D shape information to compare with the 3D shape model in memory. Then, the robot will hand over the bottle to its user. The robot vision system can also help the robot to explore and learn unknown environments. It can enter an unknown environment and attend all salient objects inside it. If there exists something new, the robot can navigate around this unknown object, track and learn it from different perspectives, and eventually store the object model in memory.

Our robot head can be used to implement more advanced bio-inspired human visual features. Fovea can be added to attend salient areas and objects. More different eye movements can be added and various cues of depth perception can be tested. We are able to propose a cognitive model based on observations on human vision systems, however, we are not able to test the proposed model since our vision systems are blind systems with past experiences which are different for different persons. Therefore, advanced cognitive

reasoning can also be tested and implemented on top of our system. Thus, the knowledge about humans will push forward the knowledge on humanoid robots and the knowledge gained from humanoid robots will also push forward the knowledge about ourselves.

It can be used in machine vision applications as well. For instance, for a robot to pick up tomatoes in a green house. Active vision is to look all around for a potential tomato. The 3D information and obtained segmentation results of the tomato can be combined together for the robot arm to take and grasp the tomato and put it into the container. Since our vision set-up uses affordable devices and works in real-time constraints, it is very suitable for mass productions.

During our research, we also have a close collaboration with Augmented Reality (AR) group. Our robot head can also be applied to AR. The main goal of AR is to add virtual figures to human vision field. However, most AR nowadays focuses on tracking and mapping as well as visualization while ignoring the nature of our eyes. To enhance the real experiences, we need to take into consideration that our eyes are not just two stand-still cameras, they are moving and converging. If we adopt moving cameras instead static ones, we could make the AR experiences more real, especially when looking at an object from a nearby position.

Another application field for our robot head is security surveillance. It can actively instead passively search suspicious activities, shift the attention to suspicious invaders. Then, the robot vision system zoom in to get higher resolution images and detailed information to determine the next action whether it should alarm or not. The advantage of active vision here is that it can cover a wide range while still focus on the most important details. Therefore, a better security surveillance performance is attained.

Our robot head can also just be a simple way to express emotions. It can move the eyes, follow the face of a person in front of it, making the expression more vivid.

There are various application areas that our robot head can be applied. We believe this research should be carried on and the system should be improved.

6.3 Future research

Owing to limited research time, we could not research every aspect of active vision for such a humanoid robot. From a horizontal level, applications such as object modeling, object recognition can be implemented. From vertical level, there is even higher cognitive vision computation possible. With the

advent of more knowledge on the human visual system, more blocks can be added to the humanoid robot vision scheme.

Furthermore, the computational load is still too high. GPU or hardware based speed-up approaches can be used to improve the real-time performance.

There are other research areas which are interesting and need to be investigated such as:

1. Attentive vision vs explorative vision: with advanced saliency detection algorithms, a robot is able to attend its view towards an object or region of interest. By shifting its attention from one place to another place, a robot has the ability to build up an attention map, for the exploration of the unknown environment.

2. Top-down and bottom-up information are combined together to develop cognitive perception of a humanoid robot. It is known that humans use previous information while exploring the unknown environments. Different subjects have different saliency maps even in front of the same scene.

3. Learning is a process of long term memory. How to gain a long term memory and how to post-process such huge information and extract abstract knowledge still remains an unknown challenging field.

4. Action and perception are strongly winded up with each other. Infants show their ability to perceive the world by touching and interacting with objects and surroundings. During this process, perception provides input to effectively build up this connection.

5. Compensatory eye and neck movements reflexes are there to keep vision stability. We only researched VOR and there are still other reflexes needed to be taken into consideration such as the optokinetic response (OKR).

6. Object recognition is a very strong abstract tool for a robot to learn the world and build up its own vocabulary. It provides top down information and also input for more complicated tasks such as "bring me a bottle of beer".

7. Social emotion that is reflected by facial expression through active vision is a way to connect humans and robots. Making a humanoid robot means it can be more easily incorporated into the human world.

Bibliography

[1] R. Bajcsy, "Active perception," *Proceedings of the IEEE*, vol. 76, no. 8, pp. 966–1005, 1988.

[2] A. Dankers, "Realtime stereo active vision," http://users.cecs.anu.edu.au/~rsl/rsl_active.html, 3rd October, 2006.

[3] Z. Zhang, "Microsoft kinect sensor and its effect," *IEEE MultiMedia*, vol. 19, no. 2, pp. 4–10, Feb. 2012.

[4] J. P. Frisby and J. V. Stone, *Seeing: the computational approach to biological vision*. MIT Press, 2010.

[5] R. Berman and C. Colby, "Attention and active vision," *Vision Research*, vol. 49, no. 10, pp. 1233 – 1248, 2009, visual Attention: Psychophysics, electrophysiology and neuroimaging.

[6] D. S. Z. R. John Leigh, *The neurology of eye movements*, 4th ed. Oxford University Press, 1999.

[7] S. Chen, Y. Li, and N. M. Kwok, "Active vision in robotic systems: A survey of recent developments," *International Journal of Robtics Research*, vol. 30, no. 11, pp. 1343–1377, Sep. 2011.

[8] N. J. Ferrier, "Harvard binocular head," in *Applications of Artificial Intelligence X: Machine Vision and Robotics*, vol. 1708, 1992, pp. 2–13.

[9] B. Scassellati, "A binocular, foveated active vision system," Cambridge, MA, USA, Tech. Rep., 1998.

[10] S. Vijayakumar, J. Conradt, T. Shibata, and S. Schaal, "Overt visual attention for a humanoid robot," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 4, 2001, pp. 2332–2337 vol.4.

[11] L. Aryananda and J. Weber, "MERTZ: a quest for a robust and scalable active vision humanoid head robot," in *Proceedings of 4th IEEE/RAS International Conference on Humanoid Robots*, vol. 2, Nov. 2004, pp. 513–532.

[12] S. Kim, C. H. Kim, and J. H. Park, "Human-like arm motion generation for humanoid robots using motion capture database," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2006, pp. 3486–3491.

[13] R. Beira, M. Lopes, M. Praga, J. Santos-Victor, A. Bernardino, G. Metta, F. Becchi, and R. Saltaren, "Design of the robot-cub iCub head," in *Proceedings of IEEE International Conference on Robotics and Automation*, May 2006, pp. 94–100.

[14] A. Ude, C. Gaskett, and G. Cheng, "Foveated vision systems with two cameras per eye," in *Proceedings of IEEE International Conference on Robotics and Automation*, May 2006, pp. 3457–3462.

[15] B. Rasolzadeh, M. Björkman, K. Huebner, and D. Kragic, "An active vision system for detecting, fixating and manipulating objects in the real world," *International Journal of Robtics Research*, vol. 29, no. 2-3, pp. 133–154, Feb. 2010.

[16] T. Asfour, K. Welke, P. Azad, A. Ude, and R. Dillmann, "The karlsruhe humanoid head," in *8th IEEE-RAS International Conference on Humanoid Robots*, Dec. 2008, pp. 447–453.

[17] A. Bakhtari and B. Benhabib, "An active vision system for multitarget surveillance in dynamic environments," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 37, no. 1, pp. 190–198, Feb. 2007.

[18] J. Batista, P. Peixoto, and H. Araujo, "Real-time active visual surveillance by integrating peripheral motion detection with foveated tracking," in *Proceedings of IEEE Workshop on Visual Surveillance*, Jan 1998, pp. 18–25.

[19] J. Civera, O. Grasa, A. Davison, and J. M. M. Montiel, "1-point RANSAC for ekf-based structure from motion," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2009, pp. 3498–3504.

[20] E. Eade and T. Drummond, "Scalable monocular slam," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, Jun. 2006, pp. 469–476.

[21] G. Qian and R. Chellappa, "Structure from motion using sequential monte carlo methods," *International Journal of Computer Vision*, vol. 59, no. 1, pp. 5–31, 2004.

[22] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *Proceedings of 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, Nov. 2007, pp. 225–234.

[23] W. Tan, H. Liu, Z. Dong, G. Zhang, and H. Bao, "Robust monocular slam in dynamic environments," in *Proceedings of IEEE International Symposium on Mixed and Augmented Reality*, Oct. 2013, pp. 209–218.

[24] J. Engel, J. Sturm, and D. Cremers, "Semi-dense visual odometry for a monocular camera," in *Proceedings of the 2013 IEEE International Conference on Computer Vision*. Washington, DC, USA: IEEE Computer Society, 2013, pp. 1449–1456.

[25] K. Huebner, K. Welke, M. Przybylski, N. Vahrenkamp, T. Asfour, D. Kragic, and R. Dillmann, "Grasping known objects with humanoid robots: A box-based approach," in *Proceedings of International Conference on Advanced Robotics*, Jun. 2009, pp. 1–6.

[26] X. Gratal, J. Bohg, M. Björkman, and D. Kragic, "Scene representation and object grasping using active vision," Oct. 2010.

[27] G. R. Bradski, "Computer vision face tracking for use in a perceptual user interface," 1998.

[28] J. Badenas, J. M. Sanchiz, and F. Pla, "Motion-based segmentation and region tracking in image sequences," *Pattern Recognition*, vol. 34, no. 3, pp. 661 – 670, 2001.

[29] F. Jurie and M. Dhome, "Real Time Robust Template Matching," in *Proceedings of The 13th British Machine Vision Conference*, Cardiff, United Kingdom, 2002, pp. 123–132.

[30] M. Niethammer, A. Tannenbaum, and S. Angenent, "Dynamic active contours for visual tracking," *IEEE Transactions on Automatic Control*, vol. 51, no. 4, pp. 562–579, Apr. 2006.

[31] H. Zhou, Y. Yuan, and C. Shi, "Object tracking using sift features and mean shift," *Computer Vision and Image Understanding*, vol. 113, no. 3, pp. 345 – 352, 2009, special Issue on Video Analysis.

[32] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 7, pp. 1409–1422, Jul. 2012.

[33] Z. Sun, G. Bebis, and R. Miller, "On-road vehicle detection: a review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 5, pp. 694–711, May 2006.

[34] L. D. Cañamero and J. Fredslund, "I show you how i like you: Emotional human-robot interaction through facial expression and tactile stimulation," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 31, no. 5, pp. 454–459, 2001.

[35] S. Shamsuddin, L. Ismail, H. Yussof, N. Ismarrubie Zahari, S. Bahari, H. Hashim, and A. Jaffar, "Humanoid robot nao: Review of control and motion exploration," in *Proceedings of IEEE International Conference on Control System*, Nov. 2011, pp. 511–516.

[36] S. Thrun, M. Bennewitz, W. Burgard, A. Cremers, F. Dellaert, D. Fox, D. Hahnel, C. Rosenberg, N. Roy, J. Schulte, and D. Schulz, "MINERVA: a second-generation museum tour-guide robot," in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 3, 1999, pp. 1999–2005 vol.3.

[37] H. Kozima, "Infanoid," in *Socially Intelligent Agents*, ser. Multiagent Systems, Artificial Societies, and Simulated Organizations, K. Dautenhahn, A. Bond, L. Cañamero, and B. Edmonds, Eds. Springer US, 2002, vol. 3, pp. 157–164.

[38] A. van Breemen, X. Yan, and B. Meerbeek, "iCat: An animated user-interface robot with personality," in *Proceedings of the 4th International Joint Conference on Autonomous Agents and Multiagent Systems*, ser. AAMAS '05. New York, NY, USA: ACM, 2005, pp. 143–144.

[39] J.-H. Oh, D. Hanson, W.-S. Kim, I. Y. Han, J.-Y. Kim, and I.-W. Park, "Design of android type humanoid robot albert hubo," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2006, pp. 1428–1433.

[40] I. Lutkebohle, F. Hegel, S. Schulz, M. Hackel, B. Wrede, S. Wachsmuth, and G. Sagerer, "The bielefeld anthropomorphic robot head flobi," in *Proceedings of IEEE International Conference on Robotics and Automation*, May 2010, pp. 3384–3391.

[41] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. The MIT Press, 2005.

[42] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Computing Surveys*, vol. 38, no. 4, Dec. 2006.

[43] X. Li, W. Hu, C. Shen, Z. Zhang, A. Dick, and A. V. D. Hengel, "A survey of appearance models in visual object tracking," *ACM Transactions on Intelligent Systems and Technolog*, vol. 4, no. 4, pp. 58:1–58:48, Oct. 2013.

[44] T. W. Fong, I. Nourbakhsh, and K. Dautenhahn, "A survey of socially interactive robots: Concepts, design, and applications," Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-02-29, Dec. 2002.

[45] R. Bajcsy, "Active perception," *Proceedings of the IEEE*, vol. 76, no. 8, pp. 966–1005, Aug. 1988.

[46] W. Theimer and H. Mallot, "Phase-based binocular vergence control and depth reconstruction using active vision," *CVGIP: Image Understanding*, vol. 60, no. 3, pp. 343 – 358, 1994.

[47] W. Klarquist and A. Bovik, "FOVEA: a foveated vergent active stereo vision system for dynamic three-dimensional scene recovery," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 5, pp. 755–770, 1998.

[48] M. Björkman and J.-O. Eklundh, "Real-time epipolar geometry estimation of binocular stereo heads," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 3, pp. 425–432, Mar. 2002.

[49] M. Björkmann and J.-O. Eklundh, "Attending, foveating and recognizing objects in real world scenes," in *Proceedings of the British Machine Vision Conference*. BMVA Press, 2004.

[50] M. Björkman and D. Kragic, "Combination of foveal and peripheral vision for object recognition and pose estimation," in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 5, Apr. 2004, pp. 5135–5140.

[51] B. Rasolzadeh, M. Bjorkman, K. Huebner, and D. Kragic, "An active vision system for detecting, fixating and manipulating objects in the real world," *International Journal of Robotics Research*, vol. 29, no. 2-3, pp. 133–154, Feb. 2010.

[52] D. Wan and J. Zhou, "Stereo vision using two PTZ cameras," *Computer Vision and Image Understanding*, vol. 112, no. 2, pp. 184–194, 2008.

[53] J. Aloimonos, I. Weiss, and A. Bandyopadhyay, "Active vision," *International Journal of Computer Vision*, vol. 1, pp. 333–356, 1988.

[54] D. H. Ballard and C. M. Brown, "Principles of animate vision," *CVGIP: Image Understanding*, vol. 56, no. 1, pp. 3–21, 1992.

[55] E. Krotov and R. Bajcsy, "Active vision for reliable ravsing: cooperating, focus, stereo, and vergence," *International Journal of Computer Vision*, vol. 11, no. 2, p. 187–203, 1993.

[56] A. D. Bagdanov, A. del Bimbo, W. Nunziati, and F. Pernici, "A reinforcement learning approach to active camera foveation," in *Proceedings of the 4th ACM international workshop on Video surveillance and sensor networks*, ser. VSSN '06. New York, NY, USA: ACM, 2006, pp. 179–186.

[57] V. Javier Traver and A. Bernardino, "A review of log-polar imaging for visual perception in robotics," *Robotics and Autonomous Systems*, vol. 58, no. 4, pp. 378–398, Apr. 2010.

[58] C. Atkeson, J. Hale, F. Pollick, M. Riley, S. Kotosaka, S. Schaul, T. Shibata, G. Tevatia, A. Ude, S. Vijayakumar, E. Kawato, and M. Kawato, "Using humanoid robots to study human behavior," *Intelligent Systems and their Applications*, vol. 15, no. 4, pp. 46–56, jul/aug 2000.

[59] A. Ude, C. Gaskett, and G. Cheng, "Foveated vision systems with two cameras per eye," in *Proceedings of IEEE International Conference on Robotics and Automation*, May 2006, pp. 3457–3462.

[60] E. Rivlin and H. Rotstein, "Control of a camera for active vision: Foveal vision, smooth tracking and saccade," *International Journal of Computer Vision*, vol. 39, no. 2, pp. 81–96, Sep. 2000.

[61] X. Zhang and H. Wakamatsu, "A unified adaptive oculomotor control model," *International Journal of Adaptive Control and Signal Processing*, vol. 15, no. 7, pp. 697–713, 2001.

[62] Y. Song and X. Zhang, "An active binocular integrated system for intelligent robot vision," in *Proceedings of IEEE International Conference on Intelligence and Security Informatics*, Jun. 2012, pp. 48–53.

[63] S. Das and N. Ahuja, "A comparative study of stereo, vergence, and focus as depth cues for active vision," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Jun. 1993, pp. 194–199.

[64] M. Bjorkman and D. Kragic, "Combination of foveal and peripheral vision for object recognition and pose estimation," in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 5, may 2004, pp. 5135–5140.

[65] A. Mian, "Realtime face detection and tracking using a single pan, tilt, zoom camera," in *Proceedings of 23rd International Conference Image and Vision Computing New Zealand*, nov. 2008, pp. 1–6.

[66] K. Ciuffreda and B. Tannen, *Eye movement basics for the clinician*. Mosby, 1995.

[67] F. Wilbers, "Human-like stabilisation of a robot eye," Master's thesis, Delft University of Technology, 2008.

[68] E. Maini, L. Manfredi, C. Laschi, and P. Dario, "Bioinspired velocity control of fast gaze shifts on a robotic anthropomorphic head," *Autonomous Robots*, vol. 25, pp. 37–58, 2008.

[69] M. Sugathadasa, W. Dayawansa, and C. Martin, "Control of pursuit eye movement," in *Proceedings of the 39th IEEE Conference on Decision and Control*, vol. 2, 2000, pp. 1793–1798.

[70] S. Hampl, V. Cimalla, T. Polster, and M. Hoffmann, "Aln-based piezoelectric bimorph microgenerator utilizing low-level non-resonant excitation," in *Proceedings of SPIE the International Society for Optical Engineering*, 2011, pp. 1–11.

[71] J. Crawford and T. Vilis, "Axes of eye rotation and listing's law during rotations of the head," *Neurophysiology*, vol. 65, no. 3, pp. 407–423, March 1991.

[72] D. Angelaki, "Eyes on target: what neurons must do for the vestibuloocular reflex during linear motion," *Neurophysiology*, vol. 92, no. 1, pp. 20–35, 2004.

[73] H. MS, "Dictionary of eye terminology," *Archives of Opthalmology*, vol. 109, no. 9, p. 1208, 1991.

[74] D. Guitton and M. Volle, "Gaze control in humans: eye-head coordination during orienting move- ments to targets within and beyond the oculomotor range," *Journal of Neurophysiology*, vol. 58, p. 496–508, 1987.

[75] G. Gauthier, J.-L. Vercher, and J. Blouin, "Integrating reflexes and voluntary behaviours: Coordination and adaptation controls in man," in *Human and Machine Perception*, V. Cantoni, V. Gesù, A. Setti, and D. Tegolo, Eds. Springer US, 1997, pp. 189–205.

[76] S. Coren, L. Ward, and J. Enns, *Sensation and perception*. Wiley, 2003.

[77] T. Imai, "Interaction of the body, head, and eyes during walking and turning," *Experimental Brain Research*, vol. 136, no. 1, pp. 1–18, 2008.

[78] A. Lenz, T. Balakrishnan, A. G. Pipe, and C. Melhuish, "An adaptive gaze stabilization controller inspired by the vestibulo-ocular reflex," *Bioinspiration Biomimetics*, vol. 3, no. 3, 2008.

[79] N. Pugeault, F. Wörgötter, and N. Krüger, "Visual primitives: Local, condensed, semantically rich visual descriptors and their applications in robotics." *International Journal of Humanoid Robotics*, vol. 7, no. 3, pp. 379–405, 2010.

[80] T. Drummond and R. Cipolla, "Real-time visual tracking of complex structures," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 932–946, Jul. 2002.

[81] A. Davison, I. Reid, N. Molton, and O. Stasse, "MonoSLAM: Real-time single camera SLAM," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052–1067, Jun. 2007.

[82] R. Szeliski, "Image alignment and stitching: a tutorial," *Foundations and Trends in Computer Graphics and Vision*, vol. 2, no. 1, pp. 1–104, Jan. 2006.

[83] R. Szeliski and J. Coughlan, "Spline-based image registration," *International Journal of Computer Vision*, vol. 22, pp. 199–218, 1997.

[84] M. Brown, D. Burschka, and G. Hager, "Advances in computational stereo," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 8, pp. 993–1008, Aug. 2003.

[85] M. Pollefeys, L. Van Gool, M. Vergauwen, F. Verbiest, K. Cornelis, J. Tops, and R. Koch, "Visual modeling with a hand-held camera," *International Journal of Computer Vision*, vol. 59, pp. 207–232, 2004.

[86] N. Snavely, S. Seitz, and R. Szeliski, "Modeling the world from internet photo collections," *International Journal of Computer Vision*, vol. 80, pp. 189–210, 2008.

[87] R. Brunelli and T. Poggio, "Face recognition: features versus templates," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, no. 10, pp. 1042–1052, Oct. 1993.

[88] D. Swets and J. Weng, "Using discriminant eigenfeatures for image retrieval," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 8, pp. 831–836, Aug. 1996.

[89] R. Szeliski, *Computer Vision: Algorithms and Applications*, 1st ed. New York, NY, USA: Springer-Verlag New York, Inc., 2010.

[90] F. Attneave, "Some informational aspects of visual perception," *Psychological Review*, vol. 61, no. 3, pp. 183–193, May 1954.

[91] C. Schmid, R. Mohr, and C. Bauckhage, "Evaluation of interest point detectors," *International Journal of Computer Vision*, vol. 37, no. 2, pp. 151–172, Jun. 2000.

[92] J. Li and N. M. Allinson, "A comprehensive review of current local features for computer vision," *Neurocomput.*, vol. 71, no. 10-12, pp. 1771–1787, Jun. 2008.

[93] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1615–1630, oct. 2005.

[94] H. P. Moravec, "Obstacle avoidance and navigation in the real world by a seeing robot rover," Ph.D. dissertation, Stanford University, Stanford, CA, USA, 1980, aAI8024717.

[95] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proceedings of of the 4th Alvey Vision Conference*, 1988, pp. 147–151.

[96] J. Shi and C. Tomasi, "Good features to track," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Jun. 1994, pp. 593–600.

[97] M. Zuliani, C. Kenney, and B. Manjunath, "A mathematical comparison of point detectors," in *Conference on Computer Vision and Pattern Recognition Workshop*, Jun. 2004, p. 172.

[98] D. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the 7th IEEE International Conference on Computer Vision*, vol. 2, 1999, pp. 1150–1157.

[99] Y. Ke and R. Sukthankar, "PCA-SIFT: a more distinctive representation for local image descriptors," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, june-2 july 2004, pp. 506–513.

[100] P.-E. Forssen and D. Lowe, "Shape descriptors for maximally stable extremal regions," in *Proceedings of IEEE 11th International Conference on Computer Vision*, oct. 2007, pp. 1–8.

[101] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (SURF)," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, Jun. 2008.

[102] E. Rosten, R. Porter, and T. Drummond, "Faster and better: A machine learning approach to corner detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 1, pp. 105–119, jan 2010.

[103] E. Mair, G. D. Hager, D. Burschka, M. Suppa, and G. Hirzinger, "Adaptive and generic corner detection based on the accelerated segment test," in *Proceedings of the 11th European conference on Computer vision: Part II*, ser. ECCV'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 183–196.

[104] S. Leutenegger, M. Chli, and R. Siegwart, "BRISK: Binary robust invariant scalable keypoints," in *Proceedings of IEEE International Conference on Computer Vision*, nov. 2011, pp. 2548–2555.

[105] M. Ambai and Y. Yoshida, "CARD: Compact and real-time descriptors," in *Proceedings of IEEE International Conference on Computer Vision*, nov. 2011, pp. 97–104.

[106] M. Brown, R. Szeliski, and S. Winder, "Multi-image matching using multi-scale oriented patches," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, Jun. 2005, pp. 510–517.

[107] E. Psarakis and G. Evangelidis, "An enhanced correlation-based method for stereo correspondence with subpixel accuracy," in *Proceedings of 10th IEEE International Conference on Computer Vision*, vol. 1, oct. 2005, pp. 907–912.

[108] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "From contours to regions: An empirical evaluation," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2009, pp. 2294–2301.

[109] L. S. Davis, "A survey of edge detection techniques," *Computer Graphics and Image Processing*, vol. 4, no. 3, pp. 248–270, 1975.

[110] R. Rakesh, P. Chaudhuri, and C. Murthy, "Thresholding in edge detection: a statistical approach," *IEEE Transactions on Image Processing*, vol. 13, no. 7, pp. 927–936, july 2004.

[111] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679–698, nov. 1986.

[112] R. O. Duda and P. E. Hart, "Use of the hough transformation to detect lines and curves in pictures," *Communications of the ACM*, vol. 15, no. 1, pp. 11–15, Jan. 1972.

[113] R. Collins and R. Weiss, "Vanishing point calculation as a statistical inference on the unit sphere," in *Proceedings of the 3th International Conference on Computer Vision*, Dec. 1990, pp. 400–403.

[114] J. du Buf, M. Kardan, and M. Spann, "Texture feature performance for image segmentation," *Pattern Recognition*, vol. 23, no. 3–4, pp. 291–309, 1990.

[115] T. Ojala, M. Pietikainen, and D. Harwood, "Performance evaluation of texture measures with classification based on kullback discrimination of distributions," in *Proceedings of the 12th IAPR International Conference on Pattern Recognition*, vol. 1, Oct. 1994, pp. 582–585.

[116] T. Ojala, M. Pietikäinen, and D. Harwood, "A comparative study of texture measures with classification based on featured distributions," *Pattern Recognition*, vol. 29, no. 1, pp. 51–59, 1996.

[117] T. Tuytelaars and K. Mikolajczyk, "Local invariant feature detectors: a survey," *Foundations and Trends in Computer Graphics and Vision*, vol. 3, no. 3, pp. 177–280, Jul. 2008.

[118] B. Triggs, "Detecting keypoints with stable position, orientation, and scale under illumination changes," in *Computer Vision - ECCV 2004*, ser. Lecture Notes in Computer Science, T. Pajdla and J. Matas, Eds. Springer Berlin Heidelberg, 2004, vol. 3024, pp. 100–113.

[119] T. Ahonen, A. Hadid, and M. Pietik?inen, "Face description with local binary patterns: Application to face recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 12, pp. 2037–2041, 2006.

[120] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, 2001, pp. 511–518.

[121] Z. Kalal, J. Matas, and K. Mikolajczyk, "Online learning of robust object detectors during unstable tracking," in *Proceedings of IEEE 12th International Conference on Computer Vision Workshops*, 2009, pp. 1417–1424.

[122] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 603–619, 2002.

[123] J. Ning, L. Zhang, D. Zhang, and C. Wu, "Robust mean-shift tracking with corrected background-weighted histogram," *Computer Vision, IET*, vol. 6, no. 1, pp. 62 –69, january 2012.

[124] S. Hinterstoisser, C. Cagniart, S. Ilic, P. Sturm, N. Navab, P. Fua, and V. Lepetit, "Gradient response maps for real-time detection of textureless objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 5, pp. 876–888, May 2012.

[125] K. van de Sande, T. Gevers, and C. Snoek, "Evaluating color descriptors for object and scene recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1582–1596, Sep. 2010.

[126] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *Proceedings of the 9th European conference on Computer Vision - Volume Part I*, ser. ECCV'06. Berlin, Heidelberg: Springer-Verlag, 2006, pp. 430–443.

[127] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, 2001, pp. I–511 – I–518 vol.1.

[128] Z. Kalal, J. Matas, and K. Mikolajczyk, "P-N learning: Bootstrapping binary classifiers by structural constraints," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 0, pp. 49–56, 2010.

[129] C. Bao, Y. Wu, H. Ling, and H. Ji, "Real time robust L1 tracker using accelerated proximal gradient approach," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2012, pp. 1830–1837.

[130] R. Collins, Y. Liu, and M. Leordeanu, "Online selection of discriminative tracking features," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1631–1643, oct. 2005.

[131] J. Ning, L. Zhang, D. Zhang, and C. Wu, "Robust object tracking using joint color-texture histogram," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 23, no. 07, pp. 1245–1263, 2009.

[132] D. A. Klein, D. Schulz, S. Frintrop, and A. B. Cremers, "Adaptive real-time video-tracking for arbitrary objects," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010, pp. 772–777.

[133] C. Rother, V. Kolmogorov, and A. Blake, "GrabCut: interactive foreground extraction using iterated graph cuts," in *ACM Transactions on Graphics*. New York, NY, USA: ACM, 2004, pp. 309–314.

[134] Z. Zivkovic, "Improved adaptive Gaussian mixture model for background subtraction," in *Proceedings of the 17th International Conference on Pattern Recognition*, vol. 2, aug. 2004, pp. 28–31.

[135] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, Jun. 2005, pp. 886–893.

[136] M.-H. Y. Kaihua Zhang, Lei Zhang, "Real-time compressive tracking," in *Proceedings of the 12th European Conference on Computer Vision*, 2012, pp. 864–877.

[137] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Comput. Surv.*, vol. 38, no. 4, Dec. 2006.

[138] S. Salti, A. Cavallaro, and L. Di Stefano, "Adaptive appearance modeling for video tracking: Survey and evaluation," *IEEE Transactions on Image Processing*, vol. 21, no. 10, pp. 4334–4348, Oct. 2012.

[139] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, "Incremental learning for robust visual tracking," *International Journal of Computer Vision*, vol. 77, no. 1-3, pp. 125–141, May 2008.

[140] F. Jurie and M. Dhome, "Real time robust template matching," in *Proceedings of British Machine Vision Conference*, P. L. Rosin and A. D. Marshall, Eds. Cardiff, Royaume-Uni: British Machine Vision Association, Sep. 2002, pp. 123–132.

[141] H. Zhou, Y. Yuan, and C. Shi, "Object tracking using sift features and mean shift," *Computer Vision and Image Understanding*, vol. 113, no. 3, pp. 345 – 352, 2009.

[142] W. Kloihofer and M. Kampel, "Interest point based tracking," in *Proceedings of 20th International Conference on Pattern Recognition*, Aug., pp. 3549–3552.

[143] B. D. Lucas, "Generalized image matching by the method of differences," Ph.D. dissertation, Robotics Institute, Carnegie Mellon University, July 1984.

[144] S. Benhimane and E. Malis, "Real-time image-based tracking of planes using efficient second-order minimization," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 1, Sept.-2 Oct. 2004, pp. 943–948 vol.1.

[145] S. Avidan, "Support vector tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 8, pp. 1064–1072, Aug.

[146] H. Grabner, C. Leistner, and H. Bischof, "Semi-supervised on-line boosting for robust tracking," in *Proceedings of the 10th European Conference on Computer Vision: Part I*, ser. ECCV '08. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 234–247.

[147] B. Babenko, M.-H. Yang, and S. Belongie, "Visual tracking with online multiple instance learning," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, June2009, pp. 983–990.

[148] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 7, pp. 1409–1422, 2012.

[149] S. Hare, A. Saffari, and P. H. S. Torr, "Struck: Structured output tracking with kernels," in *Proceedings of IEEE International Conference on Computer Vision*, Nov., pp. 263–270.

[150] S. Zhang, H. Yao, X. Sun, and X. Lu, "Sparse coding based visual tracking: Review and experimental comparison," *Pattern Recognition*, vol. 46, no. 7, pp. 1772–1788, 2013.

[151] H. Li, C. Shen, and Q. Shi, "Real-time visual tracking using compressive sensing," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, June, 2011, pp. 1305–1312.

[152] M. Björkman and D. Kragic, "Active 3D scene segmentation and detection of unknown objects," in *IEEE conference on Robotics and Automation*, 2010, pp. 3114–3120.

[153] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 11, pp. 1254–1259, 1998.

[154] G. Kootstra, N. Bergström, and D. Kragic, "Using symmetry to select fixation points for segmentation," in *Proceedings of the International Conference on Pattern Recognition*, 2010.

[155] M. Rudinac and P. Jonker, "A fast and robust descriptor for multiple-view object recognition," in *Proceedings of 11th International Conference on Control Automation Robotics Vision*, 2010, pp. 2166–2171.

[156] J. Mooser, S. You, and U. Neumann, "Real-time object tracking for augmented reality combining graph cuts and optical flow," in *Proceedings of 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, 2007, pp. 145–152.

[157] G. R. Bradski, "Computer vision face tracking for use in a perceptual user interface," 1998.

[158] X. Wang, M. Rudinac, and P. P. Jonker, "Robust online segmentation of unknown objects for mobile robots," in *7th International Conference on Computer Vision Theory and Applications*, 2012.

[159] X. Hou and L. Zhang, "Saliency detection: A spectral residual approach," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–8.

[160] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide-baseline stereo from maximally stable extremal regions," *Image Vision Computing*, vol. 22, no. 10, pp. 761–767, 2004.

[161] D. Tax, "One-class classification," phd, Delft University of Technology, Delft, Jun. 2001.

[162] N. Herodotou, K. Plataniotis, and A. Venetsanopoulos, "A color segmentation scheme for object-based video coding," in *IEEE Symposium on Advances in Digital Filtering and Signal Processing*, 1998, pp. 25–29.

[163] A. V. Vezhnevets, ""GrowCut"-Interactive Multi-Label N-D Image Segmentation By Cellular," 2005.

[164] V. Lepetit and P. Fua, "Keypoint recognition using randomized trees," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, pp. 1465–1479, 2006.

[165] Y. Boykov and M.-P. Jolly, "Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images," in *Proceedings of the 8th IEEE International Conference on Computer Vision*, vol. 1, 2001, pp. 105–112.

[166] J. J. Gibson, *The Ecological approach to visual perception*. Lawrence Erlbaum Associates, Sep. 1986.

[167] C. von Hofsten, "Eye–hand coordination in the newborn," *Developmental Psychology*, vol. 18, no. 3, pp. 450–461, may 1982.

[168] A. Bullinger, "Space, the organism and objects, their cognitive elaboration in the infant," in *Spatially Oriented Behavior*, A. Hein and M. Jeannerod, Eds. Springer New York, 1983, pp. 215–222.

[169] I. Howard, *Perceiving in Depth*, 1st ed. New York: Oxford University Press, 2012.

[170] W. Epstein and S. Rogers, Eds., *Perception of Space and Motion*. Academic press limited, 1995.

[171] (2012). [Online]. Available: http://kin450-neurophysiology.wikispaces.com/Vergence

[172] M. Z. Brown, D. Burschka, and G. D. Hager, "Advances in computational stereo," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 8, pp. 993–1008, 2003.

[173] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *International Journal of Computer Vision*, vol. 47, no. 1-3, pp. 7–42, Apr. 2002.

[174] G. Aragon-Camarasa, H. Fattah, and J. P. Siebert, "Towards a unified visual framework in a binocular active robot vision system," *Robotics and Autonomous Systems*, vol. 58, no. 3, pp. 276–286, 2010.

[175] R. Wodnicki, G. Roberts, and M. Levine, "A foveated image sensor in standard CMOS technology," in *Proceedings of the IEEE Conference on Custom Integrated Circuits*, 1995, pp. 357–360.

[176] Z. Wang and A. Bovik, "Embedded foveation image coding," *IEEE Transactions on Image Processing*, vol. 10, no. 10, pp. 1397–1410, 2001.

[177] C. Weber and J. Triesch, "Implementations and implications of foveated vision," pp. 75–85, 2009.

[178] A. Bernardino and J. Santos-Victor, "A binocular stereo algorithm for log-polar foveated systems," in *Proceedings of the Second International Workshop on Biologically Motivated Computer Vision*, ser. BMCV '02. London, UK, UK: Springer-Verlag, 2002, pp. 127–136.

[179] M. Björkman and J.-O. Eklundh, "Foveated figure-ground segmentation and its role in recognition," in *Proceedings of the British Machine Vision Conference*, W. F. Clocksin, A. W. Fitzgibbon, and P. H. S. Torr, Eds. British Machine Vision Association, 2005, pp. 819–828.

[180] S. Gould, J. Arfvidsson, A. Kaehler, M. Messner, G. Bradski, P. Baumstarck, S. Chung, and A. Y. Ng, "Peripheral-foveal vision for real-time object recognition and tracking in video," in *In International Joint Conference on Artificial Intelligence*, 2007, pp. 2115–2121.

[181] H. Sahabi and A. Basu, "Analysis of error in depth perception with vergence and spatially varying sensing," *Computer Vision and Image Understanding*, vol. 63, no. 3, pp. 447–461, 1996.

[182] P. Remagnino, J. Illingworth, J. Kittler, and J. Matas, "Intentional control of camera look direction and viewpoint in an active vision system," *Image and Vision Computing*, vol. 13, no. 2, pp. 79–88, 1995.

[183] E. Rivlin and H. Rotstein, "Control of a camera for active vision: Foveal vision, smooth tracking and saccade," *International Journal of Computer Vision*, vol. 39, pp. 81–96, 2000.

[184] M. Li, "Kinematic calibration of an active head-eye system," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 1, pp. 153–158, 1998.

[185] H. Chen, "A screw motion approach to uniqueness analysis of head-eye geometry," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1991, pp. 145–151.

[186] C. Angle, "Inv. talk at euron meeting," no. 1. Amsterdam, March 2004.

[187] R. Y. Tsai and R. K. Lenz, "A new technique for fully autonomous and efficient 3d robotics hand-eye calibration," in *Proceedings of the 4th International Symposium on Robotics Research*. Cambridge, MA, USA: MIT Press, 1988, pp. 287–297.

[188] J. C. K. Chou and M. Kamel, "Finding the position and orientation of a sensor on a robot manipulator using quaternions," *The International Journal of Robotics Research*, vol. 10, no. 3, pp. 240–254, Jun. 1991.

[189] C.-C. Wang, "Extrinsic calibration of a vision sensor mounted on a robot," *IEEE Transactions on Robotics and Automation*, vol. 8, no. 2, pp. 161–175, Apr. 1992.

[190] K. Daniilidis, "Hand-eye calibration using dual quaternions," *International Journal of Robotics Research*, vol. 18, pp. 286–298, 1998.

[191] N. Andreff, R. Horaud, and B. Espiau, "On-line hand-eye calibration," in *Proceedings of Second International Conference on 3-D Digital Imaging and Modeling*, 1999, pp. 430–436.

[192] K. Strobl and G. Hirzinger, "Optimal hand-eye calibration," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2006, pp. 4647–4653.

[193] R. Horaud and F. Dornaika, "Hand-eye calibration," *The International Journal of Robotics Research*, vol. 14, no. 3, pp. 195–210, Jun. 1995.

[194] J. Hart, B. Scassellati, and S. Zucker, "Epipolar geometry for humanoid robotic heads," in *Cognitive Vision*, ser. Lecture Notes in Computer Science, B. Caputo and M. Vincze, Eds. Springer Berlin Heidelberg, 2008, vol. 5329, pp. 24–36.

[195] M. Sapienza, M. Hansard, and R. Horaud, "Real-time Visuomotor Update of an Active Binocular Head," *Autonomous Robots*, vol. 34, no. 1, pp. 33–45, Jan. 2013.

[196] P. H. S. Torr and D. W. Murray, "The development and comparison of robust methods for estimating the fundamental matrix," *International Journal of Computer Vision*, vol. 24, pp. 271–300, 1997.

[197] P. J. Huber, *Robust Statistics*. New York: John Wiley and Sons, 1981.

[198] M. J. Harker and P. L. O'Leary, "First order geometric distance (the myth of sampsonus)," in *Proceedings of the British Machine Vision Conference*. BMVA Press, 2006, pp. 10.1–10.10.

[199] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Communicating ACM*, vol. 24, no. 6, pp. 381–395, Jun. 1981.

[200] Z. Zhang, "Determining the epipolar geometry and its uncertainty: A review," *International Journal of Computer Vision*, vol. 27, no. 2, pp. 161–195, 1998.

[201] P. Torr and A. Zisserman, "MLESAC: A new robust estimator with application to estimating image geometry," *Computer Vision and Image Understanding*, vol. 78, no. 1, pp. 138–156, 2000.

[202] P. Torr, "Bayesian model estimation and selection for epipolar geometry and generic manifold fitting," *International Journal of Computer Vision*, vol. 50, no. 1, pp. 35–61, 2002.

[203] X. Armangué and J. Salvi, "Overall view regarding fundamental matrix estimation," *Image and Vision Computing*, vol. 21, no. 2, pp. 205 – 220, 2003.

[204] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. New York, NY, USA: Cambridge University Press, 2003.

[205] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[206] H. Hirschmuller, "Stereo processing by semiglobal matching and mutual information," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 2, pp. 328–341, Feb. 2008.

[207] C. Bandera and P. Scott, "Foveal machine vision systems," in *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, vol. 2, 1989, pp. 596–599.

[208] M. E. Hansard and R. Horaud, "Cyclorotation models for eyes and cameras." *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 40, no. 1, pp. 151–161, 2010.

[209] J. Batista, P. Peixoto, and H. Araujo, "Real-time vergence and binocular gaze control," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3, Sep. 1997, pp. 1348–1354 vol.3.

[210] W. N. Klarquist and A. C. Bovik, "Fovea: A foveated vergent active stereo vision system for dynamic three-dimensional scene recovery," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 5, pp. 755–769, 1998.

[211] M. Björkman and D. Kragic, "Combination of foveal and peripheral vision for object recognition and pose estimation," *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 5, pp. 5135–5140, 2004.

[212] J. Salvi, "An approach to coded structured light to obtain three dimensional information," Ph.D. dissertation, Universitat de Girona, Departament of Electronics, Information and Automation, 1997.

[213] V. Varadarajan, *Lie groups, Lie algebras, and their representations*, 1st ed. Prentice-Hall, Inc., Inglewood Cliffs, NJ, Jan 1974.

[214] J. Weng, P. Cohen, and M. Herniou, "Camera calibration with distortion models and accuracy evaluation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 10, pp. 965–980, 1992.

[215] J. Heikkila and O. Silven, "A four-step camera calibration procedure with implicit image correction," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1997, pp. 1106–1112.

[216] R. Y. Tsai, "Radiometry," L. B. Wolff, S. A. Shafer, and G. Healey, Eds. USA: Jones and Bartlett Publishers, Inc., 1992, ch. A Versatile Camera Calibration Technique for High-accuracy 3D Machine Vision Metrology Using Off-the-shelf TV Cameras and Lenses, pp. 221–244.

[217] T. A. Clarke and J. G. Fryer, "The development of camera calibration methods and models," *The Photogrammetric Record*, vol. 16, no. 91, pp. 51–66, 1998.

[218] J. Salvi, X. Armangué, and J. Batlle, "A comparative review of camera calibrating methods with accuracy evaluation," *Pattern Recognition*, vol. 35, no. 7, pp. 1617–1635, 2002.

[219] C. DeTar, "Levenberg-marquardt method," http://www.physics.utah.edu/~detar/phys6720/handouts/curve_fit/curve_fit/node7.html, 23th November, 2009.

[220] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[221] J. Gall, N. Razavi, and L. Gool, "An introduction to random forests for multi-class object detection," in *Outdoor and Large-Scale Real-World Scene Analysis*, ser. Lecture Notes in Computer Science, F. Dellaert, J.-M. Frahm, M. Pollefeys, L. Leal-Taixé, and B. Rosenhahn, Eds. Springer Berlin Heidelberg, 2012, vol. 7474, pp. 243–263.

Appendix

Key to fully understand our active vision system is apprehension of the mathematics used in this thesis. It provides the foundation of all vision algorithm implementations. The appendices are dedicated to the introduction of the mathematical notations as well as the detailed mathematical models. They also gives a number of hands-on tips on useful mathematical tools.

It is worth noting that all the mathematical notations follow the definitions and representations of Lie Groups [213].

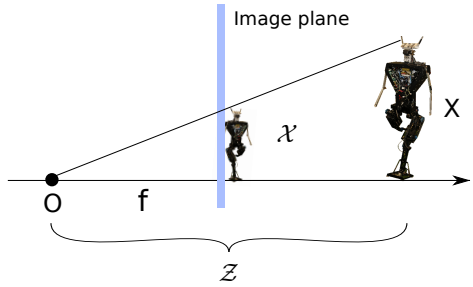


Figure 1: Pinhole model

Geometric model of image formation

Image formation

The image formation process can be regarded as a projective transformation from a 3-dimensional to a 2-dimensional projective space. This section will illustrate how this process works.

The simplest approximation of a thin lens camera is a pinhole camera model, which is shown in Figure 1. The pinhole aperture of the camera,

through which all projection lines must pass, is assumed to be infinitively small, a point. The image point (x, y) and 3D world point (X, Y, Z) are related through the ideal perspective projection

$$x = f\frac{X}{Z}, y = f\frac{Y}{Z} \tag{6.1}$$

Here f is the focal length, the distance from optical center to the principal point. Depending on the image plane position that is in front of the optical center or behind the optical center, f is either positive or negative.

In homogenous coordinates, this relationship can be written as

$$Z\tilde{\mathbf{x}} = Z\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = K_f\tilde{\mathbf{X}} \tag{6.2}$$

Since Z is usually unknown, we may write it as an arbitrary positive scalar $\lambda \in \mathbb{R}_+$.

In practice, when capturing images with a digital camera, there exists a relationship between the image coordinate frame and the camera coordinate frame, which is

$$\lambda\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} s_x & s_\theta & x_0 \\ 0 & s_y & y_0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}\begin{pmatrix} X_C \\ Y_C \\ Z_C \\ 1 \end{pmatrix} \tag{6.3}$$

In which x_0 and y_0 are the x, y coordinates of the principal point in pixels. fs_x is the unit length in horizontal pixels and fs_y is the unit length in vertical pixels. fs_θ is the skew of the pixel, often close to 0. These are also called intrinsic parameters, which describe the optical and internal geometry of the camera and define the relationship between camera coordinate frame and image coordinate frame.

The relationship of the 3D world coordinate and the camera coordinate follows the rigid transformation

$$\begin{pmatrix} X_C \\ Y_C \\ Z_C \\ 1 \end{pmatrix} = \begin{pmatrix} \mathbf{R} & \mathbf{T} \\ 0 & 1 \end{pmatrix}\begin{pmatrix} X_W \\ Y_W \\ Z_W \\ 1 \end{pmatrix} \tag{6.4}$$

in which rotation matrix **R** and translation vector **T** are called extrinsic parameters that define the relationship between the camera coordinate frame and the world coordinate frame.

Combining Equation 6.3 and Equation 6.4, we derive the geometric model of image formation as follows

$$\lambda\begin{pmatrix}x\\y\\1\end{pmatrix}=\begin{pmatrix}s_x & s_\theta & x_0\\0 & s_y & y_0\\0 & 0 & 1\end{pmatrix}\begin{pmatrix}f & 0 & 0\\0 & f & 0\\0 & 0 & 1\end{pmatrix}\begin{pmatrix}1 & 0 & 0 & 0\\0 & 1 & 0 & 0\\0 & 0 & 1 & 0\end{pmatrix}\begin{pmatrix}\mathbf{R} & \mathbf{T}\\0 & 1\end{pmatrix}\begin{pmatrix}X_W\\Y_W\\Z_W\\1\end{pmatrix} \tag{6.5}$$

$$\lambda\begin{pmatrix}x\\y\\1\end{pmatrix}=K[I|0]\begin{pmatrix}\mathbf{R} & \mathbf{T}\\0 & 1\end{pmatrix}\begin{pmatrix}X_W\\Y_W\\Z_W\\1\end{pmatrix} \tag{6.6}$$

written in another way as

$$\lambda\tilde{\mathbf{x}}=\mathbf{P}\tilde{\mathbf{X}} \tag{6.7}$$

Here **P** is called the projection matrix. Projective geometry is a fundamental mathematical model to transform a 3D space to a 2D projective space. Thus the effect of the camera is characterized by two stages: the transformation of the 3D world coordinates to the $Z = 1$ camera plane (the normalized coordinates) and the transformation of the normalized camera coordinates to the image coordinates.

Radial distortion

The perspective projection preserves the property that the straightness of a line is an invariant. However, in practice, the real cameras suffer from nonlinear distortions. The most important distortion is radial distortion. Radial distortion is performed along the radial direction from the center of distortion, causing an inward or outward displacement of a given image point from its ideal location. The negative radial displacement of the image points is referred to as the barrel distortion, while the positive radial displacement is referred to as the pincushion distortion [214]. The simplest effective model for such a distortion is:

$$x = x_d(1 + a_1 r^2 + a_2 r^4 + a_3 r^6) \tag{6.8}$$

$$y = y_d(1 + a_1 r^2 + a_2 r^4 + a_3 r^6) \tag{6.9}$$

where (x_d, y_d) are coordinates of the distorted points, $r^2 = x_d^2 + y_d^2$ and a_1, a_2, a_3 are camera parameters that model the amount of distortion.

Tangential distortion [215], resulting from the lens not being exactly parallel to the image plane, is another source of distortion. The tangential distortion is characterized by two additional parameters, p_1 and p_2

$$x = x_d(1 + a_1 r^2 + a_2 r^4 + a_3 r^6) + 2p_1 x_d y_d + p_2(r^2 + 2x_d^2) \tag{6.10}$$

$$y = y_d(1 + a_1 r^2 + a_2 r^4 + a_3 r^6) + 2p_1 x_d y_d + p_2(r^2 + 2y_d^2) \tag{6.11}$$

Camera calibration

Calibration establishes the relationship between a 3D scene point and its projected 2D image points, which is very essential when metric information is required. It can be classified into 3 groups with regard to different calibration methods used to estimate the parameters of a camera model.

- Linear techniques

  These techniques use the least squares method to obtain a transformation matrix which relates 3D points with their 2D projections. The advantage here is the simplicity of the model which consists of a simple and rapid calibration. The drawback is that linear techniques are useless for lens distortion modelling, entailing a rough accuracy of the system. Moreover, it is sometimes difficult to extract the parameters from the matrix due to the implicit calibration used.

- Non-linear optimization techniques

  By non-linear optimization techniques, camera parameters are usually obtained through iteration with the constraint of minimizing a determined function. The minimizing function is usually the distance between the imaged points and the modelled projections obtained by iterating. The advantage of these iterating techniques is that almost any model can be calibrated and the accuracy usually increases by increasing the number of iterations up to convergence. However, these techniques require a good initial guess in order to guarantee convergence.

- Two-step techniques

  These techniques combine a linear optimization to compute some of the parameters and, as a second step, the rest of the parameters are computed iteratively. These techniques permit a rapid calibration considerably reducing the number of iterations. Moreover, the convergence is nearly guaranteed due to the good linear initial guess obtained in the first step.

A widely used method was proposed by Tsai [216], which is based on a two-step technique modelling only radial lens distortion.

[217] and [218] gave a thorough survey of camera calibration methods and evaluated the performance. A good toolbox for calibration is the "Camera Calibration Toolbox for MATLAB", and the document can be also used as tutorial and reference on this topic.

Two view geometry

So far, we have described the image formation geometry with regard to a single non-moving camera. Given a sequence of images with corresponding feature points x_{ij}, taken by several cameras, i.e.

$$\lambda_{ij} x_{ij} = \mathbf{P}_i X_j, i = 1, 2, \ldots m, j = 1, 2, \ldots n \tag{6.12}$$

The camera matrices $\mathbf{P}_i$ determine the motion and the 3D points X_j the structure, under different assumptions on the intrinsic and extrinsic parameters. This is called the structure and motion problem. However, with only one camera, without knowing the actual size of the object, it is very difficult to deduce the true geometry of the scene. Therefore, a second camera is necessary.

Epipolar geometry

In our setup, we use two moving cameras. In this case, two view geometry needs to be studied. The most important property of two view geometry is the epipolar geometry, as shown in Figure 2.

As shown in the figure, the epipolar plane π intersects the image planes at epipolar line l_1 and l_2, while l_1 and l_2 intersect the image planes at epipolar poles e_1 and e_2. This shows that two projective lines can uniquely define the 3D position of a 3D point. In another way of saying, as soon as we have corresponding points x_1 and x_2 in left and right image, we can reconstruct a 3D point.

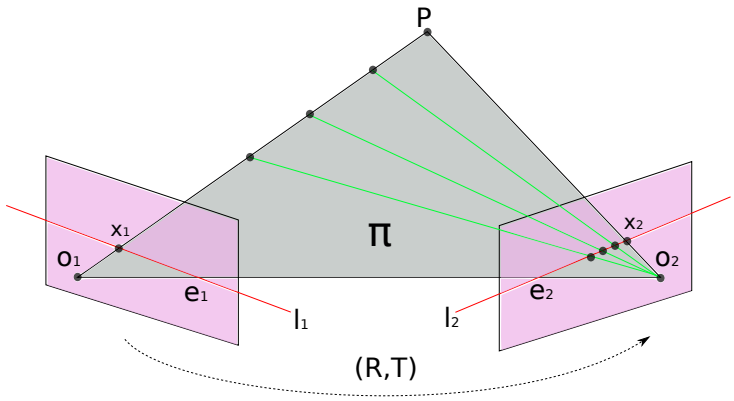


Figure 2: Epipolar geometry

Assuming that the cameras are calibrated, we denote the matrices that encompass the intrinsic parameters as K_1 and K_2. Therefore we have

$$\tilde{x}_2^T \mathbf{F} \tilde{x}_1 = 0 \tag{6.13}$$

$$\tilde{x}_2^T K_2^{-T} \mathbf{E} K_1^{-1} \tilde{x}_1 = 0 \tag{6.14}$$

in which $\tilde{x}_1$ and $\tilde{x}_2$ are the homogeneous coordinates of x_1 and x_2. The 3×3 matrix $\mathbf{F}$ is called the fundamental matrix, describing the relationship between corresponding images of the same scene that constraints where the projection of points from the scene can occur in both images. The 3×3 matrix $\mathbf{E}$ is called the essential matrix, which describes this correspondence in terms of normalized camera coordinates. It is worth noting that the epipolar geometry can be reconstructed without information on the intrinsic and extrinsic parameters.

In the last few years, several methods to estimate the fundamental matrix have been proposed, which can be classified into linear, iterative and robust methods [203]. All these methods are based on solving a homogeneous system of equations which can be deduced from Equation 6.13 and rewritten in the following way

$$\mathbf{A}\mathbf{f} = 0 \tag{6.15}$$

where

$$\mathbf{A} = \begin{pmatrix} u'_1u_1 & u'_1v_1 & u'_1 & v'_1u_1 & v'_1v_1 & v'_1 & u_1 & v_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u'_nu_n & u'_nv_n & u'_n & v'_nu_n & v'_nv_n & v'_n & u_n & v_n & 1 \end{pmatrix}$$

$$\mathbf{f} = \begin{pmatrix} F_{11} & F_{12} & F_{13} & F_{21} & F_{22} & F_{23} & F_{31} & F_{32} & F_{33} \end{pmatrix}$$

$x = (u, v)$ and $x' = (u', v')$ are corresponding points coordinates in left and right image. Eight corresponding points are sufficient to solve Equation 6.15.

Linear methods are quite good if the points are well located in the images; iterative methods can cope with some Gaussian noise in the localization of points, but they become really inefficient in the presence of outliers; robust methods use M-estimators [204], Least-Median-Squares (LMedS)[200], Random Sampling Consensus (RANSAC)[196], Maximum Likelihood Sample Consensus (MLESAC)[201] and Maximum a Posteriori Sample Consensus (MAPSAC) [202] in the presence of outliers and bad point localization. As a result it is able to cope with both discrepancy in the localization of points and false matching.

3D reconstruction

Triangulation

The problem of determining a point's 3D position from a set of corresponding image locations and known camera positions is known as triangulation. In some respect, this problem is also called the 3D reconstruction problem.

We can solve this problem by making use of Equation 6.7. We have

$$x_j = \frac{\mathbf{P}^j_{00}X + \mathbf{P}^j_{01}Y + \mathbf{P}^j_{02}Z + \mathbf{P}^j_{03}W}{\mathbf{P}^j_{20}X + \mathbf{P}^j_{21}Y + \mathbf{P}^j_{22}Z + \mathbf{P}^j_{23}W} \tag{6.16}$$

$$y_j = \frac{\mathbf{P}^j_{10}X + \mathbf{P}^j_{11}Y + \mathbf{P}^j_{12}Z + \mathbf{P}^j_{13}W}{\mathbf{P}^j_{20}X + \mathbf{P}^j_{21}Y + \mathbf{P}^j_{22}Z + \mathbf{P}^j_{23}W} \tag{6.17}$$

where x_j and y_j are a measured 2D point or feature location and $\begin{pmatrix} \mathbf{P}_{00} & \mathbf{P}_{01} & \cdots & \mathbf{P}_{33} \end{pmatrix}$ are known entries of related projective matrices.

With two corresponding feature points we can obtain 4 equations and in total we need to estimate 4 unknown parameters (X, Y, Z, W). More generally, this set of non-linear equations can be converted into a linear least squares problem by multiplying both sides of the denominator. The equation is best solved using singular value decomposition (SVD, looking for the smallest singular vector or eigenvector).

Stereo matching

Triangulation is the general form for solving 3D reconstruction. For stereo vision, the problem of finding correspondences is very important for the triangulation method to know which point in the left image corresponds to the one in the right image.

For a fixed stereo setup, the epipolar geometry can be obtained by camera calibration and stereo calibration. Both tools can be found in the MATLAB calibration toolbox. Alternatively, OpenCV provides functions that can also perform this task in a more automatic way.

As soon as the epipolar geometry is known, a more efficient algorithm can be obtained by first rectifying (i.e, warping) the input images so that corresponding horizontal scan lines are epipolar lines, which we call image rectification [205]. In this case, calculating the depth and 3D position can also be easier, which is shown in Figure 3.

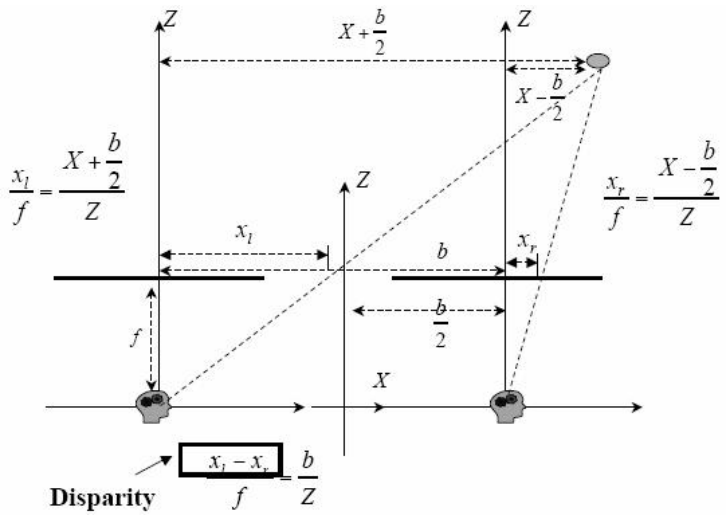


Figure 3: Stereo depth calculation with rectified images

And the function to calculate the depth is

$$Z = f\frac{b}{x_l - x_r} \tag{6.18}$$

Here we call $x_l - x_r$ the stereo disparity and there are many methods to find the correspondences. We categorize them into sparse matching methods and dense matching methods. [173] gives a thorough review on dense matching

methods and provides a benchmark for testing state-of-art dense matching algorithms.

It is worth noting that, for unknown geometry, image rectification does not work. In this case, feature based image matching is often applied to get good matching results. For subsequent frame matching, optical flow is often used.

Least squares minimization methods

Least squares minimization methods play an important role in solving many computer vision problems. For example, linear least squares methods are frequently used in homography estimation, fundamental matrix calculation, triangulation, camera calibration as well as image registration. Non-linear least squares methods are widely used in pose estimation, camera calibration, bundle adjustment, etc.

The general minimization problem is to minimize the cost function $g(\mathbf{p})$ over all the values of an unknown parameter $\mathbf{p}$, which equals to find $\mathbf{p}^*$ to ensure a global minimization for $g(\mathbf{p})$

$$arg \min_{\mathbf{p}^*} g(\mathbf{p}) \tag{6.19}$$

In the least squares minimization, the cost function is defined as the squared distance objective function

$$g(\mathbf{p}) = \frac{1}{2} \sum_i (\epsilon_i(\mathbf{p}))^2 \tag{6.20}$$

and in matrix form as

$$g(\mathbf{p}) = \frac{1}{2} \|\epsilon(\mathbf{p})\|^2 = \frac{1}{2} \epsilon(\mathbf{p})^T \epsilon(\mathbf{p}) \tag{6.21}$$

Commonly, $\epsilon_i(\mathbf{p})$ is defined as the residual difference between the measurement vector $\mathbf{b}$ and the prediction $f(\mathbf{a}, \mathbf{p})$, $\epsilon_i(\mathbf{p}) = f(a_i, \mathbf{p}) - b_i$ and the cost function becomes

$$g(\mathbf{p}) = \frac{1}{2} \sum_i (f(a_i, \mathbf{p}) - b_i)^2 \tag{6.22}$$

If $f(a_i, \mathbf{p})$ is linear with respect to unknown parameter $\mathbf{p}$, then the problem is a linear least squares minimization problem, otherwise it is a non-linear least squares minimization problem. Written into matrix form as

$$g(\mathbf{p}) = \frac{1}{2}(f(\mathbf{a},\mathbf{p}) - \mathbf{b})^T (f(\mathbf{a},\mathbf{p}) - \mathbf{b}) \tag{6.23}$$

Now we will first go into details of the linear least squares method.

Linear least squares minimization methods

Linear least squares solution to $\mathbf{A}x = 0$

As we can see from the previous discussion for solving the fundamental matrix problem; it is a typical $\mathbf{A}x = 0$ problem. With exact measurements and an exact mathematical model we can obtain an exact solution to the system. However, in the case of noise added to measurements, there exists no exact solution. Besides, with more matching points, it will become an overdetermined set of equations. In this case, a least squares solution can be found. The obvious solution $x = 0$ is not what we want since we seek a nonzero solution. Thus, a constraint $\|x\|^2 = 1$ is added. The search for a solution turns into

$$\min_x \|\mathbf{A}x\|^2, \text{subject to} \|x\|^2 = 1 \tag{6.24}$$

where $\|\cdot\|$ represents the vector norm. $\|\mathbf{A}x\|^2 = (\mathbf{A}x)^T\mathbf{A}x$ and $\|x\|^2 = x^T x$. Introducing the Lagrange multiplier λ this is equivalent to minimize the Lagrangian

$$\frac{\partial}{\partial x}(x^T\mathbf{A}^T\mathbf{A}x + \lambda(1 - x^T x)) = 0 \tag{6.25}$$

and the solution x is the last column of $\mathbf{V}$, where $\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^T$ is the singular value decomposition (SVD) of $\mathbf{A}$.

Linear least squares solution to $\mathbf{A}x = \mathbf{b}$

Assume a system of m linear equations,

$$\mathbf{A}x = b \tag{6.26}$$

for the unknown n-dimensional vector x. The $m \times n$ matrix $\mathbf{A}$ contains the coefficient of the equations, the m-dimensional vector b the data. Finding a solution x that is closest to fit $\mathbf{A}x = \mathbf{b}$ equals to find x such that $\|\mathbf{A}x - \mathbf{b}\|$ is minimized

$$\min_x \|\mathbf{A}x - \mathbf{b}\|^2 \tag{6.27}$$

If not all the components of b are null, the solution can be found by multiplying both sides of Equation 6.26 with $\mathbf{A}^T$. Then we have

$$\mathbf{A}^T\mathbf{A}x = \mathbf{A}^T\mathbf{b} \tag{6.28}$$

If $\mathbf{A}^T\mathbf{A}$ is invertible, then the solution is $x = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}$.

If matrix $\mathbf{A}$ is rank deficient, then a unique solution does not exist, which leads to an infinity solution minimizing Equation 6.26 with respect to x. In this case, the pseudo-inverse solution is given

$$x_{LS} = \mathbf{A}^+ b \tag{6.29}$$

where $\mathbf{A}^+$ is the pseudo-inverse of $\mathbf{A}$, defined by $\mathbf{A}^+ = \mathbf{V}\Sigma^+\mathbf{U}^T$. The matrix Σ^+ is related to Σ in the following way. If

$$\Sigma = diag(\sigma_1, \sigma_2, \ldots, \sigma_r, 0, \ldots, 0)$$

then

$$\Sigma^+ = diag(\sigma_1^{-1}, \sigma_2^{-1}, \ldots, \sigma_r^{-1}, 0, \ldots, 0)$$

Non-linear iterative solution

Newton iteration

Assume that we have a function $\mathbf{b} = f(\mathbf{a}, \mathbf{p})$, where $\mathbf{b}$ is a measurement vector and $\mathbf{p}$ is a parameter vector, we wish to find the vector $\mathbf{p}^*$ that satisfy $\epsilon = f(\mathbf{a}, \mathbf{p}^*) - \mathbf{b}$ for which $\|\epsilon\|$ is minimized. To solve the case where f is not a linear function, we may start with an initial estimated value $\mathbf{p}_0$, and refine the estimate under assumption that the function f is locally linear.

Assume the function $f(\mathbf{p})$ is locally linear around $\mathbf{p}_0$, using Taylor series around $\mathbf{p}_0$

$$f(\mathbf{p}_0 + \Delta) = f(\mathbf{p}_0) + J\Delta \tag{6.30}$$

where the Jacobian matrix $J = \partial f(\mathbf{p})/\partial \mathbf{p}$ is evaluated at the current estimate. We seek a point $f(\mathbf{p}_1)$, $\mathbf{p}_1 = \mathbf{p}_0 + \Delta$, which minimizes $f(\mathbf{p}_1) - \mathbf{b} = f(\mathbf{p}_0) + J\Delta - \mathbf{b}$. Let $\epsilon_0 = f(\mathbf{p}_0) - \mathbf{b}$. Then $\|\epsilon_0 + J\Delta\|$ should be minimized over Δ, and it becomes a linear minimization problem.

According to Equation 6.28, Δ is obtained by solving the normal equation

$$J^T J\Delta = -J^T\epsilon_0 \tag{6.31}$$

Then the solution $\mathbf{p}^*$ is approached by starting from the initial estimate $\mathbf{p}_0$ with successive approximation

$$\mathbf{p}_{i+1} = \mathbf{p}_i + \Delta_i \tag{6.32}$$

where Δ_i is the solution to the linear least squares problem

$$J\Delta_i = -\epsilon_i$$

However, it is possible that the iteration procedure converges to a local minimum value, or does not converge at all and the iteration algorithm depends very strongly on the initial estimate $\mathbf{p}_0$.

Gauss-Newton iteration

According to Equation 6.21, $g(\mathbf{p}) = \frac{1}{2}\|\epsilon(\mathbf{p})\|^2 = \frac{1}{2}\epsilon(\mathbf{p})^T\epsilon(\mathbf{p})$ and we expand $g(\mathbf{p})$ around $\mathbf{p}_0$ in a Taylor series

$$g(\mathbf{p}_0 + \Delta) = g + g_{\mathbf{p}}\Delta + \frac{1}{2}\Delta^T g_{\mathbf{pp}}\Delta + \ldots \tag{6.33}$$

where subscript $\mathbf{p}$ denotes differentiation.

We want to minimize the function with respect to Δ and set the derivative to 0 then we have

$$g_{\mathbf{pp}}\Delta = -g_{\mathbf{p}} \tag{6.34}$$

where $g_{\mathbf{p}} = \epsilon_{\mathbf{p}}^T\epsilon$ and $g_{\mathbf{pp}} = \epsilon_{\mathbf{p}}^T\epsilon_{\mathbf{p}} + \epsilon_{\mathbf{pp}}^T\epsilon$. Since $g(\mathbf{p})$ is linear around $\mathbf{p}_0$, $\epsilon_{\mathbf{pp}}^T\epsilon = 0$. Then we have

$$\epsilon_{\mathbf{p}}^T\epsilon_{\mathbf{p}}\Delta = -\epsilon_{\mathbf{p}}^T\epsilon \tag{6.35}$$

and this is called the Gauss-Newton update equation. Generally this is a good approximation, particularly close to a minimum, or when ϵ is nearly linear around $\mathbf{p}$.

Gradient descent

The negative (or down-hill) gradient vector $-g_{\mathbf{p}} = -\epsilon_{\mathbf{p}}^T\epsilon_{\mathbf{p}}$ defines the direction of most rapid decrease of the cost function. One way of minimization of the function g is to move iteratively in the gradient direction. To determine the length of step in the negative gradient direction, we denote $-g_{\mathbf{p}} = \lambda\Delta$, in which Δ is the parameter increment and λ controls the length of the step. In this case, the update equation is

$$\lambda\Delta = -\epsilon_{\mathbf{p}}^T \epsilon_{\mathbf{p}} \tag{6.36}$$

Gradient descent by itself is not a very good minimization strategy, typically characterized by a slow convergence, but in conjunction with Gauss-Newton it yields the commonly used Levenberg-Marquardt method.

Levenbert-Marquardt iteration

The non-linear least squares methods we described above have problems. (1) The steepest descent method has no good way to determine the length of the step. (2) Newton's method is based on solving a linear system. The matrix to be inverted can be singular. (3) Moreover, unless it is started close to the minimum, Newton's method sometimes leads to divergent oscillations that move away from the answer. That is, it overshoots, and then overcompensates, etc [219].

The Levenbert-Marquardt iteration method is a slight variation on the Gauss-Newton iteration method by augmented λ to Equation 6.35

$$(\epsilon_{\mathbf{p}}^T \epsilon_{\mathbf{p}} + \lambda I)\Delta = -\epsilon_{\mathbf{p}}^T \epsilon \tag{6.37}$$

The main advantage of this technique is rapid convergence. However, the rate of convergence is sensitive to the starting location.

Random forests for object detection

The random forests [220] is an ensemble approach that can be considered to be a form of a nearest neighbor predictor. The main principle behind ensemble methods is that a group of "weak learners" can come together to form a "strong learner". Random forests is a very fast and accurate tool for classification, clustering and regression and widely used in computer vision applications such as keypoint recognition, digit recognition, gesture recognition, object tracking, object recognition, augmented reality, etc.

Here we mainly focus on how to explain random forests for object classification.

A classifier is a mapping from a feature vector $\mathbf{f}$ to a set of discrete class labels $\mathbf{C}$. The features are composed of appearance, shape, texture, etc.

$$\begin{gathered} \mathbf{f} = \begin{pmatrix} f_1 & f_2 & \dots & f_N \end{pmatrix} \\ \mathbf{C} = \begin{pmatrix} C_1 & C_2 & \dots & C_K \end{pmatrix} \\ \mathbf{f} \rightarrow \mathbf{C} \end{gathered} \tag{6.38}$$

A standard pattern recognition method is to learn the posterior distribution over class label conditioned by features

$$P(C = C_k | f_1, f_2, \ldots, f_N) \tag{6.39}$$

and classify the objects based on the mode of the posterior

$$arg\max_k P(C = C_k | f_1, f_2, \ldots, f_N) \tag{6.40}$$

As for object detection, we treat the problem as a two classes classification problem, in which the two classes are foreground objects and background scenes. In other words, $\mathbf{C} = \begin{pmatrix} C_O & C_B \end{pmatrix}$, where C_0 stands for foreground objects and C_B stands for background.

The random forests algorithm for object detection [221] is shown in Table 1 and illustrated in Figure 4.

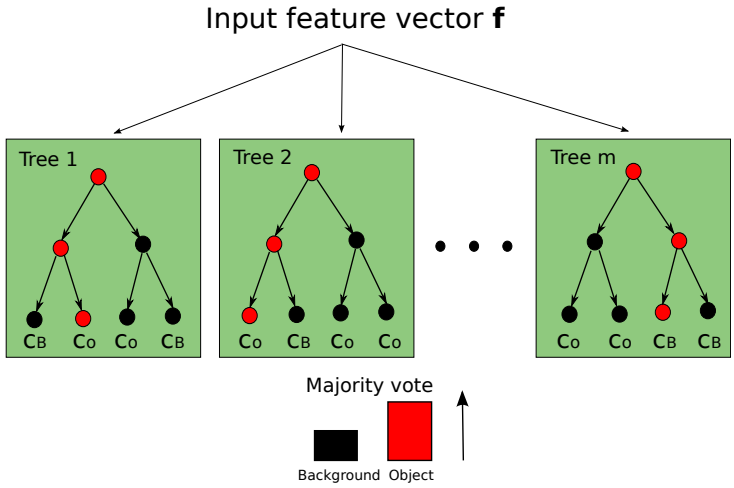


Figure 4: Random forests

Table 1: Random forests algorithm for object detection

Pre-processing:	Collect a set of images and annotate a bounding box for each object. Randomly sample a subset for training and normalize image patches within bounding boxes.
Training:	The split function evaluates one or more image features of the image patch I and passes it to the left child $p_\phi(I) = 0$ or right child $p_\phi(I) = 1$. Starting at the root node with the training set $T_{node} = T$, a tree grows recursively (parameters Φ can be depth, viewpoint, scale, aspect ratio): 1. Generate a random set of parameters $\Phi = \{\phi_k\}$; 2. Divide the set of patches T_{node} into two subsets T_L and T_R for each $\phi \in \Phi$: $$T_L(\phi) = \{I \in T_{node} \mid p_\phi(I) = 0\}$$ $$T_R(\phi) = \{I \in T_{node} \mid p_\phi(I) = 1\}$$ 3. Select the split parameters ϕ^* that maximize a gain function g $$\phi^* = \arg\max_{\phi \in \Phi} g(\phi, T_{node})$$ 4. Continue growing with the training subsets T_L and T_R if some predefined stopping criteria are not satisfied; otherwise, create a leaf node and store the statistics of the training data T_{node}.
Predict:	For detecting an object, sampled image patches from a test image and extracted feature vectors are passed through trees. In order to locate an object in the image, the probability of an object is evaluated by majority vote.

Summary

Human perception is an active process. By altering its viewpoint rather than passively observing surroundings and by operating on sequences of images rather than on a single frame, the human visual system has the ability to explore the most relevant information based on knowledge, therefore when growing up a human is able to develop cognitive perception. Comparably, for humanoid robots to develop cognitive perception, active vision is indispensable. Humanoid robot research has already nearly half a century history. There are approximately 2000 research papers on active vision published during 1986–2010 that covered a large range of research fields in robotics. Nowadays, the new trend is to use a stereo setup or a Kinect with neck movements to realize active vision. However, human perception is a combination of eyes and neck movement. In order to design such an advanced humanoid active vision system, eye movements with biological inspiration similar to human eyes should be taken into consideration. Depth perception based on pure image information can then be obtained without utilizing any advanced sensors.

This thesis presents a complete active vision system with 4 degrees of freedom that works in a similar way as human vision. It is composed of the following parts:

1. The mechanical design has 4 motors with independent vergence angle control, one tilt motor for both eyes and one pan motor for the neck.

2. The controllers simulate the eye movements as humans: saccade eye movements, pursuit eye movements, vestibulo-ocular reflex (VOR) eye movements and vergence eye movements, where motor positions and velocities are controlled with input from an Inertia Measurement Unit (IMU).

3. An optimal feature selection mechanism which is based on various properties of objects is applied before tracking.

4. In order to smoothly pursue and learn an object from different perspectives, three different trackers are used: a color based tracker, an AR marker based tracker for testing, and a robust online tracker.

5. A saliency detector segments the most dominant objects from the scenes and a robust online tracker provides refined segmentations. As a result, the robots have a self-explorative ability for unknown environments.

6. Owing to vergent eyes moving at different angles, intrinsic calibration as well as extrinsic calibration is required to ensure the accuracy of 3D perception. Here the motor positions are utilized together with a robust M-Estimator to recover the geometry between two eyes.

7. Humans utilize multiple cues for depth perception. Depth perception is strongly related to eye movements. Multi-mode depth perception is applied to perceive environment and objects in 3D for further vision tasks such as object recognition, and object grasping.

The realized system works within real-time constraints and with low cost cameras and motors. Therefore it provides an affordable solution for industrial applications.

In conclusion, active vision can be applied to various applications and it is a rapid-growing research domain. This thesis and its proposed vision system provides an insight into the research field of active humanoid robot vision.

Xin Wang

Samenvatting

Menselijke dieptеperceptie is een actief proces. Het veranderen van het gezicht en het gebruik van een reeks van beelden, in plaats van het passief observeren van een omgeving met een enkel beeld, geeft menselijk gezichtsvermogen de mogelijkheid om de meest relevante informatie te observeren op basis van kennis, waardoor een groeiende mens in staat is om cognitieve perceptie te ontwikkelen. Dit is vergelijkbaar met humanoïde robots, waar actief gezichtsvermogen nodig is om cognitieve perceptie te ontwikkelen. Onderzoek in humanoïde robotica is bijna een halve eeuw aan de gang. Er zijn ongeveer 2000 onderzoekspapers over actief gezichtsvermogen gepubliceerd tussen 1986 en 2010, hetgeen een groot scala aan onderzoeksgebieden in robotica beschrijft. Tegenwoordig is het een trend om een stereo camera of een Kinect te gebruiken, in combinatie met nek bewegingen, om actief gezichtsvermogen te realiseren. Menselijke perceptie is echter een combinatie van ogen nekbewegingen. Om een geavanceerde humanoïde, actieve zichtsysteem te ontwikkelen, moet er gekeken worden naar menselijke oogbewegingen om inspiratie op te doen uit de biologie. Diepte perceptie gebaseerd op puur beeldinformatie zal dan gerealiseerd worden zonder het gebruik van geavanceerdere sensoren. Dit proefschrift presenteert een actief zichtsysteem met 4 graden van vrijheid gebaseerd op menselijke gezichtsvermogen. Het bestaat uit de volgende onderdelen:

1. Het mechanische ontwerp heeft 4 motoren met onafhankelijke convergentie motorbesturing, een tiltmotor voor beide ogen en een panmotor voor de nek.

2. De motor besturing is vergelijkbaar met menselijke oogbewegingen: Saccade oogbewegingen, object-object-oogbewegingen, vestibulo-oculaire reflex (VOR) oogbewegingen en convergentie oogbewegingen, waar de positie en hoeksnelheid van de motoren bepaald worden door de input van een Inertia Measurement Unit (IMU).

3. Voor het objectvolgend mechanisme, worden optimale kenmerkenselectietechnieken toegepast, die gebaseerd zijn op verschillende eigenschappen van voorwerpen.

4. Om een voorwerp gelijkmatig te volgen en te leren herkennen vanuit verschillende oogpunten zijn er drie verschillende volgsystemen gebruikt: volgen op basis van kleur, volgen van een AR marker om te testen, en een robuust online volgsysteem.

5. Een saillant-kenmerkendetectie segmenteert de meest opvallende voorwerpen in een scene en een robuust online volgsysteem verfijnt deze segmentatie. Dit resulteert in de robots die in staat zijn om autonoom onbekende omgevingen te kunnen verkennen.

6. Gezien de convergerende camera's in verschillende richtingen kunnen bewegen, is zowel intrinsieke als extrinsieke camerakalibratie nodig om de nauwkeurigheid van 3D-perceptie te garanderen. De positie van de motoren kunnen samen met een robuuste M-estimator gebruikt worden om de geometrie tussen de twee ogen te achterhalen.

7. De mens gebruikt verschillende aanwijzingen voor diepteperceptie. Diepteperceptie is sterk gerelateerd aan oogbewegingen . Meervoudige diepteperceptie wordt gecombineerd om de omgeving en voorwerpen te kunnen waarnemen, wat gebruikt kan worden voor taken zoals het herkennen en grijpen van objecten.

Het gerealiseerde systeem werkt met real-time beperkingen en met relatief goedkope camera's en motoren. Het biedt daarom een betaalbare oplossing voor industriële toepassingen. Concluderend, actief gezichtsvermogen kan toegepast worden voor verscheidende applicaties en het is een snel groeiend onderzoeksdomein. Dit proefschrift en het voorgestelde zichtsysteem verschaft inzicht op het gebied van humanoïde actief gezichtsvermogen.

Xin Wang

Curriculum Vitae

Xin Wang was born in Shann'xi, China on September 13, 1983. She finished secondary school in 2005 and entered into the Northwestern Polytechnical University in Xi'an, where she studied Signal and Information Processing. With the highest score among all the students, she contiued her study as a master student in the same field. She was also leader of RoboCup Innovation Lab in her university and attended the 8th FIRA China Cup and won champion and runner-up titles in Twosome and Foursome Dance Competition. She graduated in 2009 and came to the Netherlands to become a PhD student (in Dutch: *Assistent in Opleiding* or AIO) with the Biomechanical Department at the Delft University of Technology, supervised by Prof. Pieter Jonker. She conducted her research on building up a humanoid active vision system that explores the unknown environment in a similar way as humans do. She also helped to organize the World Robocup Junior Competition 2013 in Eindhoven as local chair. Now she is a part of organizing committee in Robocup Junior. She was also active in giving robotics courses to female school students with Technika 10 and lectured in the 3D robot vision courses.