Diss. ETH No. 12004

Real Time Inverse Stereo System for Surveillance of Dynamic Safety Envelopes

A dissertation submitted to the
SWISS FEDERAL INSTITUTE OF TECHNOLOGY
ZURICH

for the degree of
Doctor of Technical Sciences

presented by
MARTIN RECHSTEINER
DIPL. EL.-ING. ETH
BORN 19 JULY 1964
CITIZEN OF REHETOBEL, AR

accepted on the recommendation of
Prof. Dr. Gerhard Tröster, examiner
Dr. Frank Ade, co-examiner

1997

Acknowledgment

This thesis was written during the course of the project "Dynamische Raumüberwachung" ("dynamic workspace surveillance") between the years 1994 and 1996. Based on the ideas of Dr. Anton Gunzinger, Bruno Schneuwly and myself, the project was initiated together with the head of the electronics laboratory, Prof. Dr. Gerhard Tröster, the former head, Prof. Dr. Walter Guggenbühl and Prof. Dr. Gerhard Schweitzer, head of the robotics laboratory at ETH Zurich.

I would like to thank my advisor Prof. Dr. Gerhard Tröster for his support and for making it possible for me to do research in this interesting area.

I also would like to thank my associate advisor Dr. Frank Ade for his suggestions and for the meticulous reading of my dissertation.

My special thanks go to Dr. Bruno Schneuwly for many fruitful discussions and advice during this project, to Dr. Markus Thaler, the head of the signal processing group, and to our system administrator Dr. Andreas Karrer who was always ready to help when difficulties or questions with the computer system arose.

My further thanks go to the company LEUTRON Vision AG, who provided the image processing hardware PRIMA*SPEED* and especially to Urs Wirth, who assisted me when problems with the hardware arose.

I would also like to thank all my colleagues at the Electronics Laboratory and the students Ruedi Widmer and Alessandro Ceccato in particular for their contributions to the image transformation module.

Zurich, December 1996 Martin Rechsteiner

Kurzfassung

Bis heute werden Roboter durch Gitter oder Lichtschranken vollständig abgeschirmt um Unfälle zu vermeiden. Zukünftige flexible Roboterarbeitsplätze sollen die Zusammenarbeit von Mensch und Roboter im gleichen Arbeitsraum ermöglichen, womit sich eine flexiblere und sinnvollere Aufteilung der Arbeit zwischen Mensch und Roboter erreichen lässt. Um solche „kooperierende Roboter“ zu ermöglichen, braucht es unbedingt hochentwickelte Überwachungssysteme.

In dieser Arbeit wird eine Methode vorgestellt, welche eine unsichtbare, frei definierbare und dynamisch der momentanen Situation anpassbare Sicherheitszone um den Roboter auf eindringende Objekte überwacht. Dazu wird das sogenannte „inverse Stereoprinzip“ verwendet, eine texturbasierte Stereomethode bei welcher die sonst notwendige rechenintensive Korrespondenzanalyse durch die Generierung und den Test einer Hypothese ersetzt wird. Objekte werden erkannt, sobald sie durch die Trennfläche zwischen den beiden Arbeitsbereichen durchtreten. Dazu wird das Bild der einen Kamera in dasjenige Bild transformiert, das die andere Kamera sehen würde, falls sich alle Objekte in der Trennfläche befinden würden. Anschliessend wird das so transformierte *hypothetische* Bild mit dem *wirklichen* Bild der anderen Kamera verglichen. Für Bildbereiche, welche in den beiden Bildern übereinstimmen, stimmt die Hypothese, womit sich dort ein Objekt in der Trennfläche befindet.

Diese neue Methode wurde entwickelt hinsichtlich einer Verarbeitung in Video-Echtzeit auf einer möglichst günstigen und kompakten Hardware, um eine spätere Implementation in eine „smart camera“ zu ermöglichen.

Im ersten Teil dieser Arbeit werden die Anforderungen an ein solches Überwachungssystem analysiert und bestehende Sensoren bezüglich derer Eignung für Sicherheitsanwendungen verglichen. Ein Schwerpunkt der Arbeit wurde auf die Charakteristiken der verschiedenen Korrelationskriterien, deren Eignung für diese Methode und deren Einfluss auf die Leistung des Gesamtsystems gelegt. Weiter werden geeignete Methoden zur Kalibration der Kameras und Transformation der Bilder untersucht. Im letzten Teil der Arbeit werden die Resultate der erfolgreichen Hardware-Implementation und einige mögliche Erweiterungen zur Steigerung der Robustheit der Methode vorgestellt.

Abstract

Providing advanced robotic systems with the capacity of sharing their workspace with humans requires equipping them with an adequate security system. For lack of sophisticated surveillance systems, robot workspace must currently be strictly separated from human workspace by fences or light barriers.

A vision based security system that monitors an invisible but clearly defined, dynamically adjustable safety envelope around the robot for intruding objects is presented. A texture based stereo vision method that does not require a time consuming correspondence search, the so-called "inverse stereo principle", is used. Objects are detected when they penetrate the separation skin between the two workspaces. To detect objects in the separation skin, one camera image is transformed into the image the other camera would see if all objects were in the separation skin. This hypothetical image is subsequently compared to the real image of the other camera. Corresponding regions belong to objects situated in the separation surface.

This new method was aimed at a processing at video rate on cheap, compact hardware with the possibility of future implementation on a smart camera.

In the first part of this thesis the requirements for a surveillance system in robotics were analyzed and existing sensors were compared in regard to security applications. Emphasis was placed on characteristics of correlation criteria, the consequences for the system performance and the selection of a suitable method. Furthermore, methods of calibrating the cameras and transforming the images are investigated. Finally, results of the successful hardware implementation and some possible extensions for making the method more reliable are presented.

Leer - Vide - Empty

Contents

1 Introduction 1

2 Flexible Monitoring Systems 3
- 2.1 Security in Robotics . . . 3
- 2.2 Human Robot Cooperation . . . 6
  - 2.2.1 Cooperative Production Systems . . . 6
  - 2.2.2 Service Robots . . . 7
  - 2.2.3 Increasing Safety of Traditional Robot Systems . . . 8
- 2.3 Requirements for Monitoring System . . . 8
- 2.4 Consequences for the Workplace . . . 9
- 2.5 Other Applications . . . 10

3 Sensors 11
- 3.1 Tactile Sensors . . . 12
- 3.2 Proximity Sensors . . . 12
  - 3.2.1 Mechanical Sensors . . . 12
  - 3.2.2 Field Based Sensors . . . 13
- 3.3 Photo-electric Sensors . . . 13
  - 3.3.1 Detection of Reflected Light . . . 13
  - 3.3.2 Detection of a Break of Light Beam . . . 14
- 3.4 Microwave . . . 15
- 3.5 Infrared Radiation Sensor (Pyroelectric Sensors) . . . 15
- 3.6 2D Vision . . . 15
  - 3.6.1 Change and Motion Detection . . . 16
  - 3.6.2 Object recognition . . . 17

4 3D Vision Methods 19

4.1 Active Vision 19
  4.1.1 Ultrasonic Sensors 20
  4.1.2 Laser Telemetry 21
  4.1.3 Triangulation 21
  4.1.4 Shape from Shading 23
4.2 Passive Vision 23
  4.2.1 Stereopsis 24
  4.2.2 Stereo from Motion 27

5 Inverse Stereo Principle **29**
5.1 Idea 30
  5.1.1 Horopter 31
  5.1.2 The Method 32
5.2 Algorithms 34
5.3 Software Implementation 38

6 Camera Calibration **39**
6.1 Coordinate Systems 40
6.2 Pinhole Camera Model 41
6.3 Geometrical Lens Distortions 44
6.4 Calibration Methods 47
  6.4.1 Linear optimization 47
  6.4.2 Full Scale Nonlinear Optimization 47
  6.4.3 Two Step Methods 48
6.5 Modified Tsai's Camera Calibration 48
6.6 Subpixel Target Localization 50
  6.6.1 Subpixel Edge Detection by Interpolation 52
  6.6.2 Parametric Curve Fitting 52
  6.6.3 Edge Detection by Image Reconstruction 53
  6.6.4 Moment-Based Edge Detection 54
  6.6.5 Maximum Likelihood Edge Estimation 56
6.7 Results 56
  6.7.1 Performances on Ideal Edges 57
  6.7.2 Influence of Noise 60
  6.7.3 Performance on Real Images 61
  6.7.4 Conclusion 62
6.8 Implementation 64
  6.8.1 Calibration Pattern 64
  6.8.2 Automatic Target Localization 65
6.9 Results 65

7 Transformation **69**

7.1 Geometric Transformation 69

7.1.1 Plane 71

7.1.2 Ellipsoid 72

7.2 Mapping Methods 73

7.2.1 Forward Mapping 73

7.2.2 Inverse Mapping 74

7.3 Eliminating Blurring and Aliasing 75

7.3.1 Area Sampling 75

7.3.2 Adaptive Area Sampling 76

7.3.3 Supersampling 76

7.4 Interpolation Methods 76

7.4.1 Nearest Neighbor Interpolation 77

7.4.2 Bilinear Interpolated Transformation 77

7.4.3 Higher Order Interpolation 78

7.5 Measures to Simplify Transformation 78

7.6 Change of Image Resolution 82

7.7 Alternative Calculation of Transformation 82

8 Correlation **83**

8.1 Correlation Methods 83

8.1.1 Correlation on Brightness Images 84

8.1.2 Correlation on High-pass Filtered Images 87

8.1.3 Correlation on Direction Images 87

8.1.4 Binary Correlation 88

8.2 Requirements 88

8.3 Behavior on Unwelcome Influences 92

8.3.1 Brightness 93

8.3.2 Noise 94

8.3.3 Blurring and Scaling 94

8.4 Segmentation Capability 94

8.4.1 Experimental Setup 95

8.4.2 Definitions 96

8.4.3 Influence of Prefilters 100

8.4.4 Behavior of the Various Correlation Criteria . . . 104

8.4.5 Influence of Filtering Correlation Results 109

8.5 Dependence on Object Size 110

8.6 Required Computational Resources 111

8.7 Conclusion 112

9 Texture Analysis **113**
- 9.1 Texture Measures . . . 114
- 9.2 Required Texture . . . 115
- 9.3 Methods for Coping with Weakly Textured Scenes . . . 117
  - 9.3.1 Increase Texture . . . 117
  - 9.3.2 Decrease Noise . . . 118

10 Hardware Implementation **119**
- 10.1 Estimation of Required Computational Resources . . . 119
  - 10.1.1 Image Transformation . . . 120
  - 10.1.2 Subsampling . . . 121
  - 10.1.3 Lowpass Filtering . . . 121
  - 10.1.4 Sobel Filtering . . . 122
  - 10.1.5 Correlation . . . 123
  - 10.1.6 Texture Analysis . . . 124
  - 10.1.7 Segmentation . . . 124
- 10.2 Hardware Architectures . . . 126
  - 10.2.1 High Performance Processors . . . 127
  - 10.2.2 Multi-processor Systems . . . 128
  - 10.2.3 Conclusion . . . 130
- 10.3 Architecture of PRIMA*SPEED* . . . 130
- 10.4 Implementation of the "Inverse Stereo Principle" . . . 132
  - 10.4.1 Implementation Possibilities of Image Transformation . . . 134
  - 10.4.2 Implementation of the Transformation Module . . . 137
  - 10.4.3 Subsampling and Packing . . . 139
  - 10.4.4 Lowpass Filtering . . . 139
  - 10.4.5 Image Correlation . . . 140
  - 10.4.6 Image Segmentation . . . 142
- 10.5 Resources Used . . . 143

11 Implementation and Tests **145**
- 11.1 Geometric Setup . . . 145
- 11.2 Experimental Data About the System . . . 148
  - 11.2.1 Thickness of Separation Skin . . . 149
  - 11.2.2 Slope of Separation Skin and Object . . . 150
  - 11.2.3 Problems . . . 153

11.3 Assessment of the Method 155

12 Possible Extensions **157**
12.1 Multi-camera Systems 157
12.2 Multi-Surface System 158
12.3 Detecting Objects Outside the Separation Skin 159
12.4 Using Color Images 159
12.5 VLSI Integration 162

Chapter 1

Introduction

Today it is almost impossible to imagine modern industry without robots. Thanks to robots, the mechanization suitable for mass production was supplemented by automation for medium or even small batch size. In the future robots will not only be used in industrial environments, but also in service (industrial cleaning, household, aids for handicapped [1]) and this will raise the requirements for robots and their sensors.

Under some circumstances robots can be very dangerous to humans near them because of their long, very strong arms and the rapid and sometimes "unpredictable" movements. At present robot workspace must therefore be strictly separated from human workspace. This is usually done with fences or light-barriers around the robot.

In order to facilitate the robot's reacting in an intelligent way to the environment, robots are given sensors which provide them with information about the environment. Many sensors, including 2D-vision, have left stages of research and are being applied in industry. Other sensors, especially 3D-sensors, are still in their infancy. Robots still lack sophisticated sensors and artificial intelligence, which facilitate either fairly autonomous operation even in unstructured environments or showing cooperative behavior so that humans and robots can work in the same environment.

Vision is one of the most powerful senses of living creatures, and vision sensors have the advantage of being non-intrusive or even completely passive. Much research has been done in the field of vision. The enthusiasm felt at the beginning was replaced by the insight that vision is not only a powerful but also a very complex and difficult sense.

An enormous drawback of vision systems is that such huge amounts of data have to be processed that a lot of vision algorithms cannot be processed in video real time. In order to speed processing up, one can either build faster processors, systems with parallel processors and specialized architectures for image processing, or apply new algorithms that are geared to an efficient processing for a given hardware.

In order to facilitate cooperation of robots with humans, real time sensors which supervise the robot's workspace are necessary. Vision proves to be very well suited for such an application, but its high computational requirements are a severe drawback.

In this work a special, very efficient algorithm was combined with an implementation on specialized hardware in order to attain video-rate processing.

In the beginning of the thesis the requirements for a robot workspace surveillance system are analyzed and known sensors which could be d for supervision purposes are presented and rated. Then the new method, called "inverse stereo principle", and the algorithms used are presented in more detail. Various subpixel estimation methods for camera calibration were investigated and the performance of the implemented camera calibration analyzed. Several spatial image transformation methods are discussed and the performance of inverse remapping with bilinear interpolation presented. A main emphasis was placed on the correlation methods. Experiments with real images were carried out to find a suitable correlation method and their computational requirements are analyzed. Finally, various hardware platforms and implementation variants are discussed and the successfully implemented hardware solution and some results with the experimental system presented. To conclude, possible extensions of the system and an outlook is given.

Chapter 2

Flexible Monitoring Systems

Up to now, robots have been used mainly in industrial environments and are usually separated by perimeter fences or light barriers from human workspace. Future robots will be able to work in less structured environments and will have the ability to cooperate with humans. For safe operation, cooperating robots must be equipped with a sophisticated security system. The industrial environment, the speed of the robot and the staff working with the robots have special requirements for a security system.

2.1 Security in Robotics

On the one hand, the introduction of industrial robots into dangerous and harmful environments has reduced risks. On the other hand, industrial robots themselves are a source of accidents. The powerful and automatically moving manipulator which moves in a space outside the machine body makes robots especially dangerous. In addition, its direction, speed and sequence of movements cannot easily be predicted by a worker.

Isaac Asimov has stated the “Three Laws of Robotics (1942)” [2] a robot must obey in order to be no harm to people:

1. A robot may not injure a human being, or, through inaction, allow a human being to come to harm.

2. A robot must obey orders given it by human beings except where such orders would conflict with the First Law.
3. A robot must protect its own existence as long as such protection does not conflict with the First or Second Law.

The laws still hold true today and will be even more important in the future, because robots and humans will work closer together than they have so far.

Whereas the first part of the First law is very general, the second part is more fictitious. The Second Law can be applied to robot teaching situations, where a safety-system should prevent accidents due to manipulation errors by an operator.

Therefore a robot must be equipped with one or more of the following safety measures, which prevent the operator from coming in contact with an operating robot:

- **Enclosures** and **fences** prevent humans from entering the robot's workspace while it is in operation and also prevent injuries by 'flying' objects released by a robot. The opening of the door of the enclosure triggers a stop of the robot unless this mechanism is bypassed. Fences have two severe disadvantages: they obstruct the often necessary view to the robot and restrict access to the robot. Theoretically, a complete enclosure of the robot is the safest technique for providing safety. However, in practice the situation often arises that for maintenance work, teaching, testing or only to adjust a misaligned object, an operator must go near the robot, and there is no mechanism to prevent this operator from being hurt.
- **Ropes, railings** and **chains** cannot really prevent humans from entering the robots' workspace, but only aggravate it. It is a cheap measure for harmless robots.
- **Light barriers** and **safety mats** detect the intrusion of humans into the robot's workspace so that it can be stopped before any harm occurs. Neither measures obstruct the view nor the access to the robot and the robot is stopped automatically without the intervention of the operator. However, light-barriers allow only plane separation surfaces and safety mats only detect the feet and give no information about the entire body of the operator. This results in larger safety zones because it must be designed with the worst case in mind. In addition, changing the dimensions

of the workspace results in a new installation of the barriers or mats. Light curtains must be positioned at a distance from the dangerous zone, such that there is enough time to stop the robot before a collision can occur. However, if this area is so large that it is possible to stand between the curtain and the dangerous machine, human-robot cooperation is not possible. The reason for this is that an object between the light barrier and the machine cannot be detected and therefore the complete robot has to be stopped and remain stopped until it is manually restarted.

- **Collision detection sensors** detect collisions after they have occured or only a short time before. With capacitive sensors the proximity of an object or a human body is detected such that the robot can be stopped before a collision occurs. Cushions with integrated pressure sensors detect a collision and at the same time prevent any harm because of the cushioning effect. However, these measures are only possible if the robot doesn't move too fast. Because any collision results in an emergency stop of the robot, this system is not suitable for cooperative robots.

- With **camera surveillance** it is possible to detect changes in the workspace or objects intruding the workspace. There are various methods possible, from simple two-dimensional change detection to sophisticated three-dimensional recognition of intruding objects. Vision is the most versatile surveillance method, but also the most complicated (see Chapters 3 and 4).

Analyses of accidents in Japan [3] and Sweden [4] showed that most accidents occurred during the repair, maintenance or teaching of a robot or while adjustments were being made in the course of normal operation. In part safety systems were switched off due to negligence or to a misunderstanding between two operators, in other cases they had to be switched off in order to teach or test the robot. It often happens that the operator misjudges the situation (e.g. when the robot has stopped or moves very slowly just before it begins to work again at normal speed) and enters the robot workspace when the safety system is switched off. Sugimoto concludes in [3] that “only when robots themselves are able to detect the approach of humans and perform appropriate actions to avoid accidents, will safety in the human-robot workplace be assured”.

2.2 Human Robot Cooperation

There are various applications where cooperation between humans and robots occurs.

2.2.1 Cooperative Production Systems

A short response time to changes in consumer preferences is expected from modern manufacturing systems. This results in shorter product life, more frequent change of product parameters and more models of the same product. In addition, there is the need to automate production lines, which were too difficult to automate some years ago.

Fully automated manufacturing and cooperative systems represent two contrasting paradigms for automated production. While safety is one of the major problems in cooperative systems, it is often very difficult to replace all human abilities by machines and fully automate a production line. In addition, there are other serious drawbacks to fully automated systems, mainly from a human-oriented point of view.

In fully automated systems human workers are eliminated as a source of friction, and in case of machine failures they should intervene immediately and without making mistakes [5]. However, operators only sitting in front of a control panel observing and supervising the system risk to get deskilled and loose experience of their job if they are only challenged in cases of emergency or maintenance [6]. This is a real conflict of fully automated systems.

In addition, "as techniques and skills are created and advanced by humans, and they are transferred from human to human", "if fully automatic unmanned systems are realized, further progress of technique and manufacturing technology stops" [7]. Nakazawa concludes that "Manufacturing systems of next generation should aim for the fusion of automation technology and human abilities.".

Therefore operators must have the opportunity to decide upon process-parameters in order to develop comprehension for the process which enables them to react in a flexible and adequate way in case of unforeseen situations [8]. However, flexible task allocation where operators can decide themselves and at any time about how the tasks are divided results in close interaction between man and machine and therefore a workspace surveillance system is necessary in order to avoid accidents (see Figure 2.1).

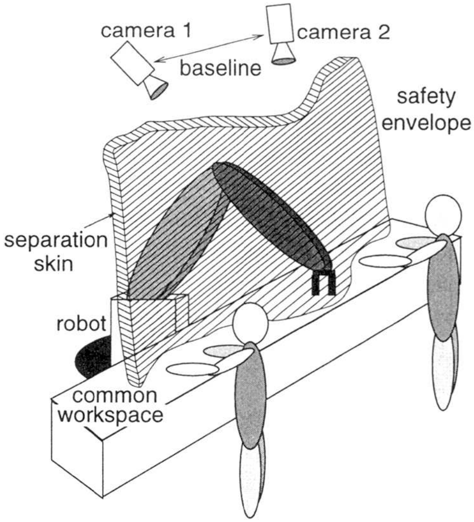


Figure 2.1: *Workspace of a cooperating robot with a safety envelope: human workspace is separated by the separation skin from the robot workspace.*

2.2.2 Service Robots

Up to now robots have mostly been used in production, where it is possible to enclose robots within a fence. However, if robots are used in service tasks, they must work in environments where humans work or live. One can imagine many service tasks in which a robot could help [1]:

- **Commercial cleaning robots** for hotel, office, hospital and industry. A floor-cleaning robot could look like a conventional scrubbing machine, but without a human driver. Control and vision must prevent the robot from colliding with humans and other obstacles in its way.
- **Paranurse** and **aids for handicapped people**. A paranurse could do more routine tasks such as serving meal trays and transporting goods between different stations. Handicapped people could be more independent thanks to a robot that can accomplish certain tasks the person can no longer do [9].
- Various services such as a **gas-filling robot** at a filling station, **household robots** or robots for guard service become possible.

For service robots working in the presence of humans, a sophisticated security system is essential.

2.2.3 Increasing Safety of Traditional Robot Systems

It is often necessary to manually adjust the robot, the material or an auxiliary machine due to the misalignment of material, disrupted flow of materials or failure in a peripheral machine . With a safety system, personnel may work within the robot's working area during operation without any risk of injury. Even in traditional robot systems such a system increases productivity because collisions are prevented and emergency stops are reduced to an absolute minimum. In addition, teaching of robots loses its danger.

2.3 Requirements for Monitoring System

An analysis of possible applications (especially in robotics) shows that a flexible monitoring system should fulfill the following requirements [10]:

- **Good machine accessibility:** A monitoring system must not hinder the operators at work and must allow easy access to the machine such that real cooperation with the robot/machine is possible. For robot maintenance and teaching, no bypassing of the safety system nor removal and re-installation of the system must be necessary (as is the case for fences).
- **Simple adaptability:** In modern manufacturing, product parameters very often change. This can result in a change of workspace dimensions and robot cycle. In such a case a monitoring system should need only very few adaptions.
- **Allow cooperative work:** It should allow an operator to work safely in close interaction with a robot.
- **Flexible definition of workspace:** It must be possible to arbitrarily define the boundary (= separation skin) of the workspaces. Plane separation skins alone are not adequate for many applications.
- **Real time processing:** Processing in real time allows for a fast response time. The necessary safety distance (distance between location where the intruding object is detected and the danger

zone) is proportional to the response time and the objects speed (standard hand speed = 160 cm/s [10]). A short safety distance reduces the space requirements and makes the system more flexible: the workspace boundary can be adapted more precisely to the real danger zone and less space is wasted and cooperation is improved.

- **Robust:** The method must be highly immune to false triggers such as high ambient light, shadows from other machines or people and other radiation (e.g. visible, infrared, sonar) often present in industrial areas: e.g. industrial strobe light, weld flash or other photo-electric devices.
- **Reliability:** The system must be very reliable and must be hard to bypass. In order to achieve this reliability, it must be possible to perform a self-check of the system and to implement redundancy.
- **No dangerous radiation:** Because such a system works in the proximity of humans, it must not radiate dangerous radiation such as laser (danger for eyes) or strong electro-magnetic fields.
- **Restricted space requirements:** Floor space represents significant costs for manufacturing lines. Therefore a security system should occupy little space for installation and it should allow for a very compact installation of the supervised machines.
- **Low costs:** Both system costs and installation costs must be low. Optical systems usually need less installation than mechanical devices (e.g fences, doors).

2.4 Consequences for the Workplace

A robot workplace can profit in the following ways from a sophisticated surveillance system:

- **Human-robot cooperation:** With a sophisticated surveillance system, real cooperation between humans and robots becomes possible. This facilitates more flexible production and automation where fully autonomous production is not possible. In addition, cooperative production has many advantages from an industrial-psychological point of view.
- **Increased safety:** In contrast to today's safety systems, a sophisticated workspace monitoring system is also able to protect

operators working near the robot for teaching and maintenance purposes; this is exactly where most accidents happen.

- **Reduced space requirements:** Because fencing is no longer necessary and the virtual separation between different workspaces can be dynamically changed, there is less unused space and multiple robots or robots and humans can work in the same workspace.
- **Reduced costs:** Because changing production parameters, teaching robots and maintenance do not require security-system re-installation, less installation costs and costs associated with machine standstill arise.
- **Faster change of topology of manufacturing cell:** The change of the dimensions of the manufacturing cell can be done by software and need only minor re-installations.

In many robot applications, necessary human-robot cooperation must be restricted to a minimum for lack of sophisticated safety systems. This results in decreased productivity and a source of robot accidents.

2.5 Other Applications

A monitoring system which can detect objects intruding into an arbitrarily defined room is not restricted to use in robotics. Many other areas such as protection of valuable goods, control of automatic doors or elevators represent possible uses.

Because robot applications generally make the most demanding requirements, the main goal in this project will be to fulfill the requirements of robotic applications.

Chapter 3

Sensors

Sensor technology plays an important role in robotics and will play an even more important role in future robotic applications. Sensors used in robotics are briefly presented and their applicability for use in security systems discussed. 3D-vision sensors are discussed in Chapter 4.

Sensor technology plays a more and more important part in robotics. Thanks to sensors, a robot is able to perceive its environment and react in an appropriate way. Only with sophisticated sensors will a robot be in the position to operate autonomously or in cooperation with humans. For example, with a vision sensor the location of an object can be recognized and the robot can grab it without having to be programmed for a specific object at a predefined location.

Sensors in robotics can be divided into two groups according to their tasks:

- **Intrinsic:** This kind of sensor gets information about the robot itself such as position, angle, velocity or acceleration. They are usually inside the robot and are used as inputs for the robot controller. These sensors are not discussed in this work.
- **Extrinsic:** Extrinsic sensors get information about the robot's environment. The robot uses this information for path planning, navigation, collision avoidance and for doing its specific task.

Some sensor techniques commonly used in robotics for collision avoidance and detection will be discussed briefly in the following.

3.1 Tactile Sensors

Tactile sensors collect information about the environment through direct contact with objects within that environment. The simplest form of a tactile sensor consists only of a single touch probe and provides only the presence and the force of an object at the sensor. More sophisticated sensors consisting of an array of sensing elements get additional information about the shape, location and orientation of an object. Such a sensor consists of materials where some material parameters (e.g. optical refractive index, resistance, dielectric constant) are a function of the pressure in the material.
The information of a tactile sensor is of a very local nature and the sensing involves physical interaction which may result in a deformation of the sensed object [11]. Such sensors play an important role in robot grippers.

3.2 Proximity Sensors

Proximity sensors detect the presence of objects in the sensor's vicinity. There are a variety of methods [1, 12].

3.2.1 Mechanical Sensors

The most common mechanical sensors are limit switches (microswitches), bumpers and safety mats.

Limit switches are one of the simplest and cheapest sensors. They are used, e.g. as safety stops with slow moving robots. Use is restricted to applications where the braking distance is smaller than the spring length of the switch and the contact point is well defined.

Bumpers are usually used for moving vehicles and robot arms (collision detectors) and consist of some cushioning supplied with sensors. The sensors are used to signal if contact has been made along the bumper. If the bumper is divided into segments with individual sensors, this information can be used for navigation. Bumpers have disadvantages: a collision has to occur before an object can be sensed and the foam or spring of the bumper must be as thick as the braking distance of the robot.

Safety mats [13] are put on the floor in the dangerous zone. They are pressure sensitive and detect people in the danger zone. The most common mats are based on a pneumatic principle. In contrast to mechanical guards which limit accessibility and visibility, they have the advantage of rarely interfering with production and permitting free access to the machine/robot for loading or unloading material and supervising the process. It is possible to divide the mat into several individual mats such that the active mats can be selected according to the action of the robot.

The drawback of such safety mats is that only the feet are located and no information about the position of other parts of the body is available.

3.2.2 Field Based Sensors

These sensors are based on the fact that electro-magnetic fields are influenced by metal objects and electric fields are influenced by conductive or dielectric objects. They emit a high frequency electro-magnetic field or electric field and detect changes in this field. Typical sensing ranges for inductive sensors are 0.8 to 60 mm, while for capacitive sensors the maximal switching distance depends on the material (dielectric constant) and varies between 5 and 20 mm [12].

Only capacitive sensors are suited to sensing the proximity of humans, because in contrast to inductive sensors, organic materials can also be detected. However, in order to protect the robot against collisions, the whole body of the robot must be equipped with capacitive sensors and the sensing distance is very short and this requires a short robot reaction time.

3.3 Photo-electric Sensors

Photo-electric sensors consist of a light source and a light detector. In order to overcome problems of ambient lighting, pulsed infrared light is usually used. There are two common modes of operation:

3.3.1 Detection of Reflected Light

The object is detected by the light it reflects. If no object is in the vicinity of the sensor, no light is reflected. Usually, near-infrared is used to reduce the effects of ambient lighting. The amount of reflected light depends on the distance between sensor and object and the object's surface characteristics. The sensor measures the reflected light and on if it exceeds a given threshold it signals that an object has been detected. If, e.g. a robot's gripper is equipped with such a short range sensor the robot can be stopped if the gripper moves towards an obstacle [14]. It is mainly useful for local collision avoidance. It is not sensible to furnish the whole surface of a robot with such sensors.

3.3.2 Detection of a Break of Light Beam

There are two common modes of operation: either the beam is projected to a facing photo-electric detector or it is reflected by a retro-reflector, which facilitates having the emitter and detector in the same housing. Both methods can detect an object on a straight line (spreading direction of light) by sensing a break of the beam. Light barriers (or light curtains) are based on the same principle. They can be used to detect objects or persons entering a danger area formed by planes (see Figure 3.1). If the light beam is interrupted, the sensor initiates a machine stop. Restrictions associated with mechanical fencing can be removed by using such invisible light curtains. However, due to the straight spreading of light, only plane fencing is possible and a change of location requires new installation. In addition, attention must be paid so that light emitted by the light barrier and reflected by an object does not reach the receiver.

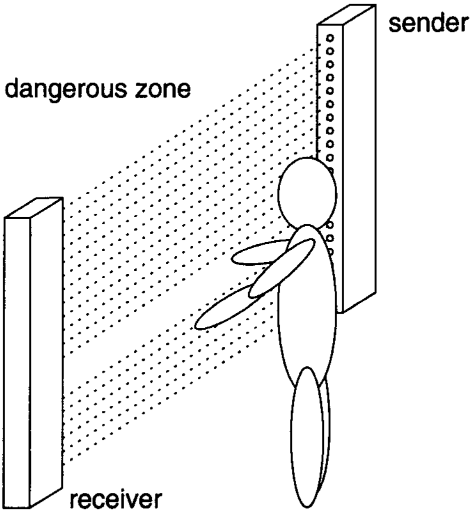


Figure 3.1: *Light-barrier*

3.4 Microwave

Microwave sensors based on the Doppler principle sense the velocity of moving objects; they are commonly used for automatic doors and intrusion detection. The disadvantage of such velocity sensors is that velocity components tangential or lateral to the sensor are not measured, low velocities are not accurately measured, small objects can be masked by large ones because the reflected signal is proportional to the area of the reflected object and interference from outside can occur [15]. Microwave sensors can also cause problems in other electronic machines because of the emission of high frequency electro-magnetic waves. Therefore it has severe drawbacks for cooperative robot systems with humans and electronic equipment present.

3.5 Infrared Radiation Sensor (Pyroelectric Sensors)

All objects with a temperature above absolute zero emit radiation. Pyroelectric sensors can be designed to be sensitive to the infrared range where humans emit radiation. Such sensors are often used to detect intruders in an office or home environment. However, standard light bulbs or even robots emit infrared radiation in the human range. Further, objects that have the same temperature as the environment (e.g. tools) are not detected. Investigation [15] showed that pyroelectric sensors would not work in a generic fashion.

3.6 2D Vision

In principle there are several methods by which a two-dimensional image of the environment can be produced. It is possible to scan a scene with a laser or an electron beam as in an electron microscope and measure the reflected energy. However, the most common method is to produce an image with a photo- or video-camera. Standard video cameras with CCD sensors or MOS-photo-diodes (random addressable) sensors are commonly used in machine vision. 2D vision has versatile applications in machine vision: it is used for the inspection of manufactured parts, measurement of location and dimension of parts for subsequent gripping or for recognizing objects. It is also used for supervision purposes, where an object in a predefined zone must be detected and/or tracked: e.g. traffic scene analysis, security installations, military reconnaissance.

3.6.1 Change and Motion Detection

The detection of moving objects and changing areas in images is a very important task in computer vision. Many applications such as traffic control, remote wide area surveillance, target tracking or supervision applications are based on it. The result of change detection algorithms are usually further analyzed (e.g. motion prediction) to get final information. By applying change detection, the amount of raw data can be drastically reduced for subsequent high level processing, which only has to process those regions where a change occurred.

Change detection can be made either at object or pixel level. Object level change detection methods compare high level features which are generally difficult to extract from real scenes. This makes such methods computationally intensive. Pixel level change detection is usually based on simple computation and therefore facilitates very fast change detection, but is more prone to false triggers. Ideally a change detector should detect only structural changes in the scene (motion, addition or removal of objects). However, change detectors also detect changes in lighting and shadows of moving objects outside the supervised zone.

Change detection algorithms either compare two consecutive frames of a video sequence (inter-frame differencing) or the current frame with a reference image (reference frame differencing). In the latter case the reference image must be periodically updated in order to cope with low frequency changes such as a change in illumination. There are several schemes to try to guarantee that the image is only updated when there is no (moving, additional) object in the scene.

The most straight-forward change detection method is simple differencing of two images ($D(x,y) = I_1(x,y) - I_2(x,y)$). This method is very susceptible to image noise. Better noise behavior shows methods calculating a metric based on the neighborhood of a pixel such as mean, variance or a parametric model of an image patch. However, all these methods have the drawback of not only detecting moving objects but also changes in lighting. In order to improve the robustness to change in illumination, researchers have proposed several methods [16].

Some are based on the fact that small illumination changes (e.g. shadows, clouds) can be closely modeled as a change in the average pixel value over an affected region and therefore texture is preserved.

Therefore the performance is improved if the original images or the resulting difference image are **highpass filtered** (e.g. Sobel-filter) [17, 18] or only the sign of the sobel-filtered images are compared [19]. Another category of approaches is based on a shading model

$$I(x,y) = L(x,y) \cdot r(x,y)$$

I: intensity L: illumination
r: reflectance

and the fact that the reflectance remains constant if illumination changes. Skifstad and Jane proposed in [16] a method that divides the intensity values of the two images

$$\frac{I_1(x,y)}{I_2(x,y)} = \frac{L_1(x,y)}{L_2(x,y)} \cdot \frac{r_1(x,y)}{r_2(x,y)}$$

and then calculates the variance in an image region of 5×5 pixel. If the scene did not change, the quotient of the reflectance values equals one and, under the assumption of constant illumination within the region, the resultant variance is zero. In [20] Fu and Chang proposed a method also based on the shading model, but using moment invariants and therefore reducing the amount of divisions. Both methods have about the same performance.

After the change detection, some object tracking or matching, statistical methods [21], or other high level image processing are usually performed.

Since all these methods are based on 2D vision, no information about the object's distance from the camera is available and therefore all changes, including those very far away and outside the supervised zone, are detected. Such changes must be eliminated with an additional processing step.

3.6.2 Object recognition

In order to make change detection more reliable and possibly to get information about the distance of an object from the camera, it is possible to include some object recognition. If objects and their dimensions are know, it is possible to derive the approximate distance. If the object recognition algorithm is only performed in regions where changes occurred, necessary computation is reduced.

Method	Machine accessibility for visual and manual control	Adaptable to new workspace dimensions without re-installation	Cooperative work possible	Susceptible to being bypassed	Space requirements for installation	Real time processing (fast response time)	Immune to false triggers (shadows, flashes, ...)	Emits dangerous radiation	Suitable for sophisticated monitoring system
Fences	bad	no	no	yes	great	—	—	no	no
Tactile sensors	good	partly[1]	limited[2]	no	small	yes	yes	no	limited[3]
Field based sensors	good	partly[1]	limited[2]	no	small	yes	yes	no	limited[3]
Light barriers	good	no	no	yes	medium	yes	no	no	no
Microwave	good	yes	yes	no	small	yes	no	yes	limited
Pyroelectric sensors	good	yes	yes	no	small	yes	no	no	no[3]
Visual change detection	good	yes	yes	no	small	limited	no	no	no[3]
Object tracking	good	yes	yes	no	small	limited	no	no	no[3]

1) e.g change of gripper may need adaption
2) detection distance too short
3) lack of 3-dimensional information

Table 3.1: *Comparison of Sensors in Regard to Security Applications*

Chapter 4

3D Vision Methods

We live in a three-dimensional world, but technical cameras. like our eyes, produce only a two-dimensional image of the environment. There are several methods for collecting 3-dimensional data. These methods are rated in regard to application in a security system.

In the process of projecting the real three-dimensional scene onto an image plane, the third dimension, the distance from the objects to the camera, is lost. The field of three dimensional vision deals with the reconstruction of the lost depth information. Automatic inference of depth information has proven to be a complex and difficult process. Human vision is based on a combination of various effects, whereas artificial vision methods mainly use only one of the effects such as texture gradient, dimensional or binocular perspective, time of flight, effects of occlusion or variation in surface reflectivity. The methods used to obtain 3D data can be grouped into active and passive vision.

4.1 Active Vision

A method is called active if energy is projected into the scene to be imaged. The information carried by this energy is transformed by the scene and received by the sensor. The depth information lies in the change the emitted energy underwent. Active methods are adequate for use in restricted and controlled environments: e.g. inspection, localization or identification in industry [22]. In active methods the correspondence problem (see Sec. 4.2.1) is inherently solved, which makes these methods less computationally intensive. Most active sensors emit either light, microwaves or ultrasonic waves. Microwaves (RADAR) have the longest wavelength (1 GHz $\rightarrow \lambda \approx 30\,\text{cm}$; 25 GHz $\rightarrow \lambda \approx 1.2\,\text{cm}$) and therefore the obtainable resolution is restricted. In addition, strong microwave radiation is harmful to humans. Ultrasonic waves have a slightly shorter wavelength (20 kHz $\rightarrow \lambda \approx 1.5\,\text{cm}$; 50 kHz $\rightarrow \lambda \approx 0.6\,\text{cm}$) but the minimum beam angle is relatively large. In contrast, light has the advantage of having a much shorter wavelength, being focused to a much narrower beam and that less technical surfaces show specular reflection. Various light sources, such as bulbs, LEDs or lasers, are used depending on the application. Methods that only measure relative length, like interferometry, are not discussed here.

4.1.1 Ultrasonic Sensors

Ultrasonic sensors produce a high frequency sound wave (above 20 kHz) and use triangulation, time of flight or phase shift measurement to collect range information. By measuring the Doppler shift in the frequency of the reflected wave, the velocity of a moving object can also be determined. For many applications (e.g. focus control in cameras) ultrasonic sensors are well suited and very cost-effective due to the simple construction. Depending on the application, a wide or narrow beam is preferred. With a wide sonar beam ($\approx 90^{\circ}$) a safety aura around a robot can be built with only a few transducers. An object in this aura can be detected and its distance measured by the reflected sonar energy. Narrow beams ($\approx 10^{\circ}$ - 30°) are used for mapping an environment because of the higher angular resolution obtained.

However, especially for indoor robot applications, specular reflections are a serious problem. If the direction of beam propagation is not close to perpendicular to the smooth surface of an object, the beam is specularly reflected and probably bounces around the room. Consequently, no echo or multiply reflected echo is received and an object is not detected at all or is estimated to be at a completely wrong distance. A smooth surface is defined as one having texture irregularities less than a quarter wavelength of the ultrasonic frequency ($\approx$ 3 mm for 25 kHz); most technical surfaces therefore produce specular reflections. A solution to this problem is using a wide beam, but this has the drawback that the resolution decreases in angle and distance. Another drawback of ultrasonic sensors is the quite low resolution due to the relatively large beam angle ($> 10^{\circ}$).

Due to specular reflections and the wide beam angle, ultrasonic sen-

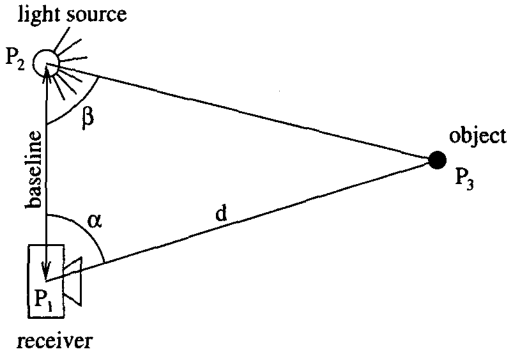


Figure 4.1: *Active triangulation*

sors are less suited to generating range maps. They are better suited to detecting objects in a safety aura around a robot.

4.1.2 Laser Telemetry

In laser telemetry, the distance to an object is obtained from the time light needs to travel from the transmitter to the observed object and back to a receiver. There are two versions of laser telemetry. (1) In pulsed laser telemetry the time of the light pulse travel is measured. (2) In phase modulation telemetry a modulated laser beam is emitted and the phase of the emitted beam is compared with the received beam; the phase shift is a function of the distance. A precision of about 0.5 - 1 cm can be reached. A 3D description of an entire scene could be produced by scanning. As a consequence such sensors are quite slow and need mechanical parts for the deflection of the beams. Thanks to progress in micro-mechanics, integrated laser deflection devices are now available. However, good receivers are much more difficult to integrate.

4.1.3 Triangulation

Triangulation represents a simple trigonometric method for calculating the distances and angles needed to determine the object's location. All active triangulation methods have at least one thing in common: a light source P_2 and a sensor P_1 are pointed at the same point P_3 in space and form an imaginary triangle. If the distance between the transmitter and detector (baseline) and the orientation angles α and β are measured, the object distance d can be calculated (see Fig. 4.1). A major drawback of all triangulation methods (active and passive) is the situation where points illuminated by the light source cannot be seen by the camera or vice versa (so-called occlusion or missing parts).

Laser-Scanning

In laser range sensors a laser beam is aimed at the scene to be imaged and the reflected beam is captured with a camera with a one- or two-dimensional array detector (e.g. CCD). If the distance between the camera and the laser source and the position of the laser spot on the sensor is known, the distance of the object can be calculated by triangulation. In order to obtain three-dimensional information for an entire scene, the scene is scanned. Either the entire sensor (laser and detector) can be moved mechanically, the laser and received beam can be deflected by separate mirrors, or only the laser is moved and a static camera with a two-dimensional sensor array is used. The main drawback consists in the fact that moving mechanical parts are involved.

Structured Light

It is possible to project a pattern onto the scene instead of a single point. There are various methods that differ in the projected pattern. The projection of a plane of light onto a scene results in a single line in the resulting image of a camera. By analyzing this image, three-dimensional data of the points on this line can be obtained by simple triangulation. By sweeping the plane over the scene, a three-dimensional representation of the entire scene is produced. In order to decrease the time needed to acquire the 3D image, it is possible to project a bundle of planes or a grid, such that a disperse depth-map is obtained. However, the main problem caused by the simultaneous projection of multiple light planes is resolving the correspondence between the projected light-plane and the image of the light-planes, as ambiguities can arise.

A solution to this problem is coding the different light planes by means of Gray code [23, 24] or color [25]. In order to produce 2^n coded light stripes, only n patterns are projected, coding each line with a Gray code of n bit (see Fig. 4.2). The input images are binarized depending on whether or not a pixel belongs to the illuminated region in order to extract the projected code. After the n planes have been projected, each stripe is coded with a n-bit code. The only constraints on the scene are that the surface reflectance must be high enough and that the projected light must be stronger than the ambient lighting. Because

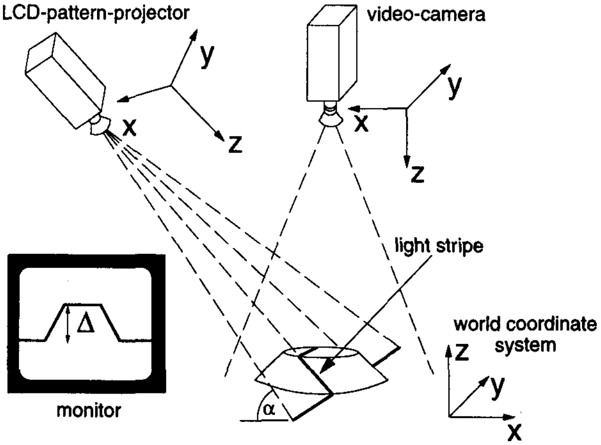


Figure 4.2: *Example of a range-sensor with structured light*

only simple operations are needed, fast processing can be achieved with adequate hardware [26]. Color coding has the advantage to require only one pattern to be projected, but its disadvantage is that strongly saturated colors in the scene make the identification of the coding difficult. However, because multiple images are used to generate one depth image, it is not suited to applications where depth images at video rate are necessary.

4.1.4 Shape from Shading

The image intensity of an object is a function of the object's surface-reflectance and orientation, the position and intensity of the light source and the position of the viewer. Provided the other parameters are given, the object's position can be estimated from one image. In case of multiple images with varying position, wavelength or polarization of the incident light source, the reflectance map and the orientation of a surface can be estimated as well. This method is strongly limited by the fact that the lighting must be completely controlled and it is most useful in applications where the lighting can be modeled by a single point source.

4.2 Passive Vision

Passive vision only uses the ambient light reflected by the scene. It is an adequate method for open spaces (e.g. geodesy) and less controllable environments (e.g. manufacturing hall). Passive methods have the advantage of producing no detectable signature and freeing the sensors from signal interference with other sensors, which is very important, if several sensors work close together. There are a number of techniques that provide three-dimensional information from a scene.

Several methods use only a single static visual image, such as "shape from (known) texture" and "distance from (known) target size". In "shape from texture", 3D information is obtained by analyzing the distortion a pattern undergoes through the imaging process. Because these methods need some a priori knowledge about the scene, they are not practical for the unconstrained scenes that are found in industrial environments.

In the "swept focus" technique a single lens with a very short depth of field is focused at different distances (or a fixed focus lens is used and the sensor moved) and the image is analyzed to get its in-focus areas at various distances. In spite of the fact that this method is not computationally intensive, it is not suited for many real time application because it requires many (depending on the attainable resolution) images.

However, with two (or more) distinct views of the same scene, more reliable 3D information can be produced by triangulation without any a priori knowledge.

4.2.1 Stereopsis

The basic principle of stereopsis is to take images from different viewpoints and triangulate the range using the position of identical scene features from different viewpoints [27]. The most difficult and time consuming task is solving the correspondence problem, namely, identifying features in the images taken from different viewpoints as images of identical features in the scene. The features can be divided into low- and high-level features:

- **Low level features:** local intensity, magnitude or direction of intensity gradient. The most direct approach is to compare the image intensity. However, this only yields reliable results if the "intensity constancy constraint" is fulfilled: this implies that an object point produces the same intensity in all images. Because of different cameras, frame grabber and viewing angle this is seldom true. Therefore either a mean or variance normalized correlation method is used, or, instead of intensity, a derived measure such as the gradient is used (see also Chap. 8). Correlation on low level

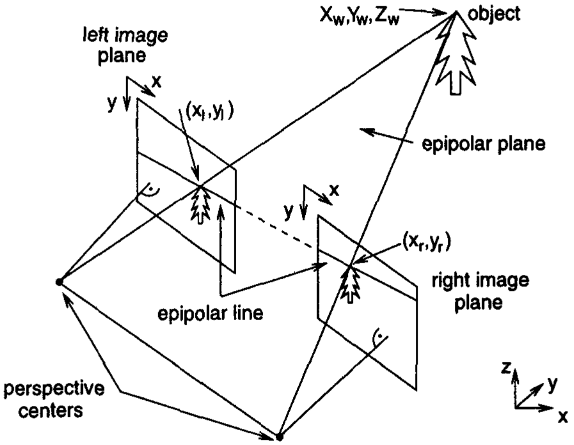


Figure 4.3: *Triangulation in passive stereo-vision (corresponding points are located on the epipolar line).*

features results in dense depth maps, but requires extensive computation because of the huge amount of potential correspondences which have to be checked.

- **High level features:** line segments, lines, corners, other geometric features or entire objects. First the high level features are extracted and parameterized. Subsequently these features are matched. High level feature matching is advantageous in that the extracted stable symbolic tokens do not presume the "intensity constancy constraint" and the correspondence search is less computationally intensive than for low level features, because the high level features are less frequent than image pixels. On the other hand, such methods only produce sparse depth maps.

It often happens that a feature in one image is matched by more than one feature in the other image. In order to reduce such ambiguity, (semi-) global constraints, such as the assumption of smoothness of the disparity map, are used. With such an additional constraint the number of ambiguous matches can be decreased. After the corresponding points have been established, the distance between the two locations in the images of the corresponding feature (= disparity) is calculated, and this is a measure of depth.

Stereopsis has two main problems: the occurrence of false and ambiguous matches and the huge amount of computational power which is used for searching for corresponding points. As a consequence, "real time" stereo vision for robot applications is very difficult to achieve. In order to make stereo vision amenable to real time, researchers have made several assumptions:

- Ideal lens: an ideal image process without any kind of distortion is assumed. This can be approximated with very good lenses with long focal lengths. However, with wide-angle lenses the effect of (radial) distortion is so dominant that it must be corrected.
- Epipolarity: Identical cameras with identical focal lengths and with coplanar alignment are assumed. This results in the ability to reduce the correspondence search to a one-dimensional search along a scan-line. This drastically reduces computation and makes hardware implementations much easier.
- Identical camera/digitizers: only with the use of identical cameras and digitizers, does the "intensity constancy constraint" hold true.

Research in the field of "real-time" stereo vision is very active. The expression "real time" does not determine a fixed speed, it only implies that all necessary processing has to be completed within given time limits. Some 'real time' systems that have been realized are sketched in the following:

- A stereo imager based on the extraction and matching of line segments with a performance of about 12 3D images with approximately 100 segments per second was presented in [28]. The system consists of an array (3×36) of FPGAs[1] for edge detection and tracking and a cluster of DSPs[2] for segment matching and false segment match elimination. Thanks to the correlation algorithm, no constraints on the camera setup are necessary and the method is robust to differences in the response of the cameras to illumination. However, the method supplies only sparse depth maps and, for scenes with a higher density of line segments (>100), the performance of the system decreases.
- A system producing dense depth maps but at an even slower speed is presented in [29]. The correspondence search is based on intensity correlation in a 5×5 window and subsequent elimination of false matches by assuming a smooth disparity map. In order to reach the "real-time" performance, the following assumptions were made: identical cameras, lenses and frame-grabbers are used and the cameras are parallel aligned such that the epipolar lines coincide with the scan lines. To achieve this high speed a Datacube has been used.

---

[1] Field Programmable Gate Array

[2] Digital Signal Processor

- A fast correlation module for stereo vision and object tracking is presented in [30]. It is based on binary correlation of the sign of Laplace-of-Gaussian (LoG) filtered images. The dedicated LoG convolver makes video rate convolution of an image possible and the correlator can produce 36 binary correlations in an 32×32 window in parallel within 100 μs. It is a very versatile system, useful at various combinations of speed and resolution: e.g. 16×20 stereo disparities plus confidence at a disparity range of 32 steps are calculated at a video rate of 30 frames/s.
- In [31] a system that combines change detection algorithms with stereopsis was implemented. In a first step, areas where a change occurred are detected. Subsequently a correspondence search is performed only in those areas in order to determine the 3-D position of moving objects. As long as only a small amount of the image changes, the amount of computation is decreased. However, changes far away from the interesting zone as well as shadows are detected by the change detection algorithm and trigger the 3-D analysis. In industrial environments, where neither the background nor the lighting can be controlled, this is a considerable drawback because it drastically increases the computation requirements.

4.2.2 Stereo from Motion

Stereo from motion is similar to stereopsis in many ways. It can be divided into two categories: (1) the camera moves and the environment is steady, (2) the camera is steady and the imaged objects move. In both modes an object is viewed from different viewpoints and therefore the distance can be calculated by triangulation. The first step in motion analysis is usually computation of optic flow, which is similar to the correspondence problem. It differs from stereopsis in that the search is not constrained (any movement is possible) unless the time between two consecutive images and the velocity of the movement is small, in which case the search is constrained by a prediction of the flow. By partitioning the movement of the camera into many small steps, the correspondence problem gets easier to solve [32]. For surveillance purposes it is possible to use a steady camera which calculates the image flow and calculates the position of moving objects. However, if no additional information about an object is available, it is not possible to calculate its distance because a slow moving small object near the camera cannot be distinguished from a fast moving large object far from the camera.

Method	Machine accessibility for visual and manual control	Adaptable to new workspace dimensions without re-installation	Cooperative work possible	Susceptible to being bypassed	Space requirements for installation	Real time processing (fast response time)	Resolution	Immune to false triggers (shadows, flashes, …)	Emits dangerous radiation	Suitability for sophisticated monitoring system
Ultrasonic sensors	good	yes	yes	no	low	yes	low	no[4]	no	no
Laser telemetry	good	yes	yes	no	low	limited[1]	high	yes	yes	limited
Laser triangulation	good	yes	yes	no	low	limited[1]	high	yes	yes	limited
Structured light	good	yes	yes	no	low	no[2]	high	partly[6]	no	no
Shape from shading	good	yes	yes	no	low	no[2]	high	no	no	no
Stereopsis	good	yes	yes	no	low	partly[3]	high	partly[5]	no	limited
Change driven stereo	good	yes	yes	no	low	partly	high	partly[5]	no	limited
Stereo from motion	good	yes	yes	no	low	partly[3]	high	partly[5]	no	limited

1) scanning of scene necessary
2) needs several images per 3D image
3) high processing power needed
4) is very susceptible to moving objects outside interesting area
5) problem with low textured scenes
6) depends on projection technology

Table 4.1: *Comparison of Sensors in Regard to Security Applications*

Chapter 5

Inverse Stereo Principle

Passive stereo vision has many advantages, but it is so computationally intensive that image processing at video rate is very difficult and expensive to implement. The proposed "inverse stereo principle" is much better suited to some applications, because it is much less computationally intensive than conventional stereo vision. The idea behind this approach and the necessary algorithms are presented in this chapter.

Only a three-dimensional remote sensing system is able to fulfill the requirements of a flexible monitoring system (see Chapter 2). The scene must be remotely sensed because a monitoring system must neither obstruct the view nor hinder the operators from their work. Only a 3-dimensional remote sensor makes an arbitrarily defined and dynamically adaptable workplace boundary possible. Passive 3D vision methods do not emit radiation and multiple systems do not interfere with each other: these are advantages over active vision. And in contrast to structured light approaches, it is not necessary to grab a series of images and therefore processing at video rate is only a question of available computing power, processing hardware and algorithms. In addition, there is no harm for humans as there is in strong laser scanning systems.

Apart from its many advantages, passive stereo vision has one crucial drawback: it is very computationally intensive because of the necessary search for corresponding points. Therefore processing at video-rate is at the very limit of today's computing power and only expensive supercomputers can cope with this computational load. In order to arrive at near real time, many researchers had to impose restrictions such as

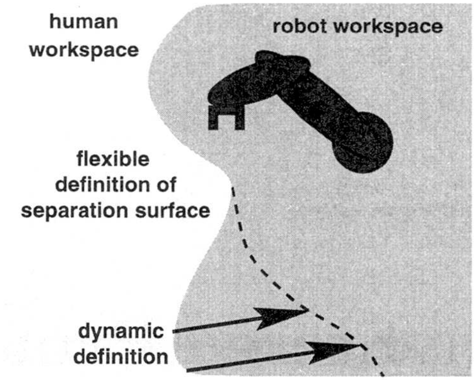


Figure 5.1: *Workspace of cooperative robot with separation envelope*

exact parallelism of cameras (in order to achieve epipolar lines parallel to scan-lines) and the use of telelenses to reduce lens distortions. However, such restrictions and the supercomputer requirement make use in industrial environments more difficult.

5.1 Idea

In many security applications, attention is focused on a small area in 3D space, e.g. in order to detect objects located in or entering the workspace of a robot. Therefore if one is interested in whether an object is at a specific location, it is inefficient to first produce a complete description of the scene and then analyze it for objects at specific locations.

In this thesis a method that overcomes these drawbacks is presented [33, 34]. It is a purposive method in that attention is focused on the location which is to be supervised and in that the low level image processing algorithm directly produces the desired result as output. This simplifies processing.

In order to supervise a robot and to allow cooperative work between a human worker and the robot, the robot is wrapped in a safety envelope consisting of a separation skin defined between the two workspaces and supervised for objects intruding the robot's workspace. The system then gives an alarm whenever an object penetrates the separation skin. The proposed method makes a flexibly defined and dynamically adjustable safety envelope possible. As illustrated in Fig. 5.1 it is possible to define

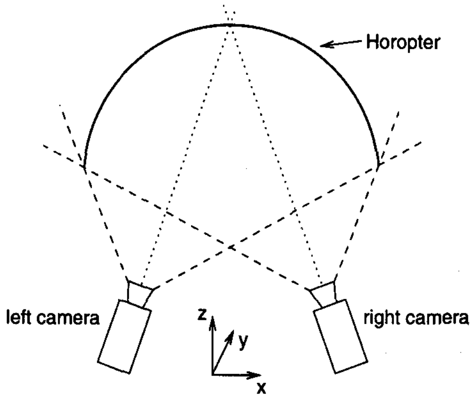


Figure 5.2: *Horopter of a camera setup*

the safety envelope as any shape and to change the shape according to the space requirements of the robot such that as much space as possible is available for human workers.

This method is much less computationally intensive than conventional stereopsis since the correspondence search is replaced by the generation and verification of a hypothesis, so that we can see if points of interest are located at a predefined surface. This method is related to the horopter idea and therefore the horopter principle is introduced in the following.

5.1.1 Horopter

There is, for two non-parallel cameras with parallel adjusted scan-lines, a locus of points in space (so-called horopter) which produces zero disparity between the two cameras and therefore results in corresponding images for both cameras (see Fig. 5.2). All objects located at the horopter come to identical locations in the images (because disparity is zero) and can therefore easily be extracted by zero disparity filters (ZDF). ZDFs could be used to extract objects in the horopter of a stereo camera rig [35] and play an important role in binocular gaze holding [36].

However, this horopter only exists if there is no rotation around the x- and z-axis of the cameras in respect to each other and in the case of an ideal perspective projection. Because the geometry of the horopter depends on the camera parameters and the setup of the two cameras, it is fixed for a given camera setup.
An appropriate transformation of one of the images allows us to change

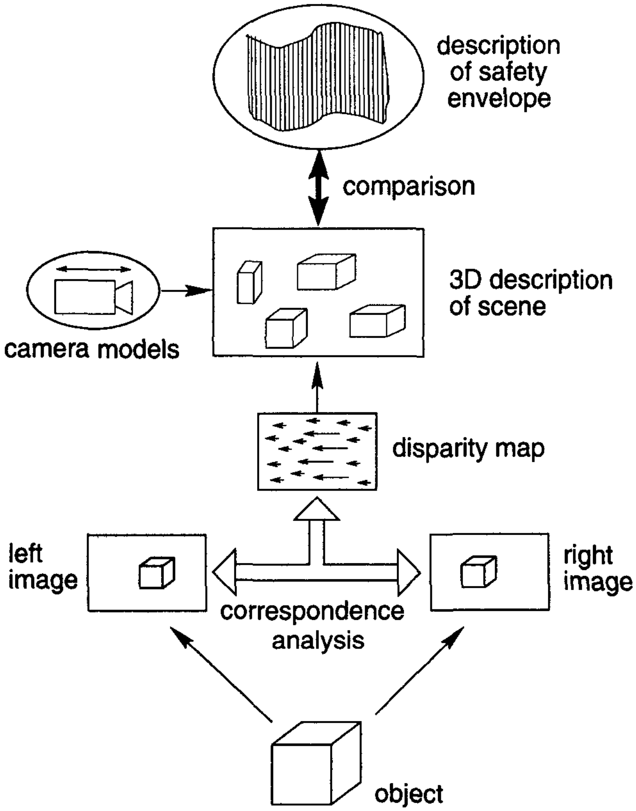


Figure 5.3: *Flowchart of Stereopsis: a complete 3D description is first produced and then compared with the safety envelope.*

the geometry (form, distance from camera) of the horopter and to define a pseudo-horopter of any geometry. In this way, a virtual horopter was laid on the ground in order to detect obstacles on a road in [37]. In addition, with an appropriate transformation, the multiple restrictions on the camera setup are no longer necessary.

5.1.2 The Method

As illustrated in Fig. 5.3, in stereopsis, the disparity map is first generated by a search of corresponding points (= points in the distinct images resulting from identical points in the object space). Given the camera model and this disparity map, the 3D description of the scene is then produced. In order to recognize the objects entering the workspace, the

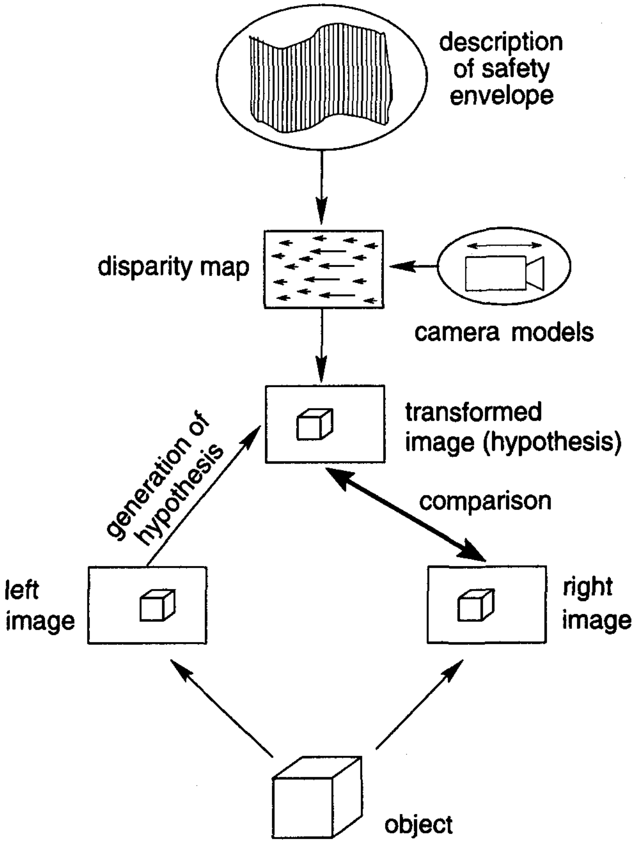


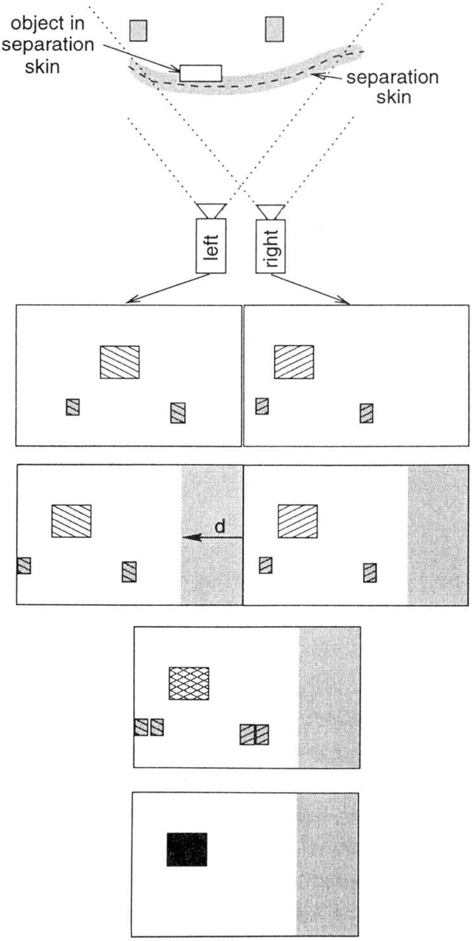
Figure 5.4: *Flowchart of Inverse Stereo Principle: a hypothetical image is produced and compared with the real image of the other camera.*

calculated object coordinates must be compared with the description of the safety envelope.

In contrast, in the "inverse stereo principle", there is no need for a correspondence analysis (see Fig. 5.4). A hypothetical disparity map is generated using the camera model and a 3D description of the separation skin (= location of hypothetical objects). Given this disparity map one of the camera images is geometrically transformed into the hypothetical image the other camera sees under the assumption that all objects are located within the separation skin (= pseudo-horopter). In order to extract objects located within the separation skin, the hypothetical image has only to be compared with the real camera image.

5.2 Algorithms

Figure 5.5 illustrates the basic principle with a simple scene consisting of three boxes of which one is located within the separation skin. The most important algorithms are presented in the following in the order of execution.



Setup with stereo camera rig and two objects outside and one within the separation skin.

Left and right images of the three objects.

The left image is transformed into a hypothetical right image (translation about *d* in the case of coplanar cameras).

The hypothetical image is compared with the real right image. The big white object is at identical positions because it is located within the separation skin.

The white object within the separation skin produces high correlation values and can be extracted by an appropriate threshold of the correlation values.

Figure 5.5: *Basic principle*

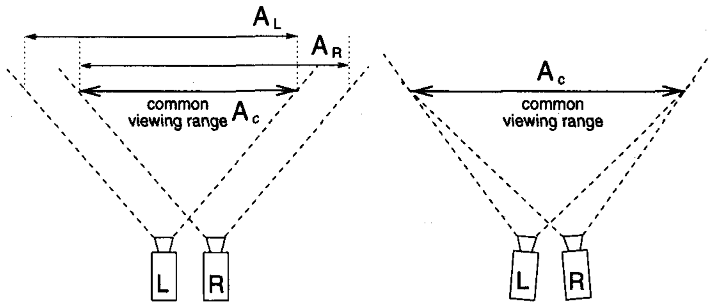


Figure 5.6: *Viewing range of parallel and non-parallel camera setup: with a non-parallel setup cameras can be adjusted such that $A_c = A_L$.*

Geometric Image Transformation

In the first step, one of the images is transformed to produce a pseudo-horopter which coincides with the supervised safety envelope, i.e. the image is geometrically transformed such that the hypothetical image coincides with the real camera image for all points at the separation skin. This transformation determines the geometry of the safety envelope. Any safety envelope that results in an unambiguous transformation could be defined. In addition, with the same transformation, lens distortion can be corrected and all geometric parameters of the camera setup (position and orientation) included. This has many advantages:

- wide angle lenses (often used due to restricted space) and inexpensive lenses with increased distortion can be used. This reduces system cost and broadens the application range.
- the camera setup is not restricted by the algorithm in contrast to many stereo vision systems which require parallel adjusted cameras. This, for instance, allows for the setting up of cameras such that the common viewing range (A_c) is maximized (see Fig. 5.6).

Correlation

After one image has been transformed into the hypothetical image, the real image is compared with this hypothetical image. Because calculating the difference between the images is not reliable and is too susceptible to noise, the images are compared by two-dimensional image correlation. A similarity (or dissimilarity) measure is computed for each 'point' (including its neighborhood) of both images.

Segmentation

Objects located within the separation skin correspond to regions with a high similarity measure. Since the correlation method was chosen such that it is mainly a function of the correspondence and translation of two templates, objects can be extracted with simple thresholding. However, due to noise, there will be erroneous pixels that lead to small non-existent objects or holes in objects. Thresholding with hysteresis or morphological operators could be used to eliminate isolated pixels and very small objects. The most common morphological operators, dilation and erosion with a 3×3 neighborhood, are defined as follows:

$$\begin{aligned} \text{Dilation}(x,y) &= \max_{i,j=-k}^{k} I(x+i,y+j) && (5.1)\\ \text{Erosion}(x,y) &= \min_{i,j=-k}^{k} I(x+i,y+j) && (5.2) \end{aligned}$$

Dilation and erosion have the disadvantage of only eliminating lonely high or lonely low pixels. Combined operators also exist: the so-called opening (erosion before dilation) and closing (dilation before erosion) operators. An operator that deletes all clusters that are completely within a rectangle of $(2e-1) \times (2e-1)$ pixels ("clusterlet elimination") performs much better than these operators. This operator can be mathematically described as [38]

$$\begin{aligned} h(i,j) &= u_0 \, \forall\, i,j : -e < i,j < e \quad \text{if} && (5.3)\\ & \quad f(-e,j) = f(e,j) = f(i,-e) \\ & \quad = f(i,e) = u_0 \, \forall\, i,j : -e \leq i,j \leq e \end{aligned}$$

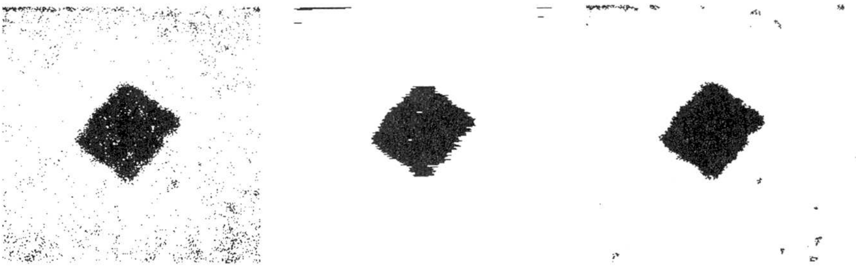
An operator that is easier to implement in hardware but produces better results than opening and closing is based on a majority decision and is described for the binary image $I(x,y)$ as follows:

$$M(x,y) = \begin{cases} 1 & \text{if } \sum_{k}^{i,j=-k} I(x+i,y+j) > 1/2\,(2k+1)^2 \\ 0 & \text{otherwise} \end{cases} \qquad (5.4)$$

In Figure 5.7 the results of various morphological operators on a very noisy thresholded image are presented.

Interpretation

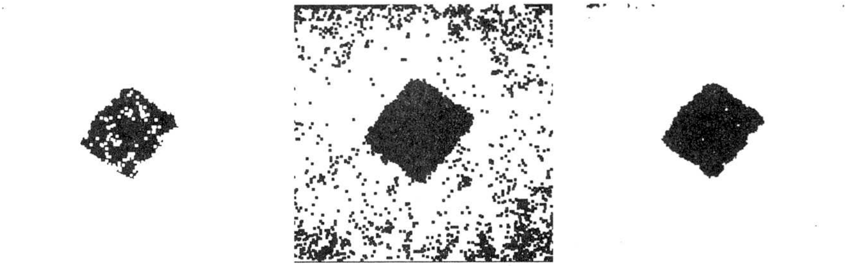
In order to make the results from image segmentation more reliable and robust, it is useful to include additional data (e.g texture intensity) in



Binarized image

Binarizing with hysteresis

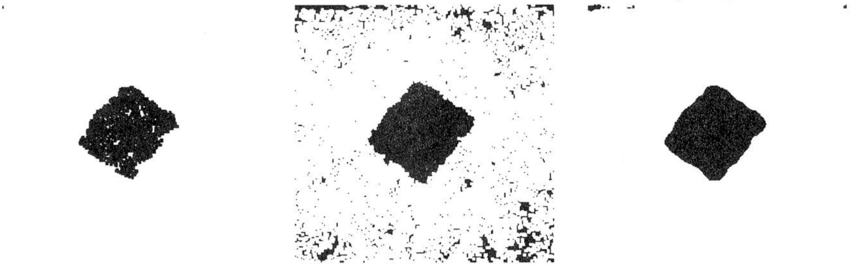
Clusterlet elimination



Dilation

Erosion

Majority 3×3



Closing

Opening

Majority 7×7

Figure 5.7: *Results of various morphological operators on a binarized image with high noise*

a subsequent process or to perform an image sequence analysis to make use of the dependence of subsequent images. A high level algorithm could calculate size, position or shape of detected objects.

5.3 Software Implementation

As a first step the method was implemented using the image processing development system KHOROS[1]. Several correlation and image transformation methods were implemented in order to test their suitability. Necessary resolutions of the data in the various processing steps were also evaluated in these tests. All the results of the investigations presented in the following chapters were obtained by using this software implementation.

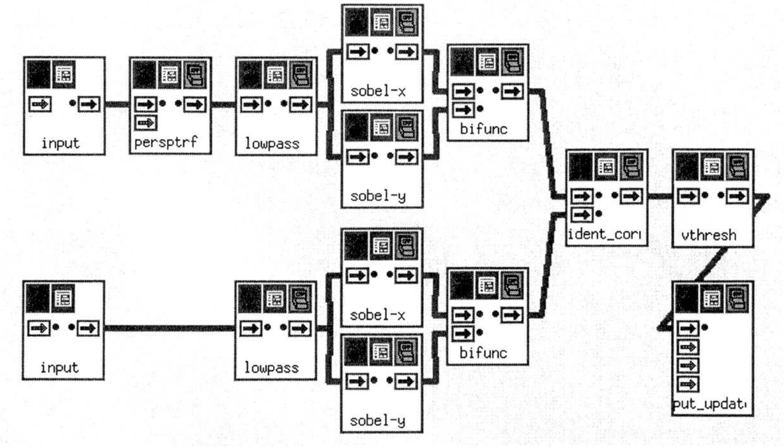


Figure 5.8: *Example of dataflow graph in Khoros: spatial transformation ('persptrf'), sobel-filtering ('bifunc' calculates argument of sobel gradient) and direction-correlation ('ident-corr') of image pair.*

---

[1] KHOROS is a registered trademark of Khoral Research, Inc., New Mexico.

Chapter 6

Camera Calibration

The camera rig needs to be calibrated in order to calculate the necessary image transformation. Various camera models and calibration methods are discussed and the method used is presented in detail. In addition, various subpixel estimation methods are presented and an analysis of the accuracy achieved is given.

In order to get the relation between an object and the corresponding image point (i.e. its coordinates), the imaging process is described by a mathematical model. In this application, this model is used to determine the necessary transformation for a given safety envelope.

These model parameters are determined in the calibration. There are various calibration methods and camera models suited to different applications. For this application a calibration method that meets the following criteria is needed:

- **Accuracy:** overall accuracy should be better than 0.2 pixel[1] for off-the-shelf wide-angle lenses. In order to achieve this accuracy,

---

[1] This accuracy depends on the translation tolerated by the correlation but not on the camera resolution unless the image is subsampled before the correlation. On the one hand it is of no use if the calibration accuracy is orders of magnitude smaller than other introduced errors. On the other hand, the errors introduced by the calibration should not be larger than errors introduced by the camera noise ($\cong$ equivalent position uncertainty < 0.15 pxl) and image transformation ($\cong\, < 0.2$ pxl). The calibration error can be split into two components, one of which is in the direction of the epipolar line and the other of which is perpendicular to it. The component parallel to the epipolar line only changes the location of the safety envelope, whereas the other component adds to the other errors. Because the calibration is performed on images with full resolution, the calibration error is divided by two.

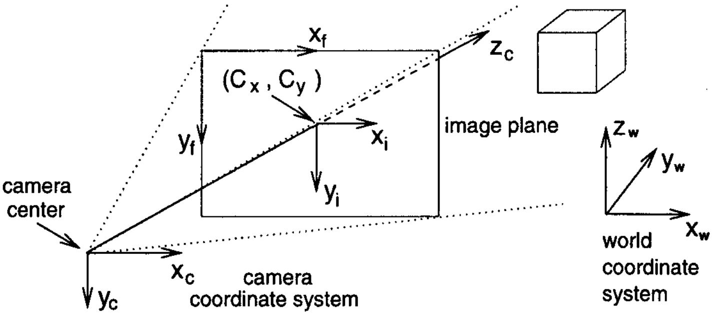


Figure 6.1: *Definition of coordinate systems*

the model should include lens distortion and the target must be measured with subpixel accuracy.

- **Versatility:** the method should be usable for a variety of camera setups and lenses, using off-the-shelf CCD cameras.
- **Autonomous operation:** The calibration procedure should not require any operator intervention such as providing an initial guess for certain parameters or selecting calibration targets in the camera images. An operator without special knowledge of calibration should be able to calibrate the setup.
- **Efficiency:** Calibration is done at the time of installation of the system and therefore need not be done in real time.

6.1 Coordinate Systems

Several coordinate systems are usually involved in the imaging process (see Fig. 6.1 for their definition):

- The world coordinate system (x_w, y_w, z_w) in which the safety envelope is described.
- The robot coordinate system which usually coincides with the world coordinate system.
- The camera-centered coordinate system (x_c, y_c, z_c) which is oriented such that the x-axis is parallel to the scan-lines and the z-axis coincides with the optical axis of the camera and points towards the scene. The origin of the camera-centered coordinate-system coincides with the optical center (= perspective center) of the camera.

- The image plane coordinate system (x_i, y_i) in which the metric image plane coordinates are given. It is a two dimensional coordinate system that is coplanar with the x-y-plane of the camera coordinate system, but is translated by f in the z-direction in order to coincide with the sensor plane.
- The framegrabber coordinate system (x_f, y_f) in which the pixel coordinates are given. This coordinate system has the same orientation as the image coordinate system but is translated and scaled such that the origin coincides with the upper left-hand corner of the image and the pixels come to lie on integer positions.

6.2 Pinhole Camera Model

The simplest camera model is the parallel projection model. However, this is only useful if the object distance is much larger than the object dimensions. A more precise and more frequently used model is the "pinhole camera model". It is based on an ideal perspective imaging process, modeling the lens as a pinhole and is based on the collinearity constraint, where object point (P), image point (P_u) and the camera center (O) lie on a straight line (see Fig. 6.2).

In the following, the "pinhole camera model" is derived. Let (x_w, y_w, z_w) represent the 3D coordinates of an object point **P** in world coordinates and (x_c, y_c, z_c) represent the same point in the camera-centered coordinate system (see Fig. 6.2). The world coordinates of point **P** are transformed into the camera coordinate system by a ro-

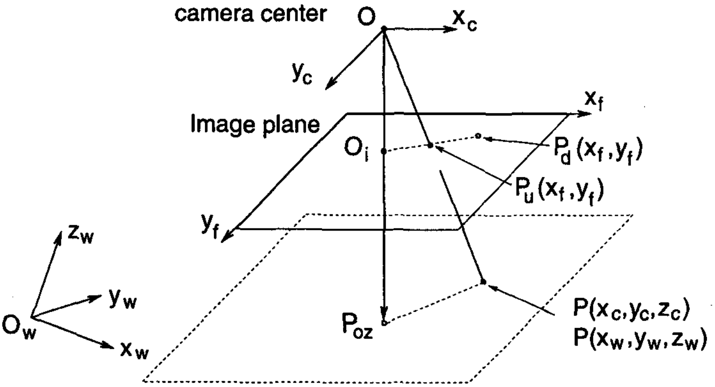


Figure 6.2: *Camera geometry with perspective projection*

tation and subsequent translation[2]. The relationship between the two coordinate systems is given by:

$$\begin{aligned} \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} &= \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \\ &= \mathbf{R} \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} + \mathbf{T} \ . \end{aligned} \tag{6.1}$$

In the next step the point is transformed from the three-dimensional camera coordinates to ideal (undistorted) image coordinates (x_u, y_u). This transformation is an ideal perspective projection:

$$x_u = f\,\frac{x_c}{z_c} \quad \text{and} \quad y_u = f\,\frac{y_c}{z_c} \ . \tag{6.2}$$

In the equations above the following parameters are used:

R	the 3×3 rotation matrix $(r_{i,j})$, expressed in world coordinates. It defines rotation around the x, y and z-axis (3 degrees of freedom).
T	the translation vector (t_x, t_y, t_z), expressed in the camera centered coordinate system.
f	the effective focal length, also called the camera constant. It defines the scaling of the image.

In the next step the metric image coordinates (x_u, y_u or x_d, y_d when considering lens distortions) are transformed into the frame-grabber coordinates (X_f, Y_f) by scaling and cropping (translating by (C_x, C_y)):

$$X_f = \frac{s_x}{d'_x}\,x_u + C_x \qquad Y_f = \frac{1}{d_y}\,y_u + C_y \tag{6.3}$$

$$\text{with} \quad d'_x = d_x \frac{N_{cx}}{N_{fx}}$$

---

[2] It is also possible to first translate and then rotate the point; **T** is then expressed in world coordinates, whereas here **T** is expressed in camera-centered coordinates.

where

(X_f, Y_f)	row (X_f) and column (Y_f) represent the number of the image pixel in the computer frame memory
(C_x, C_y)	coordinates of the true image center (so-called piercing point), expressed in frame-grabber coordinates
d_x	center to center distance (or pixel spacing) of adjacent CCD sensor elements in x-direction (scan line). This measure is calculated from sensor specifications. It is usually not equal to the sensor element size due to fill-factors less than 100%!
d_y	center to center distance of adjacent CCD sensor elements in y-direction
N_{cx}	number of sensor elements in x-direction
N_{fx}	number of pixels in a line as sampled by the computer. In case of pixel-synchronous sampling, N_{fx} is equal to N_{cx}.
s_x	uncertainty image scale factor that is to be calibrated. d_x and d_y are usually not known with absolute precision. True sensor spacing is expressed as $\tilde{d}_x = \frac{1}{s_x}\, s\, d_x$ and $\tilde{d}_y = s\, d_y$. The factor s is a scaling of the entire image and has the same effect as a change of the focal length and therefore can be included in f. Consequently only the factor s_x must be introduced as additional parameter.

Combining Equations (6.1), (6.2) and (6.3) leads to the following equations that transform world coordinates into pixel coordinates (distortions ignored):

$$X_f = \frac{s_x}{d'_x} f \frac{r_{11}x_w + r_{12}y_w + r_{13}z_w + t_x}{r_{31}x_w + r_{32}y_w + r_{33}z_w + t_z} + C_x \tag{6.4}$$

$$Y_f = \frac{1}{d_y} f \frac{r_{21}x_w + r_{22}y_w + r_{23}z_w + t_y}{r_{31}x_w + r_{32}y_w + r_{33}z_w + t_z} + C_y \tag{6.5}$$

This set of equations may be expressed in matrix form, using homogeneous coordinates[3]:

$$P_i = \begin{bmatrix} wX_f \\ wY_f \\ w \end{bmatrix} = M \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \tag{6.6}$$

with $M =$

$$\begin{bmatrix} r_{11} f \frac{s_x}{d_x} + C_x r_{31} & r_{12} f \frac{s_x}{d_x} + C_x r_{32} & r_{13} f \frac{s_x}{d_x} + C_x r_{33} & f \frac{s_x}{d_x} t_x + C_x t_z \\ r_{21} f \frac{1}{d_y} + C_y r_{31} & r_{22} f \frac{1}{d_y} + C_y r_{32} & r_{23} f \frac{1}{d_y} + C_y r_{33} & f \frac{1}{d_y} t_y + C_y t_z \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} .$$

The parameters **R** and **T** are called extrinsic parameters (6 degrees of freedom) and the parameters f, d_x, d_y, s_x, C_x, C_y intrinsic parameters.

6.3 Geometrical Lens Distortions

Imperfections in the design and assembly of cameras and their lenses lead to a variety of aberrations which affect image color, intensity, focus or geometry. For geometrical measurements camera distortion is a main concern because it directly affects the position of image features. The formulae for the pinhole camera must therefore be extended by a distortion term. Since only the distorted coordinates (x_d, y_d) are observable and one is usually interested in the undistorted coordinates (x_u, y_u), correction is expressed as a term $\delta(x, y)$ which is added to the observable distorted image coordinates $(x_d,\ y_d)$ to produce the undistorted coordinates $(x_u,\ y_u)$:

$$\begin{aligned} x_u &= x_d + \delta_x(x_d, y_d) && (6.7) \\ y_u &= y_d + \delta_y(x_d, y_d)\ . && (6.8) \end{aligned}$$

---

[3] The use of homogeneous coordinates is advantageous because perspective transformations can be expressed with the usual matrix algebra. In homogeneous coordinates a single component itself has no geometric meaning. The position of a point is defined by some quotient of the homogeneous components. Therefore a scaling of these coordinates does not change the position of the point. A point in three-dimensional space is defined in homogeneous coordinates as follows:

$$P = [X\ Y\ Z]^T \quad \rightarrow \quad P = [pX\ pY\ pZ\ p]^T\ ,$$

where p is any real scale factor. In order to get the point coordinate from homogeneous coordinates, all the components must be divided by the last component (p) of the coordinate.

To calculate the non-observable distortion-free coordinates, various distortion parameters are determined during calibration. All distortions are expressed in the metric camera-centered image coordinate system[4]. The three main sources of distortion are presented in the following.

Radial Lens Distortion

Radial distortion is strictly symmetric around the optical axis and causes an inward or outward displacement of the image points from their ideal position. A positive radial displacement is referred to as barrel distortion (a in Fig. 6.3), a negative one as pincushion distortion (b in Fig. 6.3). The radial distortion of a perfectly centered lens obeys the following equation [39]:

$$\delta_{radial} = \kappa_1 \varrho_d^3 + \kappa_2 \varrho_d^5 + \kappa_3 \varrho_d^7 + \cdots \tag{6.9}$$

where $\varrho_d = \sqrt{x_d^2 + y_d^2}$ is the distance from the principal point of the image plane and $\kappa_1, \kappa_2, \kappa_3$ are the coefficients of the radial lens distortion. Ignoring terms of an order higher than 3, the radial lens distortion is expressed by

$$\varrho_u = \varrho_d(1 + \kappa_1 \varrho_d^2) \tag{6.10}$$

or in Cartesian coordinates as

$$\begin{aligned} x_u &= x_d\,(1 + \kappa_1(x_d^2 + y_d^2)) \\ y_u &= y_d\,(1 + \kappa_1(x_d^2 + y_d^2)) \;. \end{aligned} \tag{6.11}$$

Decentering Distortion

Non-colinearity of the optical center of lens elements results in decentering distortion. This distortion has both radial and tangential components, which are described by the following expressions [39]:

$$\begin{aligned} \delta_{\text{decentering radial}} &= 3\,(j_1\varrho^2 + j_2\varrho^4 + \cdots)\sin(\phi - \phi_0) \\ \delta_{\text{decentering tangential}} &= 3\,(j_1\varrho^2 + j_2\varrho^4 + \cdots)\cos(\phi - \phi_0) \end{aligned} \tag{6.12}$$

---

[4] It is also possible to apply the correction on the framegrabber coordinates, but because the lens distortions does not depend on the pixel dimensions, the correction is usually applied on the metric camera coordinates.

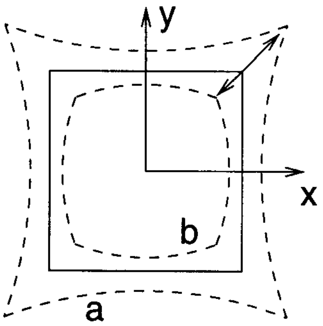


Figure 6.3: *Effect of radial lens distortion for negative (a) and positive (b) distortion parameter κ*

where ϕ_0 is the angle between the positive x axis and the axis of maximal tangential distortion. In Cartesian coordinates and by ignoring terms of an order higher than 3, Eq. (6.12) can be rewritten as

$$\begin{aligned} \delta_{\text{decentering } x} &= p_1(3x^2 + y^2) + 2p_2xy \\ \delta_{\text{decentering } y} &= 2p_1xy + p_2(x^2 + 3y^2) \ . \end{aligned} \tag{6.13}$$

Thin Prism Distortion

Thin prism distortion arises from imperfections in the lens as well as camera assembly (e.g. slight tilt of the image sensor). This type of distortion got its name from the fact that it can be adequately modeled by the adjunction of a thin prism to the optical system and is expressed as

$$\begin{aligned} \delta_{\text{thin prism radial}} &= (i_1\varrho^2 + i_2\varrho^4 + \cdots)\sin(\phi - \phi_1) \\ \delta_{\text{thin prism tangential}} &= (i_1\varrho^2 + i_2\varrho^4 + \cdots)\cos(\phi - \phi_1) \ . \end{aligned} \tag{6.14}$$

In Cartesian coordinates along the u and v axis and ignoring terms with an order higher than 3 it is expressed by

$$\begin{aligned} \delta_{\text{thin prism } x} &= s_1(x^2 + y^2) \\ \delta_{\text{thin prism } y} &= s_2(x^2 + y^2) \ . \end{aligned} \tag{6.15}$$

Radial lens distortion is usually the most important distortion component. The correction of radial lens distortion is more important for wide angle lenses (short focal length) than for telescopic lenses, as it increases with shorter focal length.

6.4 Calibration Methods

There are a huge variety of calibration methods and it goes beyond the scope of this thesis to give more than a short overview. A good overview of calibration methods is presented in [40].

Emphasis will be placed on methods used in robotics. Calibration methods will be divided into three groups according to the optimization methods:

6.4.1 Linear optimization

The nonlinear Equations (6.4) and (6.5), characterizing the transformation from 3D to 2D, can be treated as a linear set of equations if lens distortion is ignored and the coefficients[5] of the homogeneous transformation matrix (Eq. (6.6)), instead of the real geometric parameters of the model, are regarded as intermediate unknown parameters. Therefore, the over-determined linear system can be solved by a non-iterative least squares method such as singular value decomposition (SVD) [41]. If necessary, it is possible to recover the geometric camera parameters from the transformation matrix [42].

Besides the fact that distortions cannot be treated, the accuracy potential is limited in noisy situations. This is due to the fact that the number of unknowns (11) is larger than the degree of freedom (6 exterior and 2 internal parameters) and therefore the unknown parameters are linearly dependent. In the presence of noise, such redundant parameterization can lead to a good fit even for an erroneous combination of parameters.

6.4.2 Full Scale Nonlinear Optimization

It is possible to calculate the parameters of any arbitrarily complex camera model covering many types of distortions with an iterative optimization algorithm. However, a good initial guess is crucial because otherwise iteration may end up with a bad solution (local instead of global minimum). Therefore such methods violate the demand for autonomous calibration. In addition, the interaction between the distortion parameters and the external parameters can lead to divergence or to false solutions unless the process of iterations is properly designed [43].

---

[5] One of the 12 components should be set to a fixed value because the homogeneous transformation matrix is only determined up to a scaling factor. Therefore only 11 parameters must be determined.

6.4.3 Two Step Methods

Two step methods are a combination of the previously mentioned methods. These methods involve a direct solution for most of the parameters and either some iterative solution for the remaining parameters or an iterative solution for all parameters, using the direct solution as an initial guess. Because the approximate solution is used as an initial guess, the number of iterations is reduced and the optimal solution is reliably reached. The chance of finding only a local optimum is strongly reduced compared to methods with a manually provided initial guess. Tsai [40] and Weng [43] presented two different approaches. In this project an extension of Tsai's camera calibration [40] was used and it will be described in more detail in the following section.

6.5 Modified Tsai's Camera Calibration

In order to use a two step method, the mathematical model must be decomposed into two sets of parameters, one that can be solved using a direct algorithm and a second set that is solved by nonlinear optimization. In Tsai's calibration the "radial alignment constraint" (RAC) is used to establish this decomposition.

The radial alignment constraint[6] results from the observation that the vectors $\overline{O_iP_d}$, $\overline{O_iP_u}$ and $\overline{P_{oz}P}$ (see Fig. 6.2) are parallel to each other. P_{oz} is the intersection point of the camera coordinate z-axis with the plane parallel to the image plane and going through object point $\mathbf{P}$. It is important that not only $\overline{O_iP_u}$ but also $\overline{O_iP_d}$ is parallel to $\overline{P_{oz}P}$, because only $\mathbf{P_d}$ is observable. The point $\mathbf{O}$ must be known in advance and this is not trivial since the framegrabber center does not usually coincide with the lens center. However, since only a first guess of the model parameters is computed, taking the framegrabber center[7] proves to be sufficient. The radial alignment constraint is equivalent to $\overline{O_iP_d} \times \overline{P_{oz}P} = 0$, where × is the vector outer product. Therefore

$$(x_d, y_d) \times (x_c, y_c) = x_d\, y_c - y_d\, x_c = 0 \ . \tag{6.16}$$

Substituting x_c, y_c from Eq. (6.1) and replacing $x_d = \frac{1}{s_x} x'_d$ yields the

---

[6] For more detailed information about the radial alignment constraint and its proof refer to [40, 44].

[7] Consequently $C_x = 1/2 N_{cf}, C_y = 1/2 N_{cy}$.

following equation:

$$
\begin{aligned}
x'_d \frac{1}{s_x}(r_{21}x_w + r_{22}y_w + r_{23}z_w + t_y) &= y_d(r_{11}x_w + r_{12}y_w + r_{13}z_w + t_x) \qquad (6.17)\\
\text{with} \quad x'_d &= d_x(X_f - C_x)\\
y_d &= d_y(Y_f - C_y)\,.
\end{aligned}
$$

By rearranging the terms, the following over-determined system of linear equations with the unknowns $s_x r_{11}/t_y$, $s_x r_{12}/t_y$, $s_x r_{13}/t_y$, $s_x t_x/t_y$, r_{21}/t_y, r_{22}/t_y, r_{23}/t_y is obtained:

$$
\begin{bmatrix}
y_{d_1}x_{w_1} & y_{d_1}y_{w_1} & y_{d_1}z_{w_1} & y_{d_1} & -x'_{d_1}x_{w_1} & -x'_{d_1}y_{w_1} & -x'_{d_1}z_{w_1}\\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots\\
y_{d_n}x_{w_n} & y_{d_n}y_{w_n} & y_{d_n}z_{w_n} & y_{d_n} & -x'_{d_n}x_{w_n} & -x'_{d_n}y_{w_n} & -x'_{d_n}z_{w_n}
\end{bmatrix}
\begin{bmatrix}
t_y^{-1}s_x r_{11}\\
t_y^{-1}s_x r_{12}\\
t_y^{-1}s_x r_{13}\\
t_y^{-1}s_x t_x\\
t_y^{-1}r_{21}\\
t_y^{-1}r_{22}\\
t_y^{-1}r_{23}
\end{bmatrix}
=
\begin{bmatrix}
x'_{d_1}\\
\vdots\\
x'_{d_n}
\end{bmatrix}
\qquad (6.18)
$$

The parameters t_x, t_y, s_x and the 3D rotation matrix **R** are determined from these intermediate parameters (see [40] for a detailed description). In a second stage an approximation of the effective focal length f and the position t_z are determined by solving the following over-determined linear equation system:

$$
\begin{bmatrix}
r_{21}x_{w_1} + r_{22}y_{w_1} + r_{23}z_{w_1} + t_y & -y_{d_1}\\
\vdots & \vdots\\
r_{21}x_{w_n} + r_{22}y_{w_n} + r_{23}z_{w_n} + t_y & -y_{d_n}
\end{bmatrix}
\begin{bmatrix}
f\\
T_z
\end{bmatrix}
=
\begin{bmatrix}
y_{d_1}(r_{31}x_{w_1} + r_{32}y_{w_1} + r_{33}z_{w_1})\\
\vdots\\
y_{d_n}(r_{31}x_{w_n} + r_{32}y_{w_n} + r_{33}z_{w_n})
\end{bmatrix}\,.
\qquad (6.19)
$$

Equation 6.19 is derived from Eq. (6.5) with $\kappa = 0$ and substituting $y_d = d_y(Y_f - C_y)$.

Subsequently f, t_z and κ_1 are determined by minimizing[8] the sum of squares of errors of the nonlinear Equations (6.4), (6.5) and (6.12). The

---

[8] A modified Levenberg-Marquardt algorithm of the "minpack-project" (Argonne National Laboratory) is used.

solution received so far may be poor because the estimation of the image center (C_x, C_y) may be inaccurate and therefore the "radial alignment constraint" may not hold true.

To improve the solution, a better estimation of the image center should be used. One possibility is to search for a better estimation of the image center by minimizing the residual error of the radial alignment constraint, Eq. (6.17). This step has to be done before f, T_z and κ_1 are determined.

However, the method presented above has the disadvantage of providing a non-optimal solution because by using the "radial alignment constraint" only the tangential components of the points have been utilized and the radial components were completely discarded [43]. Therefore Tsai's method was modified by including a full optimization which calculates and improves *all* parameters of

$$\frac{d_x}{s_x}(X_f - C_x)(1 + \kappa_1 r^2) = f\frac{r_{11}x_w + r_{12}y_w + r_{13}z_w + t_x}{r_{31}x_w + r_{32}y_w + r_{33}z_w + t_z} \quad (6.20)$$

$$d_y(Y_f - C_y)(1 + \kappa_1 r^2) = f\frac{r_{21}x_w + r_{22}y_w + r_{23}z_w + t_y}{r_{31}x_w + r_{32}y_w + r_{33}z_w + t_z} \quad (6.21)$$

$$\text{where} \quad r = \sqrt{((X_f - C_x)\frac{d_x}{s_x})^2 + ((Y_f - C_y)d_y)^2}$$

using the approximate solution of the previous steps. This optimization is done in two steps. In the first step, all the points except those near the image center are used for the optimization. In the second step, the optimization is performed using all image points. The results of the calibration are presented in Chapter 6.7.

6.6 Subpixel Target Localization

For calibration, a set of targets with known positions in world coordinates and their corresponding image coordinates must be provided. The calibration should yield subpixel resolution and therefore the targets are measured with subpixel accuracy. Two different targets are mainly used for subpixel localization:

- Circular targets, where the center is measured by binary or grey-level centroid estimation or by elliptical contour fitting. One general problem with centroid-based techniques is that projection will cause the centroid to shift unless the object is coplanar with the image plane. Using grey-level centroid estimation, irregularly illuminated patterns may cause a shift of the location. Detailed information about centroid estimation is found in [45] and [46].

- Corners of square- or diamond-shaped targets. Because the corners of the shapes are measured, there is no error induced by perspective projection. In addition, such targets are very robust to irregularly illuminated patterns because the subpixel position is *locally* estimated.
  In order to measure the corner points, straight lines are fitted through the edges and the intersection of each two such lines yields the corner points. In case there is strong nonlinear distortion, the edges cannot be approximated by lines and an approximation with polynomial curves[9] should be used.

For the above mentioned reasons, diamond-shaped targets were chosen as fiducial marks the calibration. Since edge estimation with subpixel resolution is the most important task in target localization, subpixel estimation methods will be presented in more detail.

Subpixel or super-resolution estimation methods are used to estimate the position of an image feature (e.g. edges) to greater precision than that attainable within the restriction of discretisation. All methods use some filtering, model fitting, image reconstruction or interpolation techniques applied on the grey-level values defined at discrete locations. Common to all methods is that the edge location is first located to pixel precision. All subpixel estimation methods are based on the fact that the edge position is identical to the inflexion point of the edge profile. This follows from the fact that the point spread function (PSF) of the imaging process is symmetric around its intensity axis and the object intensity profile is symmetric around the (straight) edge location (see proof in [47]).

A subpixel estimation method for the purpose of target localization should fulfill the following requirements: It should

- be applicable to edges of any direction.
- yield good results for ramp edges of different widths (blurred edges !) and not only for ideal step edges.
- be invariant to multiplicative and additive changes in image intensity.

---

[9] The polynomial should be of adequately low degree such that noise is not modeled.

- be insensitive to noise. Therefore methods using derivatives (e.g. interpolation or center of mass of gradient operator) are disadvantageous.
- have inexpensive implementation. Therefore closed form solutions are preferable because they simplify calculation and reduce computational load.

In the following, several subpixel edge estimation methods are discussed.

6.6.1 Subpixel Edge Detection by Interpolation

Due to the fact that the inflexion point cannot be determined with linear interpolation, the subpixel location is assumed to be the intersection of the linearly interpolated intensity values[10] with the mean value of the ramp ($= (I_{top} + I_{bottom})/2$). I_{top} and I_{bottom} are the mean grey-level values at either side of the edge. Because interpolation is sensitive to noise, lowpass filtering of the image data should be considered. This method yields the best performance for smooth linear ramp edges.

6.6.2 Parametric Curve Fitting

An intensity edge can be reconstructed by fitting an appropriate parametric model to the sampled grey-level data. The (subpixel) location of the edge is equal to the inflexion point of this fitted curve. The following curves have been used to model edges [46, 47]:

- **Polynomials:** A polynomial curve or surface is fitted to the image data $f(x, y)$ in the neighborhood of the edge in a least squares sense:

  $$p(x,y) = \sum_{i=0}^{n} a_i(-x\sin\phi + y\cos\phi)^i = p(\mathbf{a},\phi) \ . \qquad (6.22)$$

  By minimizing $|f(x,y) - p(x,y)|$, the parameters of the model edge are obtained. The angle and direction of the edge are then derived from this model. Because a full polynomial model (degree of freedom equal to number of supporting values) can model anything including noise, a model of a lower degree is used [46].

---

[10] In practice there is no need to interpolate the entire profile and the edge position can be calculated directly from the two discrete intensities on either side of the subpixel edge position.

- **Sigmoids:** The parameters α, β, a and b of the equation

  $$S(x) = a\frac{e^{\alpha x+\beta} - 1}{e^{\alpha x+\beta} + 1} + b \tag{6.23}$$

  are determined by nonlinear least squares optimization. The inflexion point is given by $-\beta/\alpha$.

- **Arctangent:** In the same way the arctangent function

  $$A(x) = a\,\mathrm{atan}(\alpha x + \beta) \tag{6.24}$$

  can be fitted to the sampled image data.

- **Cubic Splines:** By fitting a spline, a smooth curve can be constructed from noisy points. In [47] polynomials of degree 3 were fitted. However, any other function differentiable up to the second order can be used as well.

Tests performed in [47] showed that the cubic spline interpolation yields the best results (according to maximal error), followed by the sigmoid method. However, all curve fitting methods are computationally intensive because they involve some non-linear optimization. ***

6.6.3 Edge Detection by Image Reconstruction

A sampled edge can be reconstructed using the sinc $(\sin(2\pi x)/(2\pi x))$ function. However, because of the slow decrease of the sinc function, many terms are required in the summation. By reconstructing[11] a *low-pass filtered* signal using the Gaussian reconstruction function

$$G(x) = \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{\frac{-x^2}{2\sigma^2}}, \tag{6.25}$$

performance in the presence of noise is increased and the computational expense decreased [46]. The subpixel edge position is obtained by calculating the center of mass of the (discrete) derivative of the reconstructed signal. The σ of the Gaussian filter should be chosen such that the reconstructed signal has no ripple; in the simulations $\sigma = 0.7$ was chosen. It is important that the support region is chosen large enough and that the area of the boundary decay is excluded for the calculation of the

---

[11] For realization, it is sensible to reconstruct a discrete image with an appropriate high resolution.

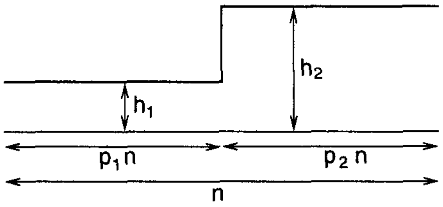


Figure 6.4: *Model of a one-dimensional edge profile*

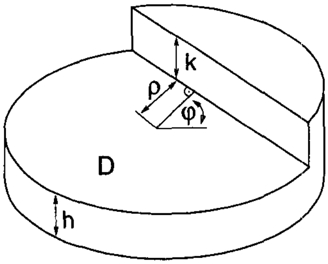


Figure 6.5: *Model of a two-dimensional edge profile on unit circle*

subpixel position.
According to tests presented in [46], results obtained with Gaussian reconstruction are much better than those obtained with polynomial fitting.

6.6.4 Moment-Based Edge Detection

All moment-based edge detectors fit a parametric model edge to the empirical edge data such that some moments are preserved. The main advantage of moment-based subpixel estimations is that they are entirely based on integration, which is advantageous over derivative type detectors because this reduces the effect of uncorrelated noise. The methods presented in the following differ in the moments used.

Tabatabai-Mitchell Operator

There are two versions of this operator: a one-dimensional and a two-dimensional one. In the one-dimensional case the parameter h_1, h_2, p_1 and $p_2 = 1 - p_1$ of the model (see Fig. 6.4) are calculated such that the following moments are preserved

$$\overline{m}_i = \frac{1}{n}\sum_{j=1}^{n} x_j^i = p_1 h_1^i + p_2 h_2^i \qquad i = 1, 2, 3\ . \tag{6.26}$$

In [48] it is shown that the subpixel position is invariant to grey-level offset and scaling. In addition, it is shown theoretically that noise and lowpass-filtering[12] of the edge profile moves the expected edge towards

---

[12] Nevertheless it is useful to lowpass-filter noisy images.

the center of the estimation window. Image blurring produces a similar effect on the edge position.

The one-dimensional method can be extended to a two-dimensional operator. The operator is applied on a grid of 69 pixels approximating the area of a unit circle. The model edge (see Fig. 6.5) is defined over a unit circle with two brightness values h_1 and h_2 separated by the line

$$y \sin\phi + x \cos\phi = \varrho \ . \tag{6.27}$$

The first three sample moments are calculated over the unit circle (D)

$$\overline{m}_i = \frac{1}{\pi} \int\int_D I^i(x,y) dx\, dy = h_1^j p_1 + h_2^i p_2 \qquad i = 0, 1, 2, 3 \tag{6.28}$$

where p_1 and p_2 are the fraction of the circle covered by intensity h_1 and h_2, respectively. The integral becomes a weighted sum if $I(x,y)$ is constant over one pixel. Analog to the one-dimensional operator, edge parameters are calculated by preserving the moments. The additional direction of the edge is calculated using the first geometric moments in the x and y directions. For a proof and derivation of this operator refer to [48].

Zernike Moment-Based Edge Detection

Zernike moments are special moments that use the circular polynomials of Zernike

$$V_{nm}(\varrho, \phi) = R_{nm}(\varrho)\, e^{jm\phi} \tag{6.29}$$

with
$$R_{nm}(\varrho) = \sum_{s=0}^{(n-|m|)/2} \frac{(-1)^s (n-s)!\, \varrho^{n-2s}}{s! \left(\frac{n+|m|}{2} - s\right)! \left(\frac{n-|m|}{2} - s\right)!}$$

which are orthogonal in the interior of the unit circle. These moments depend on the image data and properly chosen Zernike moments can be used to describe edges [49]. To calculate the Zernike moments A_{nm}, the neighborhood of an image pixel is mapped onto the interior of a unit circle[13] ($f(x,y)$ = mapped image) and then projected onto a set of complex polynomials

$$A_{nm} = \frac{n+1}{\pi} \int\int_{x^2+y^2 \leq 1} f(x,y)\, V^*_{nm}(\varrho, \phi)\, dx\, dy \tag{6.30}$$

---

[13] A neighborhood of any size can be mapped onto the unit circle.

$$= \frac{n+1}{\pi} \sum_x \sum_y f(x,y) \int_{p(x,y)} V^*_{nm}(\varrho, \phi)\, dx\, dy \qquad (6.31)$$

where $p(x,y)$ is the area of pixel (x,y) within the unit circle. For discrete images the moments A_{nm} can be expressed as convolution with a kernel M_{nm}, which is calculated according to Eq. (6.31). The step edge is defined as shown in Fig. 6.5. Using the polynomials

$$\begin{aligned} V_{00} &= 1 & V_{11} &= x + jy \qquad (6.32)\\ V_{20} &= 2x^2 + 2y^2 - 1 \end{aligned}$$

the Zernike moments A_{00}, A_{11} and A_{20} are obtained by convolving the image elements with the mask.

The edge parameters are calculated by preserving the three moments. For further information and a proof see [49].

6.6.5 Maximum Likelihood Edge Estimation

The likelihood of all possible edge patterns Φ_i is calculated in a region Ω around the edge

$$p(I|\,\Phi_i) = \frac{1}{(2\,\pi\sigma)^{N/2}} e^{-\frac{1}{2\sigma}\sum_{\Omega}(I(x,y)-\Phi_i(x,y))^2} \qquad (6.33)$$

and the most likely edge yields the subpixel position. Results presented in [50] show that this method is considerably superior to the Tabatabai-Mitchell operator under moderate and high noise conditions (SNR $=$ $\Delta H/\sigma < 10$) and comparable at low noise. However, the edge pattern can only be calculated if the PSF[14] and the shape of the edge are known a-priori. The assumption of a wrong edge shape especially deteriorates the result considerably: when a step edge is assumed the RMS error of 0.04 pxl increases to 0.1 pxl for a ramp width of 3 pxl or even 0.4 pxl for one of 6 pxl.

6.7 Results

Most step estimation methods make some assumptions about the form of the edge. The most restricting are the moment-based and maximum

---

[14] Point Spread Function of the imaging process.

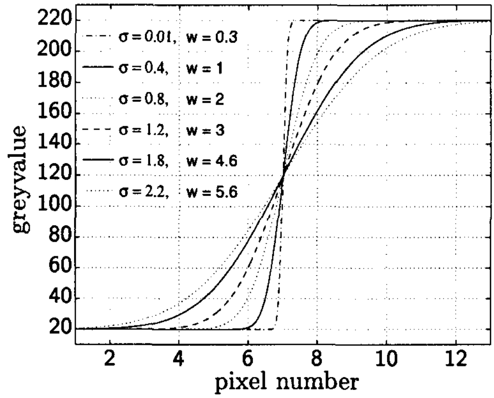


Figure 6.6: *Continuous simulated edge profiles, produced by filtering an ideal step edge with a Gaussian lowpass with σ = 0.01, 0.4, 0.8, 1.2, 1.8, 2.2 (w = width of ramp in pxl).*

likelihood methods, since they assume the edge to be an ideal step edge or a ramp of defined width. However, real edges are always ramp-shaped edges of various steepness due to diffraction and blurring. Especially in camera calibration for 3D applications there will most likely be some blurring since targets at different distances will usually not be in focus at the same time.

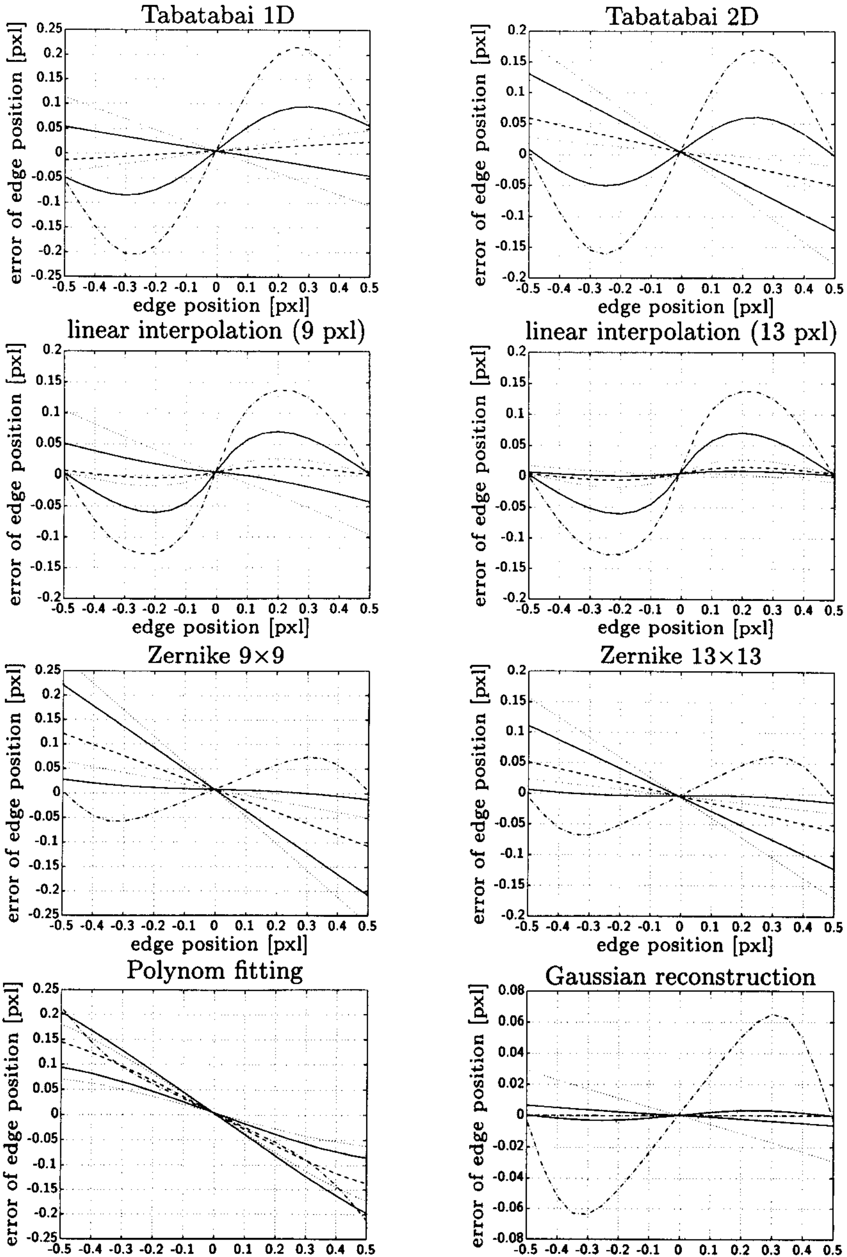
Because all operators produce erroneous results if the edge is not entirely within the support region, the allowed ramp width is restricted because of the limited support region on which the operators are applied to.

Therefore it is important to know the sensibility of various methods to non-ideal edges. First, some of the methods were tested for systematic errors by simulation with noise-free edge profiles of different ramp widths, then the operators were applied to real images of different blurring.

6.7.1 Performances on Ideal Edges

Artificial edge profiles are produced by convolving an ideal step edge with a Gaussian lowpass of varied σ (see Fig. 6.6). This lowpass models the effects of diffraction and blurring due to out-of-focus edge patterns. The discrete pixel values are then obtained by sampling the continuous edge profile. The sampling by the CCD sensor is simulated by sampling with a finite pulse (integration over pixel area) instead of an ideal impulse:

$$I(p) = \int_{p-f/2}^{p+f/2} I(x)dx \quad \text{with fill-factor } f = 0\ldots1\,. \tag{6.34}$$


Tabatabai 1D

Tabatabai 2D

linear interpolation (9 pxl)

linear interpolation (13 pxl)

Zernike 9×9

Zernike 13×13

Polynom fitting

Gaussian reconstruction

Figure 6.7: *Error of estimated edge position for various ramp widths (in pxl) as a function of true edge position (notice the different scalings of the error-axis !)*

In Figure 6.7 the errors of the estimated subpixel position of the tested methods are plotted against the true edge position[15]. Two different effects are visible:

- **steep ramp edges** (ramp width < 2 pxl, steepness > 100 intensity levels/pxl) produce position errors with a sinusoidal shape, which results from under-sampling of the true edge. Such steep edges were only observed in the y-direction (orthogonal to scanline direction) for analog cameras[16] when the camera settings are such that diffraction is minimal. Performance for such edges depends on the capability of reconstructing sparsely sampled signals. The Gaussian reconstruction and the Zernike methods show the best performance for steep edges, with a maximum error of less than 0.08 pxl for the extremely steep edges. Linear interpolation yields moderate error, whereas Tabatabai's methods yield the worst results, with an error of more than 0.15 pxl.
- with **smooth ramps** the problems that either the real edge does not agree with the assumption of an ideal step edge as a model for Zernike's and Tabatabai's methods or that part of the ramp is outside the support region of the method may arise. Both problems result in a shift of the estimated edge position towards the center of the support region for smooth ramps, which coincides with the effect of lowpass filtered edge profiles derived theoretically in [48]. Therefore in the case of smooth ramps the model of model-based methods should be changed or the support region of reconstruction-based methods enlarged.

  Enlarging the support region of moment-based methods also improves the result for smooth ramp edges and is easier to implement than an adaption of the model. The improvements result from the fact that by mapping an enlarged support region onto the unit circle the ramp gets scaled, which makes it steeper and therefore its shape closer to that of the ideal model edge. The results indicate that as a rule of thumb the support region should be three times the ramp width (compare results of 9 and 13 pixel-wide regions for Zernike's moments methods).
  Both the linear interpolation and the Gaussian reconstruction with large support regions produce very good results for smooth

---

[15] Equal to the edge position of the ideal simulated edge.

[16] This is due to the limited bandwidth of the signal path in analog cameras and framegrabbers. Digital cameras can produce such steep edges in both directions.

ramps. The Gaussian reconstruction yields good results because it is not model-based and is only based on the fact that the subpixel position is identical to the inflexion point. Linear interpolation anticipates smooth linear ramps and yields good results as long as the entire ramp is inside the support region (see worse results for a support region of 9 pixels).

The results of the Gaussian reconstruction are better than those obtained by other methods. The Gaussian method has the advantage of not needing a model edge and therefore this method is valid for all edge patterns. In addition, the lowpass filter characteristic of the Gaussian reconstruction results in good noise suppression. However, the main disadvantage is the great computational load due to the fact that a two-dimensional part of the image must be reconstructed. This method is especially suited to applications where high precision is needed and/or the edge shape is not known a priori.
It was not further evaluated in this project because of the great computational load.

6.7.2 Influence of Noise

The data presented so far was produced using synthetic data without noise. However, real images have noise from various sources:

- Shot noise of CCD sensor.
- Thermal noise of camera and framegrabber amplifiers.
- Noise induced by line jitter, which depends on the local derivative of the image signal in the scanline direction. As a consequence noise is especially high at edges perpendicular to the scanline and therefore edge location is greatly influenced by this noise source. With pixel-synchronous grabbing, line-jitter is reduced but not eliminated.

In order to measure the subpixel position error induced by noise, a series of 'identical' real images were taken and analyzed. The total grey-level noise[17] of the images proved to have approximately a Gaussian distribution with a standard deviation $\sigma = 1$.
Table 6.1 presents the mean standard deviation of all measured corners (x, y) and of the center of the fiducials (cx, cy) obtained with the various

---

[17] A "noise-free" image was produced by averaging about 100 images and the noise was then defined as the difference of the individual images to the averaged image.

Method	lin. interpolation	Tabatabai 1 D	Tabatabai 2 D	Zernike 9×9	Zernike 13×13
RMS error x	0.0178	0.0187	0.0258	0.0224	0.0247
RMS error y	0.0043	0.0068	0.0085	0.0089	0.0126
RMS error cx	0.0106	0.0111	0.0115	0.0113	0.0113
RMS error cy	0.0016	0.0026	0.0031	0.0027	0.0032

a) Results with square calibration targets

Method	lin. interpolation	Tabatabai 1 D	Tabatabai 2 D	Zernike 9×9	Zernike 13×13
RMS error x	0.1346	0.1651	0.0239	0.0229	0.0261
RMS error y	0.1310	0.1638	0.0175	0.0131	0.0151
RMS error cx	0.0262	0.0383	0.0151	0.0171	0.0188
RMS error cy	0.0209	0.0343	0.0070	0.0058	0.0058

b) Results with diamond shaped calibration targets

Table 6.1: *Position errors of real images (with noise): two-dimensional methods produce slightly better results with diamond-shaped fiducials.*

methods. It is obvious that the x-position has a higher error than the y-position. This is due to the fact that line jitter only influences edges perpendicular to the scanline, whereas only shot noise and thermal noise influence edges along both axes in a similar way.

The linear one-dimensional interpolation and the one-dimensional Tabatabai method were evaluated on a profile along the x- or y-axis. This yields good results as long as the edges are parallel to either the x- or y-axis. In the case of diamond-shaped fiducials this is no longer true and the results deteriorate as can be seen in Table 6.1.

6.7.3 Performance on Real Images

At a second stage the accuracy of real calibration targets using real camera images was investigated. For that, a calibration target was fitted on a linear robot and a series of images was taken with consecutive images being separated by about $1/20$ of a pixel in the y-direction. The targets have a size of 30×30 mm which corresponds to 64×64 pxl in the image. A ground truth for the target positions was fitted to the measured image coordinates of the target points by linear regression. Since the target is moved linearly, this produces a reliable ground truth against which the measured positions are compared. In order to test the

	one image series		all image series	
maximal residual of	x	y	x	y
mean incremental step	1.3%	0.04%	8%	0.8%
absolute value	0.1 pxl	0.1 pxl	1 pxl	1 pxl
fiducial width	0.1 pxl	0.1 pxl	0.35 pxl	0.3 pxl

Table 6.2: *Maximal deviations of calculated ground truths for each tested method and differently blurred image series.*

influence of blurred images[18] on the subpixel estimation, image series with different focus settings were produced and analyzed. The ramp steepness (intensity step per pixel) was manually measured. The steepness of ramps in the y-direction was shown to be approximately 60% to 80% higher because of reasons mentioned earlier in this chapter.

In Table 6.2 the residuals of this ground truth (start value and increment) among the different methods and image materials are presented. It can be seen that the residuals of one image series with different methods is much smaller than those among different images series. This shows that the various subpixel estimation methods produce consistent results and that the main source of errors of the ground truth is different blurring and slight changes in the scaling of the images (→ fiducial width) and inexactness in producing the image series (→ error of absolute position is larger than error of relative fiducial widths). The larger relative errors in the x-direction are due to the much smaller translation in the x-direction. Figure 6.8 shows the position error in relation to the calculated ground truth as a function of ramp-steepness.

In all results presented so far, 16 subpixel positions were estimated along each edge of the squares. The accuracy of the subpixel estimation depends on the number of subpixel estimations carried out per edge. From Figure 6.9 it can be seen that more than 16 subpixel estimations produce only minor improvements of accuracy.

6.7.4 Conclusion

It can be seen that the discussed subpixel estimation methods do not perform very differently and there is no method that is the best for all applications. Although the Gaussian reconstruction seems to outperform other methods, it has the drawback of being much more compu-

---

[18] Blurring reduces ramp-steepness.

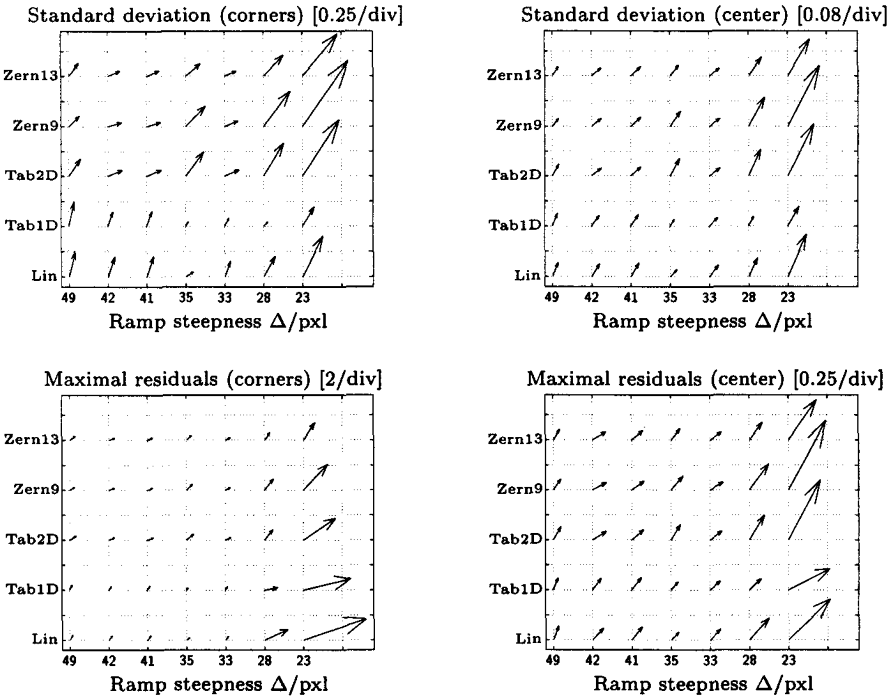


Figure 6.8: *RMS (in x- and y-direction) of position error (reference is calculated ground truth) as a function of method and blurring (i.e. ramp steepness* [$\Delta_{intensity}/pxl$]*).*

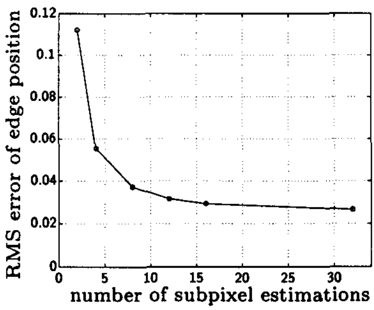


Figure 6.9: *RMS position error and position variation for different number of subpixel estimations per edge (all values given in pixel)*

tationally intensive.

However, the most important difference exists between 1D and 2D methods, which should be used for edges of arbitrary direction. The good results of the linear interpolation methods with large support regions are only obtained as long as the edges are perpendicular to either axis or the linear profile is laid perpendicular to the edge. The profile values must then be obtained by resampling, which increases the computational load. The same holds true for the 1-dimensional Tabatabai method.

In the implementation, the Zernike method with a support region of 13×13 was chosen because it performed well in relation to necessary computation.

6.8 Implementation

6.8.1 Calibration Pattern

In order to be able to calibrate all parameters of the camera model (including uncertainty scale factor s_x) a non-coplanar set of calibration points must be provided. Either a pattern with non-coplanar points or a pattern with coplanar points that is moved to several heights can be used. Whereas the first needs only one image and facilitates easier manipulation during calibration, the latter is easier to fabricate and was used in this project.

If the parameter s_x is known exactly (e.g. from an earlier non-coplanar calibration), it is possible to use only a coplanar set of points, which simplifies the calibration procedure. However, in this case it is very important that the plane with the calibration pattern is not parallel to the image plane[19].

In both, the coplanar and non-coplanar calibration, the world coordinate system should be positioned such that the world coordinate origin is set away from the origin and y-axis of the camera centered coordinate system. This so that $t_y \neq 0$, which avoids treating the case of $t_y = 0$ specially.

A pattern of 360 mm × 480 mm with 6 × 8 square white fiducials of 33 × 33 mm on a black background was produced. The precision of the targets is better than 0.1 mm. For non-coplanar calibration the calibration pattern is positioned at three different heights by using different distance pins (0, 50, 100 mm). The cameras are positioned about 1 m above the calibration pattern. The $1/2$ cameras are equipped with wide angle lenses of a focal length of 10 mm.

---

[19] In the case of a calibration pattern parallel to the image plane, the values of focal length and distance to the pattern cannot be resolved since both parameters scale the image in the same way.

The camera focus was set such that the middle calibration plane was slightly out of focus and the lens aperture was set such that no brightness clipping arises, which would very much deteriorate the results of the subpixel target estimation.

6.8.2 Automatic Target Localization

Manual selection of targets in an image is labor-intensive and prone to error. Therefore an automatic procedure where only the search window must be manually selected was implemented. The program then searches the square- or diamond-shaped targets and calculates the subpixel position of their corners.

In the initial phase, edge points of the squares are found by applying a kind of linear high pass filter along a grid of the image. Then a special search algorithm clusters and sorts the edge points belonging to a single fiducial. The initial guess for the four corner points is produced by calculating the intersection points of the straight lines fitted through these edge points by a least squares approximation.

Subsequently the edge position is determined with a subpixel estimation method along each edge at several positions. The final corner points are the intersection of the lines fitted to these subpixel edge points.

6.9 Results

In order to judge the accuracy and to compare different calibration methods with each other, an adequate measure is needed. A measure to indicate the attained accuracy of a calibration must not be influenced by change of focal length, object distance or stereo baseline. Weng et. al. [43] have proposed a new measure for calibration error which overcomes these problems. This measure, the normalized stereo calibration error (NSCE), is the ratio of the lateral error in 3D space and the standard deviation of the lateral digitization noise:

$$\mathrm{NCE} = \mathrm{NSCE} = \frac{1}{n}\sum_{i=1}^{n}\sqrt{\frac{(\hat{x}_i - x_i)^2 + (\hat{y}_i - y_i)^2}{\frac{r_{sp}^2 z_i^2}{12\,f^2}\left(\left(\frac{d_x}{s_x}\right)^2 + d_y^2\right)}}\;. \tag{6.35}$$

Lateral error is defined as the distance between the backprojected points $(\hat{x}_i, \hat{y}_i, \hat{z}_i)$[20] and the known coordinates of the calibration tar-

---

[20] All coordinates expressed in the camera-centered coordinate system.

gets (x_i, y_i, z_i), projected onto the xy-plane of the camera-centered coordinate system. In order to derive lateral digitization noise, imagine projecting each image pixel back onto a plane $(z = z_i)$ coplanar to the image plane and going through the backprojected point. This area indicates the uncertainty due to sampling at this distance. The uniform digitization noise in this rectangle of $a \times b$ has a variance of

$$(a^2 + b^2)/12 = z^2((\frac{r_{sp}s_x}{fd_x})^2 + (\frac{r_{sp}}{fd_y})^2)/12 \tag{6.36}$$

where r_{sp} is the attainable subpixel resolution (f, s_x, d_x, d_y according to Section 6.2).
For systems with only one camera the NSCE is not directly applicable, as z_i cannot be calculated from the image data. Therefore $(\hat{x}_i, \hat{y}_i, \hat{z}_i)$ is evaluated as the back-projection of the image point onto the plane $z = z_i$, and therefore $\hat{z}_i = z_i$.

The calibration was performed using three planes with calibration targets at different heights. Usually two of the planes provide the calibration points and one the test points. Five subpixel estimation methods (linear interpolation, Tabatabai's moment-based methods and the method based on Zernike moment) were used. In addition, two different targets, square-shaped targets with their edges approximately parallel to the image boundary and diamond-shaped targets, were used.

The following can be concluded from the calibration:

- The subpixel methods using one-dimensional profiles parallel or perpendicular to the scanline yield good results only when the edges are also adjusted along the image boundary. Therefore these simple methods cannot be used with diamond-shaped fiducials.
- Calibration results using diamond-shaped fiducials are slightly better than those with the square fiducials (only in the case of two-dimensional edge estimation !).
- The best accuracy of the test pattern is produced for the test-pattern between the two calibration planes. This shows that the calibration patterns should be positioned in about the same area as the camera model, for which the calibration was performed, is intended to be applied. However, additional error is relatively small.

selected Test-pattern	Methods lin. interpolation	Tabatabai 1 D	Tabatabai 2 D	Zernike 9×9	Zernike 13×13
highest	1.71	1.52	2.19	1.91	1.86
middle	1.17	1.07	1.72	1.46	1.26
lowest	1.68	1.44	1.66	1.42	1.53
all	1.12	1.02	1.67	1.38	1.16
difference	46 %	42 %	27 %	31 %	48 %

Results with square calibration targets

selected Test-pattern	Methods lin. interpolation	Tabatabai 1 D	Tabatabai 2 D	Zernike 9×9	Zernike 13×13
highest	7.07	3.46	1.77	1.68	1.38
middle	5.31	2.10	1.62	1.41	1.38
lowest	6.89	2.52	1.94	1.60	1.48
all	5.79	1.91	1.28	1.06	0.98
difference	33 %	65 %	20 %	19 %	7 %

Results with diamond-shaped calibration targets

Table 6.3: *Normalized Calibration Error (NCE) for various methods*

- When all the patterns were included in the calibration and the accuracy of all calibration patterns measured, the errors were reduced by about 25-40% and 0-35% for diamond- and square-shaped fiducials respectively. This is due to the fact that the parameters were optimized for the test-points, too.

In Table 6.3 the NCEs of the performed camera calibration are given. In addition, the image plane error is presented in Table 6.4. This is the difference between the measured image plane coordinate and the projection of the world coordinate onto the image plane, using the parameters of the camera model.

The NCE value obtained with our calibration is about the same as Weng et. al. [51] reported for the calibration for a wide angle lens[21] when only radial distortion was considered and a subpixel resolution of 0.2 pixel was assumed.

---

[21] Wide angle lenses (short focal length) usually have higher distortions. Therefore they normally produce worse results than long focal lenses if lens distortion is not perfectly modeled (→ uncorrected systematic errors).

	Methods				
selected Test-pattern	lin. interpolation	Tabatabai 1 D	Tabatabai 2 D	Zernike 9 × 9	Zernike 13 × 13
highest	0.14	0.13	0.18	0.18	0.15
	0.07	0.07	0.10	0.09	0.08
	0.34	0.35	0.46	0.42	0.43
middle	0.10	0.09	0.14	0.12	0.10
	0.05	0.05	0.07	0.06	0.06
	0.25	0.23	0.37	0.36	0.31
lowest	0.14	0.13	0.14	0.13	0.14
	0.06	0.06	0.07	0.07	0.06
	0.30	0.30	0.39	0.31	0.31
all	0.09	0.08	0.13	0.11	0.09
	0.05	0.04	0.07	0.06	0.05
	0.16	0.16	0.20	0.19	0.15

Results with square calibration targets

	Methods				
selected Test-pattern	lin. interpolation	Tabatabai 1 D	Tabatabai 2 D	Zernike 9 × 9	Zernike 13 × 13
highest	0.57	0.28	0.14	0.14	0.11
	0.19	0.09	0.06	0.08	0.06
	1.46	0.54	0.33	0.38	0.38
middle	0.43	0.17	0.13	0.11	0.11
	0.30	0.09	0.07	0.06	0.05
	1.48	0.44	0.29	0.32	0.29
lowest	0.56	0.20	0.16	0.13	0.12
	0.46	0.13	0.07	0.06	0.05
	2.03	0.68	0.34	0.32	0.24
all	0.47	0.15	0.10	0.09	0.08
	0.35	0.10	0.05	0.05	0.05
	1.06	0.36	0.16	0.16	0.16

Results with diamond-shaped calibration targets

Table 6.4: *Absolute mean value, standard deviation and maximal image plane error (in pixel) for various methods*

Chapter 7

Transformation

Image transformation is used to generate the hypothetical image. First the necessary transformations are derived for a plane and elliptical safety envelope. In the second part the possible transformation methods and their suitability are discussed.

7.1 Geometric Transformation

A spatial transformation is a mapping function that defines a geometric relation between each point in the input and output image. The mapping function can either be specified by an analytic expression such as an homogeneous transformation matrix (affine and perspective transformation), polynomial expressions (e.g. distortion correction), or by a dense grid of control points resembling a 2-D spatial look-up table (LUT), which defines any arbitrary mapping function.

In this application geometric transformation is used to transform one of the images into the hypothetical image, given the camera model and an analytical description of the safety envelope. For an arbitrary form of a safety, envelope the image may be transformed by back-projecting every point of one image plane onto the separation skin and projecting the resulting point in world coordinates onto the other image plane. However, this procedure is only necessary for geometries where no closed solution for the transformation exists. For plane and piecewise plane safety envelopes a closed form solution for the transformation can be derived.

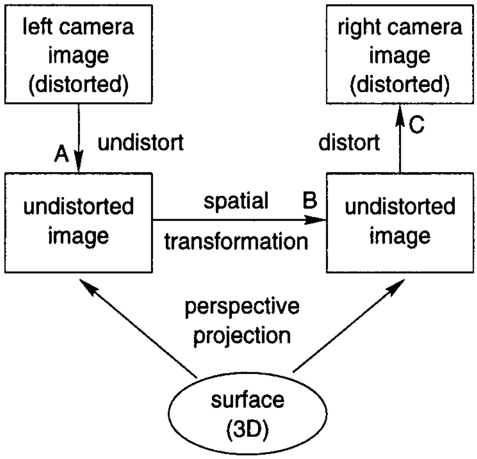


Figure 7.1: *Transformation from right to left image*

The entire transformation from the right to left camera image is broken down into the following steps (see Fig. 7.1):

A: transformation from distorted to undistorted coordinates of the right camera

B: spatial transformation according to safety envelope from right to left image

C: transformation from undistorted to distorted image coordinates of the left camera.

The transformation from distorted to undistorted image coordinates is a nonlinear transformation. Therefore it cannot be included in homogeneous image transformation and will be treated separately.
In order to calculate the distorted image coordinates (C), the equation

$$\varrho_u = \varrho_d(1 + \kappa_1 \varrho_d^2) \tag{7.1}$$

must be solved for ϱ_d which results in a cubic equation.

$$\begin{aligned} \varrho_d^3 + \underbrace{\frac{1}{\kappa}} \varrho_d + \underbrace{\frac{-\varrho_u}{\kappa}} &= 0 \\ \varrho_d^3 + p \; \varrho_d + q \quad &= 0 \end{aligned} \tag{7.2}$$

This cubic equation in ϱ_d can be solved with the Cardan method [52]. Depending on the determinant

$$D = \left(\frac{p}{3}\right)^3 + \left(\frac{q}{2}\right)^2 \tag{7.3}$$

the following real valued solutions are obtained:

$$\begin{aligned} D \geq 0: &\ \varrho_d = s + t && \text{with } s = \sqrt[3]{-q/2 + \sqrt{D}} \\ & && \phantom{\text{with }} t = -p/3s \\ D < 0: &\ \varrho_d = 2\sqrt[3]{\varrho}\cos((\varphi + 4\pi)/3) && \text{with } \varrho = \sqrt{-p^3/27} \\ & && \phantom{\text{with }} cos\varphi = -q/2\varrho \ . \end{aligned} \tag{7.4}$$

7.1.1 Plane

It is possible to derive an algebraic description of the transformation for a plane safety envelope. The arbitrarily positioned cameras are described by the homogeneous matrices $\mathbf{M_1}$ and $\mathbf{M_2}$ for the perspective transformation of the world-coordinates (X, Y, Z) into the image coordinates (b_1, b_2):

$$b_1 = \begin{bmatrix} w_1 u_1 \\ w_1 v_1 \\ w_1 \end{bmatrix} = \mathbf{M_1} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad b_2 = \begin{bmatrix} w_2 u_2 \\ w_2 v_2 \\ w_2 \end{bmatrix} = \mathbf{M_2} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} . \tag{7.5}$$

The plane $\mathbf{E}$ is described by

$$\mathbf{E} : aX + bY + Z = d \ . \tag{7.6}$$

Substituting Z from Eq. (7.6) into Eq. (7.5) yields

$$b_1 = \mathbf{M_1} \begin{bmatrix} X \\ Y \\ d - aX - bY \\ 1 \end{bmatrix} . \tag{7.7}$$

Combining the coefficients of X and Y we get a new transformation matrix describing the projection of a point on the plane $\mathbf{E}$ given by two-dimensional coordinates (X, Y):

$$b_1 = \mathbf{P_1} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \tag{7.8}$$

$$b_1 = \begin{bmatrix} a_{11} - a_{13}a & a_{12} - a_{13}b & a_{14} + a_{13}d \\ a_{21} - a_{23}a & a_{22} - a_{23}b & a_{24} + a_{23}d \\ a_{31} - a_{33}a & a_{32} - a_{33}b & a_{34} + a_{33}d \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} . \qquad (7.9)$$

For any given image coordinate (u_1,v_1) the world-coordinate (P_E) on the plane **E** projecting onto (u_1,v_1) may be calculated by computing the inverse of the square matrix P_1:

$$P_E = \begin{bmatrix} wX \\ wY \\ w \end{bmatrix} = \mathbf{P}_1^{-1} b_1 \ . \qquad (7.10)$$

Calculating the projection of this point P_E onto the other image plane is straightforward:

$$b_2 = \mathbf{P_2}\, P_E = \mathbf{P_2}\, \mathbf{P}_1^{-1} b_1 = \begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{bmatrix} b_1 = \mathbf{P}\, b_1 \ . \qquad (7.11)$$

Given this affine perspective homogeneous transformation the images of the two cameras may be transformed into one another.

7.1.2 Ellipsoid

A general ellipsoid with center (Z_x, Z_y, Z_z) and axis (A, B, C) is described by

$$\frac{(P_x - Z_x)^2}{A^2} + \frac{(P_y - Z_y)^2}{B^2} + \frac{(P_z - Z_z)^2}{C^2} = 1 \ . \qquad (7.12)$$

The corresponding transformation is non-linear and therefore cannot be expressed in matrix form. The transformation is produced by back-projecting every pixel onto the ellipsoid and projecting the resulting point onto the image plane of the other camera. A point on the line of sight going through the perspective point and an image point is described in parametric form in camera-centered coordinates by

$$^{c}P = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} + t \begin{bmatrix} (X_f - C_x)d_x/s_x \\ (Y_f - C_y)d_y \\ f \end{bmatrix} . \qquad (7.13)$$

Since the ellipsoid is given in world coordinates, ^{c}P must be transformed from camera-centered to world coordinates:

$$^{w}P = -R^{-1} \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} + tR^{-1} \begin{bmatrix} (X_f - C_x)d_x/s_x \\ (Y_f - C_y)d_y \\ f \end{bmatrix} = {}^{w}\mathbf{T} + t\,{}^{w}\mathbf{V} \quad (7.14)$$

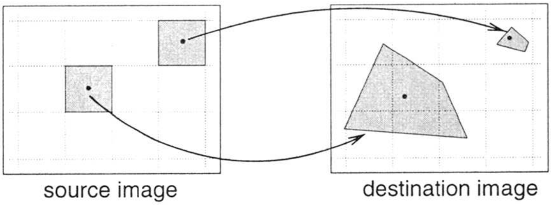


Figure 7.2: *Forward transformation by 4-corner mapping*

where R^{-1} is the inverse of the rotation matrix.
Substituting P_x, P_y, P_z in Eq. (7.12) with ${}^{w}P$, the resulting equation may be solved for t. Substituting the obtained solution for t in Eq. (7.14) yields the image point back-projected onto the ellipsoid.

7.2 Mapping Methods

The general mapping function can be given in two forms: either mapping the input coordinate onto the output (forward mapping) or vice versa (inverse mapping). For both methods at least one of the images (destination for forward mapping, source for inverse mapping) must be temporarily stored in order to facilitate arbitrary transformations.

7.2.1 Forward Mapping

Forward mapping consists of copying each input pixel onto the output image at positions determined by the mapping functions

$$[u, v] = [U(x, y), V(x, y)] \ . \tag{7.15}$$

Because the pixels are mapped from the set of integers to the set of real numbers[1], some additional mechanism is needed to cope with this. Most straightforward is rounding the coordinates $[u, v]$ to the nearest integer, which is called "nearest neighbor transformation". However, this results in missing values (holes) in the output image in the case of magnification or aliasing effects in the case of minification. Aliasing arises when some input pixels are discarded due to sparse point sampling. These artifacts are eliminated by transformation methods such as area sampling or supersampling.

---

[1] except in case of purely translation and rotation with a multiple of 90^{O}

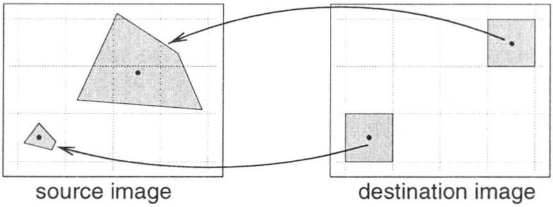


Figure 7.3: *Inverse transformation*

Forward transformation is useful when the input image must be read sequentially or when it does not reside entirely in memory (e.g. image minification). It is disadvantageous in that in case of area or supersampling the value of the destination pixel is not available until all neighboring pixels are transformed, which imposes a need for an additional accumulator array.

7.2.2 Inverse Mapping

The inverse transformation maps each output coordinate onto the input image via coordinate mapping

$$[x, y] = [X(u, v), Y(u, v)] \ . \qquad (7.16)$$

The value of the input image at point $[x, y]$ is then copied to the output image. As in the forward transformation, the coordinate $[x, y]$ is real-valued and therefore an interpolation stage must be introduced. By using inverse mapping, no holes arise in the output image, but nevertheless aliasing and blocking still occur. Analog to forward mapping, blocking and aliasing effects can be eliminated by more sophisticated sampling schemes.

Inverse mapping guarantees that all output pixels are computed and this is advantageous in that interpolation occurs in the input image which is a more convenient approach because no weighted summation in the output image is necessary and the destination pixels are directly calculated from the source image.

Whether forward or inverse transformation is a better choice also depends on whether a magnification or minification is predominant, on whether the scaling is uniform and on the hardware on which it is implemented. Spatial transformations and image resampling are discussed more accurately in [53].

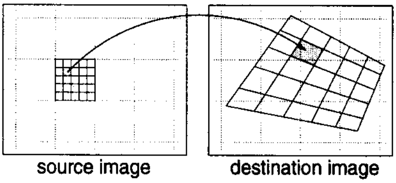


Figure 7.4: *Adaptive area sampling*

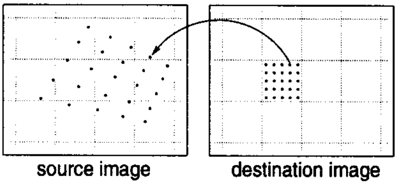


Figure 7.5: *Supersampling*

7.3 Eliminating Blurring and Aliasing

With simple point-to-point sampling without interpolation, artifacts such as blocking, aliasing and jagged lines arise. There are several methods for eliminating these artifacts. A simple extension of the point sampling method in inverse mapping eliminates blocking and jagged lines, but not aliasing: instead of taking the value of the pixel nearest to the transformed point, the value is interpolated using the neighboring pixels. In the following more methods are presented and in Table 7.1 the transformation methods and their characteristics are listed.

7.3.1 Area Sampling

In area sampling[2], instead of single points, square patches resembling the pixels are transformed into arbitrary quadrilaterals (see Fig. 7.2). In forward mapping the contributions of such quadrilaterals to each output pixel are summed up in an accumulator array[3] in order to correctly integrate the values contributed by the different source pixels. In inverse mapping the output pixel is a weighted sum of the input pixels covered by the projected quadrilateral. The weights are evaluated by an intersection test according to the pixel area covered by the quadrilateral. Thanks to the weighted sum, aliasing is eliminated but avoiding holes in forward mapping is bought at the price of blocking, since the same input value is applied to many output pixels.
The blocking effect can be resolved by two methods, as presented in the following sections.

---

[2] Also called "four-corner-mapping".

[3] Usually implemented as memory in conjunction with read-modify-write memory access.

7.3.2 Adaptive Area Sampling

Blocking in forward transformation can be eliminated by adaptive area sampling, where the input pixel is subdivided into smaller areas until the size of the transformed quadrilateral reaches some acceptably low limit (e.g. one pixel size)[4]. The intensity value of each sub-area is evaluated by interpolation.

7.3.3 Supersampling

The use of a supersampling grid is similar to adaptive area sampling but without the need of intersection tests. With supersampling more than one sample per pixel is transformed. The number of supersamples should also be chosen adaptively according to local scaling.

In forward transformation the values of the supersamples are accumulated into the appropriate output pixel. In the case of magnification a low number of supersamples may result in missing values.

In inverse transformation the value of the output pixel is calculated by averaging the interpolated values of the supersamples. If the number of supersamples is not chosen high enough, aliasing could arise because of discarded input pixels.

7.4 Interpolation Methods

In order to retrieve image intensity at an off-grid position, image reconstruction or interpolation is needed. Ideal signal reconstruction is performed with the *sinc* function. However, this is not practical, because it uses an IIR filter defined by a slowly converging infinite sum. Therefore either windowed *sinc* functions resulting in a finite sum or other interpolation techniques approximating a lowpass are used.

All methods discussed in the following assume that the inverse transformation is used. The methods differ in the quality of the result and the computational cost.

---

[4] The input must be adaptively resampled because uniformly sampling the input does not guarantee uniform sampling in the output image for non-affine (e.g. perspective) mappings.

Method	forward mapping		inverse mapping	
	scaling > 1	scaling < 1	scaling > 1	scaling < 1
Point-to-point sampling (no interpolation)	holes	aliasing	blocking	aliasing
Point-to-point sampling with interpolation	—	—	no blocking	aliasing
Area sampling	blocking	no aliasing	blocking	no aliasing
Adaptive area sampling	no blocking	no aliasing	—	—
Supersampling	blocking	no aliasing	—	no aliasing
Supersampling with interpolation	no blocking	—	no blocking	no aliasing

Table 7.1: *Overview of transformation methods and their reaction to scale changes*

7.4.1 Nearest Neighbor Interpolation

The nearest neighbour algorithm[5] is the simplest interpolation method. The value of the target point is set to the value of the point closest to the calculated exact position. It needs the least computational power of all interpolation methods as the output pixel is the function of only one input sample without further computation. However, this simple method leads to errors such as:

- blocking: since magnification is achieved by pixel replication, the image gets a blocky appearance.
- aliasing effects: minification by sparse pixel sampling leads to aliasing effects.

Therefore this method is not suited to applications where high quality is needed.

7.4.2 Bilinear Interpolated Transformation

With bilinear interpolation the target value is computed by linear interpolation in the x- and y-directions. The interpolated value is

---

[5] Other names denoting this method are: box filter, sample-and-hold-function, Fourier window.

a weighted sum of the four pixels nearest to the transformed pixel position according to:

$$\begin{aligned} I(x,y) &= (1-\Delta_x)(1-\Delta_y)\, I(x_0,y_0) \\ &+ (1-\Delta_x)\Delta_y\, I(x_0,y_0+1) \\ &+ \Delta_x(1-\Delta_y)\, I(x_0+1,y_0) \\ &+ \Delta_x\Delta_y\, I(x_0+1,y_0+1) \end{aligned} \tag{7.17}$$

where x, y $\in \mathcal{R}$
x_0, y_0 nearest integers to x, y, rounded towards $-\infty$
$I(i,j)$ value of pixel at location i, j
Δ_x, Δ_y $= x - x_0,\ \ y - y_0.$

7.4.3 Higher Order Interpolation

Interpolation by higher degree polynomials calculate the pixel value by fitting a surface of n^{th} order on the $(n+1)$ nearest pixels to the exact pixel location. Because polynomials of even degree are space variant[6] [53], usually only polynomials of odd degree are used. Experiments in [54] showed that cubic interpolation produces better results than windowed *sinc* or linear interpolation. Apart from reconstructing the image with the *sinc* function, a lowpass filtered image could be reconstructed with Gaussian reconstruction.

However, interpolation by higher order polynomials and by windowed *sinc* functions all need more source data points (up to 36 pixels for windowed *sinc* interpolation) to calculate a single interpolated pixel than bilinear interpolation. This increases both the necessary memory bandwidth and the computing power. Therefore these interpolation methods are not further considered here.

7.5 Measures to Simplify Transformation

The intersection test for determining the weights for area mapping is very computing intensive and difficult to implement in hardware. In supersampling, many points must be transformed and interpolated, which also increases the necessary computing power. However, if some a priori information about the mapping is available, the transformation can be

---

[6] This is due to the fact that the number of sampling points on either side of the interpolated point always differs by one.

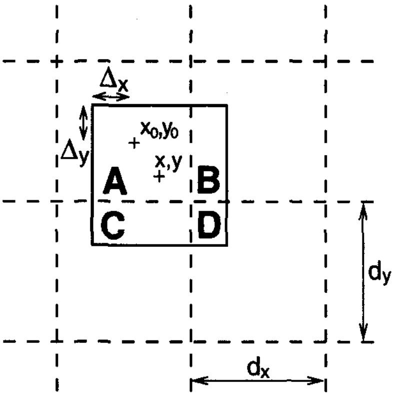


Figure 7.6: *Bilinear interpolation for a scaling invariant transformation*

simplified and the necessary computation reduced.
The transformation for the "inverse stereo algorithm" has the following characteristics on the basis of the most appropriate camera setup and smooth safety envelope:

- the scaling is ≈ 1 and therefore the transformed quadrilateral has about the same size as a pixel.
- the geometric distortion and rotation is small such that the transformed quadrilateral remains in rectangular form.
- the center of the pixel transforms to the center of the quadrilateral.

This results in the transformed quadrilaterals being approximated by a rectangle approximating a pixel. Therefore the computing intensive intersection tests are no longer necessary and the weights can be calculated as a function of the transformed center of the pixel (x, y). The partial areas in Fig. 7.6 corresponding to the weights are expressed as a function of Δ_x and Δ_y as

$$\begin{aligned}
A_A &= (d_x - \Delta_x)(d_y - \Delta_y) = (1 - \Delta_x)(1 - \Delta_y) \\
A_B &= \Delta_x(d_y - \Delta_y) = \Delta_x(1 - \Delta_y) \\
A_C &= (d_x - \Delta_x)\Delta_y = (1 - \Delta_x)\Delta_y \\
A_D &= \Delta_x\Delta_y = \Delta_x\Delta_y \ .
\end{aligned} \tag{7.18}$$

If d_x and d_y are set to 1, these factors exactly coincide with the weights of bilinear interpolation for position (x, y). Additionally, under the above mentioned assumptions, supersampling is analogous to bilinear interpolation since the number of samples within one pixel area are proportional to the area of this pixel covered by the transformed pixel. With this simplification the computing is decreased thanks to having eliminated the intersection test and having reduced the number of points which must be transformed per pixel (1 instead of 4).
This simplified method behaves like true point sampling with bilinear interpolation for scalings ≥ 1 (no blocking). However, for scalings $\leq 1/2$, some pixels are likely to be completely discarded, whereas for scalings $> 1/2$ all source pixels contribute to the output pixels, but with wrong weights. The errors of the weights are small for scalings only slightly smaller than one.
From the above it can be seen that the conditions stated at the beginning of this chapter can be relaxed and scalings greater and slightly smaller than one are acceptable.

So far it has been assumed that the weights are real numbers. The cost for hardware increases very much with the precision of the weights, especially for a real-time implementation in hardware.
However, for many applications, discrete weights with reduced resolution still produce reasonable results. The resolution necessary for obtaining sufficient results strongly depends on the application, the enlargement factor and the image material: transforming smooth images with minification or moderate magnification needs low resolution, whereas magnification of high-frequency images needs a higher resolution of the interpolation weights. Visible blocking[7] disappears when the quantization q_b of the interpolation weights is chosen according to

$$q_b \leq 1/scaling \ . \tag{7.19}$$

In order to get quantitative data about the errors introduced by bilinear interpolation, the following experiments were performed:

- An image was scaled with bilinear transformation with and without quantization of the weights. The error introduced by weight quantization compared to true bilinear transformation was measured for various scaling factors.

---

[7] Two or more pixels having erroneously the same value.

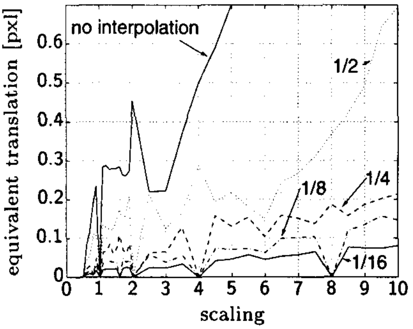


Figure 7.7: *Errors introduced by scaling with various resolutions of the interpolation factor.*

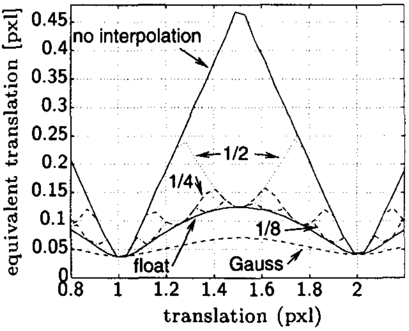


Figure 7.8: *Errors introduced by translation with various resolutions of the interpolation factor.*

- An image was translated by fractions of pixels and compared with the camera image[8] with the same amount of translation. With this procedure the error introduced by transformation is measured.

For both scaling and translation it is true that the standard deviation of the intensity-difference introduced by weight quantization is proportional to the quantization step. The measured standard deviation σ, which is a function of the high frequencies in the image, was $\sigma \approx 7$ for nearest neighbor transformation and $\sigma \approx 1.5$ for a quantization step of $1/4$. For translations and reciprocal scaling factors that are a multiple of the quantization step, bilinear interpolation with discrete weights produces the same result as true bilinear interpolation.

However, more relevant for this application is the influence of the quantization on the correlation measure. In the case of scaling, the image transformed with discrete weights was correlated with that produced with continuous weights. In the case of translation, the transformed images were correlated with the original camera images. The calculated mean value of the correlation measures of an image were compared to the correlation measures received when correlating two images translated by a certain amount of pixels. In Figure 7.7 and 7.8 the error introduced by the transformation is specified by the amount of translation that produces the same dissimilarity value.

---

[8] The image series for the correlation tests that consists of a sequence of real images (*not* shifted by calculation) shifted by fractions of pixels was used.

It can be seen that the transformation with a quantization step of $1/8$ introduces only a small additional error compared to true bilinear interpolation. Therefore in the hardware implementation the quantization step of the interpolation weights was chosen to be $1/8$. For comparative reasons, the equivalent translation of Gaussian reconstruction (the camera image was filtered with the same Gaussian filter that was used in the reconstruction) is given additionally in Fig. 7.8.

7.6 Change of Image Resolution

In order to obtain high precision, calibration was performed using images of higher resolution than those used in the monitoring system. This had to be taken into account in the transformation and the values for d'_x and d'_y were chosen accordingly. Because the transformation is applied on a single frame with half resolution in the y-direction of a full video image, $d'_y = 2d_y$ is used.

After the image is transformed, the images are subsampled in the x-direction such that the following processing steps work on images with equal resolution in the x- and y-directions, which is especially important for filters. In order to prevent aliasing effects, the images are subsampled by averaging every two pixels.

7.7 Alternative Calculation of Transformation

Besides deriving the transformation from the camera model, it is possible to determine the necessary transformation directly from an image of the separation skin. For planar safety envelopes, and when ignoring distortions, it is especially easy. The transformation for a plane of any orientation is a affine-perspective transformation:

$$\begin{bmatrix} sx_a \\ sy_a \\ s \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x_b \\ y_b \\ 1 \end{bmatrix} . \qquad (7.20)$$

If the corresponding coordinates of at least 4 points are known in both pictures, the parameters a_{ij} of the transformation can be calculated by a least squares method.

Chapter 8

Correlation

Correlation is a very important but also a very computationally intensive algorithm. First the various correlation methods and the requirements are presented. Then the correlation methods are rated according their characteristics and the results of our own investigations.

Correlation algorithms produce a measure for the similarity (according to defined criteria) between two signals (or images). Common to all methods is that it is not possible to calculate the similarity measure of a single point, but only of the neighborhood of a point[1]. There are many correlation methods that vary in characteristics and in computational requirements.

8.1 Correlation Methods

In the following the correlation between two templates, $r(x_r, y_r)$ and $s(x_s, y_s)$, is calculated, where r and s are sub-images of a given dimension of two images. u and v denote the coordinates relative to the sub-image, the so-called correlation window.

Correlation methods may consist of optional pre-processing (filtering) in order to extract special low level features such as edges or to emphasize some frequency range of the image (low-pass or high-pass filters). The correlation methods will be classified according to their pre-processing. Low-pass filtering is not treated as a special preprocessing and can be added to all correlation methods.

---

[1] Methods for symbolic template matching are not treated here.

8.1.1 Correlation on Brightness Images

There are a variety of correlation algorithms that can be applied directly to the intensity values. Many similarity measures (see Table 8.1) are based on direct or cross correlation (CCF):

$$\sum_{u}^{\text{window}}\sum_{v} r(x_r+u, y_r+v)\ s(x_s+u, y_s+v)\ . \tag{8.1}$$

However, CCF is not useful in applications with real non-preprocessed images because it very much depends on the brightness of the images and produces the highest score in bright image regions. Therefore several normalized cross-correlation functions were proposed.

In the normalized cross-correlation function (NCC) [55], the correlation is normalized with the mean of the signal energy in the two correlation windows and therefore is invariant to multiplicative intensity changes in either window[2]. However, it is still sensitive to additive changes in brightness. A correlation function that is invariant to additive *and* multiplicative intensity changes, the zero mean cross correlation (ZNCC) [56], is obtained by subtracting the mean intensity value of the corresponding correlation window from all intensity values. This algorithm is also called variance normalized correlation since the normalizing factor is the geometric mean value of the intensity values. Subtracting the local mean value is equivalent to high-pass filtering and therefore this method has similar characteristics as methods with high-pass prefiltered images (see Section 8.1.2).

Another correlation function was introduced by Moravec [57]. It is similar to ZNCC with the advantage of decreased computational requirements owing to the elimination of the square root. In addition, replacing multiplication in the denominator by addition has advantages in the case of limited resolution or for hardware implementations.

Another group of correlation functions are the dissimilarity measures based on the subtraction of the intensity values (see Table 8.2). There are two basic measures, the sum of squared differences (SSD)

$$\sum_{u}\sum_{v}[r(x_r+u, y_r+v) - s(x_s+u, y_s+v)]^2 \tag{8.2}$$

---

[2] Nevertheless, real (not computationally scaled) dark images may produce lower similarity measures because the SNR is usually lower than for bright images.

Name	Definition
NCC Normalized cross correlation	$\dfrac{\sum_{u,v}^{T} r(u,v)\, s(u,v)}{\sqrt{\sum_{u,v}^{T} r^2(u,v) \sum_{u,v}^{T} s^2(u,v)}}$
ZNCC Zero mean normalized cross correlation	$\dfrac{\sum_{u,v}^{T} [r(u,v)-\bar{r}]\,[s(u,v)-\bar{s}]}{\sqrt{\sum_{u,v}^{T} [r(u,v)-\bar{r}]^2 \sum_{u,v}^{T} [s(u,v)-\bar{s}]^2}}$
MOR Correlation according to Moravec	$\dfrac{2\sum_{u,v}^{T} [r(u,v)-\bar{r}]\,[s(u,v)-\bar{s}]}{\sum_{u,v}^{T} [r(u,v)-\bar{r}]^2 + \sum_{u,v}^{T} [s(u,v)-\bar{s}]^2}$
NISH Binary correlation according to Nishihara	$\sum_{u,v}^{T} r_{b1}(u,v) \cdot s_{b1}(u,v) \qquad (r_b, s_b = \pm 1)$

Table 8.1: *Correlation criteria based on cross correlation: high similarity produces high correlation values.*

and the sum of absolute differences (SAD).

$$\sum_{u}\sum_{v} |r(x_r+u, y_r+v) - s(x_s+u, y_s+v)| \tag{8.3}$$

These two dissimilarity measures are often used, because they are realizable with low hardware costs or computing power[3]. However, both measures are very sensitive to intensity changes. Therefore some modified difference measures were proposed: the local mean value scaled and the zero mean value versions.

---

[3] Whereas the SAD requires less resources in hardware implementation and less computing time for many microprocessors, the SSD can be implemented more efficiently than the SAD in modern DSPs since addition and multiplication can be computed in parallel (MAC operation).

Name	Definition
SSD Sum of squared differences	$\sum_{u,v}^{T} [r(u,v) - s(u,v)]^2$
ZSSD Zero mean sum of squared differences	$\sum_{u,v}^{T} [[r(u,v) - \bar{r}] - [s(u,v) - \bar{s}]]^2$
LSSD Locally scaled sum of squared differences	$\sum_{u,v}^{T} \left[r(u,v) - \frac{\bar{r}}{\bar{s}} s(u,v)\right]^2$
ZNSSD Zero mean normalized sum of squared differences	$\frac{\sum_{u,v}^{T} [[r(u,v) - \bar{r}] - [s(u,v) - \bar{s}]]^2}{\sqrt{\sum_{u,v}^{T} [r(u,v) - \bar{r}]^2 \sum_{u,v}^{T} [s(u,v) - \bar{s}]^2}}$
SAD Sum of absolute differences	$\sum_{u,v}^{T} \lvert r(u,v) - s(u,v) \rvert$
ZSAD Zero mean sum of absolute differences	$\sum_{u,v}^{T} \lvert [r(u,v) - \bar{r}] - [s(u,v) - \bar{s}] \rvert$
LSAD Locally scaled sum of absolute differences	$\sum_{u,v}^{T} \left\lvert r(u,v) - \frac{\bar{r}}{\bar{s}} s(u,v) \right\rvert$
NSAD Normalized sum of absolute differences	$\frac{\sum_{u,v}^{T} \lvert r(u,v) - s(u,v) \rvert}{\sqrt{\left\lvert \sum_{u,v}^{T} r(u,v) \right\rvert \left\lvert \sum_{u,v}^{T} s(u,v) \right\rvert}}$

Table 8.2: *Correlation criteria based on sum of squared and absolute differences: these dissimilarity measures produce low values (>0) for good correspondence.*

A combination of the ZSSD with the normalizing term of the ZNCC is the normalized zero mean SSD (ZNSSD, developed at INRIA [58]).

8.1.2 Correlation on High-pass Filtered Images

Instead of applying a correlation function that is robust in regard to intensity changes it is possible to preprocess the image data in order to suppress undesirable frequency parts of the signal. This usually makes the use of simpler correlation functions possible. By applying a zero-mean high-pass filter such as the zero-mean Laplacian operator

$$H_{\mathrm{Laplace}} = \begin{array}{|c|c|c|} \hline -1 & -1 & -1 \\ \hline -1 & 8 & -1 \\ \hline -1 & -1 & -1 \\ \hline \end{array} \quad , \tag{8.4}$$

all correlation functions become inherently robust in regard to additive intensity changes, since filtering with a (zero mean) high-pass suppresses the DC component of the signal. Because multiplicative intensity changes still influence the correlation measure, correlation on high-pass filtered images should be combined with correlation criteria which are invariant to brightness scaling[4]. In addition, the correlation gets more sensitive to position error between the templates because the emphasis of the correlation is placed more on the rapidly changing parts of the image. Highpass-based methods usually have low performance for small templates and strong noise [59].

8.1.3 Correlation on Direction Images

The application of correlation functions on the direction of the intensity gradient is very promising, as this makes the methods inherently robust to additive and multiplicative intensity changes [60]. Therefore, the correlation function itself must no longer be robust in regard to offset or scaling and one of the very simple correlation functions (SSD, SAD) can be used[5]:

$$\mathrm{DSAD} \quad = \quad \sum_{u}\sum_{v} |(r(x_r + u, y_r + v) - s(x_s + u, y_s + v)|_c \tag{8.5}$$

---

[4] The criteria must not be robust in regard to the sign of the values, because this is a property of the image and is no result of brightness scaling. This especially applies to locally scaled correlation criteria, where the factor $\bar{r}/\bar{s}$ must not take negative values. In case of different signs of $\bar{r}$ and $\bar{s}$, this factor must be set to 1.

[5] The use of more sophisticated functions is not sensible and even deteriorates the results.

$$\text{DSSD} = \sum_u \sum_v |(r(x_r+u, y_r+v) - s(x_s+u, y_s+v)|_c^2 \tag{8.6}$$

where $|\,|_c$ is the smallest absolute difference in direction, the cyclic difference.

8.1.4 Binary Correlation

Nishihara proposed in [61] a binary correlation. The binarization is done by filtering the image with the Laplace of Gaussian[6] (LoG) filter

$$\begin{aligned} \text{LoG}(x,y) &= \sum_{i,j=-k}^{k} c(i,j)\, f(x+i, y+j) \qquad (8.7) \\ \text{with} \quad c(i,j) &= \left(1 - \frac{4(i^2+j^2)}{w^2} e^{(-\frac{4(i^2+j^2)}{w^2}}\right) \end{aligned}$$

and using the sign $(+1, -1)$ as binary information. Because of the high-pass filter the result becomes robust in regard to additive intensity changes and the binarization makes it robust in regard to intensity scaling. The main advantage is the small computational requirements of the binary correlation (only 1 bit !). However, pre-filtering with the LoG is very costly due to the large filter kernels (up to 29 × 29). Consequently, the method is very promising for object localization, where the pre-processing is done only once per image, whereas the binary correlation is done for every potential template position. This advantage is not relevant for template matching where both LoG filtering and binary correlation is done only once. Nack proposed another binary correlation method, which Aschwanden [59] modified in order to improve the results. The binarization is done with an adaptive threshold on the high-pass (Roberts operator) filtered image. Experiments in [59] showed that Nack's correlation is outperformed by Nishihara's correlation.

8.2 Requirements

There are two main areas of application for correlation that make different requirements on the correlation methods:

---

[6] Also known as Marr-Hildreth or Mexican-Hat operator.

- In **template matching** an absolute measure for the correspondence of two templates is calculated. Because the result depends on a single value, it is important that the measure depends only on the image characteristics relevant for correspondence in a given application.
- With **template registration** the position of a template in an image is calculated. For that, the correlation measure is computed for various template positions in the image and the maximum similarity value yields the estimated template position. In contrast to template matching, only the relative correlation value is used and a scaling or offset of all correlation values[7] does not influence the final result.

Correlation measures have to cope with the fact that two templates are never identical[8] due to noise and other irregularities. Depending on the application, "correspondence" is differently defined and this leads to different requirements being placed on the correlation method. For one application, invariance in regard to rotation, smoothing or change of mean brightness is important whereas another application requires correlation criteria that is sensitive to these characteristics. Consequently, there is no single, "best" correlation for all applications.

The correlation measure is a function of many image characteristics. In the following the characteristics influencing the correlation measure are discussed with regard to the implementation of the inverse stereo algorithm.

- **Noise:** The images contain noise from different sources:
  - shot noise of the CCD sensor
  - thermal noise of the amplifiers
  - quantization noise of the AD-converter
  - noise resulting from line jitter, which is proportional to the local gradient in scanline direction.
  - noise induced by geometric transformations

---

[7] E.g. resulting from a global offset or scaling of the intensities of one or both images.

[8] This is why simple differencing is not useful for template matching or registration.

To examine the influence of noise, noise is modeled as additive zero mean Gaussian noise. Experiments have shown that total noise is approximated by this model. Noise of a high-contrast image with minimal camera gain[9] proved to have a Gaussian distribution with $\sigma \approx 1.2$.
Since noise is always present in real images, all correlation methods must have a high immunity to image noise.

- **Blurring:** Two cases must be distinguished: blurring which is equal in all stereo images and blurring which differs between an image pair. The same amount of blurring in both pictures has a similar effect as a lowpass filter and increases the tolerable disparity (→ increases thickness of separation skin), whereas different amounts of blurring decrease the similarity.
  The differently blurred images result from different lenses and focus settings, and mainly from different distances of an object to the cameras and scaling in image transformation[10]. The latter two effects occur only with non-coplanar cameras, where the same object might be in focus for one camera but out of focus for the other. Since the distance of an object to the individual cameras is similar (thanks to a small baseline in relation to object distance) and wide angle lenses with a large depth of focus are used, the difference in blurring between individual stereo images is small.

- **Scaling:** It is likely that an object is a different distance away from the cameras of a stereo-rig and therefore will be differently scaled. This is a problem in common stereo imaging systems and requires correlation methods which are insensitive to scaling.
  An advantage of the inverse stereo principle is that different scalings are corrected for objects located at the safety envelope. However, the scaling difference of an object increases with its distance from the separation skin. This results in a slight change of scaling for objects within the separation skin but not exactly at the separation surface. This change of scaling is very small: e.g. for a stereo baseline of 470 mm, an object distance of 1 m and a thickness of the separation skin of 50 mm, this results in a scaling difference of less than 1%.
  Therefore correlation criteria for the "inverse stereo principle"

---

[9] *High gain amplifies shot noise.*

[10] The blurring difference might be decreased by the spatial scaling of one image if the less blurred image is enlarged or increased in the other case.

must only tolerate a slight difference in scaling, which makes the use of simpler methods possible. An additional decrease of the similarity measure resulting from scaling difference is even desirable for objects outside the separation skin.

- **Rotation and perspective distortion:** In template matching for general stereo methods the correlation must be invariant to rotation[11] and scaling. Because the geometric transformation of the inverse stereo method corrects any rotation and perspective distortion between the cameras for objects at the separation surface, the correlation need not be invariant to rotation.
- **Change of brightness:** Different settings of the lens aperture, different gain of amplifiers and AD-converters and different reflection angles of an object to the cameras may result in multiplicative change of brightness. In addition, different offsets of the amplifiers and AD-converters result in an additive change. Therefore correlation methods must be robust in regard to such brightness differences between the stereo images. However, the correlation method need not be fully invariant to multiplicative changes of brightness since the difference in reflection factor (except in the case of specular reflection) and of the other parameters are bounded by physical restrictions or camera adjustments (similar aperture and gain for all cameras). Consequently, a correlation method that is only to some extent invariant can show an even better performance because templates with completely different brightness but similar information in the higher frequencies produce low similarity values.
- **Translation:** When two identical templates are translated against each other, the similarity measure decreases. This decrease of the similarity measure is a function of the translation and the shape of this function (see Fig. 8.1) depends on the correlation criterion used. The most suitable shape depends on the application:
  - coarse-fine or subpixel template registration needs a well-known function such that the subpixel position can be estimated from correlation values in the neighborhood.
  - template matching either needs a very narrow rectangular shape such that only exactly positioned templates produce

---

[11] It is possible to correct rotation by image registration and consequently the correlation does not need to be rotation invariant.

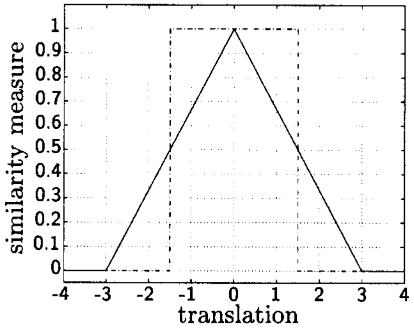


Figure 8.1: *Two ideal shapes of correlation functions: the triangle-shaped function represents a well-defined function, which can be used for estimating the translation from the correlation value. In contrast, the rectangular function provides no quantitative information about the translation.*

high correlation values or a wide rectangle when templates shifted by some amount should still be classified as corresponding.

However, the previously mentioned shapes do not occur in reality. In the "inverse stereo algorithm" a correlation criterion with a shape that allows for adjusting the required range of tolerated translations is necessary. This should be either a rectangle-shaped function width adjustable width or a triangle-shaped one where the tolerated translation is chosen by a threshold. The capability of a correlation method to separate templates according to their translation against each other will be called "separation capability".

- **Influence of brightness changes on other irregularities:** The same amount of correspondence (e.g. same translation, scaling, blurring) could produce different correlation values depending on the mean brightness common to both templates (= "common brightness"). For example. brighter images produce a higher dissimilarity measure when using SAD correlation on grey-level images, because the difference between neighboring pixels is also scaled by the global scaling in brightness. A correlation method that depends on the mean brightness of an object results in the thickness of the separation skin varying with the brightness of the object, which is intolerable.

8.3 Behavior on Unwelcome Influences

In order to decide on the correlation method most suited to the "inverse stereo algorithm", the robustness in regard to unwelcome influences must be analyzed. Some were analyzed theoretically, others by simulations and some could be taken from experiments carried out by Aschwanden [59]. Due to the fact that the performance of correlation methods depends on the image material, the absolute performance can only be predicted with exact knowledge about the expected scenes.

8.3.1 Brightness

Since in the "inverse stereo algorithm" the absolute correlation value is used, neither a common change in brightness of both images nor a change of brightness of one camera should alter the correlation measure.

The theoretical reaction to additive and multiplicative change in brightness is given in Table 8.3. We make a distinction between

	pre-processing		invariant to			
			common		different	
Criteria	LP	HP	offset	scaling	offset	scaling
NCC	×		no	yes	no	yes
NCC		×	yes	yes	yes	yes
ZNCC	×	×	yes	yes	yes	yes
MOR	×	×	yes	yes	yes	no[1]
NISH	BIN		yes	yes	yes	yes
SAD, SSD	×		yes	scaled	no	no
SAD, SSD		×	yes	scaled	yes	no[2]
DSAD, DSSD	DIR		yes	yes	yes	yes
ZSAD, ZSSD	×	×	yes	scaled	yes	no[3]
LSAD, LSSD	×		no	scaled	no	scaled
LSAD, LSSD		×	yes	scaled	yes	scaled
NSAD	×		no	yes	no	no
NSAD		×	yes	yes	yes	no
ZNSSD	×		yes	yes	yes	no[4]
ZNSSD		×	yes	yes	yes	no[4]

LP = no pre-processing or lowpass filtering, HP = highpass filtering
DIR = direction image, BIN = binarization by sign of LoG filtering
1) a scaling of 10% (LP) or 20% (HP) is tolerable
2) for SSD a scaling of 10% is tolerable
3) for ZSSD a scaling of 10% (LP) or 20% (HP) is tolerable
4) a scaling of 10% (LP) or 20% (HP) is tolerable

Table 8.3: *Behavior of correlation methods to brightness changes*

the case where the same change of brightness occurs in both images ("common") and when only one image is changed in brightness ("different"). A correlation criterion can either be robust in regard to brightness changes, change unsystematically or the result can be scaled by the same amount as the brightness of the image that has been scaled. That a decrease of brightness also decreases the signal-to-noise ratio[12] (SNR) and that therefore brightness scaled images may produce worse correlation measures, even for correlation criteria which are theoretically immune to brightness changes, is not taken into account.

8.3.2 Noise

Robustness in regard to noise mainly depends on the prefilter and on the size of the correlation kernel. In general, large kernels and lowpass-filters reduce noise whereas highpass-filters emphasize noise since noise is usually high frequency (see Table 8.4).

8.3.3 Blurring and Scaling

Blurred images result in a smoothed correlation function[14] and therefore in an increased tolerance for translation of one template against the other. This effect is dominant for highpass-based and zero-mean criteria since this reduces the available image information in addition to the reduction resulting from blurring. The results of experiments carried out in [59] are presented in Table 8.4.

8.4 Segmentation Capability

For the "inverse stereo algorithm" it is important that the correlation algorithm not only produce a high score for two exactly coinciding templates, but a translation between the templates below a given amount should also produce high similarity values. In addition, it must be possible to adjust the tolerated translation (by means of prefilters or thresholds) in order to adapt the method to various applications. The possibility of segmenting an image into regions with disparities below and above a given limit is especially important. Therefore it is very

---

[12] The dependency of the SNR on brightness very much depends on the camera characteristics (dynamic range, amplifiers) the images were produced with.

[14] $\cong$ correlation value in function of disparity.

Method	noise	scaling	blurring
intensity based	best performance	good performance	good performance, slightly worse for zero-mean and mean scaled methods.
highpass based	worst results up to kernel size of 17×17; slightly better for larger kernels	absolute worst results[15]	worst results
NISH	depends on w, medium performance for $w = 5$, else worse	bad performance, especially for low and high w.	worst results for $w > 6$, medium performance for $w < 6$.
DSAD	slightly worse than LP-based	good performance for medium sized kernels; slightly worse for small kernels	medium results

Table 8.4: *Behavior of correlation methods in regard to noise, image scaling and blurring.*

important to know the behavior of the correlation measure in regard to translations.

On the one hand this knowledge allows us to decide on the most suitable correlation method and on the other hand one can see whether a certain thickness of the separation skin ($\cong$ tolerated translation) is possible. We can also determine which correlation parameters must be used and what segmentation-error is to be expected.

8.4.1 Experimental Setup

The dependency of the correlation criteria on the translation must be evaluated for subpixel values and not only for discrete pixel translations since

- the tolerated translation of most criteria is between 0 and 2 pixel
- it must be possible to set the thickness of the separation skin to values which correspond to subpixel values

---

[15] Due to the fact that slight misalignment leads to completely misaligned edges.

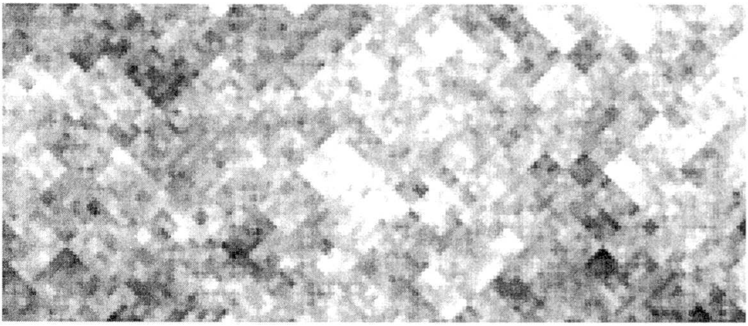


Figure 8.2: *Textured pattern used for experiments (original scale)*

- the various correlation criteria can be better rated if the translation function is not only known for discrete values.

Therefore a series of images shifted against a reference image in subpixel increments was produced by fitting a synthetic pattern of medium texture (see Fig. 8.2) on a linear robot. In this way a series of 700 images[16], shifted against one another by ≈ 0.04 pxl, was produced. In order to reduce noise in the images, five images of the same translation were averaged[17].

The middle image of this series was taken as a reference and correlated with all images from which the histogram, the mean value and the standard deviation were calculated. All the correlation criteria discussed in Section 8.1 with the optional lowpass filter (3×3) were used.

8.4.2 Definitions

Since the correlation values depend on the properties (texture intensity, image frequency, ...) of an image, which vary within the image, a single correlation value is not meaningful and therefore a statistical analysis of the correlation values of an entire image must be performed. However, since the histogram of the correlation values is not strictly Gaussian, the quality of the segmentation cannot be analyzed with only the knowledge of the statistical values (mean, standard deviation) Therefore the following new measures, based on the histogram of the correlation values, are introduced:

---

[16] Size of the images: 630×470 pixel, 8 bit intensity.

[17] Such that the standard deviation of the noise was reduced from $\sigma \approx 1.2$ to $\sigma \approx 0.6$.

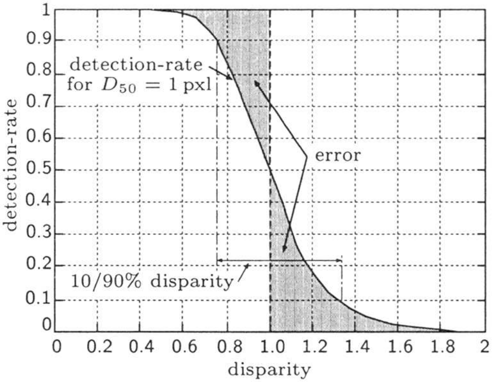


Figure 8.3: *Definition of new measures for a single detection-rate: the segmentation-error is defined as the difference from an ideal segmentation; the 10/90%-disparity is the disparity range where the detection-rate increases from 10 to 90%.*

- The **detection-rate** is defined as the percentage of pixels with a correlation value above a given threshold (T), which corresponds to pixels classified as being within the separation skin (disparity between hypothetical and real image below a given limit). It is given as a single percentage for one image (d). For a correlation criteria with Gaussian distribution it is expressed as

$$\begin{aligned} DR(d) &= \int_{c=T}^{\infty} H(c)dc = \frac{1}{\sqrt{2\pi}\,\sigma_d}\int_{T}^{\infty} e^{-\frac{1}{2}\frac{(x-\mu_d)^2}{\sigma_d^2}}\,dx \\ &\approx \frac{1}{2}\left(1-\frac{2}{\sqrt{\pi}}\int_{0}^{\frac{T-\mu_d}{\sqrt{2}\sigma_d}} e^{-t^2}dt\right) . \end{aligned} \tag{8.8}$$

- The **detection-rate function** is defined as the detection-rate in function of the disparity for a given threshold T (see Fig. 8.3). With this measure it is possible to estimate the quality of segmenting an image at a given disparity (= "segmentation-disparity") with the corresponding threshold value. The segmentation-disparity D_{50} is defined as that disparity where the detection-rate reaches 50% for a given threshold (T).

- The **segmentation-error** is a measure for the wrongly segmented pixels for a given threshold (≅ segmentation-disparity) and all disparities and is illustrated in Fig. 8.3. The segmentation-error function is defined as the segmentation-error as a function of the disparity at which the images are segmented. For correlation criteria with Gaussian distribution it is expressed as

$$SE(D_{50}) = \Delta_d \sum_{d=0}^{D_{50}} (1 - DR(d)) + \Delta_d \sum_{d=D_{50}}^{d_{max}} DR(d) \ . \qquad (8.9)$$

- The **10/90%-disparity** is the disparity for an increase of the detection-rate from 10% to 90%. This measure is not used in the analysis and for a linear detection-rate it is just $^{16}/_{5}$ of the segmentation-error.

Experiments have shown that the sobel-direction correlation (DSAD), the ZNSSD method and mainly the correlation criteria on highpass filtered images produce a histogram which is almost Gaussian such that the above defined measures could be calculated using the statistical values or the histograms. However, the locally scaled versions of SSD (LSSD, LSAD) and NSAD produce, even on highpass filtered images, wrong results when using the statistical values.

In the following these measure are discussed using the best (DSAD) and worst (NSAD with highpass) correlation method concerning their capability of segmenting images according to their disparity (called "segmentation-capability").

For all methods the mean correlation value linearly increases with disparity in a range near zero disparity (see Fig. 8.4). The size of this range and the standard deviation depend on the correlation method and prefiltering. A high ratio of the derivative of the mean in function of the disparity to the standard deviation

$$\frac{\frac{\delta mean(disparity)}{\delta disparity}}{\sigma(disparity)} \ . \qquad (8.10)$$

is important for a good segmentation-capability. The densely grouped histograms with their long tails towards positive values and the fact that they overlap to a great extent are the reasons for the high standard deviation and segmentation-error of the HP-NSAD correlation (see Fig. 8.4). This is also represented by the fact that there is no threshold where the detection-rate goes from 100% to 0% and therefore the minimal segmentation-error is very high (> 0.15).

In the following the relation between the histograms, the detection-rate functions and the segmentation-error is discussed in relation to the

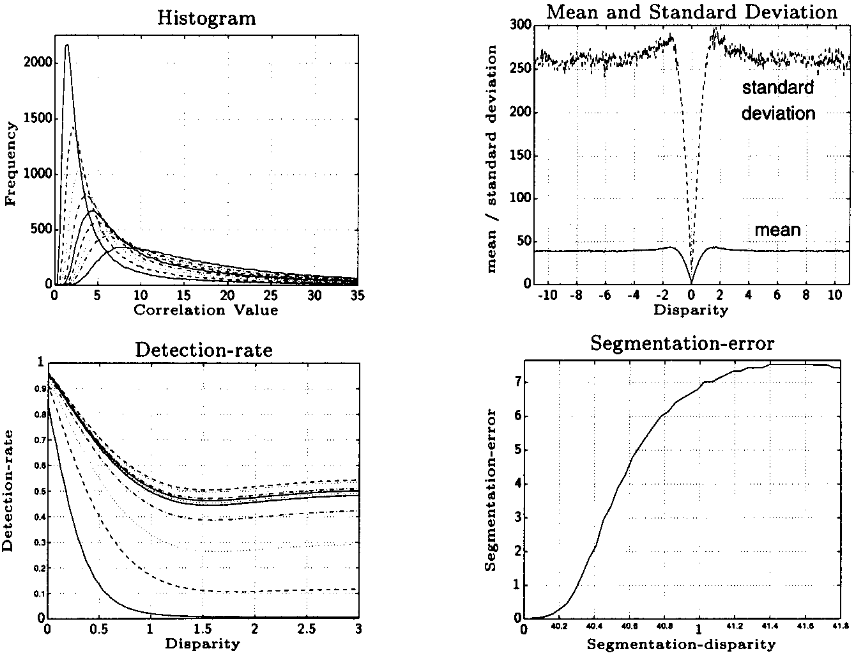


Figure 8.4: *Histogram, mean value and standard deviation of correlation values, detection-rate and segmentation-error for NSAD correlation on highpass filtered images (HP-NSAD).*

DSAD correlation (see Fig. 8.5). The histograms of the DSAD correlation are less densely grouped than those of HP-NSAD and standard deviation is smaller, which is also represented by smaller segmentation-error. In the plot of the mean and standard deviation it can be seen that the gradient of the mean as a function of the disparity is almost constant up to 3 pxl, whereas the standard deviation linearly increases in this range, which results in an increase of the segmentation-error.

The segmentation-error is approximately in inverse proportion to the gradient of the detection-rate and reaches very high values if the detection-rate does not converge to 0% for large disparities. It can be seen that segmentation-errors up to $\approx$ 0.1 show a very good segmentation-capability (see the detection-rate for D_{50}=1 pxl, which crosses 50% limit at 1.0pxl in Fig. 8.5). For larger segmentation-disparities the tail of the detection-rate gets longer and the segmentation-error increases very fast. A segmentation above 1.5 pxl (segmentation-error > 0.3) gives intolerable results (see Section 11.2.2).

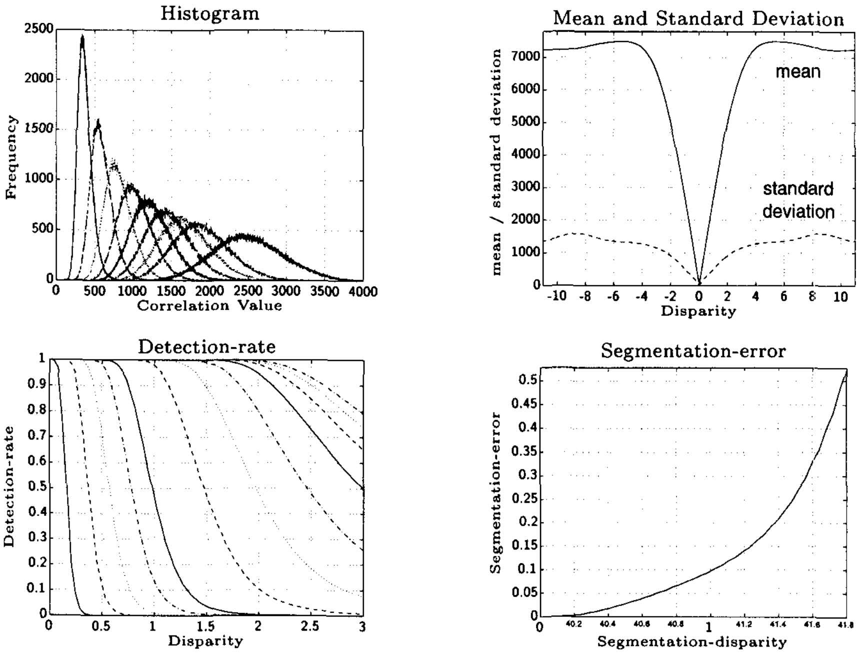


Figure 8.5: *Histogram, mean value and standard deviation of correlation values, detection-rate and segmentation-error for sobel-direction correlation with lowpass prefiltering (DSAD).*

To conclude, segmentation-errors below 0.2 show a good segmentation-capability, whereas segmentation-errors larger than 0.3 show bad performance. This is not a hard threshold because the segmentation-error slightly depends on the shape of the detection-rate function (e.g very small but tolerable detection-rate for large disparities increases the segmentation-error) and on the image material.

8.4.3 Influence of Prefilters

Most correlation methods may be applied to either intensity or highpass filtered images. The SAD correlation also works on the direction of the intensity gradient (DSAD). For all these methods the images can be optionally prefiltered with a lowpass-filter. An exception to this scheme is the binary correlation according to Nishihara (NISH), where the images are filtered with a Laplace of Gaussian filter (LoG) with a different parameter w (see Section 8.1.4).

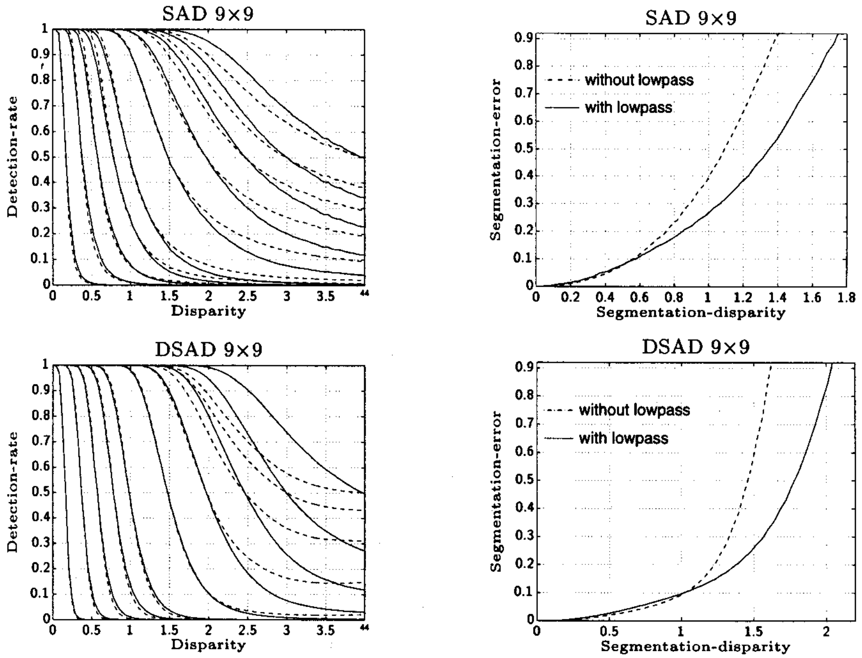


Figure 8.6: *Comparison of detection-rates and segmentation-errors for intensity-based correlation criteria and correlation on direction images with and without lowpass filtering.*

Lowpass prefiltering has a similar effect for all correlation methods on intensity images. As an example, the effect of a 3×3 lowpass on SAD correlation is shown in Fig. 8.6. It can be seen that a lowpass decreases the curvature and gradient of the segmentation-error function. For small segmentation-disparities the segmentation-error is slightly higher, but for larger disparities the segmentation-error is significantly lower because the detection-rate converges faster to 0%.
The effect of lowpass prefiltering with the DSAD method (lowpass *before* sobel filter) is greater than with methods on intensity images, but smaller than with methods on highpass filtered images. The improvement of a lowpass filter is significantly above a segmentation-disparity of 1 pixel, as can be seen in Fig. 8.6.

All methods on highpass filtered intensity images (see Fig. 8.7) show an increase of the curvature and gradient of the segmentation-error function and some decrease of the error for small disparities. The dissimilarity criteria which suppress the mean value (LSAD, LSSD, NSAD)

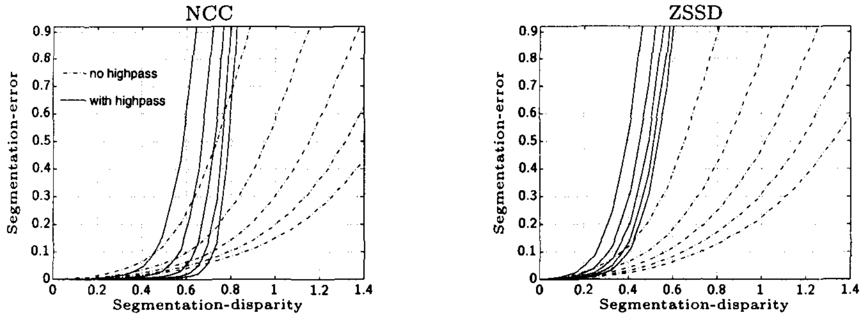


Figure 8.7: *Segmentation-error for NCC and ZSSD correlation with and without highpass filtering for various kernel sizes.*

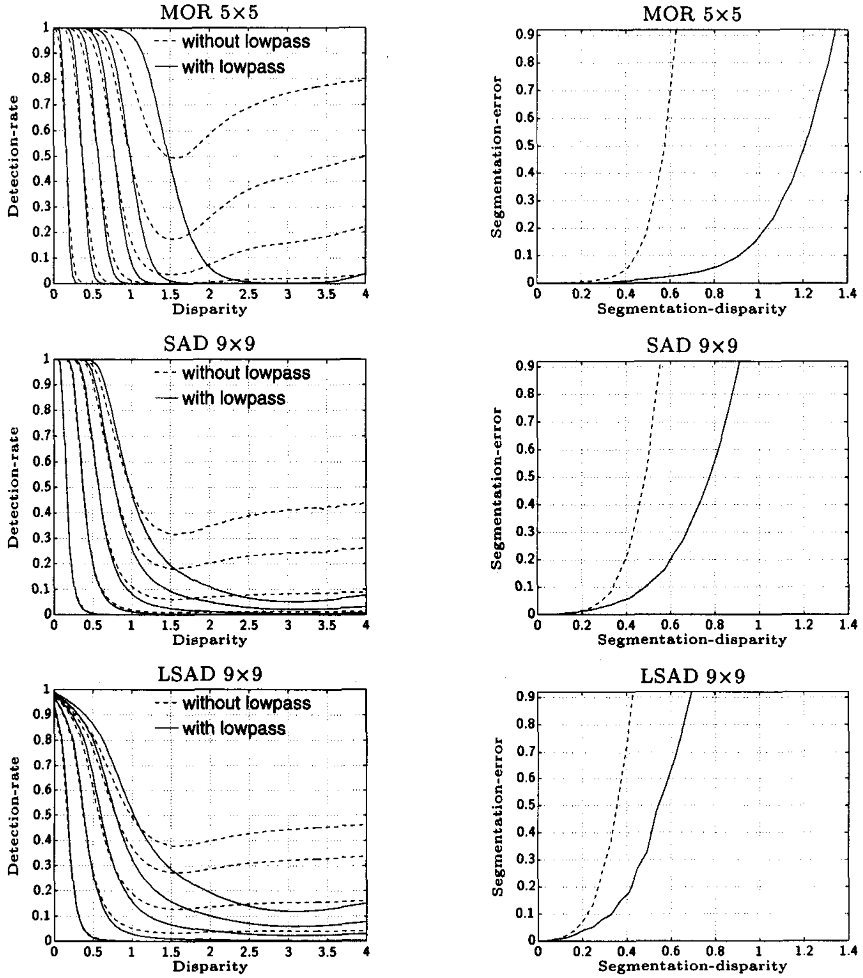
show a worsening of the behaviour with highpass filtering.

A lowpass prefilter in conjunction with highpass based correlation methods improves the segmentation-capability by increasing the gradient of the detection-rate function and converging to 0% while the method without lowpass filtering does not.

However, highpass-based methods do not show a consistent behavior in regard to lowpass filtering and may be split up into three groups with almost identical behavior:

- The methods based on cross-correlation (NCC, ZNCC, **MOR**) and the normalized zero-mean difference measure (ZNSSD) show the highest improvement although they show acceptable results without a lowpass prefilter. The gradient and curvature of the segmentation-error function is decreased and the flat zone (zone of very low error) is increased as can be seen in Fig. 8.8.
- **SAD**, SSD and its zero-mean versions show a slight improvement with additional lowpass filtering.
- The locally scaled and normalized difference measures (**LSAD**, LSSD, NSAD) show a slight improvement, but the minimal segmentation-error is still very high because no detection-rate reaches 100% for any disparity.

In the binary correlation according to Nishihara (NISH) the images are filtered with LoG filters, which show a combination of lowpass and highpass characteristics. In the segmentation-error plots in Fig. 8.11 it can be seen that small w shows a behavior like highpass based methods with small segmentation-errors for low disparities and a sharp point of inflection. With large w, the lowpass characteristics predominate with


MOR 5x5

MOR 5x5

SAD 9x9

SAD 9x9

LSAD 9x9

LSAD 9x9

Figure 8.8: *Comparison of detection-rates and segmentation-errors for highpass-based correlation criteria with and without lowpass filter.*

their smoother point of inflection at higher disparities, which is due to the fact that a dominant lowpass characteristic makes the correlation more tolerant to translations as can be seen in the plot of the mean and standard deviation in Fig. 8.9 and in the example images in Fig. 8.10. This phenomenon holds for all kernel sizes, but the degree increases with smaller kernel size.

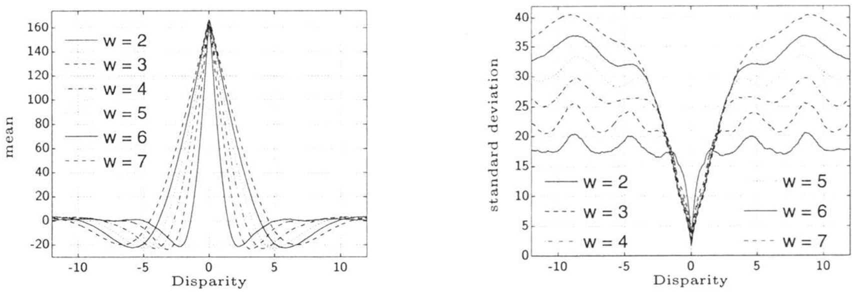
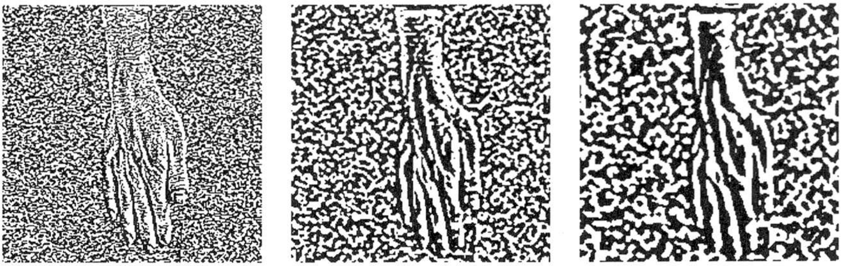


Figure 8.9: *Mean value and standard deviation of correlation values for binary correlation according to Nishihara.*



$w = 2$ $w = 4$ $w = 6$

Figure 8.10: *Sign of LoG filtered images (before correlation) with different parameters w.*

8.4.4 Behavior of the Various Correlation Criteria

In the following the behavior of all correlation methods with optional lowpass filters will be discussed based on the results obtained with a correlation kernel size of 9×9.

In Figures 8.12 and 8.13 the segmentation-error for correlation methods on intensity images with optional lowpass prefiltering are presented. In the relevant range of segmentation-errors $(0 \ldots 0.3)$ the methods can be divided into three groups:

- DSAD[18] shows the best performance.
- the correlation based on normalized cross-correlation (NCC,

---

[18] Although DSAD belongs to highpass based methods, it is shown here to ease comparison.

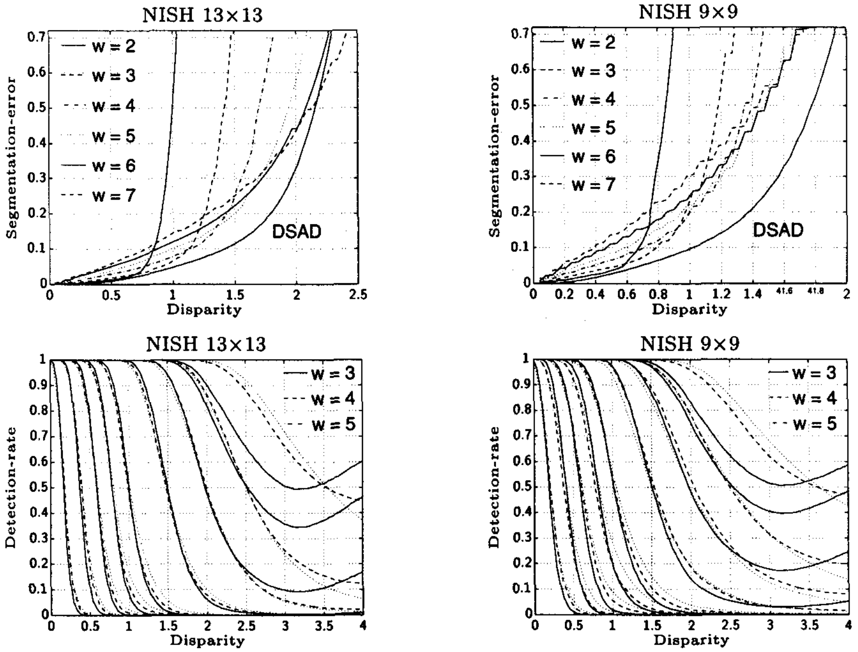


Figure 8.11: *Comparison of detection-rates and segmentation-errors for binary correlation with LoG prefiltering (NISH).*

ZNCC, MOR) and the normalized difference measures (ZNSSD, NSAD) show a good behavior with tolerable segmentation-error up to a segmentation-disparity of 0.9 - 1.0 pixel.

- the difference measures without normalization terms (SAD, ZSAD, LSAD, SSD, ZSSD, LSSD) still show tolerable performance up to segmentation-disparity of 0.7 pixel.

Except for DSAD an additional lowpass filter decreases the difference in performance for the various methods.

In Figures 8.14 and 8.15 the segmentation-errors of correlation criteria on highpass filtered images are presented. It is easily seen that the difference between the various correlation methods is much larger than for the intensity based methods. The methods can be divided into five groups:

- DSAD shows the best segmentation-capability and is only surpassed by the second group for segmentation at small disparities (up to 0.55 and 0.85 pxl for correlation with and without lowpass prefiltering, respectively).

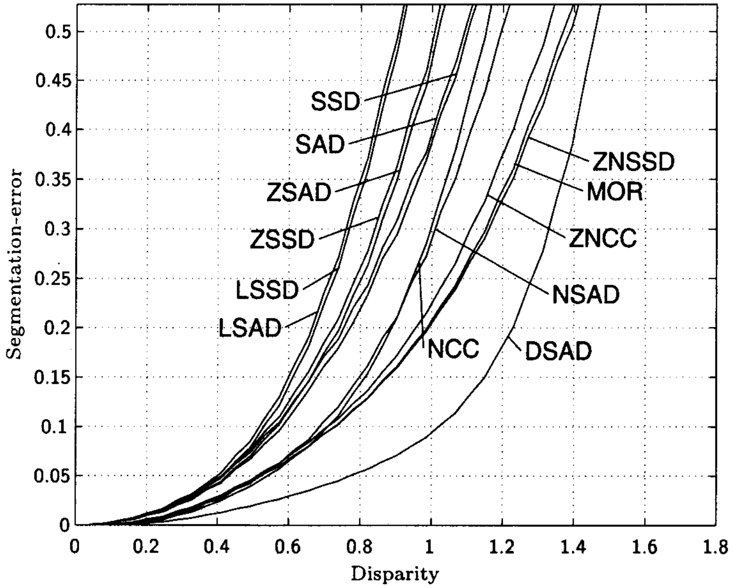


Figure 8.12: *Comparison of segmentation-errors of all correlation methods on intensity images without prefiltering prefilters.*

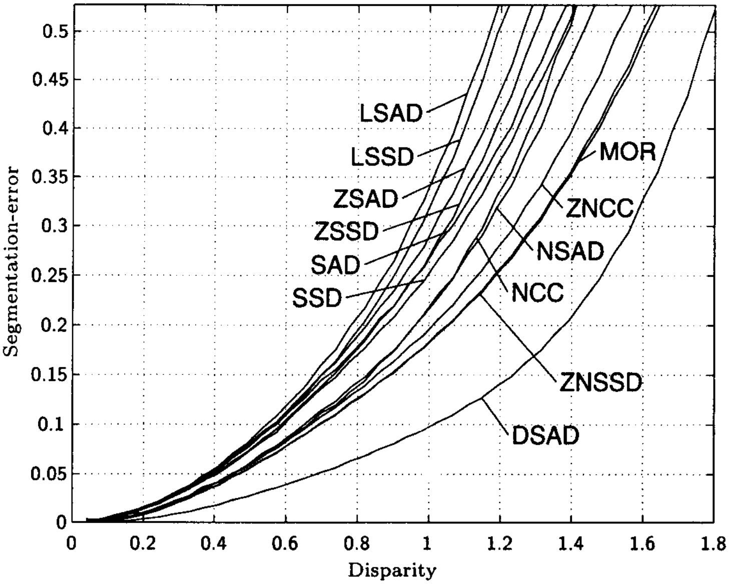


Figure 8.13: *Comparison of segmentation-errors of all correlation methods on lowpass (3×3) filtered intensity images.*

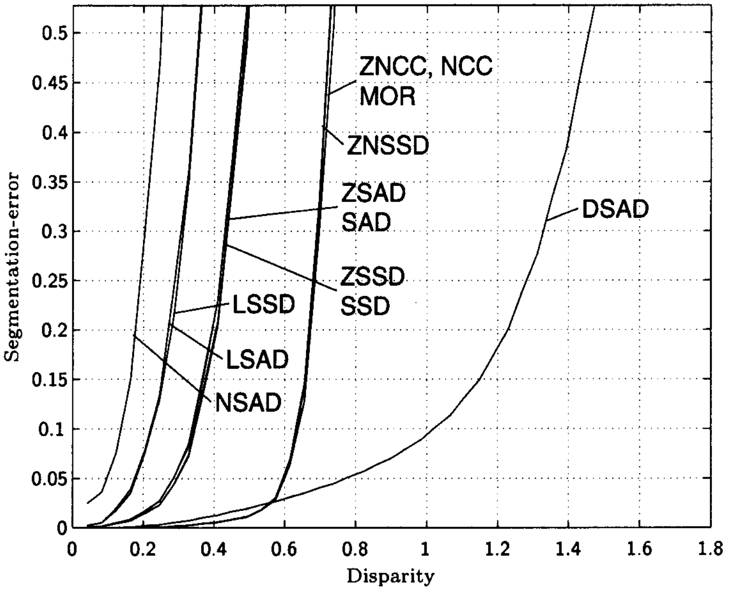


Figure 8.14: *Comparison of segmentation-errors of all correlation methods on highpass (3×3) filtered images.*

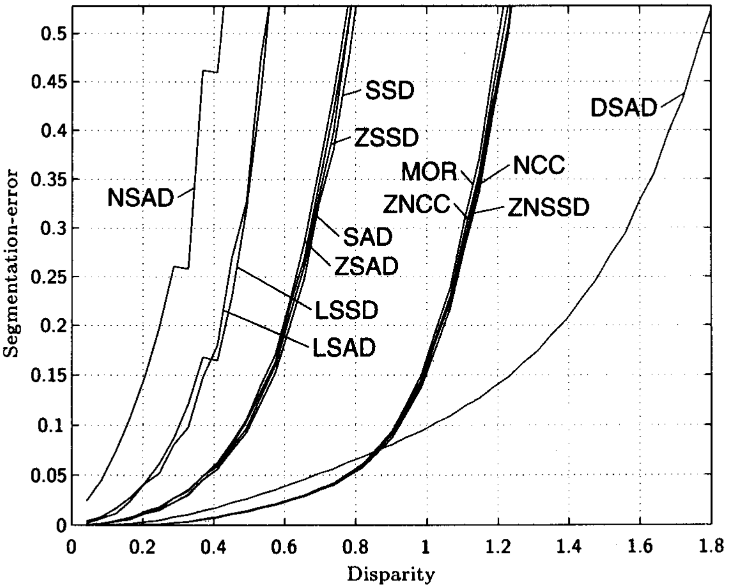


Figure 8.15: *Comparison of segmentation-errors of all correlation methods on highpass filtered images with lowpass prefiltering (3×3).*

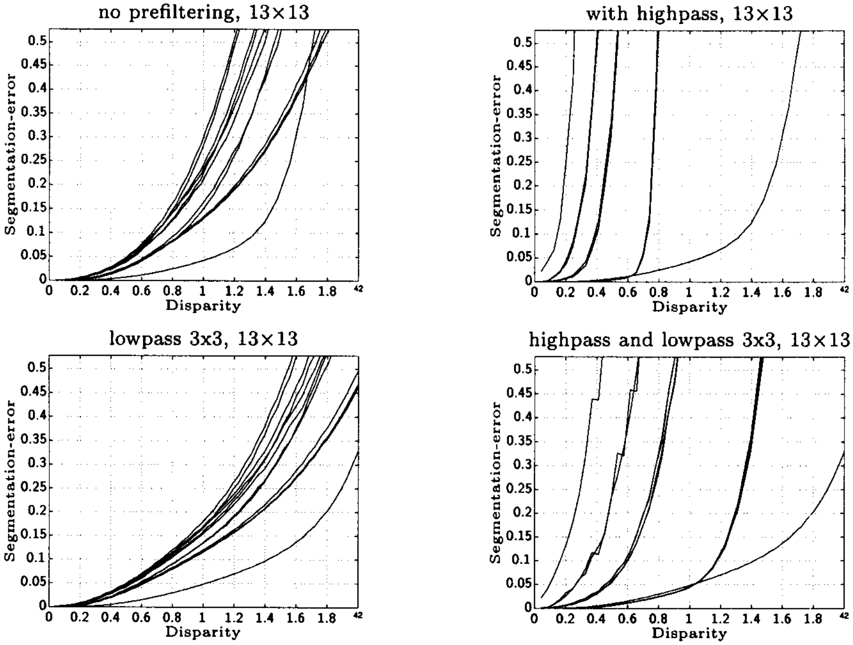


Figure 8.16: *Comparison of segmentation-errors for all methods with kernel size of 13×13.*

- The methods based on cross correlation (NCC, ZNCC, MOR) and the normalized zero-mean difference measures (ZNSSD) show good performance, especially for small disparities.
- The SAD and SSD and its zero-mean versions show a medium performance at small disparities and are not well suited to disparity segmentation.
- The locally scaled and normalized difference measures (LSAD, LSSD, NSAD) show very bad performance and should not be used for disparity segmentation together with highpass filters.

In Figure 8.16 the same segmentation-errors, but for a kernel size of 13×13, are presented. It can be seen that an increased kernel size has a similar effect as lowpass filtering and the ordering of the correlation methods remains the same. The segmentation-error functions of the various correlation methods on intensity images get more densely grouped with larger kernel sizes.

It is interesting to see that lowpass prefilters in conjunction with small kernels only improve the behavior of bad methods, whereas good

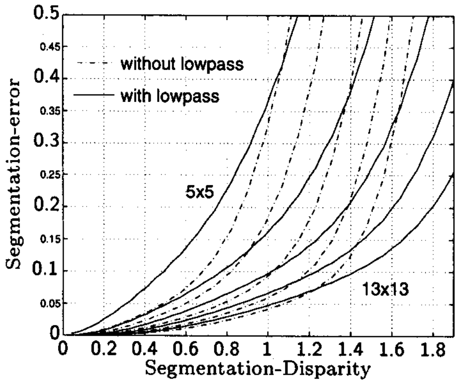


Figure 8.17: *Segmentation-error plots for sobel-direction correlation (DSAD) with kernel sizes between 5×5 and 13×13 pxl with and without lowpass prefiltering.*

methods (especially DSAD, see Fig. 8.17) even get slightly worse. Large kernels on highpass based correlation methods enlarge the flat zone of the segmentation-error function and decrease the minimal segmentation-error. In contrast to intensity based methods, lowpass filters in conjunction with highpass filtering improve the behavior for all kernel sizes. The improvement by an increase of the kernel by 2 pxl outperforms that of an additional lowpass for small kernels, whereas large kernels a lowpass prefilter is advantageous.

In summary, one can say that the difference between the highpass based methods is larger than that between the intensity based methods. The order of the methods concerning the segmentation-capability is the same for all prefilters with one exception: NSAD performs well on intensity images, but very poorly on highpass filtered images.

In Figure 8.17 the detection-rate of DSAD correlation with all kernel sizes and with optional lowpass prefilter is presented in order to give a basis on which to choose the appropriate correlation parameters. It can be seen that correlation with small kernels shows better performance when applied to raw images instead of lowpass prefiltered images. For a kernel size of 9×9 it is advantageous to use a lowpass prefilter for segmentation-disparities above 1 pxl.

8.4.5 Influence of Filtering Correlation Results

The result of the correlator may be filtered with a lowpass or median filter (3×3). In this way standard deviation is reduced and therefore the performance is improved in respect to segmentation.

Correlation on Intensity Images
For small correlation kernels the segmentation-capability is improved, but with large kernels (> 11×11) filtering has no effect. The performance increase of lowpass filters exceeds that of median filters. In general, filtering the correlation results is advantageous compared to prefiltering for small kernels (up to 7×7) with segmentation at small disparities (up to 0.6 - 1.0 pxl depending on the method).

Correlation on Highpass Filtered Images
For most correlation criteria the filter has a similar effect as for intensity images. For SAD, SSD and their zero-mean versions (ZSAD, ZSSD) the improvement is only visible up to a kernel size of 9×9. For the locally scaled and normalized difference measures (LSAD, LSSD, NSAD) the median filter produces much better results than the lowpass filter and for large kernels (larger than 9×9) the lowpass filter even worsens the results.

8.5 Dependence on Object Size

Correlation methods calculate a measure for correspondence for an image *area* and not a single pixel. Therefore small objects are suppressed by the correlation and the algorithm is incapable of detecting small objects. In general, large correlation kernels and lowpass filtering suppress small objects to a higher degree (see Fig. 8.18) than small kernels. This is due to the fact that the region within the object where pixels are influenced by non-corresponding pixels of the background is larger for large kernels.

In order to measure the smallest detectable object, images with objects of different sizes were correlated. These images were synthetically produced by inserting image areas of different sizes into a background image. The background has a disparity of 2 pxl and the inserted object image originates from two real camera images, grabbed at the same camera position. In this way two images which resemble as much as possible real camera images (with real camera noise) were produced. The threshold for the binarization was chosen such that 50% of the pixels with a differential disparity of 0.8 pxl were classified as corresponding. The results are presented in Fig. 8.18.

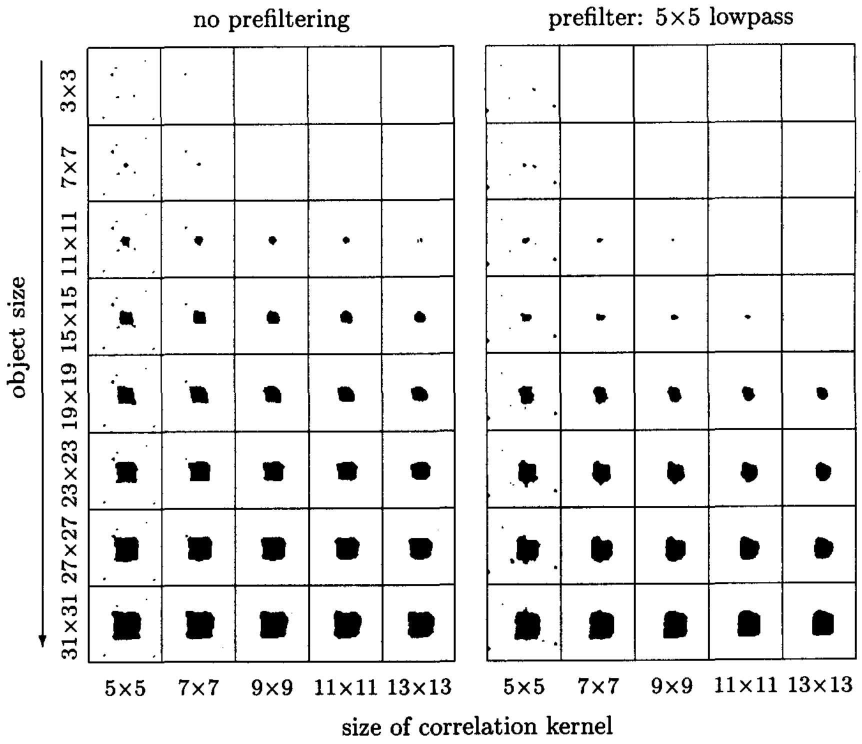


Figure 8.18: *Detection of different object sizes as a function of correlation kernel size with and without prefilters.*

8.6 Required Computational Resources

The computational resources required for a correlation method are very important since the algorithm must be processed at video rate. The required resources very much depend on the hardware used and code optimization. In Table 8.5 an estimation for implementation on a DSP and for a hardware implementation is given. The estimation for the DSP is only a rough estimation of the number of MAC[19] cycles per image. Partial sums[20] were temporarily stored (see Section 10.1.5) and computation for address calculation, loop controls and initializations is ignored. Calculating the square roots and dividing

---

[19] Multiply and accumulate: multiplications, shifting and addition can be processed in parallel.

[20] For CCF, NCC, SSD, SAD, DSAD, NSAD, NIS and for calculating mean value in ZNCC, MOR, ZSSD, ZSAD, LSSD. LSAD.

by the normalization term was avoided by adequately multiplying the threshold[21] and divisions were counted as 6 MAC operations. The zero-mean and mean scaled correlation criteria need much higher computational resources because the correlation measures in a window depend on the mean value for that window ($\bar{r}$, $\bar{s}$) and therefore the calculation of the sum cannot be optimized by calculating partial sums.

Method	HW	Computing power on DSP
CCF	low	0.4
SSD	very low	0.5
SAD	very low	0.6
CCFH	low	0.9
NCC	high	1.6
DSAD	low	3.0
NISD	medium	6.5
LSSD	high	19.2
ZSSD	medium	26.9
LSAD	high	27.4
ZSAD	medium	35.1
NIS	medium	44.5
MOR	very high	59.9
ZNCC	very high	60.1
ZNSSD	very high	60.2

Table 8.5: *Necessary computation (in Mega MAC operations) on a signal processor (DSP) for an image of 378×286 pixel and estimated resources for hardware implementation (HW).*

8.7 Conclusion

It can be seen that the correlation performs very well on the direction of the sobel intensity gradient (DSAD). It is robust in regard to brightness changes, has a medium robustness in regard to other unwelcome influences, a good segmentation-capability, low computational requirements and is very well suited to hardware implementation. Therefore the DSAD correlation with lowpass prefiltering was chosen for the hardware implementation.

---

[21] $\frac{A}{\sqrt{AB}} \geq T \;\Rightarrow\; A^2 \geq BCT^2$

Chapter 9

Texture Analysis

Correlation inherently needs texture. Some possible measures for texture intensity are discussed and the minimal necessary texture intensity is analyzed.

Image correlation inherently needs a spatial grey-level variation, also called texture. In the case of weak texture, incorrect correlation results are obtained:

- zero mean normalized correlation (ZNCC, MOR, ZNSSD) and correlation methods working on highpass filtered images produce higher dissimilarity values[1] in weakly textured regions because the "texture" introduced by noise, which is different in both pictures, outweighs the real texture.
- correlation methods working directly on intensity images tend to yield low dissimilarity values even when the patterns do not really match, but have a similar grey-level value.

Whereas the former may lead to missed objects, the latter may lead to false alarms of the surveillance system. A stereo vision system only functions reliably if either it is certain that the scene has enough texture or that the system itself measures the existing texture and reacts in an appropriate way in the case of weak texture. In the "inverse stereo principle" it is intended to either permanently measure the texture and/or to project an artificial texture onto the scene, since high texture in unstructured industrial scenes can not be guaranteed.

---

[1] Lower similarity values, respectively.

Texture Measure	scale factor	relative std dev	error 6.2 / 12.2	error 6.2 / 18.5
Std dev 7×7	1.0	0.125	0.003	0
Std dev 3×3	0.9	0.275	0.19	0.044
Kirsch compass	17	0.42	0.44	0.216
Sobel	6.8	0.525	0.53	0.32
Prewitt	4.7	0.525	0.53	0.32
Roberts	2.3	0.56	0.54	0.36
Prewitt Compass	4.2	0.6	0.55	0.35
Laplace	2.6	0.72	0.65	0.5

Table 9.1: *Data of texture measures: mean value, relative standard deviation and resulting discrimination error*

In order to test if the correlation measure at a specific point is reliable and to specially treat regions with weak texture, we must have an adequate texture measure and must know the minimally tolerable texture.

9.1 Texture Measures

The local variance (or standard deviation $\sigma = \sqrt{var}$) is a good basis for measuring texture intensity. However, the variance var has quadratic terms and depends on the mean value μ of the image intensities $I(m,n)$:

$$\begin{aligned} var &= \frac{1}{mn} \sum_{i,j=1..m,n} (I(m,n) - \mu)^2 \qquad (9.1) \\ \text{with} \quad \mu &= \frac{1}{mn} \sum_{i,j=1..m,n} I(m,n) \ . \end{aligned}$$

Therefore the calculation of the local variance requires a lot of hardware resources. Gradient operators represent another measure and are easier to calculate. In addition, it is more appropriate to use a gradient operator to measure texture for gradient based correlation criteria.

In order to compare the various texture measures (local variance and gradient operators) the texture of random noise patterns with different standard deviation was measured. As shown in Table 9.1, the mean of the gradient operators are proportional to the local standard deviation,

Surface type	mean value	relative std. dev.
White paper	1.7	0.20
Brown paper	3.0	0.20
Dark jeans fabric	5.6	0.15
Light jeans fabric	8.8	0.17
Human hand	7.4	0.30
Carpet	18.0	0.15

Table 9.2: *Measured local variance and relative standard deviation* (σ/μ) *of real materials.*

but the variance of the local standard deviation is smaller than that of the gradient operators. When segmenting an image into regions of weak and strong texture, as must be done in the texture analysis, measures with low standard deviation show better performance. However, the standard deviation of gradient operators is reduced by subsequent lowpass filtering. In order to illustrate the abstract texture intensities, the local variance of some sample surfaces is given in Table 9.2.

However, in spite of a high local variance (or gradient), correlation on gradient images do not work correctly if the derivative or direction of the image intensity is constant. Therefore when applying such correlation methods, the image must additionally[2] be tested for variation of the direction or derivative.

9.2 Required Texture

For gradient based correlation, texture is inherently required to calculate a reliable local gradient[3] (magnitude or direction). Texture directly competes with (uncorrelated!) image noise and in the case of low texture compared with the image noise, the gradient is mostly a function of the present noise. This results in weak texture producing higher dissimilarity values than strong texture (see Fig. 9.1).

The qualitative insight that the ratio of the variances of texture and noise is crucial for a reliable correlation was experimentally verified. An

---

[2] Only a variation in the direction/derivative is not sufficient, as this is likely because of image noise.

[3] In contrast, intensity based correlation only gets more tolerant to slight misalignments in the templates.

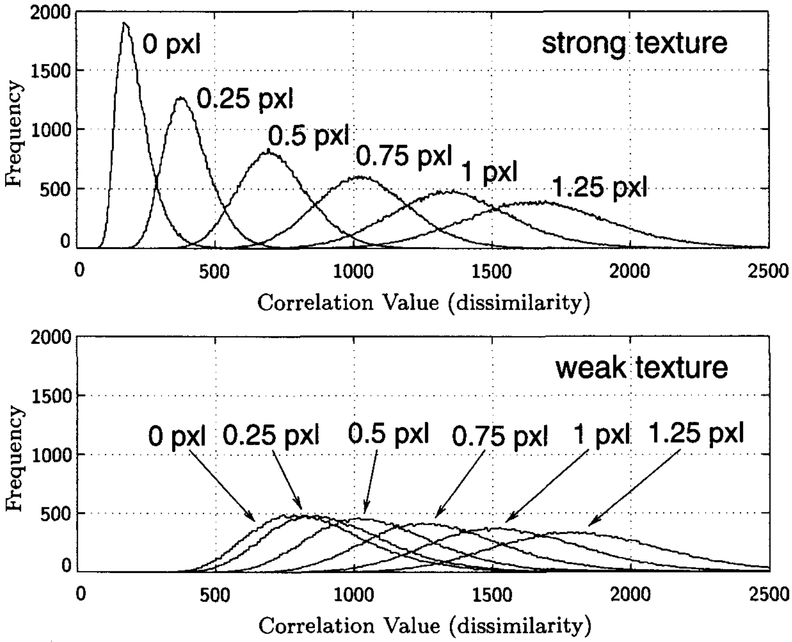


Figure 9.1: *Histograms of correlation values of weak and strong textures (low and high texture to noise ratio, respectively) for various disparities.*

experiment similar to that for investigating the separation-capability (see Section 8.4.1) was used. It differs in that a series of *artificial* patterns with different texture intensity and image noise was used. Patterns with a texture with a standard deviation $\sigma = 5 \ldots 70$ were combined with Gaussian noise of standard deviation $\sigma = 1 \ldots 20$. The mean standard deviation in a 7×7 neighborhood was used as measure for texture and noise. In Figure 9.2 the segmentation-error for DSAD correlation is plotted as a function of the texture to noise ratio (standard deviation) for various segmentation-disparities. It is clearly seen that neither texture or noise but rather their ratio is decisive for the segmentation capability.

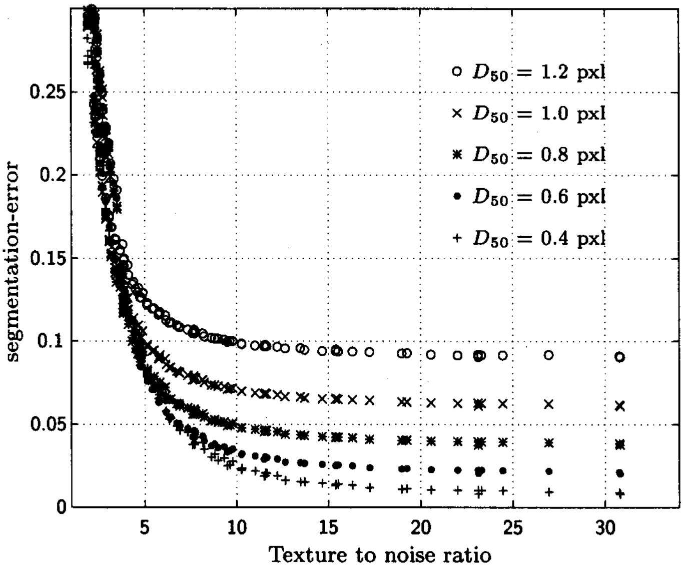


Figure 9.2: *Segmentation error in function of texture to noise ratio (standard deviation measured in 7 × 7 window): it can be seen that the segmentation error is only a function of the texture to noise ratio and is independent of texture or noise alone.*

9.3 Methods for Coping with Weakly Textured Scenes

Because the existing texture in relation to the total image noise (mathematically the ratio of the signal and noise variances) is the deciding property of an image and not the absolute texture strength of a scene, it is possible to either increase texture or decrease image noise.

9.3.1 Increase Texture

A strong texture can be produced either by special high textured clothes and paintings or by actively projecting an artificial texture onto the scene. Equipping the environment with high textured objects is often unfeasible (e.g. human skin!) or very expensive. The projection of an artificial texture onto the scene (either a random noise field [62][4] or randomly frequency-modulated sine waves [63]) is more suitable. Since CCD sensors are highly sensible in the infrared band, it is possible to project the patterns with infrared such that the patterns are invisible to humans.

9.3.2 Decrease Noise

Any noise source (see Section 8.2) could be reduced in order to increase the signal to noise ratio of the image:

- increase ambient lighting in order to reduce shot noise, i.e. the signal to noise ratio is increased
- improve camera and frame grabber circuits
- use a digital camera where the CCD sensor is directly coupled with the A/D converter in order to eliminate noise induced by jitter
- increase the resolution of the AD converter to decrease quantization noise.

When reducing the system noise, the necessary texture for reliable correlation is reduced.

---

[4] a standard slide projector was used to project the patterns

Chapter 10

Hardware Implementation

The "inverse stereo algorithm" was successfully implemented at video-rate on a dataflow processor. First the required computational requirements are estimated and some hardware architectures are discussed in regard to implementation of the "inverse stereo algorithm" at video rate. In the second part a detailed description of the implementation of the algorithm is given, starting with a description of the hardware platform "PRIMASPEED".

10.1 Estimation of Required Computational Resources

In order to decide upon the best hardware platform, it is important to know what kind of operations and which computing performance are required for implementation. In addition, the required memory bandwidth is also of great concern.

In this section the necessary computation load for this algorithm is estimated. However, this data should only be used to estimate the minimal necessary performance, because the peak performance of processors cannot be fully exploited because of the following reasons:

- limited memory bandwidth; this may be a bottleneck especially for image processing.

- a significant part of the processing power is used for program overhead (counters, conditions, branches)
- discrepancy between hardware and algorithms: e.g. a DSP optimized for multiply-and-accumulate (MAC) operations could never reach its peak performance if the algorithm has only a few MAC operations.

For reasons of comparison, the computing time used on a ultraSPARC-station is given in addition.
In order to allow for estimating the required computing power for architectures which perform several operations in parallel (e.g. address calculation, addition and multiplication), these operations were treated separately. The following assumptions were made for the estimation of the computational requirements:

- Calculation intensive functions (e.g. argument of sobel-gradient, remapping coordinates for transformation) are calculated in advance and stored in look-up-tables (LUT).
- Intermediate results are temporarily stored in off-chip memory, if more than four operations can be saved by storing one value.
- The memory data-width is 8 bit except for the mapping table memory which is 24 bit wide.
- It is assumed that the camera images are already in the memory. If the image data must first be read from the A/D converter and written to memory by the DSP, this results in additional computation.
- The computation necessary for loop counters, conditions and branches was not considered (so-called “program overhead”).

In the following sections x denotes the image width and y the image height in pixels.

10.1.1 Image Transformation

For the following analysis of the necessary operations it is assumed that the mapping coordinates and the interpolation factors for the image transformation are provided in a precalculated table. Since any arbitrary transformation is allowed, there is no possibility of precalculating some terms in order to speed up the transformation.

First, the next value of the coordinate table, containing a pointer to the upper left pixel of the source quadruple and the interpolation coefficients, is read. Then the addresses of all four pixels are calculated and the pixels read from memory

$5\,xy$	*Memory accesses.*

The interpolation coefficients are extracted from the packed data by shifting[1]. Subsequently the four pixels I_{ij} are scaled with the interpolation factors (u, v) and are summed up in order to calculate the bilinear interpolated result[2]:

$$I = ((uI_{11} + (8-u)I_{12})v + (uI_{21} + (8-u)I_{22})(8-v))/64 \qquad (10.1)$$

$6\,xy$	*Multiplications*
$5\,xy$	*Additions*
$3\,xy$	*Shift operations.*

10.1.2 Subsampling

Both, the transformed and the camera image are subsampled (see Section 10.4.3). The subsampling is done by averaging every two pixels. It is assumed that the transformed image is subsampled following the transformation, such that no additional memory access is necessary.

xy	*Memory accesses*
$2 \cdot xy/2$	*Additions*
$2 \cdot xy/2$	*Shift operations*
$2 \cdot xy/2$	*Memory accesses.*

10.1.3 Lowpass Filtering

In order to suppress noise, both subsampled images are lowpass filtered.

$$\text{Lowpass} = \begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline \end{array} = \begin{array}{|c|}\hline 1 \\ \hline 2 \\ \hline 1 \\ \hline \end{array} + 2 \cdot \begin{array}{|c|}\hline 1 \\ \hline 2 \\ \hline 1 \\ \hline \end{array} + \begin{array}{|c|}\hline 1 \\ \hline 2 \\ \hline 1 \\ \hline \end{array} \qquad (10.2)$$

The sums of the columns of the filter are calculated in advance, temporarily stored in a register and used for all three columns, which

---

[1] Assuming that a shift operation is more efficient than a memory access.

[2] Interpolation coefficients are each two bits.

results in the following reduced computation:

$$\begin{array}{rl} 2 \cdot 3(y-2)\,x/2 & \textit{Memory accesses} \\ 2 \cdot 2(y-2)\,x/2 & \textit{Additions} \\ 2 \cdot (y-2)\,x/2 & \textit{Shift operations} \end{array}$$

Subsequently for every output value the three column-sums are added and divided by 16

$$\begin{array}{rl} 2 \cdot 2(y-2)(x/2-2) & \textit{Additions} \\ 2 \cdot 2(y-2)(x/2-2) & \textit{Shift operations} \end{array}$$

10.1.4 Sobel Filtering

Both the left and right images are sobel filtered. In order to calculate the direction and magnitude of the sobel-gradient, the images are convolved with the two masks

$$S_x = \begin{array}{|c|c|c|} \hline 1 & 0 & -1 \\ \hline 2 & 0 & -2 \\ \hline 1 & 0 & -1 \\ \hline \end{array} \qquad S_y = \begin{array}{|c|c|c|} \hline 1 & 2 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -2 & -1 \\ \hline \end{array} \tag{10.3}$$

and the results are used as addresses into the LUT to read the precalculated direction and magnitude values. In order to reduce the computation, the same method as for the lowpass filter is applied. This results in the following number of operations for the calculation of the column-sums:

$$\begin{array}{rl} 2 \cdot 3\,(y-2)\,x/2 & \textit{Memory accesses} \\ 2 \cdot 3\,(y-2)\,x/2 & \textit{Additions} \\ 2 \cdot\ (y-2)\,x/2 & \textit{Shift operations.} \end{array}$$

Then for each pixel the two convolutions, as a weighted sum of the precalculated column sums, are calculated.

$$\begin{array}{rl} 2 \cdot 3\,(x/2-2)(y-2) & \textit{Additions} \\ 2 \cdot\ (x/2-2)(y-2) & \textit{Shift operations} \end{array}$$

Subsequently the results are scaled[3] in order to produce the LUT address and the values are read from the LUT:

---

[3] See Chapter 10.4.5

$2 \cdot 2\,(x/2-2)(y-2)$	*Absolute values*
$2 \cdot 3\,(x/2-2)(y-2)$	*Logic operations*
$2 \cdot 2\,(x/2-2)(y-2)$	*Shift operations*
$2 \cdot\,(x/2-2)(y-2)$	*Memory accesses.*

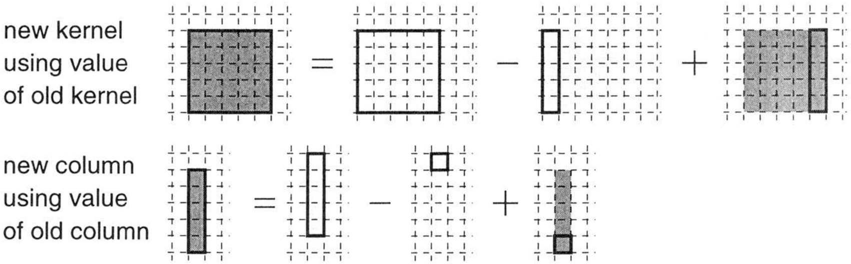
In order to reduce the number of output values which must be written to memory, the cyclic difference of the direction is already calculated at this stage.

$$\Delta_{dir}(x,y) = min(|\phi_l - \phi_r|, ||\phi_l - \phi_r| - cycle|) \qquad (10.4)$$

$2\,(x/2-2)(y-2)$	*Additions*
$2\,(x/2-2)(y-2)$	*Absolute values*
$(x/2-2)(y-2)$	*Comparisons*
$xy/2$	*Memory accesses*

10.1.5 Correlation

The cyclic direction differences are summed up in a $(2k+1)^2$ neighborhood around each pixel (x, y). In order to decrease the number of operations, new sums are calculated using previously calculated sums in the following way:



The calculation of the initial line of column-sums results in the following number of operations:

$(2k-1)\,x/2$	*Additions*
$2k\;x/2$	*Memory accesses.*

Calculating the correlation of the entire image (including the new column-sums) leads to the following operations :

$(2\,(x/2-2k)+4k)\,(y-2k)$	*Memory accesses*
$(4\,(x/2-2k)+6k-1)\,(y-2k)$	*Additions.*

10.1.6 Texture Analysis

For the texture analysis the magnitude of the sobel gradient is used

$$\text{Magnitude} = \sqrt{S_x^2 + S_y^2}\ . \tag{10.5}$$

It is also precalculated and stored in the same LUT as the direction. Since S_x and S_y were scaled, this scaling must be reversed by appropriately shifting the magnitude. Subsequently the magnitude is lowpass filtered (see Section 10.1.3) to reduce its variance.

$y\,x/2$	*Shift operations*

$(3\,(y-2)+1)\,x/2$	*Memory accesses*
$2(y-2)\,(x-2)$	*Additions*
$(y-2)\,(3x/2-4)$	*Shift operations*

10.1.7 Segmentation

First the image is thresholded and saved as a packed (16 bit words) binary image in memory.

$yx/2$	*Memory accesses*
$yx/2$	*Comparisons*
$yx/2$	*Shift operations*
$yx/32$	*Memory accesses*

In the second step small clusters are eliminated (see Section 5.2, Eq. (5.4)). It is assumed that the algorithm stores intermediate results in the processor registers (operations used for initialization are ignored).

${}^{2}/_{16}\,(2e+1)\,({}^{x}/_{2}-e)\,(y-e)$	*Memory accesses*
$3\,(2e+1)\,({}^{x}/_{2}-e)\,(y-e)$	*Shift operations*
$(3+2e)\,({}^{x}/_{2}-e)\,(y-e)$	*Additions*
$({}^{x}/_{2}-e)\,(y-e)$	*Comparisons*

Algorithm (see Section)	Memory accesses		Arithmetic operations		μP time
		MMAS[2]		MOPS	SPARC[4]
Transformation (10.1.1)	$5\,xy$	55	$16\,xy$	154	0.21 s
Subsampling (10.1.2)	$2\,xy$	22	$2\,xy$	22	0.04 s
Lowpass (10.1.3)	$3\,x\,(y-2)$	33	$2(7x/2-8)(y-2)$	76	0.10 s
Sobel Filtering (10.1.4)	$xy/2+4\,(x-1)\,(y-2)$	49	$((35/2)x-54)\,(y-2)$	190	0.15 s
Correlation[1] (10.1.5)	$x(y-k)$	11	$(2k-1)\,x/2+(y-2k)$ $(2x-2k-1))$	21	0.10 s
Texture Analysis (10.1.6)	(3y-5)x/2	16	yx/2 + (y-2)(7/2x -8)	44	0.12 s
Segmentation (10.1.7) & clusterlet elimination	17/32 xy + (2e+1)(x/2-e) (y-e)/8	9	xy + (7+8e) (x/2-e) (y-e)	136	0.21 s
Total		195		643	0.93 s

[1] Correlation kernel size 9×9 (k=4)

[2] MMAS = Mega Memory Accesses per Second

[3] MOPS = Mega OPerations per Second

[4] Time to process *one* image pair, measured on a ultraSPARC-station (167 MHz) with optimized C-code

Parameters: Image size: x×y = 756 × 286 pxl
Correlation kernel size: (2k+1) × (2k+1), k=4
Region for clusterlet elimination: (2e-1) × (2e-1), e=2

Table 10.1: *Required computing power for an image size of 756×286 pxl and 50 frames/s. Additions, shift and logic operations are summed up in 'arithmetic operations'. Memory accesses include memory address calculations. The processing time on the ultraSPARC is given for a* single *image with non-optimized code.*

The total required computing power for an image size of 756×286 and a correlation kernel of 9×9 pxl is presented in Table 10.1. Because addition, multiplications by a power of two, comparisons, shift and logic operations are of similar complexity, they were summed up to "arithmetic operations".

10.2 Hardware Architectures

There are various possible hardware architectures for an implementation of the algorithm. In order to choose adequate hardware the following considerations must be taken into account:

- as in most image processing algorithms, a huge amount of data must be processed.
- processing must be performed at video rate and the data (images) comes from the video-camera or frame-grabber synchronous to the camera clock.
- there are relatively few operations performed per data and mostly integer arithmetic (typical for low-level algorithms).
- all pixels of an image are processed with the same operations.

Image processing operations are often divided into low level and high level functions. Although there is no exact definition for low level and high level image processing, and it slowly drifts as technology changes, a rough distinction can be made in the following way:

- **Low level functions** operate on pixel level and all pixels in an image are treated with the same algorithm. Owing to their regularity they are usually well suited to a hardware implementation. Typical for low level algorithms realized in software is that the number of operations used for addressing and loop control is on the same order or even larger than those used to calculate the algorithm.
- **High level functions** usually work on features of an image and therefore the operations needed to execute the algorithm outperform those for addressing and loop control.

There are a variety of processor architectures. Here they will be divided into control-flow and data-flow processors:

- **Control-flow** is the most often used architecture, either as von Neumann or Harvard architecture. The existence of a program counter, an instruction controller and memory to store data and programs is typical. Since all input data and program code is read from memory and all results written to memory, memory access is often the bottleneck of such processors.
- In **dataflow-processors** processing is initiated by the availability of data and described by dataflow-graphs. If there is no individual data but a sequential stream (like images from a camera) there is no need for explicit operators for each datum as in the classic dataflow principle, and the data-stream can directly flow from one processing element to the next. If all data-streams flow synchronously to a global clock, such a processor is called synchronous dataflow processor.
  Since the data-stream can flow through any number of processing elements (PE) without being stored in memory, the memory bottleneck of control-flow processors is eliminated. The PEs are usually functions realized in hardware, but might also be common program controlled processors. Dataflow processors are mostly used for signal and image processing. However, dataflow-processors are usually not well suited to data dependent computations.

In the following some widely known processor architectures used for signal and image processing are discussed. However, it goes beyond the scope of this thesis to give an extensive overview and to analyze the implementation of the "inverse stereo algorithm" on other architectures.

10.2.1 High Performance Processors

There are mainly two processor types, the RISC processors and the digital signal processors (DSP). DSPs are especially designed for signal processing which uses lots of multiply and accumulate (MAC) operations. However, the availability of a MAC operation is not advantageous for the implementation of the "inverse stereo principle", as only few multiplications are used. Furthermore no floating point arithmetic is needed, because all data are integers and most multiplications are by powers of two and therefore realizable by shift operations. On the other hand, separated data and program buses, fast internal memory (e.g. ADSP-21060 has 4 Mbit on-chip SRAM) and versatile address generators (e.g. used for delay lines) could drastically increase the performance since the

algorithm is very memory intensive.
Even two years after the start of this project there are still no single processors on the market with a high enough performance for the entire algorithm. Today's single-DSPs all have a performance below 100 MIPS[4].

In recent years high performance RISC processors such as DEC Alpha (1992) and the superscalar processors MIPS R8000[5] (1992), R10000 (1996) and ultraSPARC[6] (1996) came to market (see Table 10.2). The DEC Alpha yields only about half of the necessary computational power and the other high-performance processors could only provide the indicated performance if the algorithm can be programmed such that all units are constantly used to capacity. In addition, since the algorithm uses only integer arithmetic, the relevant processing power is reduced to a peak performance of 400 Mega integer instruction per second for the ultraSPARC.

Because neither DSP nor RISC processors can produce the necessary performance, only a multi-processor solution is possible.

10.2.2 Multi-processor Systems

With multi-processors, the processing power is ideally multiplied, but memory access and communication often decrease the processing power such that it does not linearly increase with the number of processors. Multiprocessors either share memory or each processor has its own local memory. In shared memory multiprocessors, the inter-processor communication is done by using the common memory, but each processor can use only a fraction of the total memory bandwidth. Consequently, for programs with high data I/O, memory access is the bottleneck and slows the whole system down.
With local memory each processor has full access to its memory. For inter-processor communication and system I/O there is a data-communication network that can have two different topologies: systolic array with point to point connections between the individual processors or connection through a bus. At the beginning of each processed image, the results of the last cycle and the new image must be communicated

---

[4] One DSP-instruction consists of a multiplication, addition and shift operation. Consequently the SHARC (ADSP-2106x) has 40 MIPS but 120 MFLOPS.

[5] MIPS Technology Inc.

[6] Registered trademark of SPARC International, Inc; based upon an architecture developed by Sun Microsystems, Inc.

Single processors

DEC alpha	233 MIPS	
R10 000	800 MIPS	2 floating point and 2 integer units
ultraSPARC	1000 MIPS	4 integer, 3 floating point units and a graphical processor

Multi processors

TMS320C80	250 MFLOPS	consists of 4 fixed-point DSPs and one RISC
MUSIC	3.78 GFLOPS	63 DSP96002[7]; intelligent communication
GigaBooster	1.16 GFLOPS	7 DEC alpha; intelligent communication
CNAPS[8]	1280 MIPS	SIMD[9] computer on custom chip containing 64 MAC-processors
iWARP[10]	20 MFLOPS per node	processors with hardware supported message passing (320 MBytes/s per processor);
Connection-Machine-5[11]	—	MIMD[12] machine; 1 to 16 384 RISC processors

Table 10.2: *Performance of some single and multi-processors*

to each processor.

In Table 10.2 the performance and some characteristics of some multiprocessors are presented. Unlike conventional multi-processor systems where communication is controlled by the processors and decreases the system performance, GigaBooster[13] and MUSIC[14] [65] have "intelligent communication", where data communication is controlled by a hardware controller that relieves the processor from this work.

Dataflow Multiprocessors

In dataflow processors data is not stored between the processing steps

---

[7] DSP96002 by Motorola, 20 MIPS, 60 MFLOPS peak performance

[8] Fixed-point arithmetic (16 bit), by Adaptive Solutions, Inc.

[9] SIMD = single instruction multiple data .

[10] By INTEL Corporation, based on hardware and software design developed at Carnegie Mellon University .

[11] By Thinking Machines Corporation .

[12] MIMD = multiple instruction multiple data .

[13] Commercial name of Alpha7 [64].

[14] Developed at the Electronics Lab, ETH Zurich.

but flows directly from one processing element (PE) to the next. The processing elements are most often hardware-implemented algorithms but may also be common program-controlled processors.

There are many processors based on the data-flow principle. The commercial MaxVideo-250 and MaxPCI from Datacube consist of a cross-point switch (up to 3 GByte internal bandwidth) to interconnect the PEs (statistical, histogram, LUT processors and ALU) with a total performance of up to 10'000 MIPS. In addition, there are daughter-cards available for image warping and neighborhood operators. In Section 10.3 the hardware of PRIMA*SPEED*[15] is presented in more detail, since the workspace supervision system was implemented on this processor.

10.2.3 Conclusion

The necessary computation can only be provided by multiprocessor systems. Shared memory multiprocessors are not suitable due to the memory bottleneck. Multi-processors with local memory and, advantageously, with "intelligent communication" have the necessary computational power but they are not economically sound since a considerable percentage of the power is used for program overhead. Therefore the system was implemented on a dataflow processor.

Datacube's processors have the disadvantage of requiring an additional module for neighborhood operators. An image warper is available, but with mapping functions restricted to first and second order polynomials[16]. PRIMA*SPEED* was chosen as the platform for the implementation because 3×3 filters can be implemented onto the base-board and custom-designed FPGAs and daughter-boards can be used.

10.3 Architecture of PRIMA*SPEED*

The inverse stereo algorithm is implemented on the PRIMA*SPEED*[17] hardware, a synchronous dataflow processor. It is a commercial successor of the image processing system SYDAMA-2 (SYnchronous DAtaflow MAchine), which was designed and built at the electronics laboratory at ETH Zurich [66, 67]. PRIMA*SPEED* is an add-on

---

[15] PRIMA*SPEED* is a product of LEUTRON *VISION* AG, Glattbrugg, Switzerland .

[16] Such that the necessary undistortion and redistortion of the image is not possible.

[17] Formerly also called PC-SYDAMA.

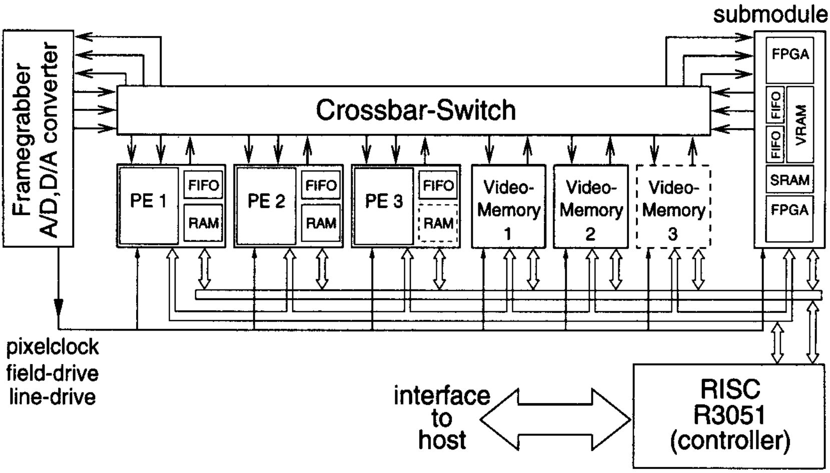


Figure 10.1: *Overview of PRIMASPEED architecture (direct access from host (EISA bus) to video memory is not shown in this figure).*

card for personal computers, especially designed for low level image processing at video rate.

The systolic array is synchronous (clocked with the pixel clock) and consists of 3 processing elements (PE) and 3 video memories (VMEM), which are all connected to the video bus and the address- and data-bus of the RISC (see Fig. 10.1 and Table 10.3). The video bus is multiplexed (2:1) in order to reduce the physical bus width and consists of an integrated crossbar-switch allowing for connecting each input to all (even multiple) outputs. Filters (e.g. 3×3 lowpass, highpass, sobel), look up tables (up to 2 × 9 bit inputs) or custom designs can be implemented on the FPGA. For functions which are too complex for implementation on an FGPA, two custom-modules may be added.

Owing to the functions programmed in hardware, there is no need to fetch and decode commands, to read and write intermediate results or to have loop controls, all of which reduce the necessary hardware.

PRIMA*SPEED* is programmed with the use of a special monitor (PRIMON), which is an adapted version of SYMON [68, 69], the monitor for the low level part of SYDAMA-2. On the one hand PRIMON allows for the configuration of the systolic hardware (set crossbar switch to configure video-bus, set parameters of PEs, video-memory and cus-

Video-memory:

VMEM	512 KByte VRAM (8 bit, 1 input, 1 output), window-generator

Processing element (PE):

I/O:	2 inputs (10 bit) and 1 output (12 bit)[18]
FPGA:	XC4005 or XC4008; SRAM-based, configuration downloaded at system startup, reloadable between two image frames.
SRAM:	256 KByte usable e.g. for look-up-tables (LUT)
FIFO:	8 KByte × 9bit; used for delay-lines[19] and neighborhood-operators

Processor (RISC):

R3051	Controller for dataflow processor; usable for high-level processing thanks to direct access to VRAM.

Table 10.3: *Some data about PRIMASPEED.*

tom modules) in an easy way. On the other hand, it allows for the definition of static dataflow graphs, which can be changed at video-rate to program complex algorithms.

10.4 Implementation of the "Inverse Stereo Principle"

The entire algorithm was successfully implemented on PRIMA*SPEED* and on an additional custom module, which was built during this work. The spatial image transformation and the lowpass filtering was implemented in this module, which consists of two FPGAs, SRAM, VRAM and two FIFO memories. Figure 10.2 shows the dataflow graph of the algorithm and the allocation of the individual functions to the hardware resources. The entire algorithm is processed at video-rate (50 images/sec), with a pipeline delay of 40 ms. The result is either a binary image or a transition-coded image for subsequent image interpretation.

In the next section some possibilities for the implementation of the spatial transformation in hardware are analyzed. In the following sec-

---

[18] Each two bits can be individually connected.

[19] This is used if one image was delayed against another, e.g. by neighborhood operators.

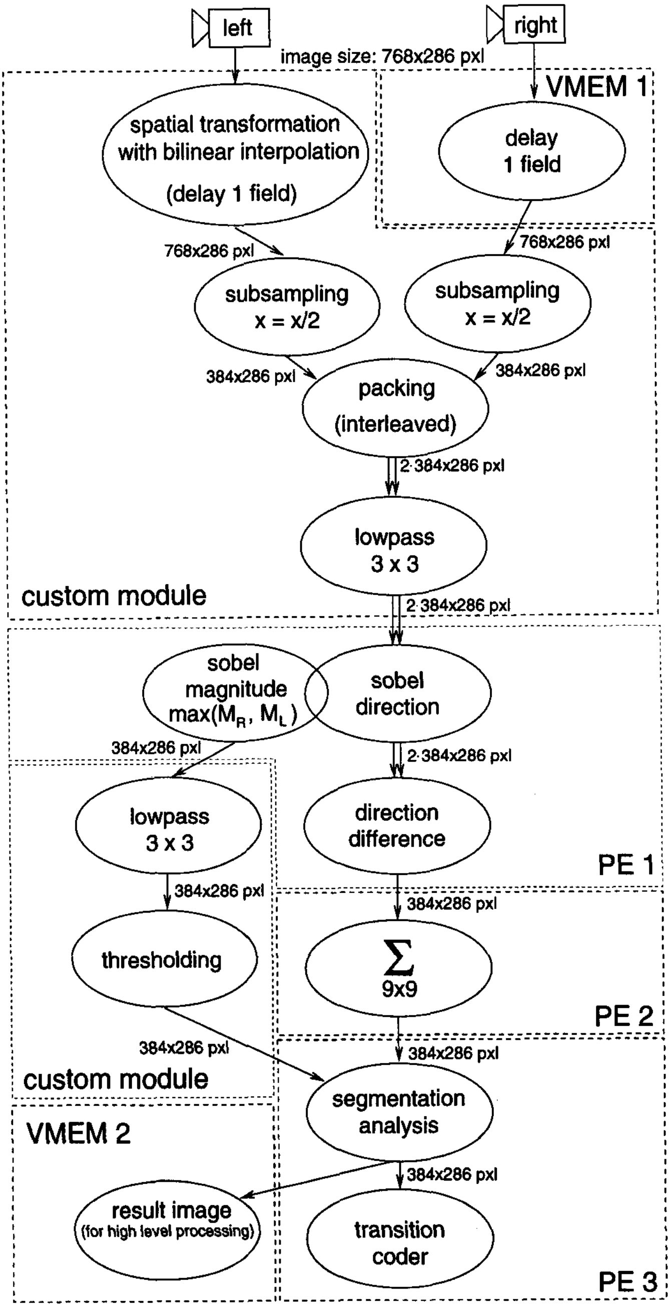


Figure 10.2: *Dataflow graph of "inverse stereo algorithm" with allocation to hardware modules on PRIMASPEED.*

tions the implementation of all algorithms is presented in detail.

10.4.1 Implementation Possibilities of Image Transformation

The module for spatially transforming images has to satisfy the following requirements:

- transformation in real time (50 fields/sec; see Section 10.4.3)
- arbitrary transformation possible (including distortion correction)
- switchable between different transformations (safety envelopes) during operation
- image resampling with bilinear interpolation or better.

Because the SRAM of the PEs are too slow for an implementation of bilinear interpolation and the PRIMA*SPEED* has too few PEs to implement all necessary functions on it, we decided to build a custom module for image transformation. In the following, several possibilities for implementation are discussed[20].

Arbitrary Mapping with DSP

With a digital signal processor (DSP) it is possible to calculate any transformation and distortion correction in real-time [21], but with the price of huge processing power. In Table 10.4 the estimated computational power for a perspective-affine transformation is presented.

When taking into account that a division needs 6, a square root 10 and the cubic root 37 MAC[23] cycles[24], this results in a total

---

[20] Methods that make only bilinear mapping possible (there are VLSI chips calculating bilinear mapping at video-rate, e.g. TMC2301) were not discussed, since they facilitate neither perspective transformations nor distortion correction, which are both very important for this application.

[21] Instead of using a DSP it is possible to build a custom hardware (VLSI) which calculates just the necessary mapping function at the required speed. Because no such hardware existed, this solution was not considered any further.

[22] Division by powers of two were not counted as divisions.

[23] MAC = multiply and accumulate.

[24] The cycle numbers are given for an implementation on a SHARC ADSP-21000 DSP [70], using single precision floating point calculation.

- Division calculated with iterative convergence algorithm needs 6 cycles.
- Square roots calculated with Newton-Raphson iteration with initial seed from ROM based table needs 10 cycles.

Algorithm	ADD	MULT	DIV[22]	roots	memory access
Undistortion	22	33	0	0	–
Transformation	66	66	11	0	–
Distortion	33	33	11	22	–
Resampling	66	55	0	0	44
Total	187	187	22	22	44

Table 10.4: *Estimated computational power for a perspective-affine transformation with correction of radial distortion (in MOPS).*

necessary computational power of about 880 Mega MAC operations per second[25]. This exceeds the performance of today’s fastest DSP, the TMS320C80, by factors. Therefore it is inevitable that we use a multi-DSP system, which increases system cost and programming effort.

Using Texture Mapping of Graphics-Processors

In the past years many 3D graphics-processors with fast hardware supported texture mapping facilities came to market. Applying a texture onto an arbitrary surface in 3D-graphics is analog to the transformation used in the “inverse stereo principle”. Therefore it is possible to construct a 3D surface (consisting of triangles) which corresponds to the necessary image transformation when that image is applied as texture to the rendered surface. An approximation of the distortion correction can also be included.
The core of some of these graphics-processors is also available such that it can be used for a custom VLSI design of a spatial transformer.

Using Mapping Table

Since the calculation of the mapping coordinates is very computationally intensive, the necessary computation is drastically reduced if the coordinates are precalculated and stored in mapping tables. Consequently, the transformation is reduced to reading the addresses with

---

- Cubic root calculated with the power approximation using pseudo extended-precision arithmetic needs 37 MAC cycles.

[25] It is assumed that the additions can always be calculated in parallel to a multiplication; an additional 1000 Mega MAC operations would be used for the transformation of an elliptical safety envelope.

the interpolation coefficients from memory and to calculating the interpolated pixel value. In order to allow for dynamically changing the transformation, different maps may be precalculated and stored.

Hybrid Solution

The largest part of the computational power is used for distortion correction (see Table 10.4). Since the distortion remains constant for a camera setup, it is more economically practical to precalculate the distortion factors and store them in a look-up-table[26]. In order to calculate the undistorted coordinates

$$\begin{aligned} x_u &= x_d(1+\kappa_1(x_d^2+y_d^2)) = x_d\,\lambda_u \\ y_u &= y_d(1+\kappa_1(x_d^2+y_d^2)) = y_d\,\lambda_u \end{aligned} \qquad (10.6)$$

the factor $\lambda_u(x_d, y_d)$ is stored in a LUT, which saves 2 multiplications and one addition. Even 59 MAC operations are saved in the calculation of the distorted coordinates when the solution $\lambda_d(x_u, y_u)$ of the cubic equation

$$\begin{aligned} \varrho_u &= \varrho_d(1+\kappa_1\varrho_d^2) = \varrho_d\frac{1}{\lambda_d(x_u, y_u)} \\ \Rightarrow x_d &= \lambda_d(x_u, y_u)x_u \\ y_d &= \lambda_d(x_u, y_u)y_u \end{aligned} \qquad (10.7)$$

is stored in a LUT. Using the values x_u and y_u rounded to the nearest integer value introduces only small errors[27] but drastically reduces the size of the required LUT. Consequently, $\lambda_d(x_d, y_d)$ must only be stored for integer values. This method results in a necessary computational power of 230 Mega MAC operations and might be implemented with a TMS320C80.

Continuous Coordinate Map Update

Taking into account that the transformation will hardly change every field but at most several times per second, it is possible to combine

---

[26] Instead of combining the distortion correction on the coordinate level, it is possible to combine the three transformations on the image level. However, this has the disadvantage that 3 (instead of 1) bilinear interpolations must be calculated, which not only increases the computation but also the introduced errors.

[27] For $\kappa_1 \approx 10^{-3}$ and a sensor size of 6.5 mm (756 pxl) × 4.8 mm (572 pxl), rounding results in a maximal error of 0.2 μm $\approx$ 0.025 pxl.

the previous two methods. During the period in which the transformer reads the mapping coordinates from one memory block, a DSP can calculate the coordinate table for the next transformation and store it in another memory bank. There is no need for the DSP to calculate the transformations in real time, but the transformer can switch to the newly calculated mapping table as soon as the DSP has finished. The DSP may calculate the transformation with or without LUT for the distortion correction, depending on the necessary updating cycle. With a 50 MIPS DSP (e.g. TMS320C54x), this results in almost 10 updates per second[28].

Conclusion

The fully DSP based transformation was not considered any further because of the huge computational power. Although the hybrid solution needs much lower processing power, it still has the drawback that for arbitrary transformations a multi-DSP system must be used. At the time when the decision about implementation took place, 3D graphics processors with hardware supported texture mapping were announced, but not available.

Therefore a mapping-table based transformer with multiple switchable tables was implemented. With this implementation it was possible to demonstrate arbitrary, switchable transformations with distortion correction. In addition, it is possible to calculate or reload new mapping tables by the RISC of the PRIMA*SPEED* during operation.

10.4.2 Implementation of the Transformation Module

The transformation module is capable of transforming the images at video-rate and interpolating subpixel values by bilinear interpolation. The transformer uses precalculated mapping tables and can switch between up to 15 stored tables at field-rate.

These tables consist of the address to the upper left source pixel (17 bit) and two 3 bit wide interpolation weights. In order to facilitate the reloading of these tables during operation (from file or directly calculated by the RISC) with a high bandwidth, video-ram (VRAM) was used to store these tables. This simplifies the arbitration between read-accesses of the transformer and write-accesses of the RISC, because the

---

[28] Less than two updates per second for elliptical safety envelopes.

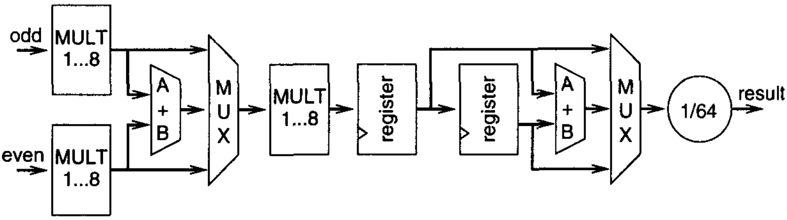


Figure 10.3: *Block diagram of bilinear interpolation circuit (clocked with double pixel clock rate)*

transformer needs to initiate a new read transfer of RAM data to the serial access memory of the VMEM only every 256 pixels; new data is then available at every transition of the serial clock. Between these read transfers the RISC has full access to the parallel databus of the VMEM, such that it can access the VRAM during more than 90% of the time. In addition, connecting the serial data bus to the transformer and the parallel data bus to the RISC bus simplifies the hardware.

In order to make the transform of each field possible, a ping-pong memory is used such that the new image is stored in one memory bank while the previous image is transformed, reading the data from the other bank. Because of the bilinear interpolation four source pixels must be read to produce one transformed output pixel. Because with FPGAs and off-the-shelf SRAM it is not possible to read 4 values within 60 ns[29], the image memory was built from two banks read in parallel to increase memory access bandwidth; in one bank the even and in the other the odd pixels are stored. In this way a read-access bandwidth of 60 MByte/s is reached.

There are two address generators in the transformer module, one produces the write addresses and the other generates the four read addresses from the address in the mapping table. The values of the source data pixels are then fed to the bilinear interpolation circuit at the double pixel clock rate. Owing to the restricted resolution of the interpolation weights (3 bit), the multiplication by the weight can be implemented by using only adders which add differently shifted pixel values. The block diagram of the bilinear interpolator is presented in Fig. 10.3.

The transformer is controlled by two finite state machines. The master

---

[29] The output propagation delay and setup time of the fastest FPGA from XILINX are together ≈10.5 ns, and consequently the read-access time of an SRAM must be below 4.5 ns.

is clocked with the pixel clock and generates all necessary signals except those that control the write access of the RISC, which are generated by a state machine clocked with the RISC clock.

10.4.3 Subsampling and Packing

In order to get a short reaction time, the entire system works with fields (50 fields/sec) instead of frames. A field has only half the resolution in y-, but full resolution in the x-direction compared to a frame. However, when applying the sobel filter, the geometric resolution in both directions should be equal. Therefore the fields are subsampled in the x-direction by averaging every two subsequent pixels on a scanline. In addition, in this way the computation of the subsequent algorithms is reduced. The subsampling was performed after the transformation to reduce errors introduced by the transformation.

The PRIMA*SPEED* is designed to process fields with full resolution in the x-direction. Therefore working with subsampled images exploits the hardware only partly. Consequently it is much better to pack two subsampled stereo images into one image by interleaving the pixels on one line. This reduces the two data-streams to a single one and simplifies the generation of neighborhoods for filters. In the subsequent algorithms this is appropriately taken into account.

10.4.4 Lowpass Filtering

As for all two-dimensional filters and neighborhood operators it is necessary to delay the data stream to have access to data that occurred earlier in the data sequence, i.e. pixels with lower x- or y-coordinates. In PRIMA*SPEED* the delay-line is implemented with one FIFO of fixed length, clocked at the double pixel clock rate. As such the data is available after one and two image lines and a neighborhood of 3 rows and any number of columns[30] can be produced. Since the input data consists of two interleaved images (transformed left and right image), a kernel of the following form, which calculates the filter values of the two images alternately is used.

$$L = \begin{array}{|c|c|c|c|c|}\hline 1 & - & 2 & - & 1 \\ \hline 2 & - & 4 & - & 2 \\ \hline 1 & - & 2 & - & 1 \\ \hline \end{array} \tag{10.8}$$

---

[30] The number of columns depends on the number of registers used to produce additional delay.

The implementation of this filter needs low hardware resources because the single weights as well as the column weights are powers of two and therefore the multiplications and the division by 16 are realized by shift operations[31].

10.4.5 Image Correlation

The correlation consists of two neighborhood operations, implemented in two separate FPGAs:

- calculation of the direction of the sobel gradient
- calculation of SAD-correlation on direction values.

First the images are filtered with two kernels applied to the same neighborhood[32] to produce the sobel-gradient in the x- and y-directions. The angle of the sobel gradient is then calculated by

$$\varphi = \mathrm{atan2}(S_x, S_y) \tag{10.9}$$

which is precalculated for all values and stored in a look-up-table.

Reduction of LUT Width

The range of both gradients is -1020 to +1020 (= 10 bit plus 1 sign bit). In order to reduce the size of the LUT from 4 MWord to 256 KWord, the resolution of the gradients is reduced. Because just ignoring the least significant bits leads to large errors for small gradients, both gradients are scaled according to the common largest absolute value such that the two most significant bits get 0 and can be left out. The error introduced by this scaling and reduction of resolution is smaller than 0.5°.

Reduction of Direction Resolution

In order to decrease the hardware complexity of the correlator, the direction resolution should be reduced before (in the LUT) or after the direction subtraction. In order to determine the error as a result of that reduction in resolution, the segmentation-error of correlations with various resolutions was investigated[33].

It is advantageous to reduce the resolution *after* calculating the direc-

---

[31] In hardware implemented by appropriate wiring.

[32] Consequently only one delay-line is necessary.

[33] The same image material as for the investigations about the correlation methods was used[71], see Chapter 8.4.1.

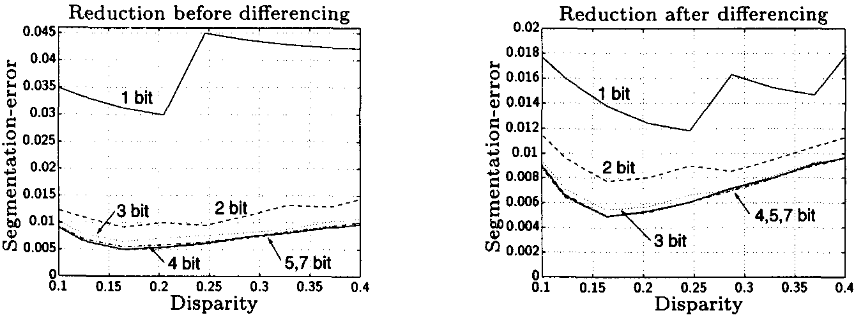


Figure 10.4: *Segmentation-error in function of disparity and different resolutions of direction difference (1,2,3,4,5,7 bit) for 5×5 correlation kernel (the lower the error at a specific disparity the better the method).*

tion difference. In this way almost the same performance is obtained as when reducing the resolution *before* calculating the difference but with a resolution increased by 1 bit. In Figure 10.4 the segmentation-error as a function of the disparity for various resolutions is shown for the correlation with a 5×5 kernel. It can be seen that there is no significant improvement for resolutions $\geq$ 4 bit when reducing *after* differencing or $\geq$ 5 bit when reducing *before* differencing. The required resolution is even lower for larger correlation kernels since the quantization noise is filtered out to a higher extent: a resolution of 3 bit and 4 bit, respectively, is sufficient. The difference in performance is for the following reason: A difference smaller than $180^o/2^n$ (n = resolution of direction difference in number of bits) always results in a difference of 0 if the reduction is made after the subtraction. However, when the resolution is reduced before subtraction, the LSB of the result is undefined, which results in higher dissimilarity values for corresponding regions. This effect is quite noticeable up to a resolution of 3 bits ($\hat{=}\pi/8$); in corresponding pictures 95% (85%) of the direction difference are smaller than $\pi/4$ ($< \pi/8$).

From the fact that a resolution of the direction of 8 and 7 bit (reduction before differencing!) produces the same result, it can be seen that a resolution higher than 7 bit does not increase performance. The difference is only noticeable at a direction resolution of less than 6 bit. This coincides with the fact that the direction difference information (resolution: 7 bits) is superimposed with noise having a mean value of approximately $\pi/16$ ($\hat{=}$ 3 bit).

The value stored in the LUT consists of the direction (8 bit) and magnitude (4 bit) of the sobel gradient, which is used for the texture analysis. However, the magnitude must be multiplied by the inverse of the scaling factor (0, 2 or 4) in order to undo the scaling of the gradients.

In the next step the cyclical difference of the direction is calculated, its resolution reduced from 7 to 4 bit and it is summed up in a 9×9 neighborhood. The reduction in resolution is necessary because of the restricted width of the FIFO. However, examination showed that a reduction[34] to 3 bit is tolerable.

The neighborhood is again produced with the FIFO, but as the data width is only 4 bit, it can be enlarged to 9×9. The correlation value is produced by summing up all difference values of this neighborhood. Subsequently this value is thresholded to produce the binary image.

10.4.6 Image Segmentation

As a first step in the segmentation the correlation value and the texture intensity are thresholded to produce binary signals. Unfortunately these signals are noisy such that measures must be taken to reduce spikes resulting from noise. Therefore, the binarization is followed by a filter to eliminate lonely pixels and very small pixel clusters which do not belong to objects (see also Section 5.2). The majority decision algorithm was implemented because it produces good results and is easier to implement in hardware than the "clusterlet elimination" algorithm. It is implemented with a FIFO to generate the 3×3 neighborhood (up to 9×9 possible), an adder to add the values and a comparator to decide whether the pixel is set to 0 or 1. The same procedure is carried out for the binary signal representing the texture intensity. It is done in the low level part in order to reduce the computation at the high level image processing, which extracts objects within the separation skin.

Due to the fact that it is more economic for a high level algorithm to work on a transition coded representation of a binary image than on the image itself, the binarized and filtered image is transition coded.

In the current system a high level image interpretation is not implemented. However, a simple algorithm which produces an alarm if an object is within the separation skin was implemented completely on the low level part of the PRIMA*SPEED*. For that, the neighborhood of the majority decision was increased to 7×7 in order to suppress small ob-

---

[34] The reduction should be performed after and not before differencing.

jects. Then the number of pixels in one field is counted and an alarm is given, if the number exceeds a predefined limit.

10.5 Resources Used

For a realization in hardware the hardware resources used are more appropriate than the computational power. Therefore in Table 10.5 the hardware resources which are used for implementation on PRIMA*SPEED* using programmable devices are presented. The number of gate equivalents is an approximation according to the percentage use of the FPGAs. The video RAM is calculated for 8 mapping tables.

Algorithm	gate count	SRAM	VRAM	FIFO
Transformation	8000	512 KByte	5 MByte	—
Lowpass	2000	—	—	2 KByte
Sobel-direction	3500	256 KByte	—	2 KByte
Correlation	2000	—	—	2 KByte
Texture Analysis	2000	—	—	2 KByte
Segmentation	1500	—	—	(2 KByte)
Transition-coding	2000	32 KByte	—	—
Total	21000	800 KByte	5 Mbyte	10 KByte
Approx. Si-area	9 mm^2	900 mm^2	—	12 mm^2

Table 10.5: *Hardware resources for implementation on PRIMASPEED. Approximate number of gate equivalents according to FPGA utilization. Estimated silicon area for an implementation with a 0.5 μCMOS-process (with macro-cells).*

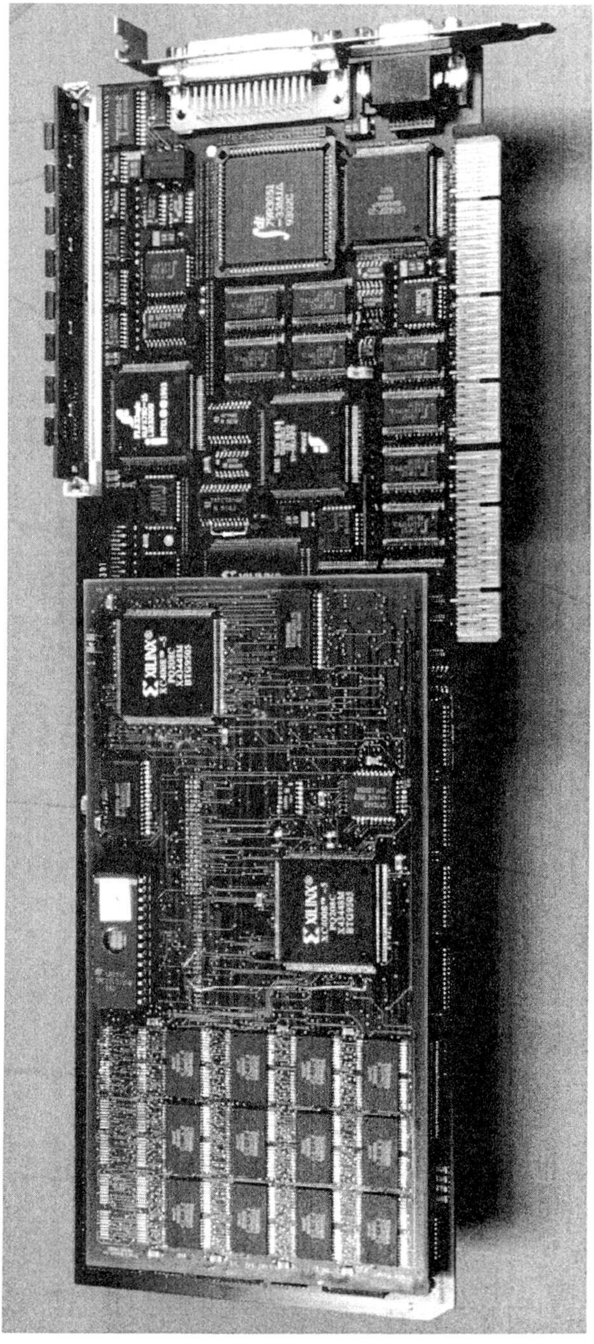


Figure 10.5: *PRIMASPEED PC card with transformation module (on left hand side).*

Chapter 11

Implementation and Tests

In this chapter some considerations about the system setup and the results of the tests of the implemented workspace monitoring system are presented.

11.1 Geometric Setup

Besides the correlation method, the geometric setup determines the performance of the system. With an adequate setup, the supervised area, as well as the thickness of the separation skin, can be tailored.

The thickness of the separation skin depends on the translation tolerance (see Chapter 8), the baseline of the stereo cameras and the distance to the surface. For coplanar cameras, the thickness of a coplanar separation skin is calculated according to

$$\text{Thickness} = \frac{2\,Dz^2\,d_x}{b\,f} = \frac{2\,D\,f\,A_c^2\,d_x}{b\,s^2} = \frac{2\,D\,z\,A_c\,d_x}{b\,s} \qquad (11.1)$$

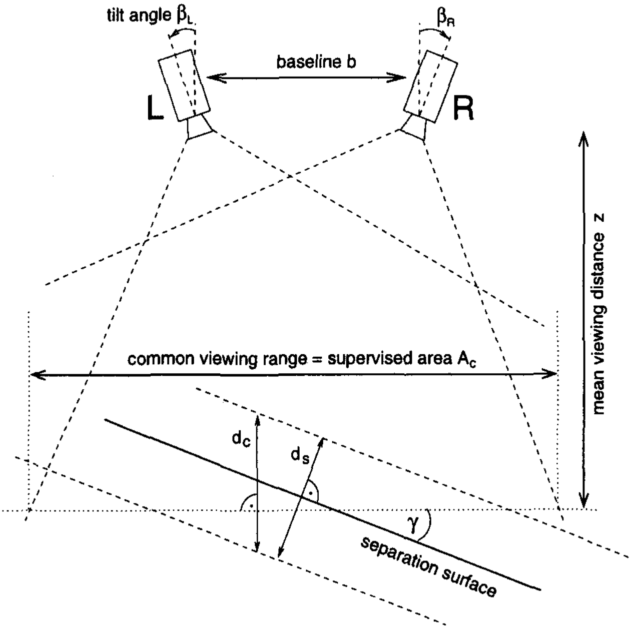


Figure 11.1: *Experimental setup with safety envelope*

with D = disparity tolerance
b = baseline of stereo cameras
f = focal length
z = distance to separation skin
d_x = pixel spacing (in x)
A_c = supervised area
s = width of sensor .

From Equation (11.1) it can be seen that for a given supervised area A_c, the thickness of the separation skin decreases with smaller focal length f and pixel spacing d_x and larger baseline b and sensor area s. Because of the dependence on distance, non-coplanar separation skins have a variable thickness.

In order to maximize the usable viewing area of the cameras, the cameras can be tilted such that their viewing areas overlap to a higher degree. However, for tilted cameras and non-coplanar camera setups Eq. (11.1) no longer holds true and the thickness depends on the tilt-angle of the cameras and is a function of its location (see Fig. 11.3). In Figure 11.1 the surface thickness and the common viewing area as

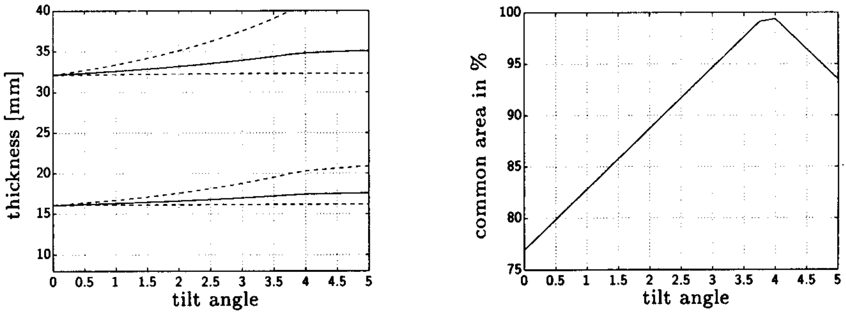


Figure 11.2: *The plot on the left-hand side shows the thickness of the separation skin (minimal, mean and maximal distance) for a segmentation-disparity of 0.7 pxl as a function of tilt-angle for an object distance of 0.8, 1.6 and 2.4 m with a baseline of 150, 300 and 450 mm, respectively. The plot on the right-hand side shows the exploitation of the camera image (common viewing range* A_c*/viewing range of a single camera* $A_{L/R}$*) as a function of tilt angle.*

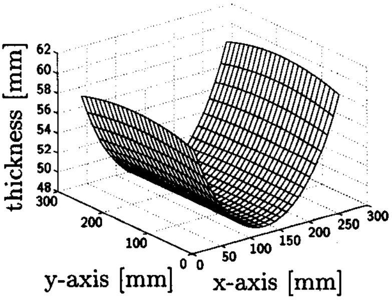


Figure 11.3: *Calculated thickness of separation skin as a function of position for tilt angle of 4°, a distance of 2.4 m and a baseline of 450 mm with a segmentation-disparity of 0.7 pxl.*

a percentage of the viewing area of a single camera are presented as a function of the tilt-angle.

The definition of the safety envelope must meet several restrictions:

- The entire surface must not be obstructed by the robot or other objects. In case this is not possible, a multi-camera approach may be useful.
- Both cameras must view the surface from the same side and under

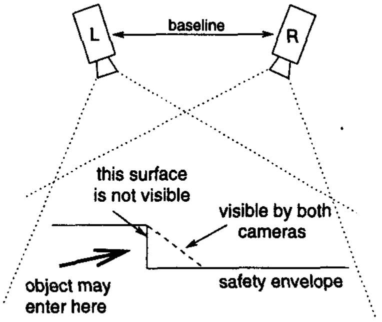


Figure 11.4: *Because the vertical surface is not seen by camera L, an object may enter the protected zone at this side of the safety envelope without being detected. This is solved by the safety envelope represented by the dashed line.*

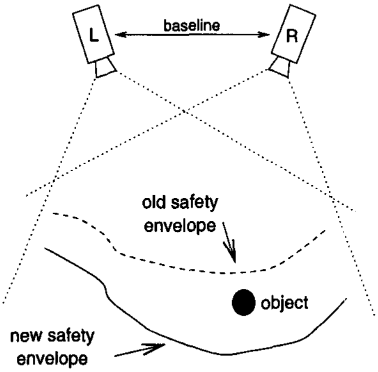


Figure 11.5: *When the safety envelope has changed, the object is unintentionally included into the protected zone. In order to prevent this, the safety envelope should be changed in small steps.*

a viewing angle which is larger than the critical angle ($\approx 70^o$; see Section 11.2).

- It must not be possible to enter the protected zone by another way than through the separation skin (see Fig. 11.4).
- When the safety envelope is dynamically changed, one must take care to change it smoothly enough such that no objects get unintentionally included in the protected zone (see Fig. 11.5).
- It must be taken into account that a separation skin tilted against the system base has a reduced thickness and consequently the maximal allowed speed of objects is reduced.

11.2 Experimental Data About the System

The data of the geometric setup of the experimental system is given in Table 11.1. In the following the results of some analysis with this system and problems are presented.

---

[1] Velocity = frame-rate × thickness of separation skin.

Characteristic	value
Resolution of camera [pxl]	756×581
Used image size (field) [pxl]	378×256
Images/sec	50
Focal length of lenses	10 mm
Mean viewing distance	800 mm
Baseline	152 mm
Tilt angle of cameras β	4^0
95%-Segmentation-disparity ($D_{95\%}$)	0.7 pxl
Thickness of safety skin (d_c) at 700 mm	$\approx$ 7.2 mm
Thickness of safety skin (d_c) at 800 mm	$\approx$ 9.4 mm
Thickness of safety skin (d_c) at 900 mm	$\approx$ 11.2 mm
Maximal velocity[1] of objects at 800 mm	0.45 m/s

Table 11.1: *Data of experimental system (thickness given for a separation skin perpendicular to the axis of symmetry of the system).*

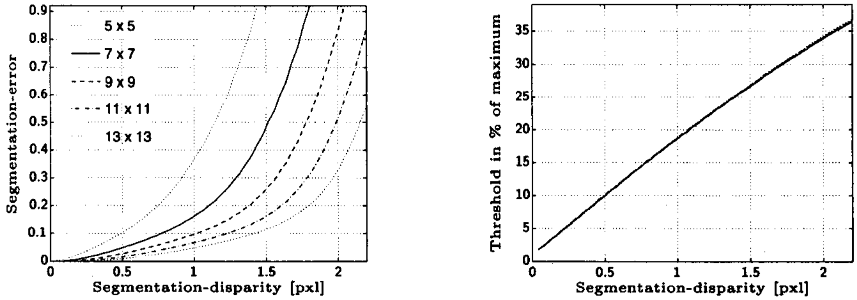


Figure 11.6: *Segmentation-error and chosen threshold as a function of segmentation-disparity for sobel-direction correlation for all kernel sizes.*

11.2.1 Thickness of Separation Skin

In the following, the thickness of the separation skin with the DSAD correlation with a 3×3 lowpass prefilter is analyzed. Besides the parameters of the geometric setup (see Eq. (11.1)), the thickness of the separation skin is a function of the correlation method and the threshold used for binarization. The threshold for binarization may be chosen by using the plot of the segmentation-error and the corresponding thresh-

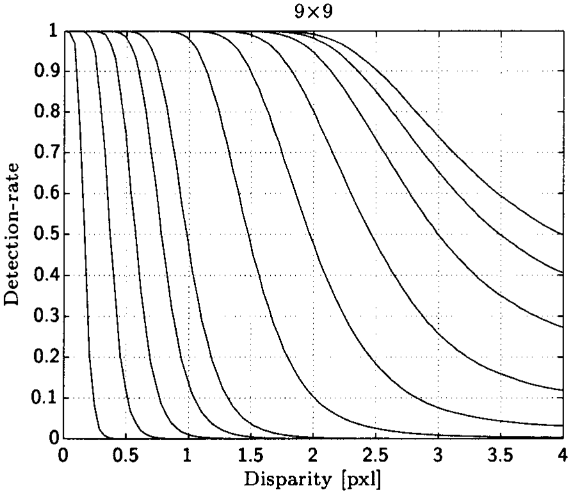


Figure 11.7: *Detection-rate for sobel-direction correlation with kernel size 9×9.*

old in Fig. 11.6. It should be taken into consideration that for a given segmentation-disparity only 50% of the pixels are above the threshold. However, with the help of the detection-rate plot in Fig. 11.7, a threshold can be selected for which a given percentage of the pixels are detected at a given disparity.

The data presented so far in the segmentation-error and threshold plots were produced with experiments using images with reduced noise (five images were averaged) and without geometrically transforming the images. Therefore the thickness of the separation skin was additionally evaluated with real images from the system and compared with the thickness estimated from the segmentation-error plots (see Table 11.2). It can be seen that the data corresponds very well (low deviation), except for the smallest threshold, which may be due to the noise introduced by the spatial transformation.

11.2.2 Slope of Separation Skin and Object

So far it was assumed that the separation skin is approximately perpendicular to the axis of symmetry (or parallel to the system base). In this case the thickness of the separation skin is equal to the tolerated translation ($T_{tol} = d_c$) in the direction of the axis of symmetry. However,

% of T_{max}	disparity		Thickness and deviation to measured data					
	95%	50%	d_c = 550 mm		d_c = 650 mm		d_c = 750 mm	
			[mm]	[%]	[mm]	[%]	[mm]	[%]
11.3	0.4	0.6	2.5	-21.3	3.5	-15.3	4.7	-15.0
15.0	0.6	0.8	3.5	0.1	4.9	2.7	6.5	-7.3
18.4	0.7	1.0	4.4	1.2	6.2	-3.2	8.2	-2.9
21.9	0.9	1.2	5.5	-0.5	7.7	-2.6	10.2	-2.3
25.1	1.0	1.4	6.4	2.3	8.9	-4.0	11.8	-6.5
28.4	1.1	1.6	7.3	2.6	10.2	-1.8	13.5	-0.2

Table 11.2: *Thickness of separation skin and deviation from measured data at three different heights. Threshold is given as percentage of maximal possible value T_{max}.*

with a separation skin tilted by angle γ against the base, its thickness is reduced to

$$d_s = d_c \cos\gamma \ . \tag{11.2}$$

This results in a reduced probability of detecting objects with surfaces not parallel to the separation skin. Experiments carried out with the experimental setup showed that the safety envelope can be tilted up to 70° against the system base, but with the consequence of reduced thickness ($\approx 1/3$). With this extreme angle only objects which enclose an angle of less than 15° with the safety envelope could be detected.

Objects entering the protected zone may enclose an arbitrary angle with the defined safety envelope. If the object surface is tilted by ω against the safety envelope, which is parallel to the ground plane, the detectable zone of the object is reduced to that portion of the object surface which is within the separation skin and projected onto the camera plane. For an object tilted about one axis by ω, the detectable object area is reduced by the factor $1/\tan\omega$.

If the object surface remains perpendicular to the axis of symmetry, but the safety envelope is tilted, as in the following experiment, the detectable object area ($\cong d$) is reduced because of the reduced thickness of the separation skin and the projection onto the image plane to

$$d = \frac{d_s}{\sin\gamma} = \frac{d_c \cos\gamma}{\sin\gamma} \ . \tag{11.3}$$

In Figure 11.8 it can be seen that an object is well detectable up to an angle of 50° with highest thresholds, but the detected zone of the object

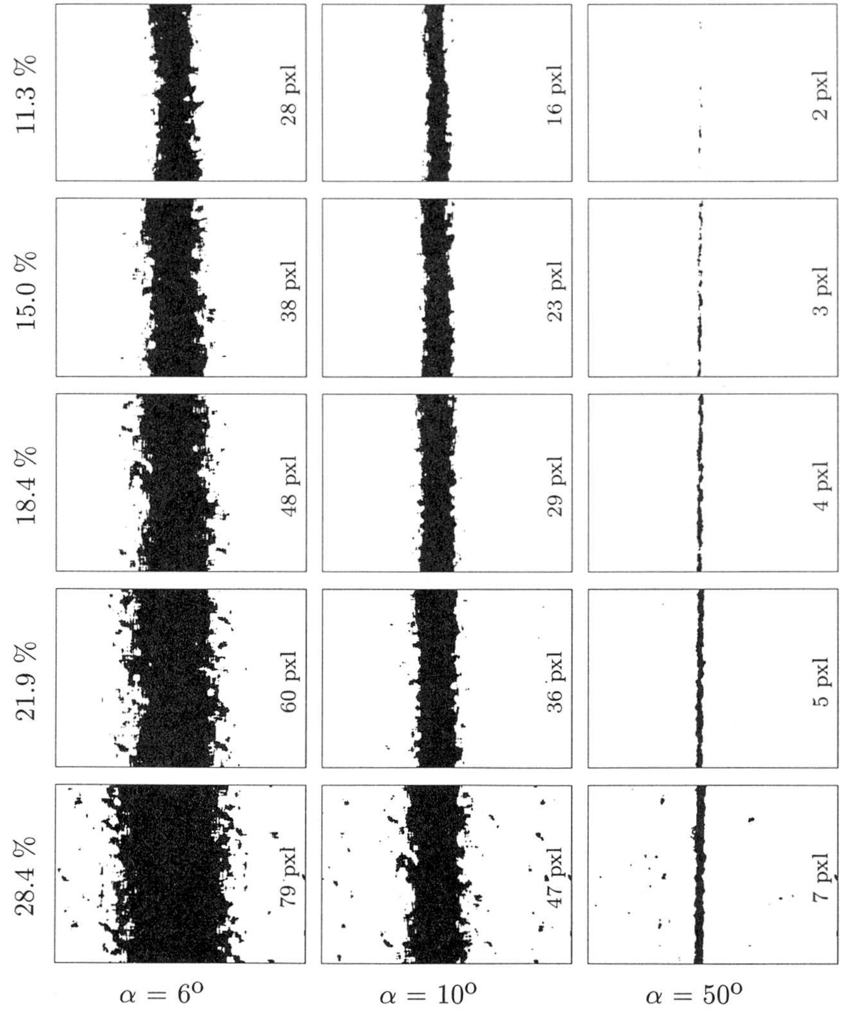
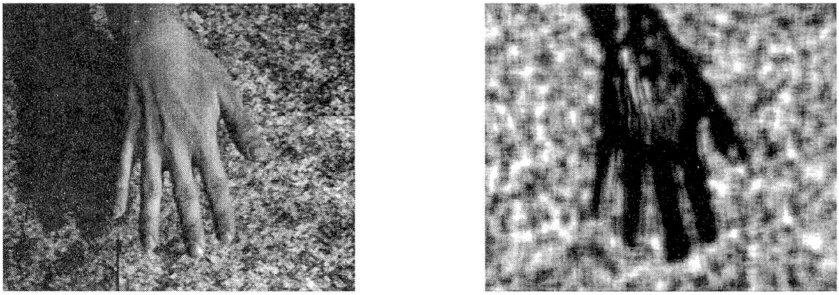


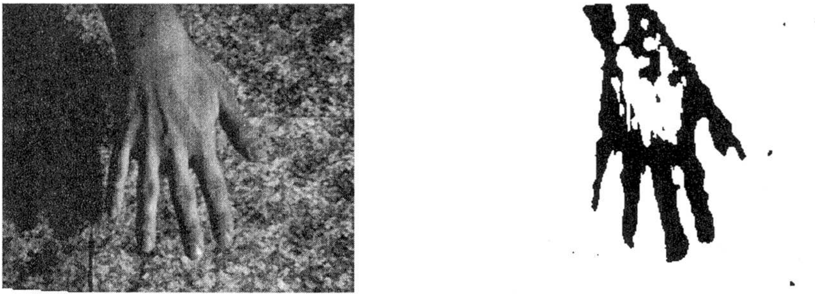
Figure 11.8: *Detected zone of objects with different slopes and processed with different thresholds (in % of maximal value). Width of theoretical detectable object area is given in the images and corresponds very well with the measured data.*

gets very small. A threshold above 25% results in erroneous segmented pixels (see image with $\alpha = 50^o$ and threshold = 25.1% in Fig. 11.8) and is therefore not applicable.



left image

correlation result



transformed image

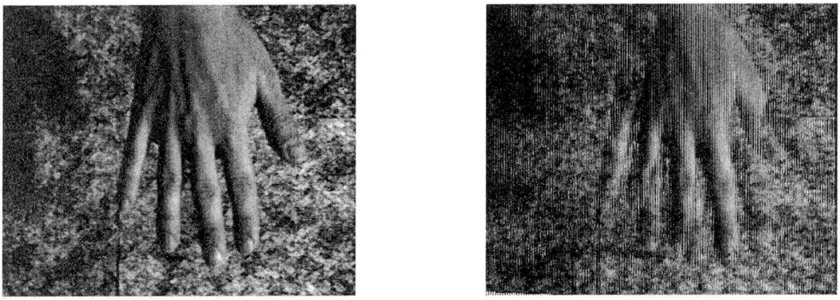
result

Figure 11.9: *Camera images and results of a hand situated in the safety envelope. The back of the hand is not detected because it is outside the separation skin.*

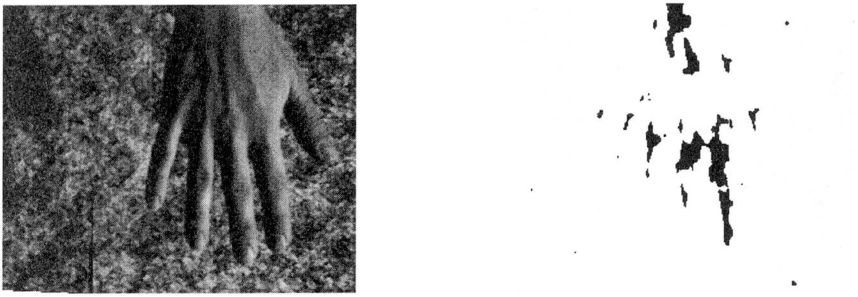
11.2.3 Problems

Besides the restrictions which must be met by the geometrical setup and by the definition of the safety envelope, the scene itself must fulfill some conditions. Not only low texture (see Chapter 9), but also repetitive texture may lead to errors.

To give a feeling for such errors, and to show that not only artificial surfaces show repetitive patterns, an example is given in Fig. 11.10. It can be seen that different fingers produce a very similar image, such that two non-identical fingers are classified as corresponding. This effect is intensified by the fact that the human hand has low texture, but a strong intensity change from dark to bright at *all* fingers (owing to shadow). In addition, in Fig. 11.9, an example where the hand is within the separation skin is presented.



left image overlaid image



transformed image result

Figure 11.10: *Camera images and results of a hand situated outside the safety envelope. As may be seen in the overlay of the left and transformed right image, two different fingers have a very similar image which results in high correlation values.*

A similar problem occurs with objects that have a monotone intensity change such that the direction of the intensity gradient is very similar in a large area. Since the DSAD algorithm only compares the direction, such an object may result in low dissimilarity values even if the intensity itself does not correspond. A solution may be to additionally include the intensity into the correlation as discussed in Section 12.4.

Under- or over-exposed images with image areas bound to the minimal or maximal intensity value also result in erroneous results. If both images have regions with constant values, these regions erroneously correspond since there is no image noise which reduces the correlation value for two non-corresponding areas like, e.g. well-exposed white paper. However, if only one image is over-exposed, objects could be missed.

11.3 Assessment of the Method

Using the "inverse stereo method" for a monitoring system has many advantages and some disadvantages. The following advantages make this method more promising for surveillance applications than usual stereo methods:

- the correction of lens distortions can be included in the transformation, which is an inherent part of the method and therefore this uses no additional computing power.
- the camera setup is not restricted by the algorithm since the transformation can include any translation or rotation. Therefore the cameras can be set up such that the common viewing area is near 100%.
- since the image is transformed such that the hypothetical image corresponds to the real image for objects at the separation surface, not only the position but also the orientation and scaling of such objects is the same in both images. This is advantageous in that the correlation method need not be invariant to scaling and rotation. The fact that objects outside the separation skin could have different scaling is even an advantage.
- because no correspondence search is necessary, a real time implementation of the inverse stereo algorithm requires less resources than that of traditional stereo methods.
- the method proved to be highly immune to changes in brightness, which mainly results from the correlation method.

On the other hand the method has some drawbacks:

- since only objects within the separation skin are detected, no information about objects before or behind the safety envelope is available. This imposes a problem at system start and whenever the separation surface is changed. Therefore whenever the workspace is enlarged, the safety envelope must be smoothly changed such that no objects are missed.
- fast objects could be missed, if they pass the safety envelope between two consecutive images.
- since only object surfaces which intersect the separation skin are detected, pointed or narrow objects which enter the protected zone with their narrow side have only a very small detectable area and

could be missed by this system. However, that is also a problem with conventional stereo systems.

- if the separation skin and the object surface have a large intersection angle, the detectable object zone gets smaller and an object could be missed.
- since an object must have a minimal size (see Section 8.5) to be detected, small objects might enter the protected zone without the knowledge of the supervision system.
- low texture may lead to 'invisible' objects and repetitive patterns to false alarms.

Chapter 12

Possible Extensions

Many possible improvements that would make the system more robust and widen its application range exist. Possibilities with multiple cameras and multiple surfaces and the use of color in image correlation are presented. Further, a VLSI integration is discussed.

12.1 Multi-camera Systems

Up to now a system with two cameras has been analyzed and implemented. Like in stereo vision where multi-camera approaches result in a more robust correspondence search [63, 72, 73, 74], in the "inverse stereo algorithm" multiple cameras make the verification of the hypothesis more reliable. Especially with periodic patterns the probability of erroneously matching the repeated pattern with the original pattern is reduced with an increasing number of cameras.

In addition, multiple cameras make it possible to look behind obstacles of one camera (see Figure 12.1). Although an object cannot impede the sight to an unambiguously defined safety envelope unless it is within the protected area, the surface may be impeded by known obstacles or the robot. In such situations multiple cameras could guarantee the visibility of all safety envelopes.

The most important advantage of a multi-camera setup is the decrease in geometric restrictions imposed on the definition of safety envelopes. To guarantee the visibility of the entire safety envelope, the angle between the separation skin and the camera direction is restricted, which

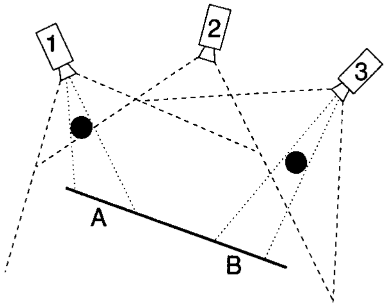


Figure 12.1: *Setup to cope with obstacles in field of view: area A is not visible by camera 1, but visible by the other two cameras, whereas area B is only visible by cameras 1 and 2.*

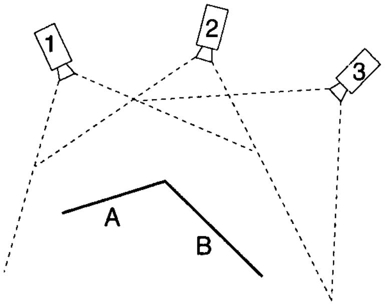


Figure 12.2: *Setup to enhance flexibility in defining safety envelopes: surface A is not visible by camera 3 but by cameras 1 and 2 whereas area B is visible by cameras 1 and 2 but not by camera 3.*

decreases the flexibility in defining safety envelopes. With multiple cameras the safety envelope can be divided into different patches separately obeying the angle condition for an individual camera pair (see Figure 12.2).

Moreover, the supervised zone can be enlarged with multiple non-overlapping views.

The necessary hardware resources (or computing power) is scaled by a factor $(n-1)$, where n is the number of cameras.

12.2 Multi-Surface System

In the application mentioned so far (robot workspace supervision) it is possible to switch between different safety envelopes in order to adapt the workspace, but only one safety envelope is active for a defined workspace. However, it is possible to use multiple safety envelopes in order to retrieve additional information. These safety envelopes can either be switched sequentially or, with an extension of the hardware, multiple surfaces can be active simultaneously. Whether multiple safety envelopes are calculated sequentially or in parallel depends on the velocity of the moving objects and the timing constraints for the algorithm. In the following some new applications based on this multi-surface approach are listed.

Spatial Scanning
By using a series of safety envelopes with different distances to the cameras, a defined space can be scanned and a three dimensional description produced.

Detection of Moving Direction
With two safety envelopes not only the presence of an object but also its direction of movement can be calculated by evaluating where it was first detected. In a system which controls an automatic door or elevator, it is possible to recognize whether a person moves towards the door or away from it.

Spatial Object Tracking
An object can be tracked in space by dynamically defining a separation skin before, after and at the position where the object was last found. If the object moves it will be detected by either the nearer or farther separation skin such that the direction of motion can be determined. Then for the next cycle the separation skins are moved into the direction the object has moved.

12.3 Detecting Objects Outside the Separation Skin

Instead of detecting the presence of objects within the separation skin it is possible to detect objects which are outside the separation skin. In this case the separation skin must be placed such that the generated hypothesis becomes true for the entire background. In this way, e.g. the floor can be supervised for objects having a height above a certain limit (see Fig. 12.3).

12.4 Using Color Images

Although our world is very colorful, color images are seldom used in image understanding. Up to now color was mainly used in image segmentation and texture classification, but very rarely in stereo vision.
Experiments have shown that color alone is not sufficient for humans to develop a three-dimensional impression of a scene, but it has a supporting effect. Most important for developing a 3D impression is the local

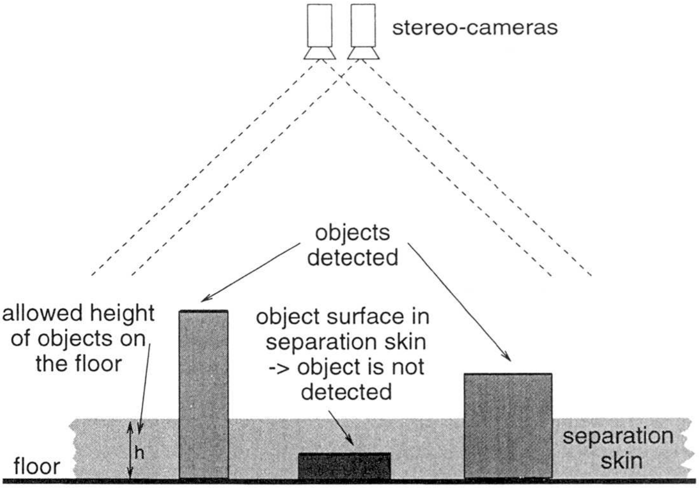

stereo-cameras
objects detected
allowed height of objects on the floor
object surface in separation skin -> object is not detected
h
floor
separation skin

Figure 12.3: *Detection of objects above the floor*

intensity variation, which is very much correlated between the three color channels. This is also reflected in the fact that the human eye has 10^8 grey-level sensitive rods and only $6.5\ 10^6$ color sensitive cones. Recently some researchers have used color information for template matching and stereo vision:

- With a photometric stereo algorithm using color and luminescence information, it is possible to get local (one pixel!) shading information in the presence of specular and diffuse reflection [75].
- In [76, 77] the dissimilarity measure (sum of squared differences) has been extended to color images by using the Euclidean distance. It could be shown that the results were as good as or slightly better than those produced without color information. This has mainly two reasons: there is slightly more information available and since the dissimilarity measure is calculated from three (highly correlated) images with superimposed uncorrelated noise, the amount of noise is reduced.
- In [78] color is used as an additional characteristic besides sign and orientation of the intensity gradient in the search for corresponding edges. Thus the number of possible edge-correspondences is reduced, which reduces the necessary computation for subsequent

analysis. Using colors results in a significant improvement for edge based algorithms.

Color information may be used in different ways in area based correlation:

- extension of dissimilarity measures to color images, using the Euclidean distance:

$$\mathrm{SSD}_{\mathrm{col}} = (R_1 - R_2)^2 + (G_1 - G_2)^2 + (B_1 - B_2)^2 \qquad (12.1)$$

- combination of a high-pass based correlation criterion with a (color) dissimilarity measure. Two templates are classified as corresponding only if both correlation measures produce a high score. In this way templates with similar high-frequency structures but very different color or intensity are not erroneously treated as corresponding. However, a similar procedure can also be applied to grey-level images.
- instead of calculating two correlation criteria, it is possible to exclude those image areas from the correlation which definitely do not correspond. Thus the entire image is partitioned into patches and for every patch a measure of the color difference calculated. This method has the advantage of reduced computation since this new measure is only calculated for patches and not for every pixel.

However, the use of colors not only slightly increases the robustness of the system, but also increases computation and hardware resource requirements:

- Because it is necessary to use a 3-chip camera (one chip for each color) the system costs are drastically increased. The use of a 3-chip color camera is important because color CCD chips with integrated color filters have reduced spatial resolution and the intensity information shows errors because the intensity is retrieved from the color information.
- The computational load is tripled for color dissimilarity measures and even higher for combined correlation criteria. When color information is used to exclude very dissimilar image areas from subsequent correlation, the computation load is only slightly higher.

To sum up, one can say that color may be used and improves the robustness of template matching, but the additional expenditure must be compared to the expenditure and improvement gained by using multiple cameras.

12.5 VLSI Integration

Implementation of the algorithm in hardware has shown that the logic needs less than 20 000 gates. Therefore the logic can easily be implemented on a VLSI chip (area < 9 mm^2 for 0.5 μm CMOS-process). The implementation of the FIFO on the same chip requires about an additional 11 mm^2. As the CMOS process used for logic implementation is not well-suited for implementation of the SRAM (necessary area ≈ 900 mm^2), it is not economically sound to implement the SRAM on the same chip. However, the entire system can be implemented on a multi-chip-module (MCM). In order to eliminate the VRAM, it is advantageous to use the circuitry for texture mapping of 3D-graphic (see Section 10.4.1) chips. In this way the entire system can be implemented in a single case for small baselines or multiple cameras with integrated processing for multi-camera systems with large baselines. Such a "smart camera" solution has the advantage of being versatile and inexpensive.

Bibliography

[1] Joseph F Engelberger. *Robotics in Service*. Kogan Page Ltd, London, 1989.

[2] Isaac Asimov. *I, Robot*. Gnome Press, 1950.

[3] N. Sugimoto, K. Kawaguchi. Systematic Robot-Related Accidents and Standardisation of Safety Measures. In *Robot Safety* (Edited by Maurice C. Bonney, Y.F. Yong). IFS Ltd, Springer-Verlag, 1985.

[4] J. Carlsson. Robot Accidents in Sweden. In *Robot Safety* (Edited by Maurice C. Bonney, Y.F. Yong). IFS Ltd, Springer-Verlag, 1985.

[5] Gudela Grote. Automaten für Menschen - nicht umgekehrt. In *Schweizerische Technische Zeitschrift*, no. 9 in 90, (pp. 46–48). Orell-Fuessli, September 1993.

[6] Dietrich Brandt. Automation in Manufacturing, Control versus chaos. In *Advances in Agile Manufacturing* (Edited by P.T. Kidd, W. Karwowski), (pp. 15–20). IOS Press, July 6–8 1994.

[7] Hiromu Nakazawa. Human Oriented Manufacturing System. In *Advances in Agile Manufacturing* (Edited by P.T. Kidd, W. Karwowski), (pp. 9–14). IOS Press, July 6–8 1994.

[8] Steffen Weik, Toni Wäfler, Eric Scherer. Komplementäre Systemgestaltung in der integrierten Produktion. *CIM-Management*, (6):43–47, 1995.

[9] K. G. Engelhardt. An Overview of Health and Human Service Robotics. *Robotics and Autonomous Systems*, 5(3):205–226, 1989.

[10] R. Mills. Selecting proper safeguarding devices for the robotic workplace. In *Proceeding of the Int. Robots and Vision Automation Conf.*, (pp. 2/39–51). Robotic Ind. Assoc, 5–8 April 1993.

[11] Howard R. Nicholls. Tactile Sensing for Robotics. In *Colloquium on Robot Sensors*, vol. 016, (pp. 5/1–5/3). IEE, London, 1991.

[12] B. Bury. Proximity Sensing for Robots. In *Colloquium on Robot Sensors*, vol. 016, (pp. 3/1–3/18). IEE, London, 1991.

[13] M.E.K. Graham. Safety Mats. In *Robot Safety* (Edited by Maurice C. Bonney, Y.F. Yong). IFS Ltd, Springer Verlag, 1985.

[14] Albert-Jan Baerveldt. Singularisation of Parcels with a Sensor-Based Robot System. In *IROS'91, IEEE/RSJ Int. Workshop on Intelligent Robots and Systems*, (pp. 523–8), Osaka, Japan, November 1991.

[15] S. Derby, J. Graham, J. Meagher. A Robot Safety and Collision Avoidance Controller. In *Robot Safety* (Edited by Maurice C. Bonney, Y.F. Yong). IFS Ltd, Springer-Verlag, 1985.

[16] K. Skifstad, R. Jain. Illumination Independent Change Detection for Real World Image Sequences. In *Computer Vision, Graphics, and Image Processing*, vol. 46, (pp. 387–399), 1989.

[17] A. T. Ali, E. L. Dagless. Alternative Practical Methods for Moving Object Detection. In *Int. Conf. on Image Processing and its Applications*, vol. 354, (pp. 77–80), Maastricht, Netherlands, 7–9 April 1992. IEE.

[18] Rebecca D. Horton. A target cueing and tracking system (TCATS) for smart video processing. In *Proceedings. IEEE 1990 Int. Carnahan Conf. on Security Technology: Crime Countermeasures* (Edited by J.S. Jackson), (pp. 68–72). IEEE New York, NY, USA, 1990.

[19] Ahmad Abdallah, Cina Motamed, Alain Schmitt. Change detection for human safety in robotic environments. In *Applications of Digital Image Processing XVII*, vol. 2298, (pp. 357–361). SPIE, 26–29 July 1994.

[20] Chand-Wu Fu, Shyang Chang. Change Detection with Moment Invariants under Time-Varying Illumination Case. In *Signal Processing V.: Theories and Applications. Proceedings of EUSIPCO-90, Fifth European Signal Processing Conf.* (Edited by L. Torres, E. Masgrau, M.A. Lagunas), vol. 2, (pp. 955–958). Elsevier Amsterdam, Netherlands, 1990.

[21] Patrick Lalande, Patrick Bouthemy. A Statistical Approach to the Detection and Tracking of Moving Objects in an Image Sequence.

In *Signal Processing V.: Theories and Applications. Proceedings of EUSIPCO-90, Fifth European Signal Processing Conf.* (Edited by L. Torres, E. Masgrau, M.A. Lagunas), vol. 2, (pp. 947–950). Elsevier Amsterdam, Netherlands, 1990.

[22] Vcenc Llario, Antonio Benito Martinez. Active Methods for Obtaining Depth Maps. In *Computer Vision: Theory and Industrial Applications* (Edited by Carme Torras). Springer Verlag, 1992.

[23] S. Inokuchi, K. Sato, F. Matsuda. Range–Imaging System for 3–D Object Recognition. In *Proc. of 7th Int. Conf. on Pattern Recognition*, (pp. 806–8), Montreal, Canada, 1984. IEEE Comput. Soc. Press.

[24] F.M. Wahl. A Coded Light Approach for Depth Map Acquisition. In *Mustererkennung 1986* (Edited by G. Hartmann), (pp. 12–17). Springer 1886, 1986.

[25] K.L. Boyer, A.C. Kak. Color-encoded Structured Light for Rapid Active Ranging. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(1):14–28, January 1987.

[26] Martin Rechsteiner, Bruno Schneuwly, Walter Guggenbühl. Fast and precise 3D sensor insensitive to ambient light. In *Optics, Illumination, and Image Sensing for Machine Vision VII* (Edited by Donald J. Svetkoff), vol. 1822, (pp. 188–199), Boston, Massachusetts, USA, November 1992. SPIE.

[27] P. Anandan. Motion and Stereopsis. In *Computer Vision: Theory and Industrial Applications* (Edited by Carme Torras). Springer Verlag, 1992.

[28] K.M. Jou, E. Belloum, et al. A reconfigurable and flexible parallel 3D vision system for a mobile robot. In *Computer Architectures for Machine Perception* (Edited by Magdy A. Bayoumi, Larry S. Davis, Kimon P. Valavanis), (pp. 215–221). IEEE Computer Society Press, December 1993.

[29] Smit Badal, Srinivas Ravela, et al. A Practical Obstacle Detection and Avoidance System. In *Second IEEE Workshop on Applications of Computer Vision*, (pp. 97–104), Los Alamitos, California, December 1994. IEEE, IEEE Computer Society Press.

[30] Keith H. Nishihara. Real-Time Stereo and Motion-based Figure-Ground Discrimination and Tracking using LOG Sign-Correlation. In *The 27th Asilomar Conf. on Signals, Systems & Computers*

(Edited by Avtar Singh), vol. 1/2, (pp. 95–100). IEEE, IEEE Computer Society Press, November 1993.

[31] N. Gouvianakis, K. Parthenis, B. Dimitriadis. A method for detection and tracking of moving objects in an industrial environment using stereo vision. In *Engineering Systems with Intelligence: Concepts, Tools and Applications* (Edited by S.G. Tzafestas), (pp. 349–56). Kluwer Academic Publishers Dordrecht, Netherlands, 1991.

[32] Peter Schaeren. *Real-Time 3-D Scene Acquisition by Monocular Motion Induced Stereo*. Ph.D. thesis, Diss ETH Nr. 10506, Swiss Federal Institute of Technology, Series in Microelectronics, Vol. 33, Hartung-Gorre Verlag Konstanz, 1994.

[33] Martin Rechsteiner, Bruno Schneuwly, Gerhard Troester. Workspace Monitoring System. In *Spatial Information from Digital Photogrammetry and Computer Vision* (Edited by H. Ebner, C. Heipke, K. Eder), vol. 30, (pp. 689–696), Munich, Germany, September 5–9 1994. ISPRS Commission III, SPIE, Washington, USA.

[34] Martin Rechsteiner, Markus Thaler, Gerhard Troester. Real Time Workspace Monitoring System: First Results. In *ISPRS Intercommission Workshop "From Pixels to Sequences"*, Zurich, Switzerland, 22–24 March 1995. ISPRS Commission III, SPIE, Washington, USA.

[35] David Coombs, Ian Horswill, Peter von Kaenel. Disparity Filtering: Proximity Detection and Segmentation. In *Intelligent Robots and Computer Vision XI: Algorithms, Techniques and Active Vision* (Edited by David P. Casasent), vol. 1825, (pp. 195–206), Boston, MA, 16–18 November 1992. SPIE.

[36] Maki Tanaka, Noriaki Maru, Fumio Miyazaki. Binocular Gaze Holding of a moving Object with the Active Streo Vision System. In *Proc. of the 2nd IEEE Workshop on Applications of Computer Vision*, (pp. 250–255), Los Alamitos, CA, 5–7 December 1994. IEEE, IEEE Computer Society Press.

[37] Stuart Cornell, John Porrill, John E. W. Mayhew. Ground Plane Obstacle Detection under variable Camera Geometry Using a Predictive Stereo Matcher. In *British Machine Vision Conf. 1992* (Edited by David Gogg, Roger Boyle), (pp. 548–559). Springer-Verlag, 22–24 September 1992.

[38] Reinhard Klette, Piero Zampani. *Handbuch der Operatoren für die Bildverarbeitung.* Vieweg Verlag, 1992.

[39] Chester C. Slama. *Manual of Photogrammetry, fourth edition.* American Society of Photogrammetry, 1980.

[40] Roger Y. Tsai. A Versatile Camera Calibration Technique for High–Accuracy 3D Machine Vision Metrology Using Off–the–Shelf TV Cameras and Lenses. In *IEEE Journal of Robotics and Automation, Vol. RA-3 No. 4*, (pp. 323–344), USA, August 1987.

[41] W.H. Press, B.P. Flannery, et al. *Numerical Recipes in C.* Press Syndicate of the University of Cambridge, 1988.

[42] Thomas M. Strat. Recovering the Camera Parameters from a Transformation Matrix. In *Readings in Computer Vision: issues, problems, principles and paradigms* (Edited by Fischler/Firschein), (pp. 93–99). Morgan Kaufmann Publishers, 1987.

[43] Juyang Weng, Paul Cohen, Marc Herniou. Camera Calibration with Distortion Models and Accuracy Evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14, No. 10, October 1992.

[44] Reimer Lenz, Roger Tsai. Techniques for Calibration of the Scale Factor and Image Center for High Accuracy 3D Machine Vision Metrology. In *IEEE Int. Conf. on Robotics and Automation Vol 1*, (pp. 68–75), Raleigh, North Carolina, April 1987.

[45] M.R. Shortis, T.A. Clarke, T. Short. A comparision of some techniques for the subpixel location of discrete targt images. In *Videometrics III*, vol. 2350, (pp. 239–250). SPIE, 1994.

[46] Robert J. Valkenburg, Alan M. McIvor, Wayne P. Power. An evaluation of subpixel feature localisation methods for precision measurement. In *Videometrics III*, vol. 2350, (pp. 229–238). SPIE, 1994.

[47] A. Hachicha, S. Simon. Subpixel edge detection for precise measurements by a vision system. In *Industrial Inspection*, vol. 1010, (pp. 148–157). SPIE, 1988.

[48] Ali J. Tabatabai, O. Robert Mitchell. Edge Location to Subpixel Values in Digital Imagery. *IEEE Transactions on Pattern Analysis and Machine Intelligance*, PAMI-6(2):188–201, March 1994.

[49] Suguta Ghosal, Rajiv Mehrotra. Orthogonal Moment Operations for Subpixel Edge Detection. *Pattern Recognition*, 26(2):295–306, 1993.

[50] Jack Koplowitz, Xiaobing Lee. Edge detection with subpixel accuracy. In *Automatic Object Recognition*, vol. 1471, (pp. 452–463). SPIE, 1991.

[51] J. Weng, N. Ahuja, T.S. Huang. Closed–Form Solution & Maximum Likelihood: A Robust Approach to Motion and Structure Estimation. In *IEEE Computer Vision and Pattern Recognition*, (pp. 381–386), 1988.

[52] I.N. Bronstein, K.A. Semendjajew. *Taschenbuch der Mathematik*. Teubner: Stuttgart, Leipzig; Nauka: Moskau; Harri Deutsch: Thun, Frankfurt/Main, 25th edn., 1991.

[53] George Wolberg. *Digital Image Warping*. IEEE Computer Society Press, California, Dept. of Computer Science, Columbia University, New York, 1990.

[54] Joseph Ward, David R. Cok. Resampling Algorithms for Image Resizing and Rotation. In *Digital Image Proccessing Applications*, vol. 1075 of *SPIE*, (pp. 260–269), Los Angeles, California, January, 17–20 1989. SPIE.

[55] A. Rosenfeld, A.C. Kak. *Digital Picture Processing*. Academic Press: New York, 1976.

[56] T. W. Ryan, B. R. Hunt. Recognition of stereo-images cross-correlation errors. In *Progress in Pattern Recognition* (Edited by L. N. Kanal, A. Rosenfeld), vol. 1, (pp. 265–322), 1981.

[57] Hans P. Moravec. *Robot Rover Visual Navigation*. UMI Research Press, Ann Arbor, Michigan, 1980/81.

[58] O. Faugeras, P. Fua, et al. Quantitative and Qualitative Comparision of some Area-based and Feature-based Stereo Algorithms. In *Proc. of the 2nd Int. Workshop on Robust Computer Vision*, (pp. 1–26), Bonn, 1992. ISPRS.

[59] Peter Aschwanden. *Experimenteller Vergleich von Korrelationskriterien in der Bildanalyse*. Ph.D. thesis, Diss ETH Nr. 10196, Swiss Federal Institute of Technology, Series in Microelectronics, Vol. 24, Hartung-Gorre Verlag Konstanz, 1993.

[60] Peter Seitz. Using local orientation information as image primitive for robust object recognition. In *Visual Communication and Image Processing*, vol. 1199, (pp. 1630–1639). SPIE, 1989.

[61] H.K. Nishihara. PRISM: A practical real time imaging stereo matcher. In *Optical Engineering*, vol. 23(5), (pp. 536–545), 1984.

[62] J.P. Siebert, C.W. Urquhart. Active Stereo: Texture Enhanced Reconstruction. *Electronics Letters*, 26(7):427–430, March 1990.

[63] Bing Kanga Sing, J.A. Webb, et al. A multibaseline stereo system with active illumination and real-time image acquisition. In *Fifth Int. Conf. on Computer Vision*, (pp. 88–93), Los Alamitos, June 1995. IEEE.

[64] Hansruedi Vonder Mühll, Björn Tiemann, et al. High Performance Multiprocessor Workstation with Intelligent Communication Network. In *Proceedings of the '94 SIPAR-Workshop on Parallel and Distributed Computing*, (pp. 1–4), Fribourg, Switzerland, October 1994.

[65] Anton Gunzinger, Urs A. Müller, et al. Achieving Supercomputing performance with a DSP array processor. In *Supercomputing '92* (Edited by Robert Werner), (pp. 543–555), Minneapolis, November 1992. IEEE Computer Society Press.

[66] A. Gunzinger. *Synchroner Datenflussrechner zur Echtzeitbildverarbeitung*. Ph.D. thesis, Diss ETH Nr. 9147, Zürich, 1990.

[67] A. Gunzinger. Concept and Realization of a Heterogeneous Multiprocessor System for Real Time Image Processing. In *Proc. of the Workshop on Computer Architecture for Machine Perception*, (pp. 263–269), Paris, France, December 1991.

[68] D. Stokar. Interactive Programming Environment for a Heterogeneous Multiprocessor. In *Conf. on Programming Environment for Parallel Computing*, Edinburgh (UK), April, 6–8 1992.

[69] Dieter Benedikt Stokar von Neuforn. *Benutzerzentrierte Betriebssoftware eines Echtzeit-Bildverarbeitungs-Rechners für die interaktive Applikationsentwicklung*. Ph.D. thesis, Diss ETH Nr. 10485, Swiss Federal Institute of Technology, Series in Microelectronics, Vol. 36, Hartung-Gorre Verlag Konstanz, 1994.

[70] ANALOG DEVICES, USA. *ADSP-21000 – Applications Handbook, Volume 1*, May 1994.

[71] Martin Rechsteiner, Markus Thaler, Gerhard Troester. Implementation Aspects of a Real Time Workspace Monitoring System. In *Fifth Int. Conf. on Image Processing and Its Applications*, vol. 410, (pp. 810–814), Heriot-Watt University, Edinburgh, UK, 4–6 July 1995. The Institution of Electrical Engineers, London, UK.

[72] Charles V. Stewart, Dyer Charles R. The Trinocular General Support Algorithm: A Three-Camera Stereo Algorithm for Overcoming Binocular Matching Errors. In *IEEE 2nd Int. Conf. on Computer Vision*, (pp. 134–138), Tampa, Florida, December 1988.

[73] Charles V. Stewart. Trinocular Stereo: Theoretical Advantages and a New Algorithm. In *Sensor Fusion II: Human and Machine Strategies*, vol. 1198, (pp. 377–391), Philadelphia, Pennsilvania, November 1989. SPIE.

[74] Jun Shen, Serge Castan, Jian Zhao. A new passive measurement method by trinocular stereo vision. In *Industrial Metrology 1*, (pp. 231–259). Elsevier Science Publishers B.V., 1990.

[75] Yingli Tian, Hungtat Tsui. Estimation Shape and Reflectance of Surfaces by Color Image Analysis. In *Image Analysis Applications and Computer Graphics* (Edited by Roland T. Chin, Horace H.S. Ip, Avi C. Naiman, Ting-Chuen Pong), vol. 1024 of *Lecture Notes in Computer Science*, (pp. 266–273), Hong Kong, December 1995. Dept. of Electronic Engineering, The Chinese University of Hong Kong, Springer.

[76] Masatoshi Okutomi, Osamu Yoshizaki, Goji Tomita. Color Stereo Matching and Its Application to 3-D Measurement of Optic Nerve Head. In *Proceedings 11th IAPR Int. Conf. on Pattern Recognition, A: Computer Vision and Applications*, vol. 1, (pp. 509–13), The Hague, Netherlands, September 1992. IEEE Comput. Soc. Press.

[77] Andreas Koschan. Dense Stereo Correspondence Using Polychromatic Block Matching. In *Computer Analysis of Images and Patterns*, (pp. 538–542), Budapest, September 1993. Springer Verlag.

[78] John R. Jordan III, Alan C. Bovik. Using Chromatic Information in Edge-Based Stereo Correspondence. In *CVGIP: Image Understanding*, vol. 54, (pp. 98–118). The University of Texas at Austin, USA, Academic Press, Inc., July 1991.

Curriculum Vitae

19.7.1964	Born in Basel
1971-1976	Primary school in Allschwil
1976-1980	Progymnasium in Allschwil
1980-1994	Gymnasium in Basel
1985	Matura Typus C (science, mathematics)
1984-1990	Swiss Federal Institute of Technology Zürich (ETH Zürich)
1990	Diploma in Electrical Engineering (Dipl. El.-Ing. ETH)
1990	working at S-TEC in Baar
1991-1996	Electronics Laboratory, ETH Zürich Research Assistant

All the work presented in this thesis has been done between 1993 and 1996 in the framework of research at the electronics laboratory at ETH Zurich.