Safe Human-Robot Interaction Using A Visual Interface

Jochen Heinzmann
Master of Computer Science (University of Karlsruhe, Germany)

THE AUSTRALIAN
NATIONAL UNIVERSITY

A thesis submitted for the degree of Doctor of Philosophy of
The Australian National University

Robotics Systems Lab
Department of Systems Engineering
Research School of Information Sciences and Engineering
The Australian National University

Submitted for examination in April 2001

Statement of Originality

These doctoral studies were conducted under the supervision of Prof. Alexander Zelinsky as advisor.

The work submitted in this thesis is the result of original research carried out by myself, except where duly acknowledged, while enrolled as a Ph.D. student in the Department of Systems Engineering, The Australian National University. It has not been submitted for any other degree or award in any other university or educational institution.

Jochen Heinzmann

Acknowledgements

Before I started this work more than $3\frac{1}{2}$ years ago it was my biggest wish to do this PhD course. After I had the pleasure to visit my supervisor Alex Zelinsky and his PhD students Gordon Cheng and David Jung earlier in 95/96 for 6 months to do my Masters in Wollongong, and I was very excited about the idea to come back to Australia to do a PhD. Meanwhile Alex had changed to the ANU and I was given the opportunity to start my PhD shortly after. I would like to thank our school director Brian Anderson, the Head of Department John Moore and particularly Alex for their support in creating this opportunity for me.

One my way many people provided valuable advice and turned me in the right direction. I should thank most of all Alex of course, but there were many others who helped me along the way. Thanks to John Kieffer, John Moore, David Wettergreen and Oussama Khatib who all provided advice at various stages of this thesis. Thanks also to all the people who helped me with my daily technical problems, James Ashton, Ali Goektogan, Tim Edwards, Luke Fletcher, Peter Brown, Chanop Silpa-Anan, Harley Truong, Chris Gaskett and of course my buddies Gordon Cheng and David Jung who provided so much help particularly in the difficult start of this work. I hope that I was able to return some of the support to them and my fellow students. I'd like to thank the Department of Systems Engineering for providing not only a well equipped, comfortable and stimulating environment but also all the support a PhD student can wish for. Thanks to all my fellow students Rochelle O'Hagan, Chris Gaskett, Simon Thompson, Roland Göcke, Gareth Loy and particularly to Jenni Watkins and Marita Rendina for making the place something more than just an office.

The time at the ANU seems to have flown past, mostly because it was so enjoyable also outside the lab. I'd like to thank my two fellow amigos Gordon Cheng and David Jung for being my friends and always making me feel welcome. Thanks also to Gordon's wife Katrina and their kids Jamie, Maddie, Andrew and Nick for their hospitality and letting me share many wonderful memories with them. Thanks also to all the other people I had the pleasure to meet and who made the past years so enjoyable, Yoshio Matsumoto, Andrew Brooks, Giovanni Bianco, Jacob Lidballe, Thomas Moor, Samer Abdallah, Frankie De Bruyne and everybody at Systems Engineering and particularly Alex for creating such a cooperative environment for everybody in the department.

I would like to thank my wife Nicole and my son Felix for their support and love during the past years and for creating a real family life despite the little time I could spent with them. Many thanks also to my parents Renate and Hermann and Nicole's parents Heidi and Hans for hosting us during the months when this thesis was written.

Abstract

The work presented in this thesis focuses on the research area of human-machine interaction, in particular the interaction between people and robots. If robots are to be introduced into the human world as assistants to aid a person in the completion of a manual task two key problems of todays robots must be solved: The human-robot interface must be intuitive to use and the safety of the user with respect to injuries inflicted by collisions with the robot must be guaranteed. Accordingly this research consists of two parts, the development of an intuitive visual human-machine interface and the formulation and implementation of a control strategy for robot manipulators which provides quantitative safety guarantees for the user.

The visual interface consists of a real-time visual eye gaze direction and facial gesture recognition system. These functionalities are based on a visual 3D head pose tracking system which requires only a monocular camera view and is completely passive. The system tracks natural facial features such as the eye brows and the mouth. Based on the image positions of the facial features the 3D head pose is determined. Robust feature tracking and 3D head pose estimation is achieved by employing probabilistic algorithms to integrate the correlation results of the vision hardware with the geometric constraints, to track the changes in the appearance of the facial features and to detect tracking failures.

The eye gaze tracking module uses the results of the 3D head pose estimation and derives the eye gaze direction from the 3D model of the head which includes the eye balls and the detection of the iris in the image. The gaze direction in combination with the 3D head pose allows the detection of the eye gaze point of the person. The facial gesture recognition detects characteristic motion patterns in the output of the face tracking system. It allows the robust detection of motion gestures such as nodding and shaking the head.

Safety guarantees for the user of a assistant robot are of paramount importance. The proposed control scheme for robot manipulators restricts the torque commands of a position control algorithm to values that comply to preset safety restrictions. These safety restrictions limit the potential impact force of the robot in the case of collision with a person. Such accidental collisions can occur with any part of the robot and therefore the impact force not only of the robot's hand but of all surfaces is controlled by the scheme.

The integration of the two subsystems visual interface and safely controlled robot allows the safe and intuitive interaction between a person and the robot. As an example application the system is programmed to retrieve gaze-selected objects from a table and to hand them over to the user on demand. All actions of the robot are controlled by gestures.

List of Publications

Journals

[1] J. Heinzmann and A. Zelinsky: A Safe-Control Paradigm For Human-Robot Interaction, Journal of Intelligent and Robotic Systems, volume 25, number 4, pp295-310, 1999

[2] A. Zelinsky and J. Heinzmann: A novel visual interface for human-robot communication, Journal on Advanced Robotics, volume 11, number 8, pp827-852, 1998

Conferences

[3] J. Heinzmann and A. Zelinsky: Visual Human-Robot Interaction, Proc. of the IEEE International Conference on Field and Service Robots FSR 2001, June 2001, to appear

[4] A. Zelinsky and J. Heinzmann: Human-Friendly Robot Design, Proc. of the International Symposium on Robotics Research ISRR'99, October 1999, pp247-256

[5] Y. Matsumoto, J. Heinzmann and A. Zelinsky: The Essential Components of Human-Friendly Robot Systems, Proc. of IEEE International Conference on Field and Service Robots FSR'99, August 1999, pp43-51

[6] J. Heinzmann, Y. Matsumoto, J. Kieffer and A. Zelinsky: Smart Interfaces + Safe Mechanisms = Human Friendly Robots, IARP Workshop on Humanoid and Human-Friendly Robotics, October 1998, Tsukuba, Japan, pp II-3

[7] J. Heinzmann and A. Zelinsky: 3-D Facial Pose and Gaze Point Estimation using a Robust Real-Time Tracking Paradigm, Proc. of IEEE International Conference on Automatic Face and Gesture Recognition FG'98, April 1998, pp142-147

[8] D. Jung, J. Heinzmann and A. Zelinsky: Range and Pose Estimation for Visual Servoing of a Mobile Robot, Proc. of IEEE International Conference on Robotics and Automation ICRA'98, May 1998, pp1226-1231

[9] J. Heinzmann and A. Zelinsky: A Visual Interface for Human-Robot Interaction, Proceedings of the International Conference on Field and Service Robotics, FSR'97, December 1997, pp572-579, represented as book chapter in: A. Zelinsky, editor, Field and Service Robotics, pp 540–547, Springer-Verlag, London, 1998

[10] J. Heinzmann and A. Zelinsky: Robust Real-Time Face Tracking and Gesture Recognition, Proc. of International Joint Conference on Artificial Intelligence IJCAI'97, August 1997, volume 2, pp1525-1530

Video Publications

[11] J. Heinzmann and A. Zelinsky: A Control Strategy for Human-Friendly Robots, Video Proc. of International Conference on Robotics and Automation ICRA'99

[12] J. Heinzmann and A. Zelinsky: Real-Time Face Tracking and Pose Estimation, Video Proc. of International Conference on Robotics and Automation ICRA'97

Naming Conventions

Throughout the thesis, I use a uniform naming convention for mathematical symbols. Small Roman letters (for example. a, b, c) denote scalar values, bold small letters (for example. $\mathbf{a}, \mathbf{b}, \mathbf{c}$) are used for vectors, and capital letters (for example. A, B, C) are used for Matrices. Greek letters are used for variable notations, which may not be consistent with other notations in other areas. In general, small Greek letters (for example θ, π) are used for scalars, capital Greek letters (for example. Θ, Π) for vectors and capital roman letters for matrices. In general, the dimension of every variable is denoted when it is introduced (for example. $\Theta \in \mathbb{R}^3$).

Contents

1 Introduction **1**

1.1 The interfaces of the future 1

1.1.1 Natural interaction 1

1.1.2 Interaction with machines 2

1.1.3 Communication channels 3

1.1.4 Physical interaction 3

1.1.5 Traditional approaches 4

1.1.6 Humanoid robots 5

1.1.7 Safety issues 5

1.2 Research objectives 6

1.3 Contributions 7

1.4 Application areas 8

1.4.1 Head pose and 3D eye gaze tracking 9

1.4.2 Gaze and gestures for robot control 10

1.5 Organisation of this thesis 11

2 Related Work **15**

2.1 Road map 15

2.2 Visual human-computer interfaces 17

2.2.1 Face detection 20

2.2.2 Facial feature detection and tracking 25

2.2.3 3D head pose estimation 29

2.2.4 Eye gaze estimation 34

2.2.5 Facial gestures 38

2.2.6 Other face-related issues 39

2.2.7 Non-face related issues in human-centred visual interfaces . . 41

2.2.8 Summary 44

2.3 Human-friendly robots 44

2.3.1 Safety issues 45

2.3.2 Physical interaction 49

2.4 Visual human-robot interfaces 50

2.4.1 Summary 51

2.5 Summary 52

3 Face Tracking **55**
3.1 System Architecture 56
3.2 Layer 1: Template correlation hardware 57
3.3 Layer 2: Kalman filter network 59
3.3.1 The basic idea 60
3.3.2 The Kalman filter in a nutshell 60
3.3.3 Kalman filter network 61
3.3.4 Improved Robustness 64
3.3.5 Network of template 65
3.3.6 Area search windows 69
3.3.7 Search window allocation 74
3.3.8 Strain integration 75
3.4 Experimental results 78
3.5 Summary 80

4 Head Pose Estimation **85**
4.1 Huttenlocher's alignment algorithm 86
4.1.1 Analysis of the accuracy and sensitivity 88
4.2 Systematic Errors 89
4.3 Tracking Error Sensitivity 97
4.3.1 Sensitivity Model 101
4.3.2 Sensitivity and Spatial Orientation Error 105
4.3.3 Effect on the face tracking system 107
4.4 Multi-triangle pose estimation 110
4.5 Experimental evaluation 115
4.5.1 Experimental setup 115
4.5.2 Features 116
4.5.3 Mannequin tracking results 118
4.5.4 Real-head tracking experiments 127
4.6 Summary 130

5 Gaze Tracking and Gesture Recognition **133**
5.1 Underlying eye model 133
5.1.1 General architecture 134
5.1.2 Eye gaze direction calculation 135
5.1.3 Merging results from two eyes 138
5.1.4 Summary 139
5.2 Experimental results 139
5.3 Gesture recognition 144
5.3.1 Gesture decomposition 145

5.3.2 Gesture definition and recognition 146
5.3.3 Gesture vocabulary 148
5.4 Summary 149

6 Human-Robot Coexistence **155**
6.1 The vision 155
6.2 A safety philosophy for human-friendly robots 156
6.2.1 Safety requirement 156
6.2.2 The perception problem 157
6.2.3 Assignment of responsibility 157
6.2.4 Defining human-friendly robots 159
6.2.5 A basic safe behaviour 160
6.3 Summary 162

7 Safe Robot Control **163**
7.1 Safe robot control 164
7.2 Impact Potential 165
7.2.1 Impulsive impact force 166
7.2.2 Poles of the impulsive impact force 168
7.2.3 Collisions while considering friction 171
7.2.4 Stationary points 172
7.2.5 Poles of the generalised impact force 172
7.2.6 Kinematics of human-friendly robots 179
7.2.7 Definition of Impact Potential 181
7.2.8 Controlling the impact force 182
7.3 Implementing Impact Potential Control 183
7.3.1 Transformation to torque space 183
7.3.2 Torque verification 184
7.3.3 Torque clipping 184
7.3.4 Stationary robot 187
7.3.5 Unsafe robot state 188
7.4 Simulation of a 2 DOF robot 188
7.4.1 Impact control point selection 189
7.4.2 Static simulation 190
7.4.3 Dynamic simulation 190
7.5 Summary 195

8 Impact Potential Control Implementation **199**
8.1 Hardware platform 200
8.1.1 The WAM robot 201
8.1.2 The WAM hand 202
8.1.3 Hardware safety features 203

8.1.4 Suitability for human-robot interaction 205
8.2 Control point selection 205
8.2.1 Poles of $\widehat{f}_e$ on the WAM 206
8.2.2 Geometry of the WAM robot 208
8.2.3 Implementation 215
8.3 Contact sensing without sensors 216
8.3.1 Methodology 217
8.3.2 Implementation 217
8.4 External position control 218
8.4.1 Example: PID control 218
8.5 Software architecture 220
8.5.1 Visual interface and application controller 223
8.5.2 Safety core implementation 223
8.5.3 Robot motion server interface 223
8.5.4 IPC server 224
8.5.5 Network communication 225
8.6 Passive motion experiments 225
8.6.1 Zero-G mode interaction 225
8.6.2 Safety envelope motion damping 227
8.7 Motion control experiment 228
8.8 Impact Experiment 236
8.9 Summary 242

9 System Integration **247**
9.1 System functions 248
9.1.1 Basic concept 248
9.1.2 Actions of the interactive pick-and-place system 249
9.2 Experimental setup 250
9.3 System structure 252
9.4 Finite state machine implementation 253
9.5 Experimental results 256
9.5.1 Place sequence 261
9.5.2 Pick sequence 265
9.5.3 Error correction sequence 265
9.6 Summary 267

10 Conclusions and Further Work **273**
10.1 Conclusions 273
10.2 Directions of future work 274
10.2.1 Face and gaze tracking 275
10.2.2 Safe robot control 276

10.2.3 Human-robot interaction 276
10.3 Closing words 277

A Asimov's Laws of Robotics **279**

B Commercial Vision Systems **281**

C Huttenlocher's Alignment algorithm **283**
C.1 The alignment algorithm in a nutshell 283
C.2 Closed form solution 284

D Gravity Compensation **287**
D.1 Gravity compensation on a serial manipulator 287
D.2 Parameter identification 290
D.3 4DOF configuration 291
D.4 Least squares estimation 291

E Video Index **295**
E.1 Visual interface 295
E.1.1 3D Head pose tracking 295
E.1.2 Gesture recognition 296
E.1.3 Eye gaze point estimation 296
E.2 Safe robot control 297
E.2.1 Basic behaviour: Zero-G 297
E.2.2 Safety core motion damping 297
E.2.3 Collision experiments 297
E.2.4 Human intervention 298
E.3 System integration: Interactive pick-and-place 298

F Vectorspace Algebra **299**
F.1 Algebraic operations with n-dimensional subspaces 299
F.1.1 Intersection of affine subspaces 300
F.1.2 Orthogonal projection into affine subspaces 300

G States of the Finite State Machine **303**

List of Figures

2.1 Categorisation of human-centred visual computer interfaces 16
2.2 Categorisation of safety measures in robotics 19

3.1 Structure of the tracking system 57
3.2 The computational cycle of the classical Kalman filter [Bozic79] . . 61
3.3 Part of the Kalman filter network 64
3.4 Change in appearance of the mouth area 66
3.5 Correlation kernel network 67
3.6 Example of the selection of templates 68
3.7 Local minima situation 69
3.8 Probability density functions for the distribution of area search windows 71
3.9 Distribution of area search windows 73
3.10 Heuristic allocation of correlation windows 75
3.11 The modules of the strain integration on links between feature trackers 76
3.12 Progressive facial feature recovery image sequence 79
3.13 Feature tracking and recovery after partial and total occlusions . . . 81

4.1 Geometry and dimensions of the setup for the case study 89
4.2 Systematic angular error (a) for Huttenlocher's original algorithm and (b) for $s = \max(s_1, s_2)$. 92
4.3 Systematic error for (a) $s = s_1$ and (b) for $s = s_2$ 93
4.4 Depth error for $s = s_1$ b) for $s = \max(s_1, s_2)$ (note the different scale). 94
4.5 Plot a) shows the systematic error of the original algorithm at a distance of 2000mm over θ_x and θ_y. Plot b) shows the error of the extended algorithm. 96
4.6 Feature velocity 97
4.7 Rotational sensitivity s_r 99
4.8 Rotational sensitivity s_r for a smaller triangle (a) and for the original triangle at a distance of 2000mm 100
4.9 The sensitivity s_t of the depth estimate for the normal model triangle and a 45° rotated triangle 102
4.10 True sensitivity (a) and the approximation (b) 104

4.11 True sensitivity (a) and the error ratio (b) 106
4.12 Effective pose estimate errors (a) for $(\theta_x, \theta_y) = (0.0, 0.0)$ and (b) for $(\theta_x, \theta_y) = (0.0, 0.3)$ 108
4.13 Effective pose estimate errors (a) for $(\theta_x, \theta_y) = (-0.8, 0.6)$ and (b) for $(\theta_x, \theta_y) = (1.2, -1.4)$ 109
4.14 Multiple solutions 110
4.15 Sensitivity of the rotation around the x-axis (a) and the z-axes(b) . . 113
4.16 Overall pose estimate error 114
4.17 Overall sensitivity 114
4.18 MARITA-system (a) and view from the vision system (b) 117
4.19 Bias (a) and standard deviation (b) of Rot_x 119
4.20 Bias (a) and standard deviation (b) of Rot_y 120
4.21 Bias (a) and standard deviation (b) of Rot_z 121
4.22 Bias a) and standard deviation (b) of the translation in x-direction . 123
4.23 Bias a) and standard deviation (b) of the translation in y-direction . 124
4.24 Bias a) and standard deviation (b) of the translation in z-direction . 125
4.25 Rot_x, Rot_y and Rot_z over 6000 frames 126
4.26 Tracking image sequence with pose triplets overlay (part 1) 128
4.27 Tracking image sequence with pose triplets overlay (part 2) 129

5.1 Schematic of the gaze vector calculation 134
5.2 The underlying model of the eye gaze direction calculation 135
5.3 Iris location correction 136
5.4 Systematic error introduced by the approximation when merging the two gaze vectors. 138
5.5 Experimental setup for the measurements of the gaze estimators accuracy 140
5.6 Trace plot of the eye gaze vector intersected with the target plane . 141
5.7 Tracking sequence 142
5.8 Eye gaze trace plot for the pan and tilt of the gaze direction, measured in mm on the left side and in degrees on the right side. 143
5.9 Trapezoid functions which define the *down* action. 146
5.10 Typical activation pattern 147
5.11 Architecture of the gesture recognition system 147
5.12 Tracking Information passed to the gesture recognition module . . . 150
5.13 Gesture recognition image sequence 151
5.14 Measurements during gesture recognition 152

6.1 Safety core and safety envelope 162

7.1 Components of the safety envelope 165
7.2 1 DOF example 169

7.3 1 DOF manipulator configuration 174
7.4 Impact force distribution on the 1 DOF manipulator 174
7.5 2 DOF example manipulator 175
7.6 Impact force $\widehat{f}_e$ on a 2 DOF manipulator arm 176
7.7 Poles at joint 2 177
7.8 Impact force $\widehat{f}_e$ on a 2 DOF manipulator arm 178
7.9 3DOF manipulator configuration 179
7.10 Singularities in $\widehat{f}_e$ for a 3 DOF manipulator 180
7.11 Example of a base design for a human-friendly robot 181
7.12 Two link robot with two point masses at the end of the links 189
7.13 Test configurations of the 2 DOF robot 191
7.14 Safe region in the torque space 192
7.15 Simulation of a 2DOF robot with IPC core 193
7.16 Simulation of the robot initially in an unsafe state 194
7.17 Safe link design 196

8.1 The WAM robot (Figures courtesy of Barrett Technology) 201
8.2 3 fingered Hand of the WAM (courtesy of Barrett Technology) . . . 202
8.3 Potential poles on the first two links 207
8.4 Potential poles on link 3 207
8.5 Potential poles on link 4 209
8.6 Experimental setup of the WAM robot 209
8.7 Configuration example of the WAM robot with a pole of $\widehat{f}_e$ on link 4 211
8.8 The pole in $\widehat{f}_e$ on link 4 212
8.9 Test configuration of the WAM robot 213
8.10 Impact force $\widehat{f}_e$ on link 3 and 4 in a general configuration 214
8.11 Control points on the WAM robot 215
8.12 Example configuration of the PID controller 221
8.13 Software architecture of the WAM control scheme. 222
8.14 Zero-G control structure 225
8.15 Image sequence of a Zero-G interaction experiment 226
8.16 Control structure for the motion damping experiment 227
8.17 Image sequence of the experiment in Zero-G mode 229
8.18 Image sequence of the experiment with safety envelope 230
8.19 Motion sequence 1 231
8.20 Safe robot motion using PID control and a 1N safety envelope . . . 232
8.21 Safe robot motion using PID control and a 3N safety envelope . . . 233
8.22 Safe robot motion using PID control and a 5N safety envelope . . . 234
8.23 Motion sequence 2 237
8.24 Safe robot motion using PID control and a 1N safety envelope . . . 238
8.25 Safe robot motion using PID control and a 3N safety envelope . . . 239

8.26 Safe robot motion using PID control and a 5N safety envelope . . . 240
8.27 Setup of the impact experiment . . . 241
8.28 Image sequence of the impact experiment . . . 241
8.29 Plasticine impactors after the impact with different limits, a) 1N, b) 2N and c) 3N. . . . 243

9.1 Transfer configuration . . . 249
9.2 Experimental scene . . . 251
9.3 Top view of the experimental setup . . . 251
9.4 Information flow . . . 253
9.5 Simplified representation of the FSM . . . 254
9.6 sub-states of the *idle* state . . . 255
9.7 Sub-states of the *pick object* state . . . 257
9.8 Safe reconfiguration pose . . . 258
9.9 Action sequence of the experiment . . . 258
9.10 Interaction experiment (8:32 minutes) . . . 260
9.11 Angles between current and target gaze angles . . . 261
9.12 Put down object sequence (part 1) . . . 262
9.13 Put down object sequence (part 2) . . . 263
9.14 *place object* sequence . . . 264
9.15 Pick object sequence (part 1) . . . 266
9.16 Pick object sequence (part 2) . . . 267
9.17 Pick object sequence . . . 268
9.18 Error correction sequence . . . 269
9.19 Error correction sequence . . . 270

D.1 Kinematic configuration and dimensions of the robot, the position of the coordinate systems and the location of the centre of mass of the links. . . . 288
D.2 Unknown estimation for the 4DOF configuration. . . . 293

G.1 Substates of the idle (object grasped) state . . . 304
G.2 Substates of the *place* state . . . 305
G.3 Substate of the *receive* state . . . 306
G.4 Substates of the *hand over* state . . . 307

List of Tables

2.1 Overview of the algorithms and their applications 18

3.1 Evaluation states 68

4.1 Parameter estimation analysis 122

5.1 Error analysis of the eye gaze tracking 141

E.1 Video index 296

List of Algorithms

3.1 Correlation kernel selection: `select (State, Size)` 67
3.2 Strain integration: `integrate`(d_1, d_2) 77
4.1 Extended pose estimation : `pose`$(\mathbf{q}_2, \mathbf{q}_3)$ 91
4.2 Multi-plane pose estimation : `pose_n-plane(T)` 115
7.1 Torque vector clipping function: `clip` $(\tau_e$, `Constraints)` 185
8.1 Command Position Generation 219

Chapter 1

Introduction

Since the beginning of electronic computers in the early 1950s, the interaction between computers and their users has grown considerably. The first digital computers were programmed by punchcards. Users had to wait for hours to get the output of their program which would allow them to modify and debug their program. The invention of the *von Neumann* machine architecture combined with the progress in semi-conductor circuits improved the program execution process so programs could be immediately run by the user and their results could be examined immediately. Today, computer users have a set of applications accessible through graphical user interfaces that they can use to provide particular services such as word processing or spreadsheet calculations.

1.1 The interfaces of the future

Generally, ease of the interaction with computers has always been associated with a reduction of the abstraction required by the user to obtain desired information or results. In essence, people wish to describe tasks to the computer using an efficient interaction mode which is intuitive and requires the minimum of learning. These ideas have led to today's graphical user interfaces (GUIs) in which the technical details are hidden from the user by symbols and procedures whose meanings are intuitively understandable, or at least easy to learn. The goal of human-machine interfaces are computer systems that allow the user to interact in completely intuitive ways.

1.1.1 Natural interaction

The most natural and intuitive form of interaction for people is the interaction with other people. It appears sensible to predict that future generations of computers and computerised machinery will allow the user to interact with the computer not only with keyboard and mouse but in similar ways as people interact with one and

another. Users could simply tell a computer what they want it to write, to display, to change in a document, which information to retrieve or what commands to send.

This vision of natural interaction with computers can be found in various flavours in many science fiction environments, for example in Arthur C. Clarke's *"2001: A Space Odyssey"* and in Gene Roddenberry's *"Star Trek"*, both published in 1964. In particular the robots of the future, as they are the most articulated and life-like of all computerised machines, are imagined by many people to be able to interact with a people in natural ways, for example in Isaac Asimov's *"I, Robot"* and countless others. Although this is only fiction, it indicates that people believe that a natural, intuitive interaction is the optimal form of communication with machines.

1.1.2 Interaction with machines

A development that started with the advent of cheap microcomputers is the evolution of devices and machines controlled by computers. This includes not only industrial manufacturing machinery, but also various kinds of household appliances such as dishwashers, washing machines, refrigerators and microwaves. Also motor vehicles have become increasingly computerised. Computer assisted car navigation is now a standard component in motor vehicles, and automobile manufacturers are developing advanced computer systems for visual lane tracking [Kosecka et al. 1998] [Lee et al. 1998], collision warning and visual systems to automatically read speed limit signs on the side of the road [Dickmanns 1999], to detect driver fatigue [Heinzmann and Zelinsky 1997]. Motor vehicles are consumer products with huge market volumes which allow the development of technology products using cutting edge research results.

Also, as computer systems become cheaper over time it will become economically viable to include more sophisticated computer systems in various consumer products such as interactive TV sets that also provide access to the Internet, multi-player games and other interactive datacast contents. As more consumer products become computerised with increasing levels of functionality, the technology will become more complicated to use. The question of how people should interact with such systems will arise. The first off-the-shelf solutions to network private homes and to control virtually any computerised appliance are pushing into the market. A speaker of SUN Microsystems which actively pursues this market has said "One of the challenges in the networked home is to keep things simple for the consumer"[1]. Products of this kind will be technically and economically feasible in the near future but the appropriate human-machine interface technology is yet to be developed.

[1] `http://www.java.sun.com/features/2000/01/homegateway.html`

1.1.3 Communication channels

The development of natural interaction systems for computers and computer controlled machinery depends on improvements in two key areas, the extension of the available communication channels and the inclusion of implicit information in the interaction. People use bilateral known contexts in their conversations to make the exchange of the new information more efficient. For example in a conversation about a particular car a person may use the term 'the radiator' and the conversation partner will assume that this refers to the radiator of this particular car. However, today's computers are nescient to anything other than the most basic context information.

Presently, the main communication channels between a user and a computer are the mouse and the keyboard. In some applications touch-screens are being used to replace the mouse. Speech recognition for word processing applications is now available[2], but the market share of speech recognition systems is still small.

The dominant communication channel between the computer and the user is the computer screen. It presents all information for the user in form of text, images, graphs or video. Limited information is provided by sound in today's computer systems. Force feedback systems in input devices have been used in telerobotics for a considerable time and recently have become available as haptic interaction devices for virtual worlds[3].

To allow natural interaction computers must be able to replicate the same communication channels that people use between one and another. For many applications natural speech recognition is the major communication channel between the user and the computer. Besides speech, gestures are an important communication channel for humans. For human-machine interaction *deictic* gestures (pointing) are of the highest interest. Deictic gestures refer to an entity in a given context. People use two different modes for pointing, arm/hand/finger and eye gaze. Arm/hand/finger pointing is explicit and occurs quite deliberately while a person's gaze fixates on an area or object of current interest. Depending on the scale of the environment one or the other mode can be preferable. Gaze fixation can be favourable in close-up/desktop situations, however, for larger scale environments, such as controlling a crane on a construction site, arm/hand/finger pointing is the preferred communication channel.

1.1.4 Physical interaction

Another mode of interaction which is particularly important for *human-robot* interaction is physical interaction. The user of a robot should be able to guide or stop a robot manipulator by simply pulling or pushing the robot. This ability can be important in teaching a robot what to do, or what not to do. There are many other

[2]For example ViaVoice `http://www.software.ibm.com/speech/index.html` and Naturally Speaking `http://www.dragonsystems.com`.

[3]See for example Sensable Technologies `http://www.sensable.com`.

situations where people use physical interaction when completing a task, and people would expect a robot to be able to understand the same sorts of physical interaction patterns.

Among those devices that may become feasible in the mid-term future are robots that work *together* with humans. Such robots could assist a person to perform a particular task or complete subtasks autonomously [Khatib 1999].

1.1.5 Traditional approaches

Robotics research in the past 30 years has been polarised into two basic philosophies in terms of the autonomy of robots. Although both directions have been driven by the idea to extend the application areas of robots beyond repetitive tasks, their approaches to reach this goal have been quite different. While most academic researchers have focused on the development of fully autonomous robots that could perform variable tasks without human intervention, the industry oriented researchers devised telerobotic systems which require an operator to initiate each and every action of the robot from a remote control post.

Teleoperated robots are used for hazardous tasks in nuclear facilities [Perret 1998], servicing of live power line [Penin et al. 1998], fire fighting [Schraft et al. 1999], bomb disposal and in mines after accidents and in inaccessible places such as in pipe inspection applications[4], deep sea[5] and Mars exploration[6].

Despite the initial high hopes, autonomous systems have never really left the research labs and have not made their way into real world applications. Deriving sensible actions from the available sensor information in low-structured environments has proved to be a much harder task than was anticipated a few decades ago. In the area of mobile robots limited success has been achieved. Today autonomous mobile robots are used for meal distribution on hospitals [Engelberger 1989], for floor cleaning tasks in airports and train stations, for in-house mail delivery in offices [Schweitzer et al. 1998] and luggage transportation in hotels [Graf and Weckesser 1998]. The autonomy of robot manipulators dispatched into real-world applications appears to be limited to making small adjustments in the location of welding seams in arc welding applications [Barborak et al. 1999] [Sicard and Levine 1989].

This situation has lead to the idea of teams consisting of a humans *and* robots working cooperatively on the same task. In this scenario, a person provides the sensory and the reasoning capabilities while the robot assistant provides precision, strength and endurance. Various names for this kind of human-robot cooperation systems have emerged including *human-friendly robots*, *personal robots*, *assistant robots* and *symbiotic robots.*

[4]For example see `http://www.roboprobe.com/`

[5]For example see `http://www.brooke-ocean.com/` and `http://www.makai.com/`

[6]For example see NASA/JPL website at `http://mpfwww.jpl.nasa.gov/MPF/index.html`

1.1.6 Humanoid robots

Closely related to the idea of human-friendly robots is the development of humanoid robots. Although the idea of humanoid robots is not new, the interest in this topic sharply increased in 1997 after the motor vehicle manufacturer Honda presented the results of 10 years of confidential R&D, the humanoid robot *P3*. The P3 humanoid walks on uneven ground, climbs stairs, pushes a trolley and steps back when it is being pushed [Hirai et al. 1998]. The predominant motivation of giving a robot an anthropomorphic shape is that the human form is the most efficient shape for operation in environments designed for humans, and therefore the robot must have the ability to interact with people. Honda's commitment to develop the P3 and its successors shows that Honda anticipates that humanoid and human-friendly robots could be technically feasible and that they have a strong market potential. What is desirable for household appliances is absolutely crucial for human-friendly robots: An interface that allows people to interact with this technology in a natural and intuitive way.

1.1.7 Safety issues

The transition from the control of physically passive computers and appliances to dynamic interaction with a robot brings up another issue. As mentioned previously, human-friendly robots will require the ability to physically interact with persons for a variety of tasks. The concept of humans and robots sharing the same workspace is orthogonal to the current workplace philosophy. Presently we protect humans from robots by isolating robots in work cells that automatically shut down if a person enters. The concept of humans and robots sharing workspaces requires a radical redefinition of the control strategies that are used to guide today's robots.

The controllers of industrial robots are designed to provide high stiffness and high path tracking accuracy while moving at high speeds[7]. Even though such characteristics would also be desirable for a human-friendly robot, other characteristics are much more important. The highest priority in the control of any kind of human-friendly robot is to prevent any of the robot's actions causing any harm to people. All executable tasks of a robot must be subject to this restriction. This logic was captured by Isaac Asimov in the first law of his famous *3 Laws of Robotics* in 1960 (see Appendix A).

Stiffness and tracking accuracy at high speeds are characteristics that are contradictory to the goal of not harming a person under any circumstances. These traditional goals of robot control systems must be abandoned as the primary goals and a new philosophy for the control of such systems is required which is compatible with the safety requirements for human-friendly robots. The physical danger posed

[7]Examples for such systems can be found on all websites of robot manufactures, for example `http://www.abb.com` or `http://www.kuka.com`.

by autonomous motion of the robot must be addressed in such a philosophy. Solutions that expand on the idea of physically making it impossible for the robot to harm a person by building robots that are light weight, weak and flexible can render the robot useless for most tasks. Therefore, the second goal of such a philosophy must be to realise robotic systems that still have the ability to perform real tasks. The implementation of a control system for a human-friendly robot must achieve both goals otherwise the robot would be either dangerous or useless. Only then people will be comfortable to work with such systems, and manufacturers might be willing to produce them[8].

Natural human-computer interfaces and a control philosophy that can guarantee the safety of humans are the technical prerequisites for the development of human-friendly robots. This thesis is directed at making a contribution in both these areas.

1.2 Research objectives

The goal of this thesis is to explore new ways of human-robot interaction which could be desirable and useful. The ultimate goal of human-machine interfaces is to allow a person to interact with machines in the most intuitive, natural and efficient way possible. This requires robots to use human communication channels to allow a person to use the robot with a minimum of training.

An inevitable side effect of this development is that machines become more anthropomorphic. Even without a humanoid shape a machine could appear to be anthropomorphic simply by being able to communicate with a person in a human-like way. However, the aim of this work is not to create human-like machines. As a matter of fact I do not see any advantages in undertaking such an endeavour nor do I believe that society would like human-like machines to come into existence in the present or in the medium future. The ultimate goal are highly sophisticated machines which assist people, that are easy to use and which allow people to expand their own capabilities or to become more independent.

In this relatively new research area which has more unsolved problems than solved ones my work focuses on two aspects: Computer vision systems that allow machines to detect the eye gaze and facial gestures of the operator and robot manipulator control schemes for safe human-robot interaction. These two building blocks are integrated to create a robot manipulator system that allows a person to pick up or put down objects by simply looking at the object or the place where it should be put down. The actions of the robot are invoked by facial gestures such as nodding and shaking the head. The vision system must therefore be able to determine the eye

[8]In retrospective it appears that the great work that has been done on systems for autonomous highway driving has never really stood a chance to get implemented by car manufacturers because of the unresolvable legal liability issues connected to such systems, for example who would be responsible in an accident situation?

gaze direction in 3D and to recognise facial gestures of the user in real-time. When prompted the robot approaches the operator to receive or deliver an object. These hand-over actions facilitate the same mechanism used by humans and require the robot to "feel" the grip of the operator. Since there can not be any barriers between the robot and an operator for such interactions, the robot must be controlled in a way that makes safety guarantees possible.

To verify the applicability of the approach the system was implemented on the WAM[9] robot. Although the overall system is operational and can be used by anybody to move objects in the robots environment, the aim has not been to develop a system with performance characteristics applicable to any particular real-world problem. The system rather is a proof of concept for human-robot interaction systems. In almost any imaginable real application the inclusion of natural language recognition is necessary to create a truly intuitive interaction system.

1.3 Contributions

The main contributions of this research work are in the area of real-time 3D head pose tracking with a monocular camera and in devising a control philosophy and algorithm for a robot manipulator that satisfies safety criteria for human-robot interaction. The contributions are summarised below:

- **Human-machine interfaces:**
  - A vision-based system to *simultaneously* measure 3D head pose and 3D eye gaze direction from a monocular camera view.
  - A non-intrusive method to head and eye gaze tracking which does not require head gear, markers, restricted head motion or infrared illumination. The system allows full natural head motion.
  - A real-time 3D head pose estimation system based on feature locations and probabilistic inclusion of tracking reliability of the individual features.
  - An analysis of the practical accuracy and robustness of the well known alignment algorithm proposed by [Huttenlocher and Ullman 1990].
  - An extension of the alignment algorithm with the following improvements:
    * Consistent results for noisy feature location measurements
    * Reduced errors for the resulting pose estimate in noisy measurements
    * Improved robustness of the overall head pose
  - Measures to improve the robustness and the efficiency of facial feature tracking and recovery from tracking failures.

[9]WAM stands for *W*hole *A*rm *M*anipulator built by Barrett Technology.

- An experimental validation of all proposed vision algorithms.

- **Human-friendly robot control:**
  - A formulation of a novel safety concept that allows safe operation of a robot manipulator with a person in the robot's work space.
  - A control strategy for manipulators which limits the potential impact force of the manipulator with static obstacles.
  - An implementation of the proposed impact force limitation scheme on a real robot manipulator.
  - An experimental validation of the proposed control method.

- **Human-robot interaction:**
  - The integration of the described components into a working system.
  - A human-friendly robot system that allows human-robot interaction both visually and through physical contact.
  - A system that allows a non-expert to interact with a robot manipulator in a natural and intuitive way while the system guarantees safety constraints.
  - An experimental evaluation of the system and proof of feasibility of safe human-robot interaction.

1.4 Application areas

The research results of this thesis have a variety of applications in many areas previously mentioned, although the time horizons are quite different. Non-intrusive head and eye tracking systems are reaching a maturity level now that makes this technology applicable to real-world problems in an efficient and cost-effective way. Companies specialising in this technology are now appearing and are offering a variety of systems. One of them, *Seeing Machines,* is selling a system closely related to the work presented in this thesis (see Appendix B). Considering that this research area attracted serious attention only in the mid 1990's the time to market has been relatively short.

The development in the area of human-friendly robots is considerably slower. and the research in this area has a longer history than the gaze tracking area ([Karwowski et al. 1987], [Etherton and Sneckenberger 1988], [Milgram et al. 1993]). As a general comment systems developed in various labs around the world appear to be far from deployment to any real-world applications (including the system presented in this research) [Johannsen 1995], [Khatib 1999], [Luh and Hu 1999]. These systems should be regarded as proofs of concept that particular aspects of such systems can be realized. Generally, the systems lack robustness, safety, speed and the ability to

react to rather unstructured environments and to understand context information to allow for efficient deployment. The medium and long term prospects of this technology are subject of debates in industry and academia. The topic has attracted considerable attention in the recent years [Brooks et al. 1998], [Cheng and Kuniyoshi 2000]. Although there is no guarantee that this technology can reach maturity in the coming decades the effort to probe for possibilities to create such systems is definitely worthwhile.

1.4.1 Head pose and 3D eye gaze tracking

There is an innumerable number of applications for visual human-machine interface systems ranging from desktop computer and interactive multimedia devices, to safety applications in the motor vehicle industry and human measurement systems for the animation and advertising industries. Other applications include more convenient and richer interfaces for support systems for disabled persons and interfaces to intelligent home appliances. For most applications it is crucial that the person is able to interact naturally with the system, in particular that the head motion is not restricted artificially and no visual markers need to be worn by the user.

Safety applications

Operator inattention warning systems have potential applications in the transportation sector, for example in heavy trucks, trains and aeroplanes. Also control post situations, such as power plants, chemical production plants, nuclear facilities and air traffic control rooms are potential application areas. Here, the detection of fatigue and inattention of the operator increases the safety level of the overall system since human error has shown to be a significant source of accidents. Such safety systems could monitor the gaze of the person to verify that the person notices warning signals, detect operator inattention to the main process (for example the road in heavy trucks), detect fatigue indicators in the behaviour of an operator and recognise if the operator has actually fallen asleep.

Human performance measurement

A persons gaze is of major interest in many psychological experiments. This includes basic academic research concerned with reaction times to visual and acoustic stimuli and the way people look at objects, pictures and video sequences. Human performance measurements are also conducted for the ergonomic designs of cockpits for cars, trucks and aeroplanes. A related application area are training systems. For example pilots undergo regular training sessions in flight simulators where a variety of critical situations are simulated. Gaze tracking systems could be used here to verify the reactions of pilots.

Advertisement validation

Successful marketing is an important success factor for many mass market products. However, little is known about how consumers actually look at advertisements. Gaze tracking systems could provide valuable information about where exactly consumers look while viewing an advertisement. This information could be used in the production stage of an advertisement to improve its effectiveness.

Interfaces for the disabled

Visual interfaces could allow people with disabilities, in particular paralysed people, to use computerised machines such as telephones, TV set, desktop computers or steer a wheel chair [Bergasa et al. 1999].

General desktop computer interface

With the advent of cheap web cams for desktop PCs, a gaze tracking system could be used as new input device. The obvious application for a gaze tracking system would be as a replacement or supplement to the mouse. Also this system could be used as an interface to virtual reality applications such as computer games and low bandwidth teleconferencing systems [Saulnier et al. 1995] [Sengupta et al. 2000] [Feng et al. 2000].

1.4.2 Gaze and gestures for robot control

Human-robot interfaces are another area of application for visual interfaces. Although human-friendly robots are still far from being applicable to any real-world problem, the development of natural interfaces as well as safety precautions in their control are prerequisites to such systems. Human-friendly robots could be used in both home and office environments to aid the humans in performing simple daily tasks. Another application area that has attracted some attention is the construction industry. Various construction machines could be computerised allowing a skilled worker to coordinate the actions of a number of machines in tasks ranging from transport of material and control of cranes [Müller et al. 2000] and excavation machines [Cohen et al. 1996] to cooperatively carry and deploy construction material [Khatib 1999] [Kim and Zhang 1998] [Luh and Hu 1999] [Hayashibara et al. 1999].

Helping hands for the disabled

One application that may be realisable relatively soon is in the area of support systems for disabled and elderly people. This is because even though such systems may appear to be too cumbersome and too slow for efficient use by an able person, such machines can make a huge difference to a disabled person. Helping hands could be used by disabled people in a similar way to the experiment conducted

with the system proposed in this thesis. The robot assistant could be used to move various objects in the environment. The robot could also be used to utilise the infrastructure designed mainly with able persons in mind such as lift buttons and traffic light buttons. Robotic arms could help elderly people to get in and out of their beds and bath tubs. Such simple capabilities would allow them to live in their homes by themselves longer and more independently.

Home and office robots

Robots helping out in homes and offices are a long way away from commercialisation. Although autonomous vacuum cleaners, lawn mowers, floor cleaning machines, mail delivery robots are reaching maturity now, robots with manipulators are considerably more difficult to control, mainly due to the much more sophisticated tasks. However, if such systems are to be build in the future they will require both natural human-robot interfaces and safety measures which guarantee safe operation at all times.

Humanoid robots

Humanoid robots are probably the furthest away from commercial use. As mentioned previously, the main purpose of humanoid robots is to be used in environments designed for humans and thus they have to be able to work in the vicinity of humans. Human-friendly robots could be used as assistants in construction, production and office environments.

1.5 Organisation of this thesis

This first chapter sets the scene for motivating the research in the areas of *face and gaze tracking* and *safe control of robot manipulators* and explains their relation in the context of human-friendly robots. The remaining chapters of this thesis are organised in the following way.

Chapter 2: Previous work

Chapter 2 reviews the research areas of visual human-machine interfaces and safety measures in human-robot coexistence. This chapter also gives an account of the different ways in which human-robot interaction systems and visual human-robot interfaces can operate.

The overview of human-machine interfaces categorises the systems reported in the literature according to their functional and algorithmic blocks rather than their overall functionality. The different functions of face-related visual interfaces such as head pose tracking, eye gaze tracking and facial gesture recognition are discussed. Most algorithms developed in this area and those imported from other areas - such as general detection of 3D pose of rigid and articulated objects - are applicable to

more than one functional block of visual interfaces. The discussion seeks to elucidate the interwoven pattern of algorithms and their applications.

The developments of human-friendly robots depends largely on two key aspects, the guarantee of safe operation and interfaces that allow the efficient use of such systems. Few systems have been developed that address these key aspects, but interest in the area recently has sharply increased and many research projects are on their way. Published results concerned with safety strategies and visual interfaces are reviewed.

Chapter 3: Face tracking

The feature based face tracking system developed in this research is described in this chapter. The system consists of three layers which extract the 3D head pose of a user from a monocular camera view. The lowest layer performs image correlations to find and track facial features such as the eyes, the mouth and the eye brows. The second layer merges the correlation results with geometric constraints provided by the 3D model of the face of the third layer. The third layer estimates the 3D pose of the head based on the robust image positions of the facial features.

Chapter 3 focuses on the first two layers which are concerned with robust feature tracking in monocular images. The actual feature tracking is performed in the first layer. However, this layer is not able to detect or recover from the loss of features, nor can it tolerate temporary occlusions of features or initialise when no feature positions are known. These issues are addressed in the second layer by mechanisms such as Kalman filters to integrate the tracking results with the geometric constraints between the features, the exploitation of the strain exerted on these geometric constraints, efficient tracking of changes in the appearance of the facial features and the balancing of resources spent between the position and the appearance tracking according to the tracking status.

Experiments demonstrate how the mechanisms achieve robust feature tracking, including feature position initialisation and feature appearance tracking.

Chapter 4: 3D head pose estimation

The third layer of the system estimates the 3D head pose of the user which is required to generate the 2D constraints for the second layer and to derive the 3D eye gaze position. The basis for the 3D pose estimation is an improved version of the alignment algorithm developed by Huttenlocher and Ullman [Huttenlocher and Ullman 1990]. An analysis of the accuracy and sensitivity of the original algorithm is presented which leads to improvements in these two crucial aspects. The introduction of a sensitivity model for fast estimation of the sensitivity of an estimate allows the integration of the results from multiple feature triplets into a single 3D pose result with a low systematic error and low sensitivity.

Simulations and experiments using a robotic mannequin head verify the validity of the approach. The implementation of all three layers is experimentally validated on real image sequences with a person performing large and rapid head motions.

Chapter 5: Eye gaze and gesture recognition

Two additional modules based on the data derived by the face tracking system complete the visual human-machine interface: An eye gaze detection system and a facial gesture recognition system. The eye gaze estimation system provides a deictic communication channel that allows the user to reference objects and locations in the environment. The chapter also presents an experimental evaluation of the performance of the eye gaze tracking and gesture recognition modules.

The gesture recognition system consists of two layers, a preprocessing layer which detects characteristic constellations of the head pose parameters at each instance and a temporal matching layer that detects the temporal patterns in the output of the first layer.

The combination of the two modules forms a powerful visual interface for human-machine interaction. The interface uses the same communication channels that are also used in inter-human communication. Untrained users can control any device connected to the visual interface after only a few minutes.

Chapter 6: Human-robot coexistence

This chapter formulates a set of rules for the behaviour of human-friendly robots which define a framework which guarantees the safety of the human operator. It is particularly important to agree on a division of the responsibilities in the interaction between the human and the robot. Such a division is presented which allows for the development of behavioural rules for a human-friendly robot. The chapter presents a definition of the basic restrictions on the autonomous actions of a human-friendly robot. Also a control architecture is presented which incorporates the safety behaviour of the robot.

Chapter 7: Safe robot control

Chapter 7 derives a formulation from the discussion in the previous chapter in terms of control constraints for a robot manipulator. The impulsive impact force is identified as the crucial system parameter for pre-collision safety and an model for the impulsive impact force is derived. Based on this model a control constraint for the robot is defined which is then transformed into the motor torque space. The chapter discusses the conditions for the appearance of poles in the impact force model. This is crucial because the control constraints are undefined if the impulsive impact force model has a pole at a particular point. Finally, the proposed formulation of the

control constraints are verified in a simulation of a 2DOF robot manipulator. The simulation shows that impact force control can be successfully implemented.

Chapter 8: Implementation and experiments

Chapter 8 presents implementation details and experimental results of the control scheme on a robot manipulator. The hardware platform is the WAM arm (Whole Arm Manipulator). This entirely cable driven 4DOF manipulator is well suited to human-robot interaction experiments because of its kinematic design, hardware safety features and light weight mechanical design. The safety scheme developed in the previous chapter is augmented with a position controller to allow the robot to move in a goal oriented manner. An outline of the software architecture used in the implementation is also presented. The implementation is tested at three different experimental levels, passive, externally generated motion, ego-motion and collision with a static obstacle. The experiments show that the impulsive impact force limitation controller has been successfully implemented.

Chapter 9: System integration

This chapter shows how the previously described components of human-friendly robots, the visual interface and the safe robot control scheme, can be used to create robotic systems that allow non-experts to interact with a robot safely and in a natural and intuitive way. The system presented allows a person to control the robot with eye gaze and facial gestures only. The robot is able to retrieve objects from the environment or to put objects down in positions the user identifies by simply looking at them. The robot is also able to hand objects over to the user or the receive objects from the user.

The results of an interaction experiment are presented which demonstrate all functionalities of the system. It shows the details of the sensory data used to drive a finite state machine which represents the system state and generates all robot commands according to the gesture commands of the user.

Chapter 10.1: Conclusions

This chapter reviews the various aspects of the work presented in this thesis and places the results in the context of the state of the art in the area of visual interfaces and human-friendly robots. This thesis represents one of the first attempts in this new research area. A discussion of the wide range of possible further directions in the ongoing development of human-friendly robots is presented.

Chapter 2

Related Work

This chapter reviews previous work in the areas related to this research. It provides a road map for visual human-computer interfaces and human-friendly robot control/human-robot interaction. The review categorises the different approaches and goals of the previous work. The review is followed by a discussion of the specific results that have been achieved in the visual human-computer interface area and the human-friendly robot control area respectively. This is followed by a review of the first attempts to build visual interfaces to allow a person to interact with a robot naturally and intuitively. The chapter ends with a summary.

2.1 Road map

Visual human-computer interfaces have had three major development thrusts; systems and methods concerned with face, hand and whole body tracking. The hand and body tracking systems are discussed only briefly since the work in this thesis is concerned with face tracking only. Figure 2.1 gives an overview over the various aspects and functions of hand and body tracking systems. While hand tracking is related to face tracking in the sense that the detailed configuration of a body part over time is of interest, the work done on human body tracking is primarily concerned with the gross posture of the whole human body. No significant work has been done on integrating the three areas. One of the main reasons for this lack of integration of these obviously related areas is the problem of acquiring images of appropriate resolution. If the whole body of a person has to fit into an image, the image areas which contain the hands and the head of the person have insufficient resolution for most hand and head pose tracking systems.

Some work has been reported which track the head and the hands of a person in upper torso image sequences, but no hand or head poses are extracted [Wren and Pentland 1998], [Wu et al. 2000]. [Hongo et al. 2000] reported a system that uses multiple cameras, a wide angle scene camera and multiple cameras for the face and the hands respectively. The system recognises directional gestures of the face

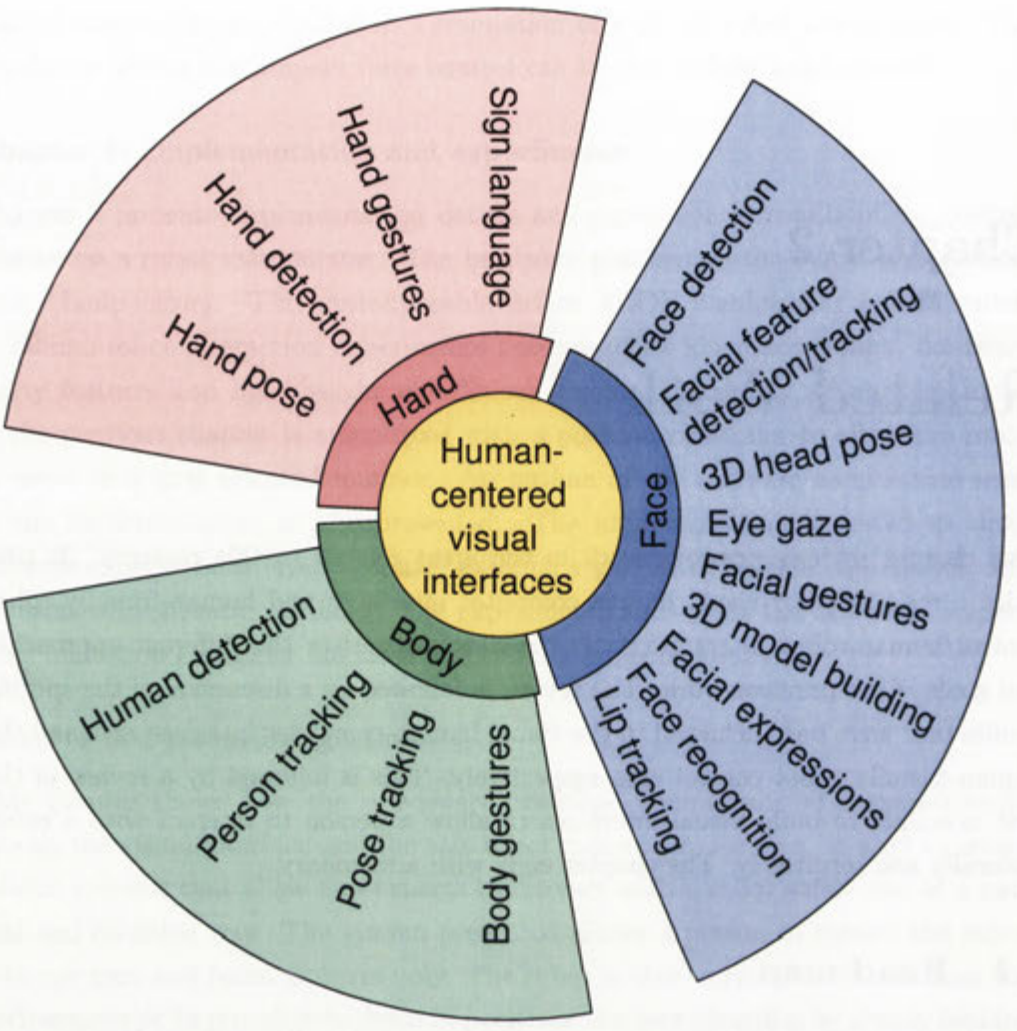


Figure 2.1: Categorisation of human-centred visual computer interfaces

(nodding etc.) and shape gestures of the hands (such as holding two fingers up), but the technology can simultaneously determine the upper torso, head and hand postures.

The categorisation shown in Figure 2.1 is based on sub-functionalities of visual human-machine interfaces. One of the future challenges will be to integrate the systems from the different areas and create interfaces that allow the tracking of the body, hand and head postures. The various sub-functions of each of the three areas shown in the outer ring are closely related, and most systems integrate more than one of those sub-functionalities. A detailed description of methods and algorithms of the hand and full body tracking systems are presented in Section 2.2.7.

Visual interfaces which are concerned with the tracking of human faces have different sub-functions as shown in Figure 2.1. The sub-functions are basic building blocks to a desired overall function. For example 3D head pose estimation can be based on feature tracking [Gee and Cipolla 1994] [Matsumoto and Zelinsky 2000] or different variants of face detection algorithms such as 3D model fitting [Zhang

and Kambhamettu 2000] [Wu and Toyama 2000], view based face detection methods [Pappu and Beardsley 1998] or exploitation of symmetry [Jacquin and Eleftheriadis 1995]. Alternatively, sub-functions which depend on other sub-functions may implicitly include them. View based systems [Pappu and Beardsley 1998] perform both, detection and pose estimation, in a single step and do not execute a separate face detection step. Most of the reported systems are no more than proof-of-concept systems and often the sub-functions that are required for initialisation are not implemented. Instead, a system may require the user to select the features in the first frame of a sequence so that the system can track the features motion in subsequent frames [Gee and Cipolla 1994]. Therefore, care is required when functionalities of different systems are compared.

The proposed categorisation of visual interfaces using sub-functions is one possible method. A categorisation could also be made with respect to the algorithms and methods used to achieve the functionality of the system. One algorithm can be used for a number of different sub-functions in different areas. For example the skin colour detection algorithm [Yang and Waibel 1996] is widely used for the detection of faces in head tracking systems, for the detection of hands in hand pose estimation systems and in the detection of the face and hands in body pose tracking systems. The resulting interweaving of algorithms and their applications is more complex than in the case of the categorisation along sub-functions. Accordingly, Section 2.2 is structured according to the sub-functions shown in Figure 2.1. Table 2.1 provides and overview of the relation between the functional blocks and the algorithms and methods used to implement the functions.

2.2 Visual human-computer interfaces

Visual human-computer interfaces have attracted considerable attention in the academic community in recent years. Some of the pioneering work in the face and head pose tracking area was done by [Azarbayejani et al. 1993] and by [Gee and Cipolla 1994]. The first non-intrusive eye gaze tracking system using neural networks was reported by [Baluja and Pomerleau 1994a].

These early systems had shortcomings since they were the "first shots" at the respective problems. The feature based face tracking systems of Azarbayejani and Gee could not recover from feature tracking errors which are inevitable with any feature tracking method. This lack of robustness caused unrecoverable (and in Gee's system undetectable) failure if the person moved too fast, turn his/her head too far from the camera or simply stepped out of the image and then returned. Gee's system also required the user to manually select the facial features in the first frame. Baluja's gaze tracking system used an artificial neural network to analyse the raw pixel values in the image region containing one eye of the user. This system could not tolerate head motion. However, all three systems were capable of real-time operation, an

	Face detection	F/feature detection	3D head pose estimation	Eye gaze estimation	Facial gestures	3D model building	Lip tracking	Facial expression recognition	Face recognition	Hand tracking	Human body tracking
Skin colour	✓	✓			✓		✓			✓	✓
3D model matching	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓
Symmetry detection	✓	✓				✓					
Image invariants	✓	✓	✓								
Eigen spaces	✓	✓					✓	✓			
Art. neural nets	✓	✓		✓							
Template correlation	✓	✓	✓	✓	✓		✓	✓	✓		
Optical flow	✓		✓		✓		✓		✓	✓	
Gabor filters	✓	✓	✓	✓		✓					
IR illumination	✓	✓	✓		✓						
Hough transform.		✓	✓		✓						
Heuristic filters		✓	✓	✓							
Kalman filters		✓	✓	✓		✓				✓	✓
Hidden Markov m.					✓						
Kinematic models										✓	✓

Table 2.1: Overview of the algorithms and their applications

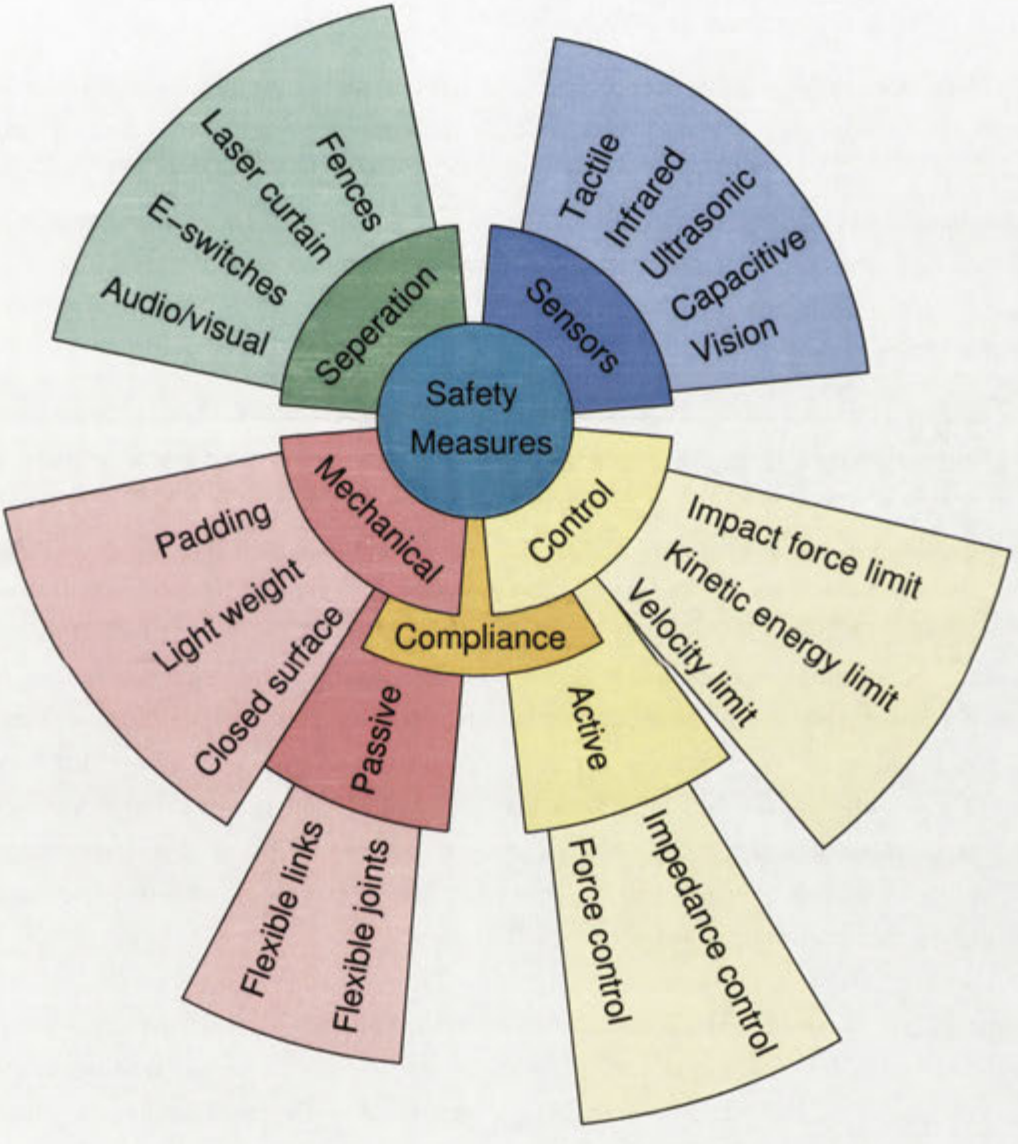


Figure 2.2: Categorisation of safety measures in robotics

important characteristic which has not been matched by many later systems, which is impressive considering the much lower computing power available at the time.

The area of visual human-computer interaction now attracts much attention, not only in the academic community, but also in the computer vision industry[1]. The important subareas of visual human-computer interfaces are discussed in the following subsections.

2.2.1 Face detection

The detection of faces in still images and image sequences is one of the basic sub-functions of face-centred visual interfaces. It is required for almost all further processing stages. As in all visual feature detection methods, a tradeoff has to be made between missed occurrences and false detections of the feature. Different applications have significantly different requirements for the two error modes.

Appearance based

One straight forward method to detect faces is image correlation, that is, comparing small apriori known images of faces with the input images. Systems with a bank of fixed templates were reported in the early 1990's [Brunelli and Poggio 1993], but their performance was unsatisfactory as they had low robustness with respect to different orientations and scaling of the faces in the image. However, this method is still used today as an initialisation stage for other face tracking systems such as face tracking systems [Matsumoto and Zelinsky 2000]. In such systems missed occurrences of faces are not critical since the detection is performed on a continuous video stream. Therefore misses in some frames do not degrade the overall system performance noticeably. This method has also been used in combination with the estimation of head pose [Pappu and Beardsley 1998]. The use of deformable templates is a variation of the fixed template method and has been applied to individual sub-features such as the eyes and the mouth (see Section 2.2.2).

Another variation of template matching is the use of eigenspaces [Sung and Poggio 1994]. The eigenspace approach assumes that all face patterns occupy a small parameterisable sub-space in the high-dimensional image space. In the training stage a representation of this sub-space, called the *canonical face* representation, is generated from a small number of face patterns. Clusters of face patterns are represented by their principal components, or eigenvectors, as multi-dimensional Gaussians. In the detection stage the cluster model is used to calculate the distance vector of a given image to the model. [Sung and Poggio 1998] uses a Mahalanobis like distance function. The resulting distance vectors serve as input to a multi-layer perceptron. The perceptron is trained with the distance vectors from the face and non-face patterns. In the face detection stage the system performs an exhaustive search of the

[1] See Appendix B for an overview on the commercially available face/gaze tracking systems.

image. For each region the distance vector is calculated and then classified by the perceptron as a face or a non-face pattern. To allow for different scalings of faces the processed regions are scaled to 19 × 19 pixels (the size of the training images) and are normalised with respect to brightness and contrast. The system achieved rates of missed occurrences between 3.7% and 20.1% for two sets of images with different quality. Although this is a good result compared to other template based methods, it requires a huge number of finely cropped face images for the training of the perceptron. The canonical model was generated from over 1000 face images and the perceptron was trained with over 47000 face images. The computation times required in the detection stage has not been reported by [Sung and Poggio 1998]. However, the system reported by [Rowley et al. 1995] which processed a 320 × 200 pixel image in 24s on a SPARC 20 claims that it is 50 times faster than the system reported by [Sung and Poggio 1998].

Today purely template based face detection methods are only used in special applications such as initialisation for face tracking systems. For general face detection tasks their performance, both accuracy and computing time, are not competitive compared to other methods. Template matching methods are also restricted with respect to the orientation of the head in the image. Most systems only consider frontal face views without head rotation and therefore are not able to detect faces from other angles.

Neural networks

A number of face detection systems based on neural networks have been reported. Sung's system described in the previous section used a neural network to classify distance vectors derived from the eigenspace model. Other systems have been reported that use the raw pixel intensities (brightness and contrast normalised) as input to neural nets [Rowley et al. 1998], [Duta and Jain 1998] and [Feraud et al. 2000]. [Rowley et al. 1995] used face images 20 × 20 pixels in size as the input to a neural network. The neural net contained 3 different types of hidden nodes: 4 that are connected to 10 × 10 pixel sub-windows, 16 which look at 5 × 5 pixel sub-windows and 6 overlapping 20 × 5 pixel sub-windows of the region to be analysed. These sizes were chosen to allow the neural network to specialise in the detection of facial features such as eyes, pairs of eyes, the mouth and so on. The system also arbitrated between different networks which had different topologies and were trained differently. The achieved accuracy and processing times in tests using the training images as test images were better than those reported by [Sung and Poggio 1998]. A similar approach was also reported by [Feraud et al. 2000]. This system used a variation of neural networks called the Constrained Generative Model (CGM). During learning each non-face example is constrained to be reconstructed as the mean of the n nearest neighbours of the nearest face example. In effect, the neural net learns the projec-

tion P of a point x of the input space on the set of faces sub-windows. The resulting distance is a measure of the likelihood that the sub-window contains a face. The system feeds the regions to be examined through several stages of increasing complexity and in the final stage through the CGMs. Different CGMs are provided for different viewing angles. This improves the generality, however, 2 frontal view CGMs and 2 side view CGMs can still not cover all viewing angles, in particular rotation around the vertical and optical axes. The results of the CGMs are post-processed by another neural network with arbitrates between the CGM results and determines the final result. The system provided similar accuracy as the system by [Sung and Poggio 1998] but took only 1s of computation time on a DEC Alpha. [Duta and Jain 1998] reported a system that used the algorithms of the two systems described above in addition to a third NN based algorithm and merged their individual results to achieve a slightly better result than each individual systems.

The approach to incorporate not only one but many algorithms into a system to exploit as many characteristics of the face as possible is taken by a number systems that are discussed in this chapter. [Triesch and von der Malsburg 2000] is one of the first attempts to formalise the integration of multiple visual cues.

Neural network based systems suffer from the same shortcomings as the template and view-based systems. The detection of faces in different scales and head poses can not or only with large computational complexity achieved. Furthermore, the training of those systems requires many (at least many hundreds) preprocessed face patterns that have been manually cropped from images. The requirement for an exhaustive search of unknown images also adds to the computation time requirements.

Structural models

Shape models and invariants are based on the assumption that certain geometric constraints hold for all faces and for all head poses. Although it may not appear that shape models and invariants are closely related, they really are two sides of the same coin. Image invariants are mathematical representations of geometric relationships between features of an object in an image. Invariants are generally used to verify a hypothesis about the location (and possibly the pose) of an object in an image or derive additional information such as 3D parameters. See [Pinto-Elias and Sossa-Axuela 1998] for example. A typical setup is to use filters to detect eyes and mouth regions in the image separately [Saulnier et al. 1995]. The invariants suggest that there can only be two eyes for each face and the mouth must be on the line which is perpendicular to the connection between the eyes and intersects it in the middle between the eyes [Shakunaga et al. 1998]. Such constraints are commonly used in facial feature detection systems (see Section 2.2.2). Shape models implicitly represent invariants of the features captured in the model. The goal of the model matching algorithm is to fit the shape model in a way that the implicit invariants are best

satisfied, or to reject the hypothesis that the image contains a face in the selected region [Pinto-Elias and Sossa-Axuela 1998]. Image invariants and shape models capture the structure of faces and use structural evidence to localise faces.

Models and invariants can be divided into two major classes: 2-dimensional and 3-dimensional models/invariants. To verify a hypothesis about the occurrences of faces based on feature localisation 2D invariants can be used [Shakunaga et al. 1998]. Such 2D invariants can also by used to derive 3D information without a 3D model. [Gee and Cipolla 1994] derived the 3D facial normal from 2D feature locations. The face detection system described by [Yokoyama et al. 1998] used a contour model to fit an elliptical model to the face, using the symmetry invariant of frontal face images. Elliptical models that exploit local symmetry have been used by [Brunelli and Poggio 1993] and [Jacquin and Eleftheriadis 1995] for face localisation. The iterative nature of these algorithms means that they have a high computational complexity, and are therefore generally used as a sub-function in conjunction with other techniques. For example, in conjunction with skin colour segmentation [Yokoyama et al. 1998].

Invariants and 3D models have been used for both face detection [Elagin et al. 1998] and head pose estimation [Wu and Toyama 2000]. The system reported by [Elagin et al. 1998] used a 3D model of the face consisting of a graph in which each node represents a facial feature identified by its response to a Gabor filter. The edges are the geometric relationship between the nodes. [Elagin et al. 1998] use hand crafted graphs, called bunch graphs, which are derived from a number of graphs of different persons. Bunch graphs representing different head poses are correlated over the whole image to detect faces. A similar approach was described by [Pramadihanto et al. 1998]. This system used a flexible graph of Gabor filters to ensure geometric consistency of the features in a detected face *after* an initial stage that detects the individual features. [Maurer and von der Malsburg 1996] describe a system that uses a automatically generated, rigid network of Gabor filters for face tracking.

The work described in this thesis is closely related to the systems described in the previous paragraph since it also uses a 3D model of filter responses of facial features to detect the face and head pose. However, unlike previously reported systems, our system is able to acquire the face pose robustly in real-time and initialise the feature and head pose tracking.

The system described by [Wu and Toyama 2000] uses a different model. Instead of generating a graph were the nodes correspond to unique facial features, they generate a head model with Gabor filter responses on regularly placed points on an elliptical head model. The resulting general head model, which captures a 360° view of the head, can then be matched on input images to localise faces and determine the head pose. The drawback of such general models is that they allow only approximate estimates. Errors of 20° are reported. Well initialised feature graph models provide accuracies in the range of 1° to 3° [Matsumoto and Zelinsky 2000].

In conclusion shape models and invariants have been applied with only limited

success to the face detection problem. They require exhaustive searches of the images and are not competitive in comparison to other techniques. However, these methods have been applied successfully to the feature and head pose tracking problems.

The work described in this thesis uses a structural model to detect the position of the face in the image. Our system achieves a high performance in face detection due to two key characteristics: The feature detection is based on a simple correlation method (see Chapter 3) which allows 6000 feature searches per second and the use of a flexible model based on probabilistic methods to exploit the partial detection of facial features.

Skin colour

Skin colour detection is the most widely used algorithm today to detect faces in images. Colour information has been long used to segment regions in images containing objects of interest [Ohta et al. 1980] [Swain and Ballard 1991]. The effectiveness of this method applied to the detection of human skin in colour images became apparent through the work of [Yang and Waibel 1995]. The key idea was to consider images in a brightness-normalised colour space (for example HUV or HSI) instead of the normal RGB representation. Skin coloured pixels occupy only a small region in the pure colour spaces, and can therefore be detected reliably. Yang and Waibel developed a simple model of the distribution of skin colour in the UV space which allowed the transformation of a colour image into a grey scale image where the brightness of each pixel represented the likelihood that the pixel was human skin. This method can be augmented to automatically accommodate for the slight differences between the skin colour of different people [Yang and Waibel 1996]. [Yang et al. 1998] describes how this method also works effectively on persons from different races such as Asian, African American and Caucasian, even though the brightness of their respective skin regions differs significantly.

This method is effective in detecting skin coloured regions with a small computational overhead, and real-time performance can be easily achieved. However, a face is not necessarily the only skin coloured surface in an image. It is well known that wooden surfaces and clothes of particular colours occupy the same region in the pure colour space and therefore can not be distinguished from skin by this algorithm. Therefore skin colour segmentation is usually followed by other methods which take into account motion [Yang and Waibel 1996], eye blinking [Schwerdt and Crowley 2000], the size and shape [Cai and Goshtasby 1999] [Heinzmann et al. 1998][2], image invariants [Terrilon et al. 1998], wavelet correlation [Kumar and Poggio 2000], or the local symmetry [Sun et al. 1998].

In general, skin colour detection is a cheap and reliable preprocessor for face

[2]It should be noted here that although I am the first author of this paper, this particular system was developed by Dr. Yoshio Matsumoto with whom I had the pleasure to work with.

tracking. The method has now become popular in the areas of face/hand detection/tracking.

2.2.2 Facial feature detection and tracking

In most applications the detection of the face is only the first step. For many applications such as head pose tracking, eye gaze estimation, 3D head model building, facial expression recognition and face recognition a more detailed analysis of the face is required. The system must extract facial features such as the eyes, the eye brows, the mouth and so on. The previous section argued that face detection and facial feature detection are strongly related. Face detection can be achieved by detecting facial features and assembling the results into a coherent face structure using a model or invariants [Pinto-Elias and Sossa-Axuela 1998], [Shakunaga et al. 1998], [Elagin et al. 1998], [Pramadihanto et al. 1998]. This method is used in the research presented in this thesis.

On the other hand, facial features can be detected based on the results of the face detection by exploiting the correspondence between a correlation kernel of the face and individual facial features [Matsumoto and Zelinsky 2000] or by examining the "holes" in the skin area of the face which are a result of the different colour of the eyes [Cai and Goshtasby 1999] [Heinzmann et al. 1998]. Whether the top-down or the bottom-up approach is favourable depends on the constraints of the application. In face tracking applications the head pose changes only slightly from frame to frame, and therefore the motion of the facial features can be easily detected. If the facial features can be tracked, therefore the location of the face can be determined. For applications that require the detection of a face in a single frame the top-down approach is faster because the rough search for the whole face quickly restricts the area in which feature detectors are to be used. However, if computational complexity is not important, for example in off-line indexing applications [Feraud et al. 2000], a bottom-up approach can be used.

Many of the algorithms used in the detection of faces which treat the face as a single entity could also be used for the detection of facial features.

Appearance based

Similar to face detection, template matching is the most straightforward method to localise facial features. The facial feature tracking system reported in [Matsumoto and Zelinsky 2000] used brightness normalised correlation of pixel intensity templates to detect and track features such as the corners of the eyes, the eyebrows and the corners of the mouth. In this system, the approximate locations of the facial features were determined by using a low-resolution facial template and the correspondence of the feature locations with that template. At the approximate feature locations individual, high resolution templates were correlated with the image to derive and

then track the exact feature locations. When the system detects a tracking failure, it switches back into the face searching mode with the low-resolution face template.

[Pappu and Beardsley 1998] described a system that used intensity gradient direction correlation instead of brightness normalised intensity correlation. Each element of the templates contained the direction of the gradient of the intensity at the location of the corresponding pixel. The correlation function summed up the difference of the gradient directions of the template and the region. Since the gradient does not depend on the absolute image intensity the resulting corresponding measure is also normalised for brightness.

The previous section described the use of eigenspaces to detect faces. The same technique can also be used to detect individual features. For each feature a eigentemplate is generated from several canonical views of the feature. The work described by [Shakunaga et al. 1998] used eigentemplates to detect feature candidates in images and post-processed the result by exploiting the geometric structure of the facial features.

A variation of template based methods is the wavelet convolution. [Maurer and von der Malsburg 1996] described a system that used Gabor wavelets to detect and track facial features. Each feature is described by a *jet*, a vector of Gabor filter responses at an image location. The system uses Gabor wavelets of 4 orientations and 8 frequencies. Therefore each jet is described by 32 complex numbers. The detection of a feature is achieved by calculating the filter responses at an image location and comparing it with the response of the original feature. The distance function of two jets is the dot product of the two filter response vectors. Because the displacement is calculated by the phase shift between the wavelet responses of the different wavelengths, subpixel accuracy can be achieved. However, the maximum translation between two consecutive frames is limited by half the wavelength of the wavelet with the lowest frequency.

This technique has proven to be successful[3] in this application domain. Other systems using Gabor wavelets have been reported by [Pramadihanto et al. 1998] and [Elagin et al. 1998]. Haar filters are a variation of the Gabor filter and have been used by [Kumar and Poggio 2000] to achieve similar results.

All appearance based methods described above use brightness independent localisation algorithms. The system described in this thesis uses pixel (colour) intensity templates to detect and track facial features similar to the system reported in [Matsumoto and Zelinsky 2000]. However, due to the limitations of the hardware vision system, the correlation is not brightness normalised. Section 3.3.4 describes the problems associated with this fact and the additional algorithms that are required to achieve robust feature tracking and overcome this limitation.

[3] A company called EyeMatic has been spun off using this technology. See `http://www.eyematic.com/` for more information.

Neural networks and genetic algorithms

The face detection system described in Section 2.2.1 reported by [Rowley et al. 1995] and [Rowley et al. 1998] used a neural network (NN). The topology of the NN was chosen such that parts of the network specialise in detecting certain facial features such as the eyes and the mouth. Although this particular system does not detect individual features, this technique can also be used to detect the location of facial features.

The system described by [Pinto-Elias and Sossa-Axuela 1998] uses a genetic algorithm to "learn" the representation of facial features in terms of Hu's invariants [Hu 1962]. The invariants are calculated with respect to the moments for each location in the image. Results of tests on 240 images showed a detection rate of 100%, with a false detection rate of approximately 7%.

Skin colour

As stated earlier, the results of skin colour detection indicate not only the location of the face in the image, but also hints at the location of facial features. The eyes, the eye brows and the mouth can be segmented in the image since they are non-skin coloured. This fact can be used to detect the location of these features when combined with a geometric model of the facial feature locations.

The work described by [Sun et al. 1998] first performed a detection of facial features without classifying them, merged the result with detection of facial symmetry and finally matched a fuzzy model to the combined bitmap to detect and classify individual features. A similar approach was described by [Cai and Goshtasby 1999]. This system tried to match multiple templates corresponding to different head poses to a skin colour detection bitmap in order to detect individual features.

The skin colour maps are used for the detection of facial features rather than for tracking features. The computation of the skin colour map and the respective feature locations is expensive compared to correlation methods and only yields a approximate position estimate. However, it provides a robust way to initialise facial feature tracking systems as described in [Heinzmann et al. 1998].

Infra-red illumination

An active method to detect facial features that should be reviewed is the exploitation of the retro-reflectivity of the human eyes through infra-red (IR) illumination. This effect is also known as the *red-eye effect* in photography which occurs when a flash light that is close to the optical axis is used. The basic idea is to switch a set of IR-LEDs on and off in synchronisation with the camera frames. The activation state of the LEDs is toggled at each vertical sync of the camera. In effect the video stream contains frames with alternating IR illumination. While the pupils appear dark in the frames without IR illumination, through the retro-reflectivity effect the pupils

are bright in the frames with IR illumination. The flickering of the pupils can be easily detected using image subtraction. This technique is robust under changing illumination conditions and it requires few computational resources. For this reason basically all commercial eye gaze tracking systems use IR illumination technology (see Appendix B).

Publications describing this technology date back more than 10 years, [Tomono et al. 1989] and [Hutchinson et al. 1989]. However, recent attention has been drawn towards the method through the *Blue Eyes* project at IBM's Almaden Research Center [Morimoto et al. 1998], [Morimoto et al. 1999] and [Morimoto and Flickner 2000].

Hough transformation

In 1962 P.V.C. Hough patented a method to detect patterns in digital images [Hough 1962]. The *Hough transformation* has become one of the standard computer vision algorithms to detect shapes such as lines, circles, ellipses or arbitrary shapes in images. The core idea is to use a voting scheme to accumulate evidence in the image that supports the hypothesis that a parametrisation of the shape appears at a particular image location. For a discussion of the variations of Hough transformation algorithms refer to [Kassim et al. 1999].

The Hough transformation for the detection of circles and ellipses are of particular interest to detect the iris. The contrast between the iris and the eye white is a strong feature and can be detected reliably. However, usually the top and bottom of the iris are occluded by the eye lids, so only two arcs on the left and right are visible. The Hough transformation is well suited to exploit this evidence and derive the location and parametrisation of the circle or ellipse. This method has been used successfully in non-intrusive gaze tracking systems such as [Klingspohr et al. 1997] and [Matsumoto and Zelinsky 2000].

The Hough transformation does have significant memory requirements to store the discrete voting space. The voting space for a circle of unknown radius is 3-dimensional (x, y of centre and radius) and 5-dimensional for an ellipse (x, y of centre, major and minor radius and the angle of major axis). To date, this method appears to be the most robust method to detect the location of the iris and allows for real-time operation [Matsumoto and Zelinsky 2000].

Heuristic filters

Until recently, real-time performance was not easy to achieve for facial feature tracking systems. The detection methods have to be efficient to allow calculations at video frame rate. This situation has led to the development of simple heuristic filters that were able to operate in real time. The system described by [Gee and Cipolla 1994] which pioneered head pose estimation used two simple heuristics to

track the eyes and the mouth. The eye tracker simply searched for the darkest pixel in a window around the previous detection. Since the eye is surrounded by skin this heuristic works reasonably well but is inaccurate since eyelashes can also produce black pixels. The method works reasonably well while the head faces the camera. The mouth was tracked using an edge detector. All the tracking filters had to be positioned by hand at the beginning of the sequence. The system described by [Saulnier et al. 1995] used similar heuristics that were implemented as dedicated software modules for each type of facial feature. The eye gaze tracking system described by [Baluja and Pomerleau 1994b] also used a heuristic filter to detect the location of the eye. This method detects bright spots within dark areas. This solution was tailored to a particular setup where the user sat in front of a desktop computer with a light source mounted below the monitor which created a reflection on the cornea.

The recent increase in computer speed has made such approaches obsolete. Other methods which require higher computational resources such as brightness normalised correlation, skin colour detection and the Hough Transformation provide higher reliability and robustness with respect to projective deformation and rotation of the features.

Summary

This section reviewed the principal techniques for facial feature detection and tracking and showed the close relationship to many face detection techniques. Often face and facial feature detection go hand in hand to achieve robust results. The choice of algorithm depends highly on the application and real-time requirements of the system. While the skin colour based methods are reliable and efficient if the whole image has to be considered (for example for facial feature tracking initialisation and model building), the tracking of facial features in real-time can be achieved most efficiently with template correlation or Gabor jet convolution methods. The robustness of systems can be increased by combining different methods, for example combining skin colour, facial symmetry and geometric constraints [Sun et al. 1998].

2.2.3 3D head pose estimation

Head pose estimation is an essential part of many visual human-machine interaction systems such as low-bandwidth video conferencing, face recognition, computer games, character animation and virtual world interfaces. The 3D head pose is sometimes referred to as the *gaze direction*, implicating that a person is generally looking where the user's nose is pointing. This is only an approximation of the persons true gaze direction and the term *gaze direction* is inappropriate. In this thesis the term *head pose* refers to the 3D translational and rotational state of the head. The term *eye gaze direction* is used to refer to the 3D gaze vector which originates in the centre of the eye ball and points through the centre of the pupil.

The various methods that are used to derive the head pose of a person can be classified into two significant approaches. The first approach considers the head as a single feature in the image and exploits the *appearance* of the face to derive the head pose. The second approach considers the head to be a conglomeration of facial features and derives the head pose from the *geometric relationships* between individual features.

Appearance based methods

The system proposed by [Pappu and Beardsley 1998] used intensity templates to detect the head pose of a person. In the initialisation stage a 3D ellipsoidal model is manually fitted to a single image of the person to be tracked. The texture from the image is then mapped onto the ellipsoidal model and then by rotating the resulting textured ellipsoid a 11×17 matrix of views that represent the head with different head orientations around the horizontal and around the vertical axes is generated. The synthetic views are then matched against the live image at a sub-sampled resolution of 32×32 pixels. A trade-off must be made between the accuracy of matching and the number of synthetic views. The granularity of the resolution which influences the computational resources required. The use of an 3D ellipsoidal model of the head introduces distortions in the synthetic views, in particular for prominent features such as the nose. The system can not tolerate changes in depth or rotations around the z-axis of the head. The reported processing speed was 6Hz. The accuracy of the system was not reported but it is reasonable to assume that it had a similar accuracy as the system by [Wu and Toyama 2000] who reported an accuracy of approximately 10°.

The approach taken by [Wu and Toyama 2000] differs only slightly from [Pappu and Beardsley 1998]. It also assumes a 3D ellipsoidal model of the face, but instead of using straight image texture, the elements of the model are the response of Gabor filters. To achieve close to real-time performance, the search space for head poses is discretised in a similar way to that suggested by [Pappu and Beardsley 1998]. The reported processing speed was 3-5Hz. The achieved accuracy varied strongly with different persons and whether the person being tracked was used to generate the original head model. The average errors around each *individual* axis differed from 5.7° to 70.5°.

The technique reported by [Basu et al. 1996] differs significantly from the appearance models above. It utilised an extension of the method proposed by [Black and Yacoob 1995] which is based on optical flow to track the motion of a planar surface. Since optical flow is a relative measure of motion between frames, convergence over time can not be guaranteed. After a sufficiently long tracking sequence offset errors emerge. To reduce this drift effect the motion of the model can be found by minimising the difference between the actually observed flow field and a synthetic flow

field generated by rotating the face model. [Black and Yacoob 1995] used a planar face model which yielded poor convergence. Better results have been achieved with the extended 3D ellipsoidal model of [Basu et al. 1996]. The system described by [Zhang and Kambhamettu 2000] uses a more precise model that described the shape of a human head by using extended superquadrics. This system also extended the optical flow method to detect partial occlusions of the face and disregard the flow in the occluded areas.

The advantages of optical flow based systems compared to appearance based systems is that all 6 rigid motion parameters of the head can be detected and that no discretisation of the pose space is required. The major disadvantage is the drift error which accumulates over time. It should be noted that in all the reported work the experimental test sequences were no longer than several hundred video frames.

The system reported by [Malciu and Preteux 2000] combined optical flow and appearance based methods. It used a model similar to the extended superquadric model of [Zhang and Kambhamettu 2000] with texture mapping. The head motion is first estimated by the optical flow field. The resulting pose is used as the starting point for an optimisation process which minimises the difference between the observed view and the appearance of the model projected into the image plane. By merging both results the drift problem can be eliminated as the texture mapping always reset the model into the correct position. The system was tested with synthetic test images. The reported accuracy in 90% of the images was better than 3° in individual rotation angles and better than 8% for the depth measurement. The speed of computation was not reported.

Feature based methods

The recovery of the head pose based on the constellation of facial features has attracted much attention, particularly for head tracking applications. Feature based approaches do not have inherent drift problems and the tracking of facial features in image sequences is well within real-time capability of today's computer systems. The accuracy of feature based systems is generally better than that of appearance based systems for comparable computation and image resolution restrictions. Feature based systems also cope better with changing scaling of the face and therefore give the user greater freedom of motion.

The basic idea of head pose estimation in feature based systems is to match the pose of an apriori known 3D model of the facial features to the 2D feature location measurements in the image. If stereo cameras are used instead of a monocular system the measurements are 3-dimensional. However, the underlying pose recovery problem is similar although the actual calculations are quite different. The facial features are modelled as point features and are characterised by the texture properties of the area. The properties depend on the chosen feature detection method and could

include the actual texture, the intensity gradient field or the response to Gabor filters. The input to the head pose algorithm is the location of the point features, which are often augmented by statistical measures of the uncertainty of the measured points. Uncertainty measures are important to the robustness of the system. Visual feature detection methods are never perfect and all fail from time to time, thus providing incorrect results. Errors and measurement noise introduce inconsistencies in the observed feature constellation and therefore in most cases a *best fit* between the model and the measurements must be determined.

Depending on the input (monocular or stereo) and the perspective model chosen (affine projection or perspective projection) a closed form solution exists for the model fitting problem or an iterative search approach must be used. The closed form solution is derived from an overconstrained system and the resulting pose minimises the discrepancies between the measured feature positions and the positions predicted by the 3D model. Iterative solutions for the model fitting problem are computationally expensive. However, iterative solutions are suitable for continuous tracking applications when the change in head pose between consecutive frames is small. In real-time systems that run at 30/60Hz this is a feasible solution since head motions will be limited. An iterative search in the 6-dimensional state space of the head pose without prior knowledge of the approximate head pose should be considered as impractical.

In many cases by making certain restrictions allow the derivation of a closed form solution for the head pose. These restrictions can apply to the number of features [Gee and Cipolla 1994], [Shimizu et al. 1998] or require all features to lie within a plane [Maurer and von der Malsburg 1996].

The first approach that used a restricted number of features was reported by [Gee and Cipolla 1994]. The system uses 4 facial features, the two pupils and the corners of the mouth which can be assumed to lie within a single plane. Two vectors, the one connecting the pupils and the one connecting the centre of the two mid-points of the eye and the mouth connections, are used to derive the facial normal in a closed form solution. Gee later extended this method by tracking more features like the nostrils and the point between the eyebrows and used the RANSAC algorithm to select three points which, fed through Huttenlocher's alignment algorithm [Huttenlocher and Ullman 1990], produced the head pose that best explained the feature location of all features [Gee and Cipolla 1996]. Huttenlocher's alignment algorithm assumes affine projection (no depth forthshortening) and recovers the pose of a model consisting of 3 points in a twofold ambiguity with a simple and efficient closed form algorithm. A variation of the algorithm is used by [Shimizu et al. 1998]. The research presented in this thesis extends the original Huttenlocher algorithm. The extensions are presented in Chapter 3 and address the two major drawbacks of the original algorithm, it's high systematic error for feature positions generated by a perspective projection which is the case in all real images from a real camera and

the output of the algorithm, the rotation matrix, is only valid if the feature coordinates are the result of a noise-free affine projection. These characteristics hamper the application of the original algorithm to real image data and make the extensions presented in this work necessary.

The system described by [Maurer and von der Malsburg 1996] used more features, but assumed that all features lie within a plane. A least squares method is used to solve the resulting overconstrained equation system under affine projection. A similar approach was used by [Shakunaga et al. 1998], however, this method does not require the assumption that the facial features lie within a plane. An arbitrary number of 3D feature points without additional constraints can be used.

A stereo camera system that uses only a restricted number of features was reported by [Xu and Akatsuka 1998]. Like Gee's original system it used both eyes and mouth corners to estimate the head pose. However, 3D measurements are obtained from the stereo camera system. A closed form solution for the head pose is obtained from 3 out of the 4 available 3D measurements. The reported accuracy of the system indicated an average error of $\sim 3^\circ$ for each individual axis of rotation. The translational error was not reported.

All the previous work discussed thus far have an important shortcoming; there is no consideration of the tracking performance of the individual features in the head pose estimation. As mentioned earlier on, every visual feature detection method fails under certain conditions, for example when the feature surface normal is almost perpendicular to the image plane. Failure to track a facial feature can cause random head pose estimation results and ultimately complete tracking failure. Therefore, robustness to partial tracking faults should be a significant consideration in the design of face tracking systems if they are to be used in real-world applications.

One way to consider uncertainty for feature localisation results is to use an extended Kalman filter to estimate the head pose. This idea was first implemented by [Azarbayejani et al. 1993] for head pose estimations and has previously been described by [Clark and Kokuer 1992] and [Reinders et al. 1992] for the calculation of general object orientations. Azarbayejani's system uses an 18-dimensional state vector that included the 3D translation and rotation, and the associated first and second order derivatives. Although the relationships between the 2D measurements from a monocular camera and the corresponding state parameters are not linear, a local linearisation approximates the true relationships sufficiently well. The variance of the measurements is derived from the correlation values of the brightness normalised template correlation used for feature tracking. The reported accuracy for the rotational state of the head was 2.4°. The same method has been used in other head tracking systems [Saulnier et al. 1995]. These systems achieve near real-time performance with a frame processing rate of 10Hz. The 18-dimensional extended Kalman filter requires significant computational resources. However, it allows for the inclusion of feature tracking confidences to be included into the calculation of

the head pose.

An alternative solution to a Kalman filter has been proposed using a spring-damper model [Matsumoto and Zelinsky 2000]. This system used stereo vision to acquire 3D measurements for each facial feature. It connects the measurements and the corresponding feature model points with springs whose strength is proportional to the confidence of the feature detection. Then an iterative search process is used to find a new model pose which balances the pull of the springs. In effect, features which could be detected reliably influence the resulting pose of the head model more than unreliably detected features. The system achieves real-time performance of 30 frames per second with a slightly higher accuracy than Azarbayejani's Kalman filter solution.

Summary

The appearance and feature based head pose detection methods differ significantly in their performance profiles. Appearance based methods provide less accurate results but require less apriori knowledge about the head pose. They are well suited for single image processing and initialisation of feature based head tracking systems. Optical flow methods suffer from drifting problems which can only be solved by adding another module based on features or appearance. Therefore, optical flow methods are not well suited as the sole method of pose estimation, however, they can augment other methods. Feature based methods provide real-time performance and high accuracy, however, they usually require an apriori known model of the head. They also offer the possibility to incorporate multiple cameras to achieve higher accuracy, which is not possible for optical flow based systems. In general feature based methods offer significant advantages for real-time head pose tracking.

2.2.4 Eye gaze estimation

One of the driving forces behind the development of face tracking system is the goal to allow machines to sense where a user is looking. It is not sufficient to consider the head pose only, the orientation of the eyes must also be measured. The vision based methods that measure the eye gaze direction of a person can be classified into two main groups, use of corneal reflections and localising the iris with respect to the centre of the eye ball. Systems of both categories have evolved from restrictive laboratory based setups to complete unintrusive and unrestrictive systems. Early systems required people to put their heads into fixation frames to suppress head motions [Spindler and Chaumette 1997] and [Klingspohr et al. 1997]. The camera(s) were mounted to point directly at the persons eye(s)[4]. With the miniaturisation of camera technology it has become feasible for a person to wear the frame and the camera. Most head mounted systems contain a magnetic motion tracker to determine

[4]Many systems consider only one eye since in general both eyes are pointing at the same target.

the pose of the frame in space, while the cameras determine the orientation of the eyes relative to the frame [Iwamoto and Tanie 1997]. Recently, a small number of systems have been devised which not require the user to wear equipment and allow natural user head motion [Matsumoto et al. 1999].

Non-vision based methods also exist to detect a persons gaze. One commercial system uses contact lenses with tiny coils embedded. Users are required to wear the lenses and keep their heads within a magnetic detection frame which measures the motion of the coils in space.

Since the work presented in this thesis is focused on non-intrusive and non-restrictive gaze tracking systems, the following overview concentrates on methods that have been or can potentially be used in the same way.

Corneal reflection based systems

Section 2.2.2 described a method for facial feature detection using IR illumination. A similar method can be used to not only localise the pupil but also to derive the orientation of the eye ball. The IR illumination does not only create a bright pupil, it also creates a reflection on the cornea of the eye, referred to as *glint*. The displacement between this corneal reflection and the bright pupil in the image directly corresponds to the orientation of the eye ball in space. Therefore, by observing only the bright pupil and the corneal reflection and with apriori knowledge of the eye ball shape and the distance between the camera and the eye, the eye gaze vector can be determined [Hutchinson et al. 1989].

This method has been used in head fixation systems [Spindler and Chaumette 1997], wearable head frame systems and, in a limited way, in unconstrained head motion systems [Hutchinson 1993]. To achieve a good noise to signal ratio with this method, a high resolution image of the eye is required since the distances between the pupil and the corneal reflection are generally only a fraction of the eye diameter. In an unconstrained head motion system the camera must be mounted in front of the user and use a large focal length to achieve high resolution images. However, since the eye must remain in the camera's view, the motion of the user's head is restricted to only a few centimetres. To allow reasonable head motion an active camera system must be used [Morimoto and Flickner 2000].

In the image the bright pupil is usually much larger in size than the corneal reflection. In frontal views of the eye the position of the corneal reflection can not be determined since it is overlaid with the much larger bright pupil. This problem is solved by using not only the coaxial illumination for the pupil, but also one or two off-axis IR-sources which create additional reflections which can also be detected precisely [Morimoto and Flickner 2000] .

The corneal reflection method is used by nearly all commercial systems (see Appendix B). Most commercial systems use non-head mounted cameras so the user

is not required to wear any head gear. The elegant aspect of the corneal reflection method is that the head pose is not required for fixed camera systems. The orientation of the head does not influence the measurements of the system. The drawback of this method is that only small translational head motions are tolerated. Motions along the optical axis of the camera can also cause problems as the scaling of the eye in the image and the geometric relation between the eye and the environment changes. Obviously such head motion can not be detected since the 3D head pose is unknown. To use the system effectively users must keep their heads steady to about ±5cm [Hutchinson et al. 1989]. Systems that use frames or head gear are suitable for laboratory experiments but are too intrusive for general human-computer interaction. The corneal reflection method is popular because the technique works robustly under varying illumination conditions, it requires few computational resources and no head pose measurement is required.

Eye ball position based systems

People have the capability to easily estimate where another person is looking. The only cues available to humans are the 3D head pose and the location of the iris relative to the surrounding facial features. These cues can also be used by a computer vision program to determine the 3D gaze direction of a person. However, these cues are much more subtle and are subject to variations in appearance due to changes in illumination and head pose. They require sophisticated feature detection and pose estimation algorithms that are designed for robust operation in varying conditions. Obviously, such systems require more computational resources than corneal reflection systems. However, recent advances in computation speed are making such methods possible.

The requirements for a passive 3D eye gaze tracking system depend on the restrictions that are imposed. If the head can be mechanically fixed, the 3D gaze tracking problem can be reduced to finding the mapping between the *image* location of the iris and the 3D gaze direction [Klingspohr et al. 1997]. The method used to detect the iris is the circular Hough transform. A simple calibration procedure can be used to determine the 3D head pose which is implicitly contained in the mapping. This system used a linear interpolation between the eye gaze direction to the corners of a monitor in front of the person to find the gaze point. Although this model does not take into account that the iris is moving on the surface of a sphere, it's accuracy is reported to be in the order of 2.3° at a frame rate of 1Hz. This method is unsuitable for applications other than laboratory experiments.

A system with similar restrictions using a different processing method was reported by [Baluja and Pomerleau 1994b], [Baluja and Pomerleau 1994a]. The reported work was the first eye gaze detection system that did not use corneal reflections or a head frame. One of the eyes was located by a heuristic filter which

located a bright spot within a dark surrounding. This filter was tailored to a setup which consisted of a desktop computer with a light source mounted below to create a bright reflection on the cornea[5]. The raw pixel values were used as inputs to a neural network which derived the orientation of the eye. The reported accuracy of the system was 1.7° for an eye image area of up to 20 × 40 pixels at a frame rate of 10Hz. This accuracy was achieved by passing the gaze direction generated by the neural network through a 2D lookup table with static correction values. Although the system tolerated small movements of the head, the 3D head position could not be determined by the system and therefore translational head motion introduced errors in the gaze vector calculation.

The only way to achieve robustness to head motions in completely passive visual eye gaze tracking systems is to also measure the head motion. Measurements of the head pose and the image location of the iris allows the calculation of the gaze direction, independent of rotational and translational head motions. This philosophy has been pursued in this thesis. The reported work and the successor stereo system developed by [Matsumoto and Zelinsky 2000] are the only systems to date that have implemented this technique. The head model includes the position of the centre of the eye balls and their radius. The measurement of the head pose therefore yields the 3D position of the eyes. The image location of the iris allows for the calculation of a ray from the camera centre through the sphere of the eye. The intersection of this ray and the 3D position of the eye ball determines the 3D eye gaze direction. The system described in [Matsumoto and Zelinsky 2000] uses a circular Hough transformation to determine the location of the iris. The reported accuracy of the gaze direction is better than 3°. The system runs at full video frame rate (30Hz).

Summary

Two classes of eye gaze detection systems have emerged thus far. The corneal reflection method has the following advantages:

- Robustness to changes in illumination
- Precise measurements
- Few computational resources
- The measurement of the head pose is not required.

These advantages have made the method successful in the commercial area since the alternative method requires complex vision algorithms to achieve similar results. However, the corneal reflection method also has the following drawbacks:

[5]This reflection was not used in the same way as in corneal reflection systems. This work raises the question to what extent the neural network learnt to associate the location of the reflection to the iris position to derive the eye gaze direction.

- High resolution images of the eyes are needed
- Head pose/facial gestures can not be recovered
- Change in the distance of the eye to the camera can not be determined and give rise to erroneous results in the eye gaze point
- Narrow camera view angles allow for limited lateral head movement
- No estimate of the gaze direction is possible if the eyes are occluded
- Since IR illumination is required implementation on existing web-camera technology is not possible

The restriction of the head movements can partially compensated for by a pan/tilt camera. However, due to the narrow view angle and the limited frame rate of normal cameras fast head motions can not be compensated for. This solution does not address the problem of accounting for the distance between the camera and the eye. Overall the corneal reflection method does not provide the level of flexibility needed for human-robot interfaces. In applications such as interfaces to digital datacasting appliances for video telephony/conferencing, computer games and virtual worlds, the head pose is a crucial piece of information.

These shortcomings in the corneal reflection were starting point of the development of the visual interface described in this thesis.

2.2.5 Facial gestures

The recognition of facial gestures, such as nodding and shaking the head, has received modest attention. Gesture recognition is a typical pattern recognition problem. In every performance of a gesture the timing and facial feature motion parameter amplitudes differ slightly from other performances of the same gesture.

This kinship to natural speech recognition was recognised by [Darrell and Pentland 1993] who devised a gesture recognition system for facial expressions and hand gestures. They used a *dynamic time warping* algorithm which is widely used in natural speech recognition [Sakoe and Edelmann 1980]. However, this method requires considerable computational resources and only two gestures could be detected in real time [Darrell et al. 1995]. Unlike words, cyclic gestures do not have a precisely defined number of cycles making them difficult to recognise. For example, a nodding gesture typically consists of 2 to 4 cycles of up-down motion. The same problem occurs with the *continuous dynamic programming* method presented by [Nishimura and Okaa 1996] and later by [Wu et al. 1998] because both are based on a predefined series of measurement vectors.

In my masters research thesis [Heinzmann 1996] I developed a gesture recognition system that is based on the decomposition of the gesture into *atomic actions*, sections

of limited variability in the parameter flow. The method used finite state machines to model and recognise gestures. This algorithm is not only efficient (12 different gestures are distinguished at 30Hz sampling rate) but also provides the ability to model gestures with variable cycle numbers as well as the differences in gesture performances. A similar concept has been used by [Hong et al. 2000] for the detection of face and hand gestures and by [Kawato and Ohya 2000] to recognise nodding and head shaking gestures.

The recognition of spatio-temporal gestures is straight forward and has satisfactory error rates. Most errors are introduced through failures of the head tracking. Recognition methods based on the finite state machines have proven to be efficient and are sufficiently accurate to model facial gestures. The system described in my masters research has been reused in the human-robot interface described in this thesis.

2.2.6 Other face-related issues

Other issues related to face-centred visual interfaces are discussed here to round of the description of the functional blocks. However, the work presented in this thesis does not touch on these areas, but is important to face and facial feature tracking technology.

Face recognition

The recognition of faces for identification purposes has been the most active area of research in visual interfaces for many years. The body of work published in this area is too diverse to review. Overviews on the different approaches have been presented in [Brunelli and Poggio 1993] and [Chellappa et al. 1995]. Face recognition is related to the three previously discussed functional blocks: Face detection, feature detection and 3D head pose estimation. Obviously the location of the face has to be initially determined. Many face detection algorithms require the detection of the location of key features in the face to fit a model to the image or to extract characteristic parameters. If the system does not expect the person to look directly into a camera at a known point in time, it is necessary to track the persons head pose to determine when the person is in a favourable position for the recognition process to begin.

Facial expression recognition

The recognition of facial expressions has attracted considerable amount of attention. In some earlier publications the term *facial gestures* has been incorrectly used instead of the term *facial expression.* Facial expressions do not have a time component and are defined only by deformations of the facial surface. The recognition of facial expressions requires a deformable model of the face in the form of either a deformable feature mesh or deformable templates of the face. The detection of facial expressions

is an important function for applications such as video conferencing, virtual world access and character animation.

Past work published in this area has used optical flow [Mase 1991], full-face templates [Darrell et al. 1994], eigenspace full-face templates [Ohba et al. 1998] and deformable feature templates [Black and Yacoob 1995] to detect different facial expressions. The system described in [Tao and Huang 1998] used a full face model composed of deformable patches. Intensity edges have been used to measure the fine details of the deformations of facial features [Saulnier et al. 1995], [Ohta et al. 1998], [li Tian et al. 2000a] and [li Tian et al. 2000b]. The analysis of facial actions (facial muscle contractions) and their relation to emotion such as happiness, sadness, surprise etc. has been termed as *Facial Action Coding System* (FACS) [Ekman and Friesen 1975], [Ekman and Friesen 1978]. FACS is the basis of many classification methods in the systems described above.

3D model building

Most face tracking systems use hand crafted models. This is convenient for laboratory experiments as it reduces the complexity of the problem and the measured accuracy does not depend on the accuracy of the model acquisition stage. Hand crafted models also can be tweaked to yield the best performance and therefore the results tend to be better.

Face models have two aspects to them: Geometry and appearance. It is possible to learn one of these aspects assuming the other aspect is known or both aspects must be learnt by the system. Learning can be performed in a dedicated model acquisition stage before the tracking or it can be performed online during tracking. The two stage approach is usually preferred since it offers the possibility of asking the user to manually identify facial features or to fit a general face model to an image [Rowley et al. 1998], [Pinto-Elias and Sossa-Axuela 1998].

The system described in [Sengupta and Ohya 1998] and [Sengupta et al. 2000] used a regression method and assumed an affine geometry to determine depth maps from multiple monocular images to build a 3D model of the head without apriori knowledge. The resulting model is then texture mapped with the input images and was then used to generate animations of the face.

The approach presented by [Feng et al. 2000] uses an apriori geometric model of an average face consisting of feature points connected with spring loaded elements. Using a series of monocular images the model is adapted to a particular person. The derived model is then texture mapped with information from the input sequence for animation purposes. A similar system using stereo cameras and projective geometry was reported by [Nagel et al. 1998]. An apriori known average face model is adapted according to the results from the 3D stereo camera measurements.

In general the use of a predefined model requires the system to solve the difficult

problem of finding the correspondence between the model features and the image features.

Lip tracking

The mouth is a highly deformable facial feature. While the eyes can only rotate and open/close, the mouth can be brought into a multitude of different configurations. Lip motion is an important feature for applications such as character animation and teleconferencing. It is also used in speech recognition, in conjunction with the audio processing, resulting in a higher recognition rates. This is desirable for applications such as automatic indexing systems for digital media archives, automatic production of movie subtitles and automatic protocol systems in a video conference system.

Earlier systems used simple 2D models based directly on the appearance of the mouth region in the image [Coianiz et al. 1995] [Colombo and Bimbo 1996]. Today more complex 3D models are used to model the lip shape [Basu et al. 1998b]. Since the lips do not provide a high contrast with respect to the surrounding skin, earlier systems used coloured lipstick [Saulnier et al. 1995] or reflective markers [Basu et al. 1998b]. With increasing computational resources and the refinement of visual algorithms artificial markers are no longer required. Most 3D lip model systems use the analysis-through-synthesis approach to fit the projection of the 3D model to the observed 2D shape [Matthews et al. 1998], [Revéret and Benoît 1998] and [Basu et al. 1998a].

Lip tracking systems that allow users to move freely, and in particular move their heads around, require a face tracking system to reliably obtain images containing the mouth. Therefore, many lip tracking systems are built upon a face tracking system [Saulnier et al. 1995].

2.2.7 Non-face related issues in human-centred visual interfaces

Although this thesis entirely focuses on face-centred interfaces, two other important directions of development in the visual human-machine interface sector should be examined.

Besides the face of a user, the hands are also of interest. Hand gestures are used extensively to enhance the understanding of the spoken language. Hearing and speaking impaired people communicate entirely through sign language. Hands are also an intuitive way to interact with virtual environments. For those reasons the tracking of hands and hand gesture recognition has received considerable attention.

The other main direction of development in visual interfaces is the tracking of the body pose. This includes arms, legs, torso and head positions, but usually excludes the hand pose, eye gaze or facial expressions. A variety of sensor based systems have been developed for this application. They are used mainly in commercial animation

applications. The academic research in this area has been focused on visual solutions which removed the need for artificial markers attached to the person's limbs.

Hand tracking

Hand tracking is probably the most difficult problem in human-machine interfaces. Although the skin colour detection method provides a reliable segmentation of the hand [Zhu et al. 2000] [Imagawa et al. 1998] [Yang and Ahuja 1998], the recovery of the hand pose is hampered by the uniform colour of the hands resulting in low contrast features, the partial occlusions caused by many hand poses, and the high bandwidth of hand motions.

However, for many applications it is not necessary to recover the precise pose of the hand. One important application of hand tracking is the recognition of spatio-temporal hand gestures [Black and Jepson 1998] [Marcel et al. 2000]. Spatio-temporal gestures also play an important role in the recognition of sign language [Yang and Ahuja 1998]. Deictic (pointing) gestures are also of interest in many real-world interface systems [Wu et al. 2000], and to command a robot [Sato and Sakane 2000]. To recognise deictic hand gestures it is sufficient to derive the 3D orientation of the hand, but not necessarily the position of each finger. Hand tracking systems have also been used as interfaces to desktop computer as a replacement for the mouse [Maggioni 1993] [Maggioni 1995]. The setup of such systems requires the mounting of the camera above the hand (for example on the ceiling) and the hand is assumed to be approximately parallel to the image plane. In such a restricted case the 2D shape analysis of the skin coloured region in the image is sufficient to recognise a range of different gestures [Triesch and von der Malsburg 1996] [Segen and Kumar 1998] [Sato et al. 2000]. Eigenspaces have also been used to classify the hand pose from 2D edge images [Moghaddam and Pentland 1997].

The reconstruction of the precise 3D configuration of the hand from 2D images is much more difficult. The system described by [Heap and Hogg 1996] used a point distribution model to derive the 12 unknowns required to define the pose of their 12 DOF model. A two-step iterative approach was reported by [Wu and Huang 1999] which derives the palm orientation first and then recovers the state of the individual fingers in the second step using a kinematic model of the hand. An analysis-through-synthesis approach has been reported in [Shimada et al. 1995] which tried to determine the 3D pose by altering the configuration of a 3D model until it best fits the observed 2D silhouette.

Human body tracking

The third mayor thrust in visual human-machine interfaces is concerned with full body pose estimation systems. This topic has been an active research areas since the early 1980's [Hogg 1983] since it has obvious applications in surveillance and char-

acter animation area. Given the fact that with comparatively simple background subtraction and shape tracking methods such as snakes [Baumberg and Hogg 1993] [Baumberg and Hogg 1994] reasonable results can be achieved with few computational resources. Todays systems can be divided into three classes according to the resolution of estimation of the body pose.

Systems of the first class only detect the presence of one or more persons and extract the silhouette. Systems based on shape similarities and snakes [Geiger and Liu 1996], optical flow [Iketani et al. 1998], Hidden Markov Models and Kalman filters [Rigoll et al. 2000] have been reported in this area. Various extensions have been proposed to these schemes including multi-camera multi-person tracking [Utsumi et al. 1998], tracking of interacting people with resolution of occlusion [McKenna et al. 2000] and the detection of people who are carrying objects [Haritaoglu et al. 1999].

The second class of systems uses 2D models of the human body. The models consist of multiple parts for the various body parts and are matched with the observed silhouette. The "cardboard people" [Ju et al. 1996] are an extension of the facial expression detection system based on optical flow [Black and Yacoob 1995]. The model of the motion consists of ten rectangular parts which are linked by rotary joints. The model motion is adapted to match the observed optical flow. Similar models have been used by [Hu et al. 2000], [Cham and Rehg 1999] and [Ioffe and Forsyth 1999]. The "Pfinder" system uses a "blob" (ellipsoid) model to approximate the silhouette which is adapted online to allow to track the torso and the limbs of the person [Wren et al. 1996] [Wren et al. 1997]. The "Ghost" [Haritaoglu et al. 1998b] and the W^4 system [Haritaoglu et al. 1998a] form a body tracking and body part labelling system based on the silhouette. The system used a number of heuristics to match a six-part model with the silhouette and then tracks the body parts according to their appearance in the image.

Similar to the hand tracking, the recovery of the precise pose of the human body is difficult due to the dexterity of the body and the occlusions that occur in many body poses. For desktop interfaces the complexity of the task can be reduced by the tracking of the upper torso, the arms and hands and the head [Wren and Pentland 1998], [Iwai et al. 1999]. For complete body tracking, more general methods are required. The "Spfinder" system is an extension of Pfinder which uses stereo cameras and 3D blob models [Azarbayejani et al. 1996a], [Azarbayejani et al. 1996b]. Using stereo vision and a full body kinematic model consisting of ten moving parts, more detailed body tracking was reported by [Munkelt et al. 1998]. A similar approach was reported by [Zheng and Suezaki 1998]. Using a monocular camera image the user must manually fit the 3D model to key frames. The system then interpolates the motion of the model in the intermediate frames. An analysis-through-synthesis approach was reported in [Delamarre and Faugeras 1999]. The system manipulates a 3D model until a best fit with the silhouette in the images occurs. A system purely

based on fitting silhouettes is described by [Brand 1999].

Similar to the hand tracking the human body pose tracking problem is still unsolved. Systems can only derive approximate estimates with only a few assumptions and minimal apriori knowledge [Azarbayejani et al. 1996a], [Wren et al. 1997]. If more accurate approximations are required then more apriori knowledge is required [Munkelt et al. 1998]. The other similarity to hand tracking systems is the wide use of kinematic models which provide constraints about the possible motions of the person in addition to the visual information derived from the images. Such additional constraints are particularly important for analysis-by-synthesis systems which have to deal with a high-dimensional search space due to the dexterity of both hands [Shimada et al. 1995] and body [Delamarre and Faugeras 1999].

2.2.8 Summary

In the previous section the various aspects of human-centred interfaces were discussed. Attention was given to face detection, facial feature tracking, 3D head pose estimation, gaze estimation and gesture recognition since these are the areas this thesis is concerned with. A wide range of related areas have also been described briefly to allow the reader to put the research in face and gaze tracking into a larger context.

The computer vision algorithms used for the various functions in visual human-machine interfaces have been addressed in the sections describing the respective functional blocks.

Of all the past work referenced in this chapter only one system covers more than one of the three major areas of human-machine-interaction [Hongo et al. 2000]. All other systems are either dedicated head, hand or body tracking systems. Even within the group of systems concerned with facial interfaces most systems are tailored towards one function such as gesture recognition or gaze tracking. The area of visual human-machine interfaces currently lacks integration of the various aspects of visual interaction modes. The system presented in this thesis implements three major functions, 3D head pose tracking, 3D eye gaze estimation and facial gesture recognition which have not been tackled by other researchers. Our interface is used to control a real-world system, a human-friendly robot manipulator. The manipulator is controlled by gaze and gestures to pick up specific objects and hand them over to the user.

2.3 Human-friendly robots

Since *human-friendly robots* is a relatively new research area many problems are still open. As stated in Chapter 1, the idea behind human-friendly robots is to create manipulator systems that are able to complete tasks in close collaboration with

humans, in particular in physical proximity of humans and in environments used by and designed for humans. This is radically different to traditional robot applications and gives rise to two mayor issues which must be addressed:

- The safety of the humans in the environment of the robot: If people and robots are to share the same workspace, constraints must be placed upon the robots actions to prevent injuries occurring to the user.
- To create interfaces that make human-friendly robots usable: Existing interfaces are too cumbersome to allow intuitive and efficient interaction between robots and people.

Both both of these issues have attracted minor attention, however, the interest is beginning to grow. Section 2.3.1 reviews developments in the area of safety strategies for human-friendly robots and the issues related to the physical interaction between robots and humans.

Visual interfaces are also well suited to natural interaction between robots and humans. Section 2.4 describes past research work in this area.

2.3.1 Safety issues

The safety of humans in the environment of the robot is of paramount importance and this issue must be addressed in a robotics system that is designed for physical interaction with humans. The various approaches to achieve safety in traditional robotic systems and interaction systems are illustrated in Figure 2.2.

Separation

The traditional approach used in industrial environments is to separate robots and humans whenever the robot is active. This is achieved by physically isolating the area in which the robot is located using barriers. In this case robots and humans work in separate and designated areas. The human world and the robot world are completely separated.

Other safe-guarding methods include laser fencing and visual acoustic signals to indicate the condition of the robot [Engelberger 1980], [Dhillon and Anude 1993]. Safety measures are tailored to industrial applications and are not suitable to such human-friendly robot systems.

Sensors

A variety of sensors have been developed to prevent or detect collisions in robot manipulators. The most common way to detect collisions is to use a combination of force and torque sensors mounted in the wrist of the robot. These sensors are well suited to detect collisions, however, they are restricted to detecting the collisions of

the hand only and are therefore not suitable as general safety devices. A standard way to achieve sensitivity on all surfaces of a robot is to use tactile sensors. Tactile sensors can be incorporated into the path planning process to create passable paths in unstructured environments [Cheung and Lumelsky 1988]. Various systems with tactile sensors have been proposed [Hills 1982], [Howe and Cutkosky 1993]. A common problem with tactile sensors is that it is often difficult to mount the sensors on the moving parts of the joints [Lozano-Pérez 1985]. Also, tactile sensors can *not* prevent collisions but only *detect* them *after* they occur.

Non-contact proximity sensors can be used to detect collisions before they occur. Infrared and ultrasonic sensors have been used to detect objects in the close proximity of a robot manipulator [Graham and Millard 1991], [Lumelsky and Cheung 1993]. Capacitance based systems allow a good coverage of the robot with fewer sensors [Novak and Feddema 1994], [Vranish and Chauhan 1990]. A variety of computer vision approaches towards collision avoidance in robot manipulator systems have been reported [Matthies and Elfes 1988]. However, the reliability of vision systems is much lower than that of other sensor classes.

In general the reliability of collision avoidance systems decreases with their detection range. Vision and ultrasonic system have reasonable reach, but their reliability is low. Systems with higher reliability such as tactile and capacitance based systems can only detect objects in the close proximity and therefore restrict the robot to slow motions and strong braking forces are required to avoid collisions.

In the context of human-friendly robots collision detection systems are a necessity and collision avoidance systems are desirable. However, no sensor system exists that can guarantee collision avoidance with sufficient reliability without imposing strong restrictions on the speed of motion of the robot. This is why there is a tendency to accept the possibility of a collision with a human and then aim to ensure that collisions are harmless. Such non-sensor based solutions increase a robot's safety while not depending on sensors and also reducing the cost, complexity and weight of the system [Morita et al. 1999] [Lim and Tanie 2000].

Mechanical safety measures

The most obvious mechanical safety measure is to use padding, it is simple and effective. Padding of the robot arm with an deformable substance reduces the impact forces markedly [Yamada 1997]. Padded systems ensure safe operation in human-robot coexistence applications [Suita et al. 1995], [Morita et al. 1999]. The most common problem with padding is covering the joints and the hand of the manipulator such that no gaps appear in the padding and the robot is not restricted in its mobility. In general padding is advantageous since it decreases the risk of injury in a collision and should therefore be considered wherever possible.

Another important mechanical feature of human-friendly robots are enclosed sur-

faces that prevent the user from getting body parts being caught between moving parts of the robot. Industrial robots do not fulfil this criteria in most cases. To increase rigidity robot arms are often constructed from several parts that move with respect to each other. A good example for a manipulator with enclosed surfaces are the Robotic Research arms and the Barrett Whole Arm Manipulator[6] (WAM).

The weight of the manipulator arm is also of importance. A heavy arm produces much higher impact forces at the same speed than a light weight robot. To allow a human-friendly robot to move at reasonable speeds the arm needs to be light. The choice of light materials such as aluminium, magnesium or even plastics where possible is preferable. Such materials increase the power-to-weight ratio of the robot and therefore, heavier objects can also be manipulated.

Compliance

Compliance protects the user from excessive forces during contact with the arm and allows the user to override the actions of the robot to a certain extent. This is physiologically important since a rigidly controlled arm can give users the impression that they are not in control. Physical compliance can be achieved in two ways, either by making the robot mechanically compliant through flexible elements in its structure (*passive compliance*) or by a control method that provides compliance (*active compliance*).

Passive compliance can be achieved by adding compliant elements in the joints. A mechanical compliance adjustment system based on springs mounted inside the joints has been reported in a human-robot interaction system [Morita et al. 1998], [Morita et al. 1999]. A similar compliant effect can be achieved by using electro-rheological (ER) fluid in clutches mounted between the actuator and the joint. ER fluid changes is viscosity when a electric field is applied. Systems using ER clutches for human safety have been reported [Arai et al. 1998], [Sakaguchi et al. 2000]. A different approach towards human-safety through mechanical compliance is described by [Lim et al. 1999]. Instead of making the robot itself compliant, the robot is mounted on a spring and damper system that allows the whole manipulator to yield if a collision occurs. In addition, the base is set on coasters which gives the robot compliance in the horizontal direction [Lim and Tanie 1999] [Lim and Tanie 2000].

Two approaches have been proposed to achieve active compliance: force control and impedance control. Force control [Raibert and Craig 1981], [Craig 1986] is a widely used control method in robotics research to allow the precise application of forces by the robot. It provides a formalism to derive the joint torques required to produce a desired force and torque at a given point on the robot using the dynamic model of the manipulator. This formulation is valid only for situations in which the robot has already established contact. This methodology does not incorporate

[6]See also `http://www.barrett.com`

the means to control collision forces. The impedance control scheme proposed by [Hogan 1984], [Hogan 1985] was developed for industrial applications that require the control of response of the manipulator to external forces. One of the first applications that were described was automatic deburring. Again, the formulation was designed to control the resistance of the robot arm to external forces. Like force control this method was not designed to control collision forces [Hsu and Fu 2000].

As described by [Morita et al. 1999], safety considerations can be split into two distinct phases, pre-collision safety and post-collision safety. Non-contact sensors and padding are mostly concerned with the pre-collision safety, allowing the robot to move freely without posing a threat to the user. Tactile sensors, passive and active compliance are concerned with the post-collision safety, limiting the forces the robot can apply to a person in contact with the robot. The robot experiments described by [Lim and Tanie 2000] show that compliance only reduces the post-collision safety, while the initial collision force remains unchanged, and therefore the pre-collision safety is not increased.

Collision force control

Active compliance control methods are not suitable (and were not intended) to provide the means to control and limit the collision force of the robot with people. Padding and proximity sensors can reduce the collision forces and increase the pre-collision safety, but they do not provide a quantifiable limitation either. To allow quantitative control and limitation of the collision force a pre-collision safety control paradigm is required.

A simple solution would be to limit the joint velocity of each joint. Such a limitation would provide a maximum potential impact force for static obstacles. However, the maximum impact force would probably be reached only in small subspace of configurations, and therefore, if the resulting maximum impact force is considered safe, the robot could move faster in other configurations without exceeding this limit. The method is only a crude approximation which wastes much of the potential performance of robot system. A similar approximation can be achieved by limiting the robot's kinetic energy [Li and Horowitz 1995]. However, this method also unnecessarily limits the performance of the robot.

The research reported in this thesis develops a control scheme that allows for the quantitative limitation of the collision forces of a robot. Such a quantitative pre-collision safety measure is essential for safe human-robot coexistence. The scheme consists of a control command filter which ensures that command signals complies to control constraints which guarantee the desired limitation. This architecture allows the combination of the proposed control scheme with *any* position or force control algorithm.

The issue of controlling the potential collision force of a robot should not be

confused with what is known as *impact control*, where the primary concern is the stability of the controller during the transition between free motion and the constraint motion after contact with the environment has been established. This subject has been addressed by many researchers during the past decade [Volpe and Khosla 1993], [Mills and Lokhurst 1993], [Hyde and Cutkosky 1994], [Xu et al. 1995], [Tarn et al. 1996], [Sarkar et al. 1998].

2.3.2 Physical interaction

The ability to physical interact with the environment, in particular a human or another robot, is considered to be a basic feature of human-friendly robots and the topic has attracted much interest in recent times [Khatib 1999], [Wakita et al. 1998], [Brooks et al. 1998]. A simple example of physical interaction between people that is experienced on a daily basis is the passing of an object from one person to another. A person passing an object will keep hold of the object until they can feel the receiving person's grip on the object. Only when the receiving person is applying forces to the object which indicates that the receiving person is in control of the object, the passing person releases the object. [Wakita et al. 1998] describe a system which simulates this behaviour using a force/torque wrist sensor.

Another example of physical interaction that has attracted considerable attention is cooperative carrying of objects [Khatib 1999], [Khatib et al. 1997]. This system allows the cooperative manipulation of an object by multiple mobile manipulators and humans. The idea is to let the robot(s) support the main weight of the object, while the person applies small steering forces to the object to direct the motion of the robot. This application of physical interaction requires the robot to be able to sense and to distinguish the small steering forces from the strong gravitational and inertial forces exerted by the object. Force/torque wrist sensors were used for this purpose. Other cooperative carrying systems have been reported [Kosuge et al. 1998], [Kosuge et al. 2000], [Hirata and Kosuge 2000]. Non-mobile systems have also been reported [Kim and Zhang 1998], [Luh and Hu 1999], [Xu and Zheng 1999].

The development of cooperative robot systems has focused on the control aspect and they all depend on force/torque sensors. The robot is largely insensitive to external forces away from the wrist which could be the result of a human attempting to move the robot or of a collision with an object or a person. No consideration has been given to the design of the control algorithms and architectures of these systems to human safety, collision detection and limitation of the effects of possible collisions with the robot other than the wrist.

The research presented in this thesis focuses on the development of a control scheme which limits the potential impact force of the manipulator in a collision. Such a quantitative safety guarantee which is not restricted to the robot's hand only but is valid for the whole manipulator arm is a crucial building block for human-

friendly robots, in particular for their application in the real world. The scheme proposed in this thesis also allows the detection of external forces acting on the robot at any point. This is important for the detection of collisions with obstacles and the identifying forces applied by the operator at any point of the robot. Using this external force detection scheme the exchange of objects between a robot and a human operator are possible. The robot's actions such as opening and closing the hand are triggered by forces applied by the operator through the object.

2.4 Visual human-robot interfaces

Besides the safety and control aspects the interface between humans and robots is a crucial aspect of human-friendly robots. The robot must be able to communicate in a human-like fashion, and therefore mimic human perception to a certain extent. Vision is the most important sensor for humans and as stated earlier, humans exchange much information via this communication channel using gestures. To date few visual human-robot interfaces have been reported in the literature.

Hand gesture interfaces

The *GripSee* system reported by [Triesch and von der Malsburg 1998] and [Becker et al. 1999] consists of a stereo camera system and a 7DOF robot manipulator set up in a desktop environment. Various objects are placed on the table and the robot can be command by hand gestures to pick up an object and put it into another place on the table. The system expects two hand gestures for each operation, the first to indicate the object and the grasping direction and second indicating the destination location. This coding system which requires the user to memorise various hand gesture commands is unnatural and therefore contravenes the original purpose of visual interfaces which was to allow natural interaction.

Another hand gesture interface, called the Space-Mouse, was reported by [Kurpjuhn et al. 1999]. This system allows a robot manipulator to be steered in Cartesian space by performing different hand poses in front of a camera. The system only allows motion control of the robot. No direct interaction is allowed, the operator controlling the robot could be in another room.

A similar visual control system is reported by [Terashima and Sakane 1999]. The system uses an active desktop to control a manipulator. A 3D model of the robot and it's environment and various controls are projected onto the table surface. A hand tracking system is employed to select which controls are activated and which objects are selected. The active desktop system is similar to systems described by [Wellner 1993] and [Maggioni and Wirtz 1991]. This system can also be regarded as a telerobotic system with visual user interface.

Body gesture interfaces

A visual interaction system for a humanoid robot was presented by [Cheng and Kuniyoshi 2000]. The system tracks the face and hands of the user with a stereo camera system and derives the approximate upper body pose. The humanoid robot imitates the motions of the user. The integration of the vision system within the structure of the robot closes this gap between the robot and the command post. Therefore, the operator actually interacts with the robot. However, the interaction is restricted to simple arm motions which do not provide the dexterity required to manipulate objects. At the current stage of development the system does not allow physical interaction such as the passing of objects between the robot and the user.

A gesture interface for mobile robots was reported by [Waldherr et al. 2000]. This system recognises four different arm motion gestures which allow the user to control the actions of a mobile robot. The small gesture repertoire is sufficient to command the robot to follow the user and to stop. More complex commands such as "go to place X" are not possible. Similar to the humanoid robot described above, the human-robot interaction is restricted to giving visual commands and observation of the robot's actions.

Desktop assistant systems

[Hayakawa et al. 2000] described a robot assistant system in a desktop environment. Similar to GripSee [Triesch and von der Malsburg 1998] it uses a stereo camera system and a manipulator. The system learns to associate visual observations about the assembly status with explicit commands from the user. After the training the system automatically issues the actions appropriate to aid assembly based on the visual observations. The basic idea is not to have a visual interface between the operator and the robot but rather between the assembly and the robot. The operator is unable to issue certain robot commands via this visual interface or physically interact with the robot.

2.4.1 Summary

A small number of visual interaction systems have been developed for human-robot interaction. However, these systems like the active desktop control [Wellner 1993] and the SpaceMouse [Kurpjuhn et al. 1999] are merely remote control interfaces. The user and the robot are not sharing the same environment and therefore are not able to refer to the same physical entities in the environment. The GripSee system [Triesch and von der Malsburg 1998] and the desktop assistant robot [Hayakawa et al. 2000] are examples of system that achieve interaction in a shared physical environment. This is the direction of development which will ultimately lead to machines that allow the cooperative completion of physical tasks between a human and a robot.

The research reported in this thesis incorporates a visual interface which uses the eye gaze of the user and facial gestures such as nodding and shaking the head. Both the eye gaze and facial gestures are natural human expression modes and therefore require no additional training. The system can pick up designated objects and hand them over to the operator or put objects down in places determined by the user's gaze point. Since the gestures are natural the commands required for such operations can be explained to a person in a few minutes. The manipulator itself is controlled in a way such that the safety of the operator can be guaranteed. The system also allows physical interaction with the person as previously described. It therefore incorporates the two crucial requirements for human-friendly robots, a natural interface and the ability to physically interact with a person in a safe way.

2.5 Summary

This chapter presented an overview on the important visual human-machine interface modes, algorithms and methods used for the various sub-functionalities of interfaces. The review focused in particular on face-centred interfaces, 3D head pose estimation, eye gaze estimation and facial gesture recognition. Many different functions and interaction modes have been implemented in the various systems. However, the research area suffers from a lack of projects which have integrated the various results and deployed them in real systems, thereby showing the practicality and efficiency of the methods.

The area of human-friendly robots is relatively new and few results have been published. The related area of cooperative object handling has received much more attention and a number of these systems have been developed. The focus of the work in these systems has been the control of the robot while human safety issues have only been given scant consideration. These systems have no interaction modes other than physical interaction through forces and torques applied to the wrist. The few human-friendly robots with visual interfaces that allow the issuing of commands through gestures again do not provide any other interaction modes such as physical interaction. Again, human safety is largely ignored.

The popular safety strategies for human-friendly robots are mostly mechanical solutions such as padding and passive compliance. Control strategies such as compliance control have been used in a number of systems. However, since these control strategies have not been developed specifically with human-robot coexistence in mind, none of the systems limit the collision forces in the case of contact with a human. This crucial safety characteristic must be implemented in human-friendly robots.

The research work described in this thesis incorporates a vision system that is able to detect the 3D head pose of the user, the 3D gaze direction as well as facial gestures. The visual interface is used to control a human-friendly robot by gestures

to allow a person to pick up objects. The robot is controlled with a novel control method that limits impact forces. The method also allows the detection of collisions at any point of the robot's surface. The robot is able to physically interact with the user, without relying on force/torque sensors in the wrist.

Chapter 3

Face Tracking

Chapter 2 presented a review of methods for tracking faces and facial features and to derive the head pose. A visual face tracking and 3D head pose estimation system is the prerequisite for the tracking of the 3D eye gaze and the recognition of facial gestures. Both gaze direction and facial gestures are natural communication mediums which are suitable for non-expert users to control machines and appliances.

This chapter describes the visual face tracking system based on facial feature localisation which has been developed in this research. The system determines both the 3D head pose and the 3D gaze direction in real-time from a monocular video stream. The system requires only a single external camera and the user is not required to wear any specialised gear. Moreover, the system works with completely natural faces and does not require visual markers or high contrast lipstick [Saulnier et al. 1995].

The real-time capability of the system is achieved by the application of dedicated vision hardware which uses Sum of Absolute Differences (SAD) correlation between template bitmaps and the live images. Since the detection and tracking of individual facial features with this method is unreliable, the system uses a probabilistic framework with Kalman filters and a number of heuristic algorithms to integrate the raw correlation results to yield a reliable and robust system.

The 3D head pose is calculated using the alignment algorithm proposed by [Huttenlocher and Ullman 1990]. The algorithm uses a closed form solution for the elements of the rotation matrix and the translation vector of a feature triplets and is therefore efficient and suitable for real-time operation. However, a detailed analysis presented in this chapter shows that the algorithm has two major drawbacks. The solution has a high systematic error as a result of the differences between the assumed affine projection and the perspective projection of real cameras. The output rotation matrix is only a proper rotation matrix if the feature's image coordinates are a result of a precise affine projection.

Extensions to the original algorithm are presented which reduce the systematic error and guarantee the output of a proper rotation matrix. These extensions are

evaluated in case study with a geometric setup similar to the face tracking application. The study showed that the new algorithm reduces the systematic error up to 75% for the rotation and 90% for the translation along the z-axis.

The proposed algorithms have been tested on real image data. To achieve a precise and realistic evaluation of the system a mannequin robot was used. The experiments showed that the system can acquire robust and accurate 3D head pose estimates over a wide range of rotations.

Section 3.1 gives an overview of the architecture of the system; feature tracking, measurement integration and 3D pose estimation. Section 3.2 describes the vision hardware, the lowest of the three layers. Section 3.3 describes the middle layer which consists of a network of Kalman filters and a number of heuristic algorithms. The purpose of the middle layer is to integrate the unreliable correlation results and exploit the known geometric constraints between the facial features. The output of the middle layer are robust estimates of the feature's positions. Section 4 describes the algorithms used in the top layer, the 3D head pose estimation. Section 4.5 presents the experimental results with real image data from the mannequin robot. Section 3.5 discusses the insights gained from the study of the robust tracking and 3D pose estimation problem in monocular image streams.

3.1 System Architecture

The system consists of three main layers shown in Figure 3.1. At the lowest level the vision hardware performs bitmap correlation. The results are correlation values for selected feature positions.

The results are processed by the second layer which utilises a 2D model and takes the measured 3D head pose into account to merge the information to produce robust estimates of the feature image positions. This layer is implemented using a network of Kalman filters which is described in detail in Section 3.3. The estimated feature positions determine the location of the hardware search windows for the next image frame.

The 2D feature positions are used in the third layer to determine the 3D pose of the head. The system derives the 3D head pose from multiple feature triplets to allow robust estimates using the frontal and side views of the head. The head pose, motion and feature appearance parameters are used by other modules such as the gesture recognition and eye gaze detection module. The feature points of the 3D model are projected back into the image plane to adapt the constraints of the 2D model in the second layer. All three layers are synchronised by the frame clock and execute at 30Hz.

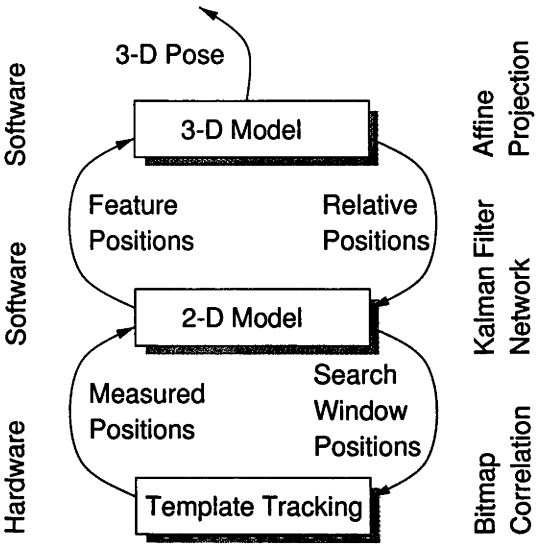


Figure 3.1: Structure of the tracking system

3.2 Layer 1: Template correlation hardware

The lowest layer consists of a real-time vision system implemented in hardware. When the research project commenced the Fujitsu Colour Tracking Vision (CTRV) vision hardware provided outstanding vision processing performance. Most of the research was done using a Motorola 68030/25MHz host CPU. The 68030 was upgraded to a Pentium/200MHz at the end of the development. The limited computational resources of the host CPU influenced many of the design decisions and enforced a minimal computation cost requirement for all algorithms.

The CTRV system was developed at Tokyo University [Inoue et al. 1993] and later commercialised and manufactured by Fujitsu. The system is an image correlation system consisting of a colour real-time NTSC digitiser, a correlation device, 3 image buffers and a NTSC output. The system is available as a VME or PCI board. Both versions have been used at different stages of the research. The video signal is digitised at a resolution of 640×480 pixels and stored in one of the image buffers. While the next video frame is being digitised into another image buffer the correlation device performs the desired image correlations. The correlation kernels are stored in a dedicated image buffer. The kernel is 8×8 pixels in size, however, oversampling factors of up to 7 can be specified to enlarge the image area, but not the number of pixels correlated. The correlation function is the sum of absolute differences (SAD) as shown in Equation 3.1.

$$D(s_x, s_y) = \sum_{x=0}^{7} \sum_{y=0}^{7} \mid i_t(x \cdot m_x, y \cdot m_y) - i_f(x \cdot m_x + s_x, y \cdot m_y + s_y) \mid \qquad (3.1)$$

where (s_x, s_y) are the image coordinates of the correlation and $i_t(x, y)$ is the intensity

of the correlation kernel and $i_f(x, y)$ is the intensity of the pixel in the previous digitised frame.

The system is able to track features as they move across the image by calculating the distortion D between the template and the live image in a grid of 8×8 positions in the image moving the search window over the live image horizontally and vertically by the oversampling factors m_x and m_y in between correlations. The lowest distortion value in the array indicates where the template fits the image best. By calculating the distortion array in consecutive frames features can be tracked as they move across the image. The system is able to track 600 features at video frame rate.

The image intensity used for the correlation is one of the red, green or blue channels. If the system is used in colour mode it calculates the distortion for each RGB channel separately. The sum of the three distortion values for one image location is used as the colour distortion value as shown in Equation 3.2.

$$D_c(s_x, s_y) = D_r(s_x, s_y) + D_g(s_x, s_y) + D_b(s_x, s_y) \tag{3.2}$$

The performance of the system in the colour mode is reduced by a factor of 3, corresponding to the increase in computational complexity, and therefore only 200 colour features can be tracked at video frame rate.

The software library also provides the option of using bigger correlation kernels such as 16×16, 24×24 and 32×32 pixels and arbitrary sizes. This functionality is generated by using multiple 8×8 kernels, so the number of features that can be tracked at frame rate is reduced.

The good performance of the correlation device is a result of the simple SAD correlation function. A drawback of this design is the sensitivity of the correlation to changes in the illumination. If the intensity of the image changes the previously recorded templates will no longer match the features in the image and the correlation results generate unreliable feature positions. This problem often occurs in face tracking applications under inhomogeneous illumination. As the head rotates the features on different sides of the head are illuminated more or less and the feature tracking becomes unreliable.

Once the feature tracking fails and the tracking window no longer contains the feature (outside the 8×8 grid) the system is unable to recover the feature location, even if it appears again at the nominal intensity. This can also occur due to the temporary occlusions of a feature. Additional methods are required to realise robust facial feature tracking. The second layer implements methods which permit reliable and consistent feature localisation which is crucial for 3D head pose estimation.

3.3 Layer 2: Kalman filter network

The vision hardware offers the capability to develop high performance vision based systems that run in real-time. However, the basic template tracking paradigm is not robust enough for applications such as human face tracking.

One of the main problems with tracking the features of a face is the change in appearance of the features during motion which can disappear completely if the person turns his/her head sufficiently far. Two main classes of changes in the appearance of feature can occur; the change in brightness caused by inhomogeneous illumination, and the projective deformation of the feature. The deformations are caused by rotations of the head around the body axis and the axis passing through the ears which compress the projection horizontally and vertically respectively, and by tilting the head which rotates templates in the image plane.

The changes in appearance of the tracked features result in a worsening of the correlation result, particularly at the correct position. Therefore the risk increases that the template matches best at a position that does not correspond to the correct feature location. When the tracking window loses a feature the resulting measurements for the feature become unpredictable. In such situations the correlation value is significantly worse than during tracking the feature, and eventually the tracking window "finds" a location in the image that shows similarities with the original feature and the tracking window locks onto the incorrect feature. In such situations it is impossible to decide with a thresholding technique whether the tracking window is tracking the wrong feature or the higher distortion is caused by the deformation of the feature or a change in illumination.

In this situation apriori information about the geometric relationships between the facial features can be used to detect inconsistencies in the feature position pattern and to guide failed search windows back to their respective features. However, in critical situations such as large head rotations nearly all the tracking windows return increased correlation values. Therefore a geometry check can determine whether the configuration is right or wrong in most cases, but it is not possible to confidently distinguish between which windows are tracking the right feature and those which are not.

The solution proposed in my earlier work [Heinzmann 1996], [Heinzmann and Zelinsky 1997] was to integrate the results of the hardware tracking system with a geometric model of the face in a probabilistic way. The basic idea was to keep the system in a stable configuration with correct geometric relationships between the search windows and low distortions. This method has has been reused in this research and is described for completeness in Section 3.3.3. The major drawback of this method is the tendency to remain in local minima where the distortion of the geometric constraints is insufficient to trigger the relocation of falsely positioned feature trackers to correct feature locations. This problem is resolved in the research

presented in this thesis which augments the basic method with a number of probabilistic algorithms which are described in Section 3.3.4. Also the integration of the robust tracking layer with the 3D head pose estimation in the top layer supports the basic correlation result integration by providing precise 2D geometric relations between the individual facial features (see also Figure 3.1).

3.3.1 The basic idea

The geometric constraints between the facial features are modelled as 2D distance vectors. The constraints spawn a 2D network with features as nodes and constraint vectors as edges. The correlation results have to be merged with these geometric constraints to derive robust position estimates. In unproblematic situations when all features are matching well, it is sensible to allow the features to deform the network in reasonable ways since it is likely that the deformation is a result of imprecise 3D pose estimation than a loss of the features. However, if some of the features are matching badly it is probably because some of the search windows are not tracking the correct feature anymore. In this situation the system should relocate the search windows which have lost their features to a position derived from the well tracking features. In these critical situations usually all features have increased distortions and thresholding is not an adequate method to classify between tracking and lost windows.

The method proposed in my Master's research used Kalman filters to estimate the feature positions from the two data sources, the correlation results and the geometric constraints of the feature network. In effect, the Kalman filters arbitrated between the feature positions suggested by the image correlation and by the geometric model and derived estimates of the feature position while taking the correlation confidence of each feature and all neighbouring features and their geometric relationship into account.

The mathematical background of the Kalman filter can be found in numerous references and therefore only a short description of the functionality of the filter is presented. A good introduction to Kalman filters can be found in [Maybeck 1979]. The mathematical details of Kalman and related filters are described in [Bozic 1979].

3.3.2 The Kalman filter in a nutshell

The Kalman filter is a recursive linear estimator which is used in many real world applications to merge the measurements of sensors and a predicted state derived from a system model. It is used for the navigation of planes, missiles and mobile robots where uncertain measurements from sensors which observe beacons or landmarks are used to determine the position of the vehicle.

Figure 3.2 shows the computational cycle of the Kalman filter. In the first step a prediction of the current state vector $\mathbf{x}(t)$ is calculated by multiplying system

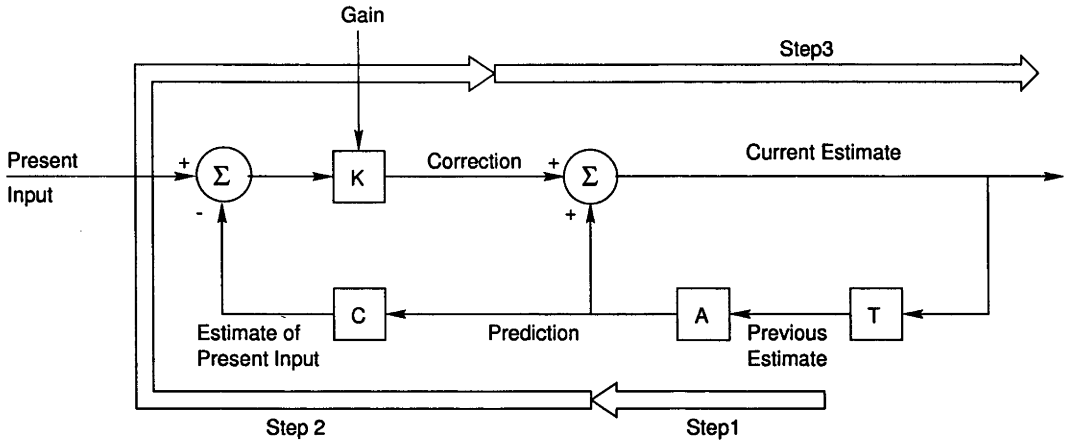


Figure 3.2: The computational cycle of the classical Kalman filter [Bozic79]

matrix A and the previous estimate $\mathbf{x}(t)$. The system matrix defines the relation of successive state vectors. In the second step a prediction for the input vector is calculated from the state prediction vector by multiplying it with the observation matrix C. The predicted input vector is subtracted from the measured input vector $\mathbf{y}(t)$ to obtain the error vector. The filter gain matrix $K(t)$ can then be calculated. In the final step the predicted state vector is corrected by a correction vector calculated from the gain matrix $K(t)$ and the error vector. The result is the new estimate for the state vector $\mathbf{x}(t)$. The relevant equations for the state estimation are given below.

Estimator:

$$\mathbf{x}(t) = A\mathbf{x}(t-1) + K(t)(\mathbf{y}(t) - CA\mathbf{x}(t-1)) \tag{3.3}$$

Filter gain:

$$K(t) = P_1(t)C^t(CP_1(t)C^t + R(t))^{-1} \tag{3.4}$$

Error covariance matrix:

$$P(t) = P_1(t) - K(t)CP_1(t) \tag{3.5}$$

$P_1(t)$ is the predicted covariance matrix,

$$P_1(t) = AP(t-1)A^t + Q(t-1) \tag{3.6}$$

$R(t)$ is the observation noise covariance matrix and $Q(t)$ is the system noise covariance matrix.

3.3.3 Kalman filter network

Kalman filters have been previously applied to the face tracking problem. The straightforward implementation using the Kalman filter is to compose the state vector

from the head pose, velocity and acceleration parameters [Azarbayejani et al. 1993]. However, this results in a high dimensional system (Azarbayejani used a 18 dimensional state vector). The computation required with such large matrices is far too high for real-time systems. Various simplifications can be used which lead to small errors but markedly reduce the calculation time. One way is to ignore the off diagonal elements of the covariance matrix if the correlation between the the variances is weak. The work reported in my Master's research used one Kalman filter for each tracked feature. This effectively ignored the off diagonal elements and the dimension of the state vector is variable (proportional to the number of features). This is similar to the method of the Variable State Dimension Filter (VSDF) described by [McLauchlan et al. 1994]. However, the adaption of the state transition function to changes in the feature network and inclusion of new features is simplified due to the distributed computation and the simple repetitive structure of each element in the network.

Each feature uses one Kalman filter to merge the feature position vector returned by the vision system with the position prediction calculated from the reference features. Figure 3.3 illustrates how the individual Kalman filters are interconnected to form the facial feature network. As an example the three filters for the features around the right eye are shown in the figure. The output from reference features is directly used as input to the position estimation of other dependent features. The position estimates based on the location of the reference features are merged (M) as a weighted sum according to the variance of the position of the respective reference feature.

Therefore, for each facial feature i the following calculations must be performed. First the position prediction $\mathbf{p}_m$ based on the position of the k reference features with indices r_{ij}, $j = 1 \ldots k$ and the 2D connection vectors $\mathbf{d}_{ij}(t)$ provided by the 3D head pose estimation layer are calculated. The variance $p(r_{ij}, t-1)$ of the error of the position estimation of each reference feature is used to weight the contribution of each reference feature to the position prediction $\mathbf{p}_m$.

$$\mathbf{p}_m(i,t) = \frac{\sum_{j=1}^{k} \frac{\mathbf{p}(r_{ij},t-1)+\mathbf{d}_{ij}(t)}{p(r_{ij},t-1)}}{\sum_{j=1}^{k} \frac{1}{p(r_{ij},t-1)}} + \frac{\mathbf{p}(i,t-1) - \mathbf{p}(i,t-2)}{1 + p(r_{ij},t-1)} \tag{3.7}$$

To accommodate fast motions of the face the prediction must consider the motion of the feature observed in the previous video frames. Under the assumption of constant velocity the second term adds a fraction of the motion observed in the previous two calculation cycles to the predicted position. It is scaled by the error variance $p(r_{ij}, t-1)$ of the previous position estimate to suppress motions based on low confidence measurements which can cause oscillations of the filter.

Because only the diagonal elements of the covariance matrix of $\mathbf{p}_m$ are considered and they are assumed to be equal, only one variance value pp_m represents the

confidence of the x- and y-component of $\mathbf{p}_m$.

$$pp_m(i,t) = p(i,t-1) + \frac{1}{\sum_{j=1}^{k} \frac{1}{p(r_{ij},t-1)}}$$

The position measurement is a result of the correlations of the various templates. The lowest correlation value $c_s(i,t)$ is selected by the selection module (S) of all correlations of all templates of feature i and e the corresponding position $\mathbf{p}_s(i,t)$ is selected as the currently measured position of the feature. This position $\mathbf{p}_s(i,t)$ is later used in the arbitration of the Kalman filter. The corresponding variance pp_s of the measurement (again the off-diagonal elements are ignored and the two diagonal elements of the covariance matrix are assumed to be equal) is calculated by an empirically determined formula which transforms the raw correlation result c_s into a suitable range for the filter.

$$pp_s(i,t) = 2500\frac{e^{5(c_s(i,t)^2-1)}}{1+e^{5(c_s(i,t)^2-1)}}$$

The maximum variance of 2500 corresponds to a standard deviation of 50 pixels which in turn corresponds to the search window size used by the vision hardware. Beyond this window the correlation result and the error variance are not related and therefore, the transformation levels off at that point.

Now the filter gain $k(i,t)$ is calculated in the standard way as

$$k(i,t) = \frac{pp_m(i,t)}{pp_m(i,t) + pp_s(i,t)}$$

Finally, the new position estimate $\mathbf{p}(i,t)$ for the facial feature and the error variance $p(i,t)$ which again represents the diagonal elements of the covariance matrix is calculated.

$$\mathbf{p}(i,t) = \mathbf{p}_m(i,t) + k(i,t)\left(\mathbf{p}_s(i,t) - \mathbf{p}_m(i,t)\right) \tag{3.8}$$

$$p(i,t) = (1 - k(i,t))\, pp_m(i,t) \tag{3.9}$$

Here the calculation loop closes and the new position estimates $\mathbf{p}(i,t)$ of all features i are forwarded to the 3D pose estimation layer. This higher layer then updates the model pose and calculates the new relative displacement vectors $\mathbf{d}_{ij}$ between the facial features i and j from the projection of the model into the image plane.

This setup allows deformations of the 2D feature net depending on the confidence of the localisation of the various features. The links to a feature that can be localised with a high confidence become stiffer and more difficult to deform while links to features with high correlation results become soft and can easily be deformed by the position measurements of a neighbouring feature. This confidence dependent

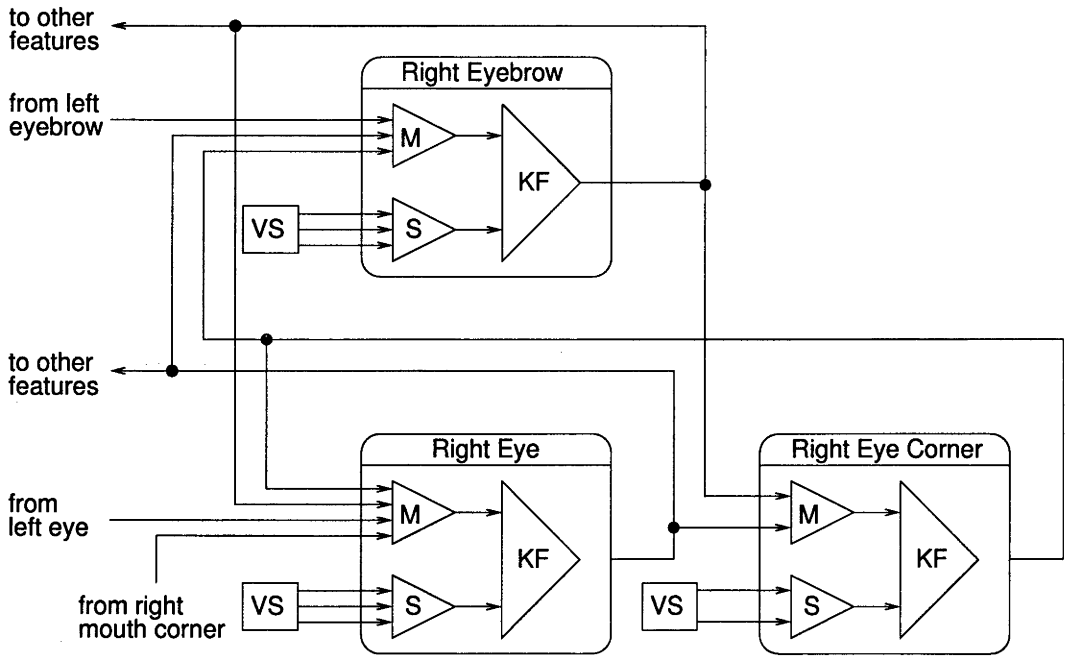


Figure 3.3: Part of the Kalman filter network

flexibility of the geometric relationships between the features is crucial during the recovery of tracking failures using the position detection of only a few facial features. Single high dimensional Kalman filter solutions, such as [Azarbayejani et al. 1993], do not provide this flexibility. Although they work well under optimal conditions, their ability to recover from tracking failures is poor because they require that all feature positions must be determined at the same time to reinitialise the filter state vector. They are not able to exploit the detection of only few facial features and then recover the position of the remaining features successively.

3.3.4 Improved Robustness

Section 3.3.3 described the feature network used to integrate the tracking results of individual features in a distributed way such that the positions of poorly tracking features are estimated by using well tracking features in a probabilistic way. The Kalman filter network provides good tracking and recovery performance under most conditions. However, the major drawback of the method is a tendency to remain in local minima. This effect can occur during the initial feature localisation after the system starts or when some of the feature positions were lost during tracking due to occlusions or deformations. Some of the feature trackers that are not tracking their respective facial features may "find" different facial features with a similar appearance as the correct feature. For example the eye corner and the eye brow and the line between the lips and the shadow at the bottom of the nose have similar appearances and the respective feature trackers can get "stuck" on the wrong feature. In particular if a falsely tracked feature is not far from the true feature the corrections through the

model estimate of the Kalman filter may not be strong enough to enforce a relocation of the feature tracker.

This effect is worsened by the fact that the vision hardware performs the correlation only within a search window which is double the size of the template. If the correct feature is outside this window the correlation results do not indicate the position of or direction to the correct feature. Therefore the limited size of the search window must be artificial enlarged by placing multiple search windows into different positions.

Besides the limited size of the search windows the application of rigid SAD template matching to the face tracking problem worsens the local minima effect of the Kalman filter network. This is because the correlation result is sensitive to changes in illumination and projective deformations and scaling of the features in the image it is not possible to make a confident decision whether a worse correlation value is the result of deformations of the correct feature or an indication that the feature tracker is in a local minima and the wrong feature is being tracked. This blurred distinction between correct and incorrect feature tracking makes thresholding impossible and was one of the motivations for the use of the Kalman filter network. This effect hinders the recovery of all features particularly when the conditions are not ideal. This is the case for example when the face is not viewed frontally or the illumination is inhomogeneous.

These problems have been solved in this research with a number of algorithms which aim to increase the robustness of the tracking and speed up the recovery of lost features, and in particularly ensure that the face tracking system is able to escape local minima. These improvements include the following algorithms and methods:

- Multiple templates for each feature organised in a probabilistic transition network
- Variance controlled allocation and localisation of area search windows
- Strain integrators in the feature links

3.3.5 Network of template

As stated in the previous section, the application of rigid and non-normalised template matching to face tracking results can result in poor correlation values due to deformations and brightness changes of the facial feature. Figure 3.4 shows the change of appearance in the right mouth corner caused when a person turns to look from left to right. Although the changes between two consecutive images are small, none of the highlighted templates provide good correlation results for all poses. The only solution to ensure that the correct feature is tracked robustly in various poses of the head is to adapt the correlation template to projective and brightness changes of the feature. However, the computational requirement increases linearly with the

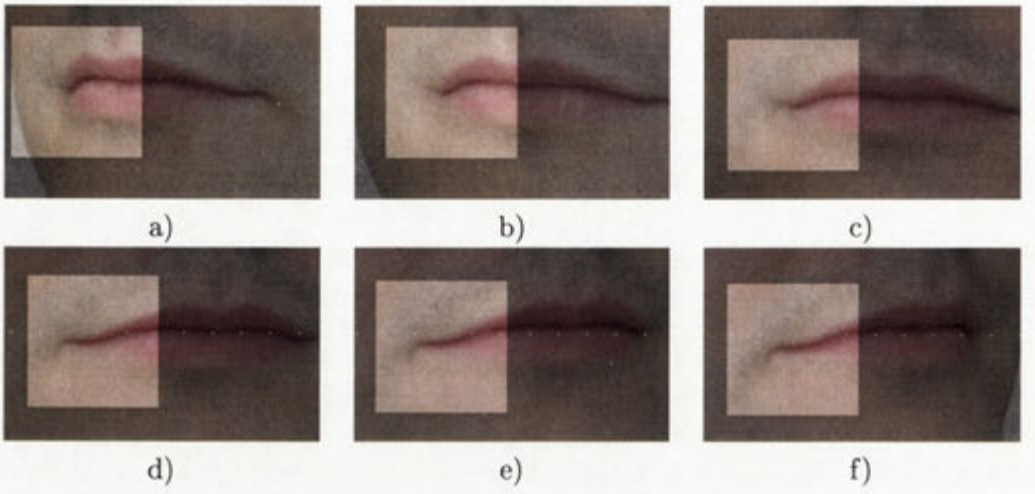


Figure 3.4: Change in appearance of the mouth area

number of correlation templates used for each feature. Since the vision hardware can only perform a limited number of correlations per frame, a subset of kernels must be selected for correlation in each frame and for each feature. The goal is to find a small set of correlation templates for each frame which maximises the chance that the best fitting template is an element of this set. Therefore a scheme is required which exploits the structure of the transitions of the appearance of the features. Assuming that each template represents the appearance of a feature in a particular configuration and the transitions between the configurations are subject to physical limitations this subset can be determined.

The basic idea of this method is that if the best matching template of all templates for a particular features is known, then a probability value can be assigned to every other template which indicates how likely a transition is from the current appearance to another appearance. These transition likelihood values depend on the similarity of the head pose under which the templates were recorded.

To exploit this structure in the transitions between the templates and to minimise the amount of templates that have to be correlated in each frame the transition likelihood between certain templates is stored in a network as shown in Figure 3.5. Each template is a node in the network and each link contains the likelihood of a transition from the source template to the destination template. The transition likelihood values in Figure 3.5 are examples only. The values used in the actual system are empirically set according to the head pose where the templates are recorded. During the setup of the system templates are recorded in many different known head orientations and the transition likelihoods from a template to the neighbouring templates are reciprocal to the angular distance between the templates. Neighbouring templates are the templates which are closest to the current template in a particular direction in the head pose/image brightness parameter space. The sum of the likelihood values of all transitions from a template equals 1.

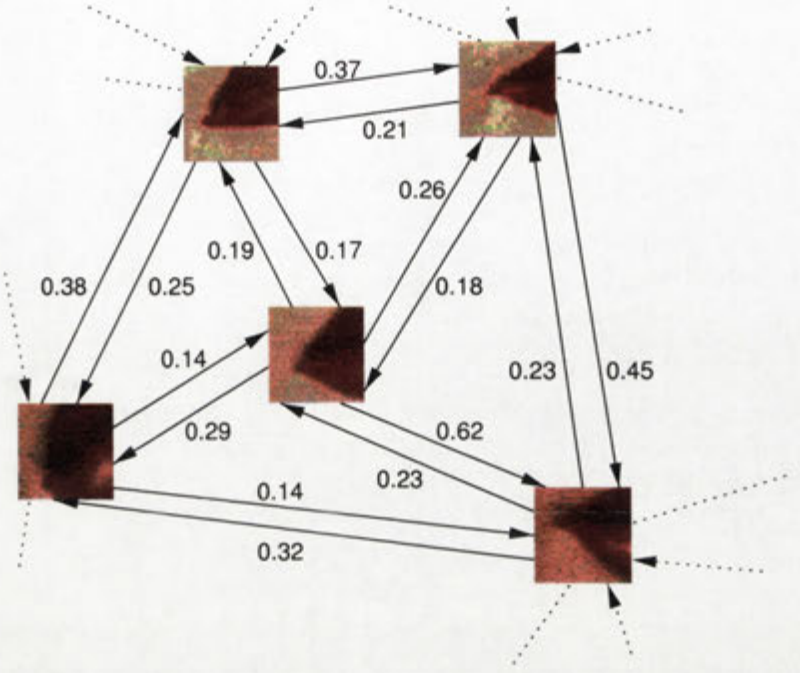


Figure 3.5: Correlation kernel network

Algorithm 3.1 Correlation kernel selection: `select (State, Size)`

```
if Size==0 then
  return ∅
end if
use Set Result(Template)=∅, Set Buffer(Template, Probability)= ∅
use BufferElement LastAdded=(CurrentTemplate, 1.0)
Result.add(CurrentTemplate)
Size–
while Size>0 do
  for all LastAdded.LinkedTemplate do
    if LinkedTemplate∉Result then
      if LinkedTemplate∉Buffer then
        Buffer.add(LinkedTemplate, Link.prob·LastAdd.prob)
      else if Buffer.prob(LinkedTemplate)<Link.prob·LastAdd.prob then
        Buffer.replace(LinkedTemplate, Link.prob·LastAdd.prob)
      end if
    end if
  end for
  LastAdded = Buffer.select_smallest_prob
  Result.add(LastAdded)
  Buffer.remove(LastAdded)
  Size–
end while
return Result
```

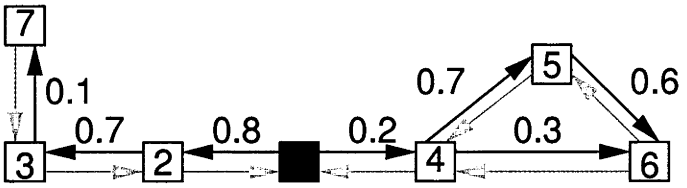


Figure 3.6: Example of the selection of templates

Step	Result	Buffer
1	{1}	{[2, 0.8], [4, 0.2]}
2	{1,2}	{[3, 0.56], [4, 0.2]}
3	{1,2,3}	{[4, 0.2], [7, 0.056]}
4	{1,2,3,4}	{[7, 0.056], [5, 0.14], [6, 0.06]}
5	{1,2,3,4,5}	{[7, 0.056], [6, 0.084]}
6	{1,2,3,4,5,6}	{[7, 0.056]}
7	{1,2,3,4,5,6,7}	{}

Table 3.1: Evaluation states

There are no links from one node to itself because the transition likelihood values refer only the likelihood of the change to a different appearance. The template which represents the current appearance is always in the set of templates to be correlated in each frame.

Algorithm 3.1 returns a set of templates of cardinality `Size` which includes the template which represents the current appearance of the feature and the `Size-1` templates with the highest product of transition likelihoods calculated from the current template. Figure 3.6 and Table 3.1 illustrate the operation of the algorithm with an example. Figure 3.6 shows part of a template network. The template t_1 represents the current appearance of the feature. Only the forward transitions are drawn in the figure for simplicity. In the first step the current template is added to the *Result* set and the two neighbouring templates t_2 and t_3 are added with their respective transition likelihood. Then the template t_2 is selected since 0.8>0.2 and added to the *Result* set and its neighbour template t_3 is added with the likelihood 0.8*0.7=0.56 and so on. Between steps 4 and 5 the likelihood of template t_6 increases from 0.06 to 0.084 because the path via template t_5 has a higher likelihood than the path via template t_4 only. This example assumes that the *Result* set is allowed to contain 7 templates. If a smaller set is desired by the system the algorithm terminates when the *Result* set has the correct cardinality.

If the links for each template are pre-sorted according to their transition likelihood, this breadth first search algorithm determines the set of features with the highest transition likelihoods in $O(n)$ where n=`Size` is the size of the set to be determined. The algorithm is well suited for a real-time system since the computation time is independent of the number of templates in the template network and the limitations of the vision hardware enforces a upper limit on the sizes of the template

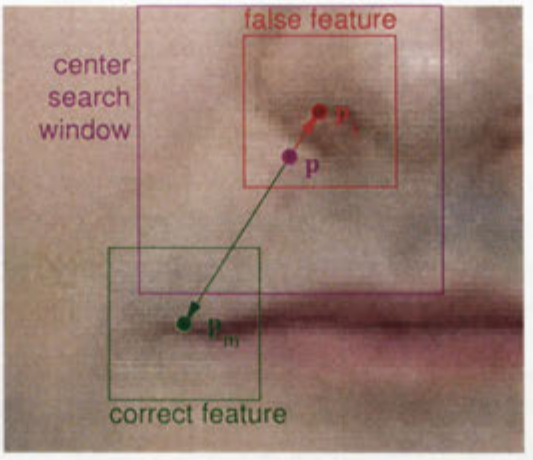


Figure 3.7: Local minima situation

sets.

3.3.6 Area search windows

The Kalman filter network provides an improvement of the tracking robustness when the tracking of some facial features fails. The position estimate provided by neighbouring features is sufficient in such cases to guide the lost feature trackers back to their respective facial feature. However, this algorithm depends on the prerequisite that only a few of the features have been lost. If all features are lost neither the vision system nor any neighbouring features are able to provide useful estimates for the location of the features. This problem usually occurs at system startup time or when complete tracking failures occur, for example when the person temporarily leaves the image and later returns, or when the head is rotated far enough that all features tracked by the system disappear. To kick-start the recovery process based on the Kalman filter network a minimal number of facial features have to be localised correctly by the system. After a complete tracking failure it is necessary to increase the fixed search window size of the vision hardware.

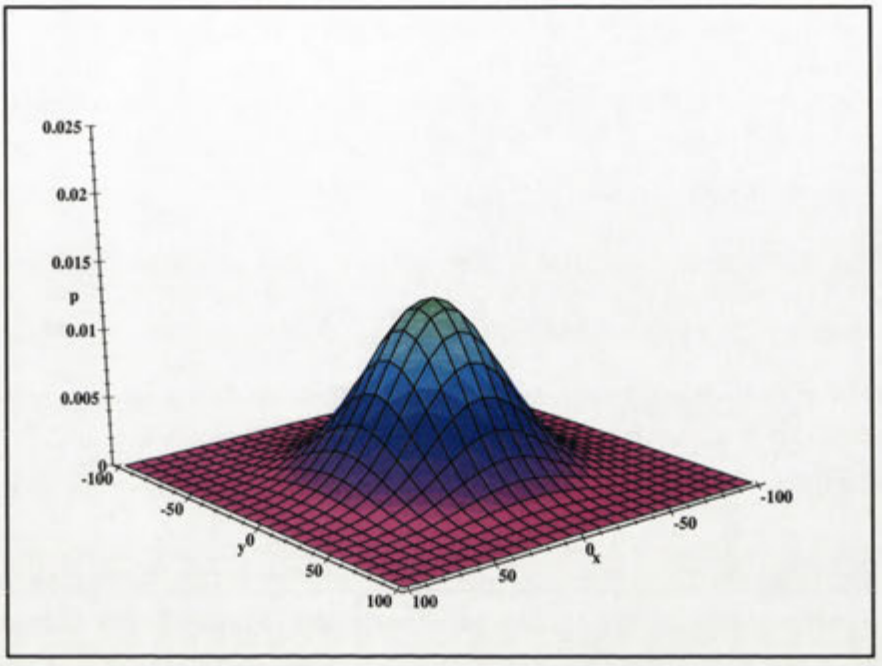
Another tracking situation with a different characteristic also requires the enlargement of the hardware search windows. As stated earlier lost feature tracker tend to get trapped at image features that have a similar appearance as the correct features. Because the correct facial feature is outside the hardware search window the measured position $\mathbf{p}_s(i,t)$ is consistently set on the false feature. The reasonably good correlation results of the false feature causes a high Kalman filter gain $k(i,t)$ and therefore, the estimated position $\mathbf{p}_m(i,t)$ offsets the measured position $\mathbf{p}_s(i,t)$ only slightly. If this offset is small enough and the false feature is still within the hardware search window the system becomes trapped in a local minima. Figure 3.7 illustrates this situation. In this tracking situation it would be advantageous to enlarge the hardware search window such that the correct feature is within this enlarged

search window. Unfortunately this tracking condition can not be reliably detected and therefore it is difficult to determine if and to what extent an enlargement of the search window is required.

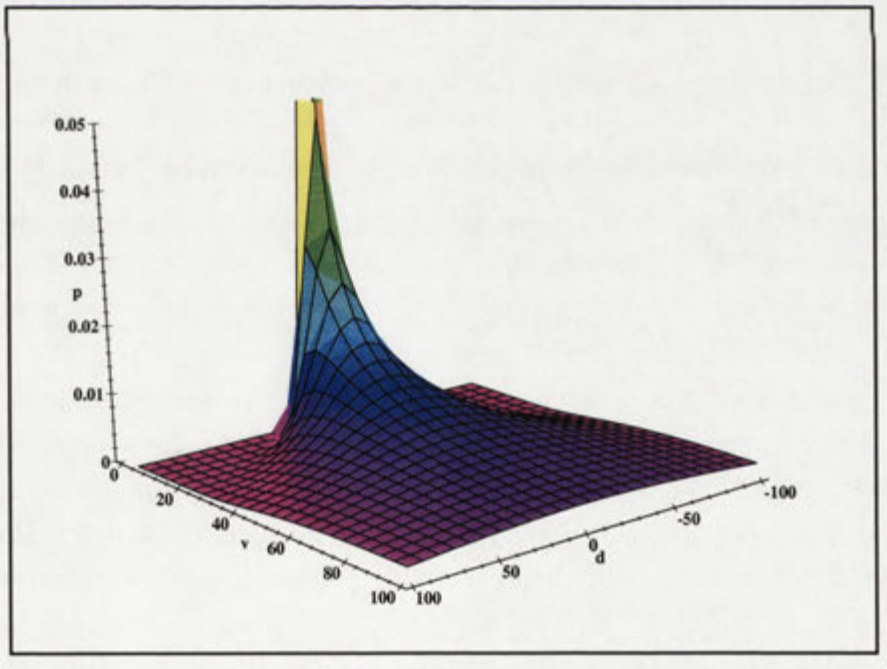
Similar to the selection of the optimal set of templates for a feature to adapt to changes in its appearance the area search for features is an optimal resource allocation problem. Searching the entire image for features with slightly worsened correlation results would solve the problem if unlimited computational resources are available. In reality this brute force method is not efficient because the position estimates $\mathbf{p}(i,t)$ often contain only moderate errors. Searching only the surroundings of the estimated position requires less resources, but may not be sufficient if a complete tracking failure has occurred.

The variance of the feature position derived by the Kalman filter is a good indicator to what extend the search window should be enlarged. If a complete tracking failure occurs the variances of all feature trackers is markedly increased indicating that no useful position estimate can be made and all image locations have similar probabilities to contain the facial feature. During recovery when some features are already found the variances decrease markedly for the identified features but also for the other features because useful position predictions $\mathbf{p}_m(i,t)$ can now be made. In this case it is sufficient to search only in a moderate radius around the position estimate $\mathbf{p}(i,t)$. If a feature tracker is trapped in a local minima and the correlation value and variance $pp(i,t)$ are increased only slightly it is sufficient to search the vicinity of the search window. If the difference between the position prediction $\mathbf{p}_m(i,t)$ and the measured position $\mathbf{p}_m(i,t)$ is large, the position estimate $\mathbf{p}(i,t)$ is sufficiently far from the false feature such that the false feature would no longer be within the hardware search window. The local minima situation would be resolved by the Kalman filter network.

The variance $pp(i,t)$ of the position estimate $\mathbf{p}(i,t)$ of feature i is used to determine a probabilistic distribution of additional hardware search windows for each feature i called *area search windows.* Inside the area search windows templates of the feature are correlated. The position of each area search window is calculated in polar coordinates with the current position estimate $\mathbf{p}(i,t)$ as the origin. A random angle α is determined from a uniform distribution. The distance d from the origin is calculated as the absolute value of a random number from a normal distribution with $\mu = 0$ and $\sigma^2 = pp(i,t)$. Through the choice of a normal distribution the area search windows are concentrated around the current position estimate $\mathbf{p}(i,t)$ while few area search windows are located a greater distances. The difference in the probability density decrease with increasing variance $pp(i,t)$. If $pp(i,t)$ is high, it indicates a complete tracking failure and the area search windows are distributed uniformly over the image. To avoid that area search windows overlapping the centre search window (this area has the highest probability density) half the search window size *sws* is added to d. For each area search window j of feature i the two random values α



a)



b)

Figure 3.8: Probability density functions for the distribution of area search windows

and d need to be determined.

$$\begin{aligned} \alpha(i,j,t) &= \text{uniform}\,[0° \ldots 360°] \\ d(i,j,t) &= |\text{normal}\,(0, pp(i,t))| + \frac{sws}{2} \end{aligned}$$

The search window size sws depends on the template size. In the current implementation the templates are 24×24 pixels in size with a resulting hardware search window size of 48×48 pixels. The position of the centre of the area search window in image coordinates is then

$$\mathbf{p}_{\text{sws}}(i,j,t) = (d\cos\alpha, d\sin\alpha)^t$$

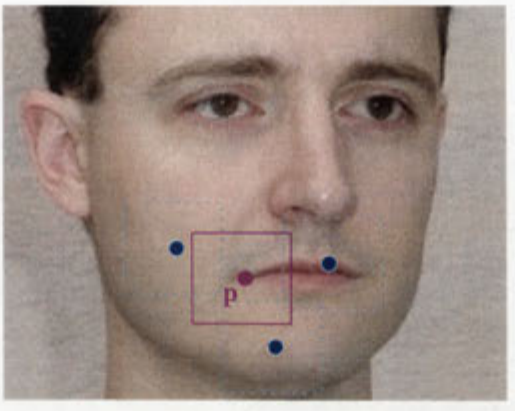
If the area search window position is outside the image boundaries, new random values for α and d are selected until a position within the image is found.

Figure 3.8a) shows the probability density plot for image locations around the estimated coordinates exemplary for $\sigma = 30$ before the overlap-adjustment. This is a typical value for a situation where a feature tracker is not tracking the correct feature but the model provides meaningful position predictions. For example, the probability density can result in a distribution of area search windows as shown in Figure 3.9b). The example distribution of area search windows in Figure 3.9a) is typical when the correct feature is tracked and the standard deviation is in the range of $\sigma = 5 \ldots 10$. Here the area search windows a concentrated around the estimated position $\mathbf{p}(i,t)$. In the case of a complete tracking failure the standard deviation is typically in the range of $\sigma = 50 \ldots 100$ and all positions in the image have almost the same probability to be selected for an area search window. Figure 3.9c) shows an example for the resulting distribution of area search windows.

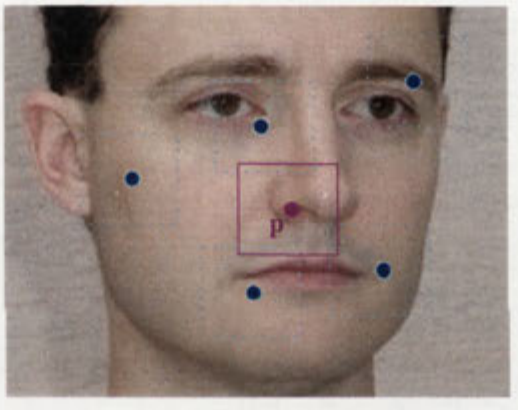
The probability density as a function of the distance d and the variance σ is plotted in Figure 3.8b). The plot shows how the probability density changes with increasing variances. For high variances the distribution of area search windows becomes almost even for the entire image.

If the result of an area search window indicates that the wrong feature has been tracked it also implies that the current template in the template network which is being used does not represent the appearance of the correct feature but the appearance of a false feature. Therefore the currently used template is insignificant for the selection of the templates to be used in the area search windows. Therefore, for each area search window a template of the respective feature is selected randomly where each template has the same probability. It is crucial not to restrict the set of templates to be used in the area search windows.

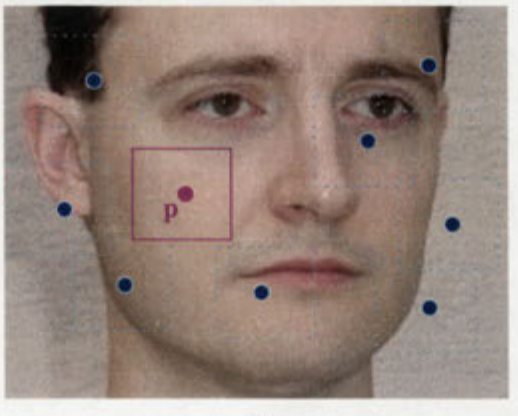
From all the correlation results of the area search windows for one feature tracker the best result c_a is selected and it is compared with the best result c_c from the templates correlated at the centre position $\mathbf{p}(i,t)$. Only if the result from the area search



a)



b)



c)

Figure 3.9: Distribution of area search windows

window is better than the result from all centre templates $c_a < c_c$ the position $\mathbf{p}_a$ where this result was achieved is used in the Kalman filter network. If the position is swapped for the area search window it is necessary to consider this in the following calculations of the Kalman filter network. Equation 3.8 and 3.9 are then replaced by

$$\mathbf{p}(i,t) = \mathbf{p}_s(i,t)$$

$$pp(i,t) = pp_s(i,t)$$

Also, the previous position $\mathbf{p}(i,t-1)$ is reset to $\mathbf{p}_s(i,t)$ to nullify the motion term in Equation 3.7 in the following iteration. This will account for the temporal discontinuity in $\mathbf{p}(i,t)$.

3.3.7 Search window allocation

The previous two sections both described ways to gain the most information from the limited computational resources. The allocation resources for centre templates and area search windows for each feature i is based on the variance $pp(i,t)$ of the position estimate.

The vision hardware is capable of correlating 200 templates within a standard size search window in each video frame. The face tracking system described in this thesis typically uses 19 facial features, and therefore, each feature tracker can allocate $n_w = 10$ search windows. The 10 windows must be divided into centre templates which detect changes in the appearance of the feature (see also Section 3.3.5) and area search windows which are required to recover from tracking failures.

Again, the variance $pp(i,t)$ is a good indicator which type of correlation are required. Large values in $pp(i,t)$ indicate that the feature tracker is likely to have lost its feature, and therefore it is reasonable to allocate more area search windows. Small variances indicate that the correct feature is tracked and the resources should be spent to ensure that the feature is tracked robustly and changes in its appearance are detected in time.

This simple but effective heuristic is expressed in the following allocation formulas. First the number of centre appearance change detection correlation windows n_c is determined and the remaining resources are allocated for n_a area search windows.

$$\begin{aligned} n_c &= \max\left(1, \min\left(n_c, \left\lfloor n_c\left(1 - 0.01\sqrt{pp(i,t)}\right)\right\rfloor\right)\right) \\ n_a &= n_w - n_c \end{aligned}$$

At least one correlation window is allocated for the centre correlation to track the feature which is currently the best guess of the system. Additionally, a number of correlation windows which depend on the standard deviation of the current position

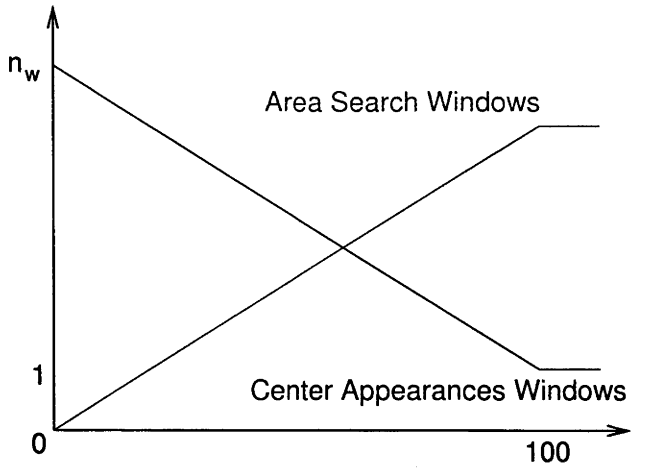


Figure 3.10: Heuristic allocation of correlation windows

estimate are allocated to track changes in the appearance of the feature. Figure 3.10 shows a plot of the values of centre and area search windows. The remaining resources are allocated for area search windows.

The techniques discussed so far are concerned with the allocation of scarce resources for a particular vision processing application. However, these techniques are well suited for vision applications other than face tracking and even non-vision applications which involve multiple costly measurements of a process.

3.3.8 Strain integration

The strain integration mechanism specifically aims at avoiding the local minima problem for feature trackers. This problem has been described in Section 3.3.6. The Kalman filter is able to pull the feature trackers out of local minima if the combination of high correlation results in the minima and large distance between the local minima and the estimated position is sufficiently favourable. This is the case in most situations. However, in some situations, particularly when all features are tracking only marginally, the position estimate $\mathbf{p}_m(i,t)$ from the network has a high variance $pp_m(i,t)$ which is not strong enough to free individual trackers from local minima. This is the drawback of the flexibility of the 2D feature network which nevertheless is important in the earlier stages of the recovery process.

To allow high flexibility of the feature network a mechanism is required to identify the remaining cases when feature trackers are trapped in local minima. The observation of the correlation results of the feature trackers *over time* makes it possible to reach a robust decision about which feature trackers should be relocated. A number of qualitative properties can be defined for such a relocation algorithm:

1. If the observed positions of two features are not consistent with the projected model at least one feature must be relocated.

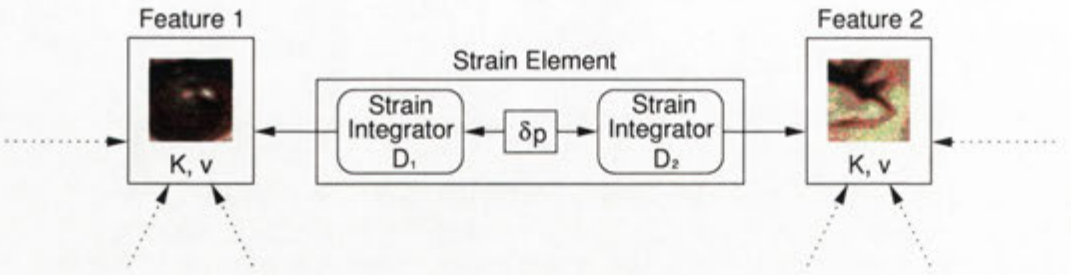


Figure 3.11: The modules of the strain integration on links between feature trackers

2. The decision which feature tracker to relocate is made on the grounds of observing the worsening of correlation results over many frames which are above the normal noise level, or because of only a few significantly worse correlation results. In other words, the confidence in the decision which feature tracker to relocate must be high.

3. Feature trackers which have higher position estimate variances $pp(i,t)$ than their neighbouring feature trackers are relocated to the predicted position $\mathbf{p}_m$ of the better tracking neighbours.

4. If two neighbouring feature trackers have low position estimate variances $pp(i,t)$, then the discrepancy required for a relocation of one of them must be higher or must be observed over a longer time than the discrepancy required to relocate one of two feature trackers with high variances $pp(i,t)$. It should be noted that such situations can be caused by an error in the 3D head pose estimation which result in erroneous projections of the feature positions. In which case the feature trackers which have lost their features will be relocated earlier because their position estimate variances are higher.

5. The number of links to neighbouring features should not affect the relocation probability of a feature tracker.

6. The integrated strain decays over time if the link is no longer strained.

Integrating the strain in the links between features over time is used to detect when feature trackers are trapped in local minima. Figure 3.11 shows the basic concept of the algorithm. After each Kalman filter network cycle the strain

$$\delta p = \|\mathbf{p}(i_1,t) - \mathbf{p}(i_2,t) - \mathbf{l}(i_1,i_2,t)\| \tag{3.10}$$

between each pair of neighbouring feature trackers i_1 and i_2 is calculated as the length of the error vector between the current link geometry and the link vector $\mathbf{l}$ derived from the projection of the 3D model. This corresponds to the first two properties defined for the relocation mechanism. The fifth property requires the

integration of the strain for each link rather than for each feature. Each link contains two integrators d_1 and d_2 which integrate the strain directed towards the respective feature according to Equation 3.11 and 3.12.

$$d_1(t) = \lambda\left(d_1(t-1) + (1-k(i_1,t))\frac{pp(i_1,t)}{pp(i_1,t)+pp(i_2,t)}\delta p\right) \quad (3.11)$$

$$d_2(t) = \lambda\left(d_2(t-1) + (1-k(i_2,t))\frac{pp(i_2,t)}{pp(i_1,t)+pp(i_2,t)}\delta p\right) \quad (3.12)$$

When one of the strain integrators d_j reaches a predefined threshold $d_{\max}$ feature tracker j is relocated with respect to the other feature. At the same time both integrators d_1 and d_2 are reset to 0. This ensures that a badly tracking feature tracker which is relocated several times can not trigger the relocation of a better tracking feature tracker. This complies to the third property. The strain integration algorithm is outlined in Algorithm 3.2.

Algorithm 3.2 Strain integration: `integrate(`d_1`,`d_2`)`

if $d_1 \geq d_{\max}$ AND $d_1 \geq d_2$ **then**
  $d_1 = d_2 = 0$
  $\mathbf{p}(i_1,t) = \mathbf{p}(i_2,t) + \mathbf{l}(i_1,i_2,t)$
else if $d_2 \geq d_{\max}$AND $d_2 > d_1$ **then**
  $d_1 = d_2 = 0$
  $\mathbf{p}(i_2,t) = \mathbf{p}(i_1,t) - \mathbf{l}(i_1,i_2,t)$
end if
return

The fourth property is implemented in the factor $(1-k(i,t))$ which reduces the effective strain on a well correlating feature with high Kalman filter gain $k(i,t)$. In effect feature trackers with well correlating templates resist relocation better than feature trackers with poorly correlating templates. In the extreme situation when $k(i,t) = 1$ no strain can be accumulated for a feature tracker i. This corresponds to a hypothetical situation where a feature has a correlation result of 0.

Finally, the sixth property is implemented in the decay factor $0 \leq \lambda \leq 1$ which scales the integrated strain in each cycle. The decay factor must be sufficiently high enough to allow the quick accumulation of real errors but low enough to allow tracking noise to be tolerated without triggering the relocation of feature trackers. The face tracking system presented in this thesis uses $\lambda = 0.98$ and $d_{\max} = 30$. Small values in λ result in no relocation if the discrepancy δp is not large enough. High values result in relocations of correctly features due to noise. The situation with the threshold value $d_{\max}$ is similar. Large values result in no relocations even if they are required. Small values can result in the relocation of correctly tracking features. The relocation of correctly tracking features is not a problem unless the relocation interferes with the operation of the Kalman filter network due to the elimination of

the motion term after a swap operation. This occurs particularly during fast head motions where the geometric constraints change quickly and the motion term plays an important role in the proper operation of the Kalman filter network. The value for $d_{\max}$ must be chosen such that local minima are always detected and the resulting relocations of correctly tracking features does not interfere with the Kalman filter network. The values used in the face tracking system were determined empirically.

3.4 Experimental results

The facial feature tracking system was tested extensively on real image data. Generally, the system quickly picks up some of the features, and based on these locations, all facial features. It tolerates temporal occlusions and the person frequently leaving the image and returning. When all the facial features are picked up by the system the tracking is reliable even during fast head motions. Features that get occluded due to extensive head motions are recovered as soon as they reappear.

The experimental results presented in this section also incorporate the functions of the third layer of the system (compare Figure 3.1) which is an integral part of the system and provides important feedback to the two lower layers. However, the experiments presented in this section focus on the functionalities of the second layer.

Figure 3.12 shows an image sequence of a person entering the camera view and the progressive recovery of the facial features. The estimated position of each feature is marked with a yellow virtual "pin" which indicates the location and the surface normal.

Initially, none of the feature locations are known to the system. The image sequence is displayed at 10 frames/s (i.e. the sequence lasts 1.8s). The first features at the left[1] eye brow are picked up by the system in image 7 due to the ongoing search with area search windows. Only 0.1s later in image 8 most of the features around the left eye are recovered by the system. The remaining features around the left eye, namely the outer eye corner and the outer end of the eye brow are pulled towards the correct location by the Kalman filter net. Another 0.1s later in image 9 the strain integrators relocated the features around the right eye and the respective features are recovered by the system. Also the centre feature of the mouth was correctly localised. The remaining unrecovered features are the features on the left and right side of the head. Their position is still estimated far too low in the image due to a large rotation of the 3D model of the head around the x-axis (pointing from ear to ear). When the subject is not visible in the image the feature tracking produces random positions and the resulting head pose moves randomly. At the time when the subject entered the image the 3D model was rotated around the x-axis. The strong influence of the well tracking features on the 3D pose estimate slowly

[1]To make the reading more intuitive, left refers to the left side of the image, not the actual left side of the person's face.

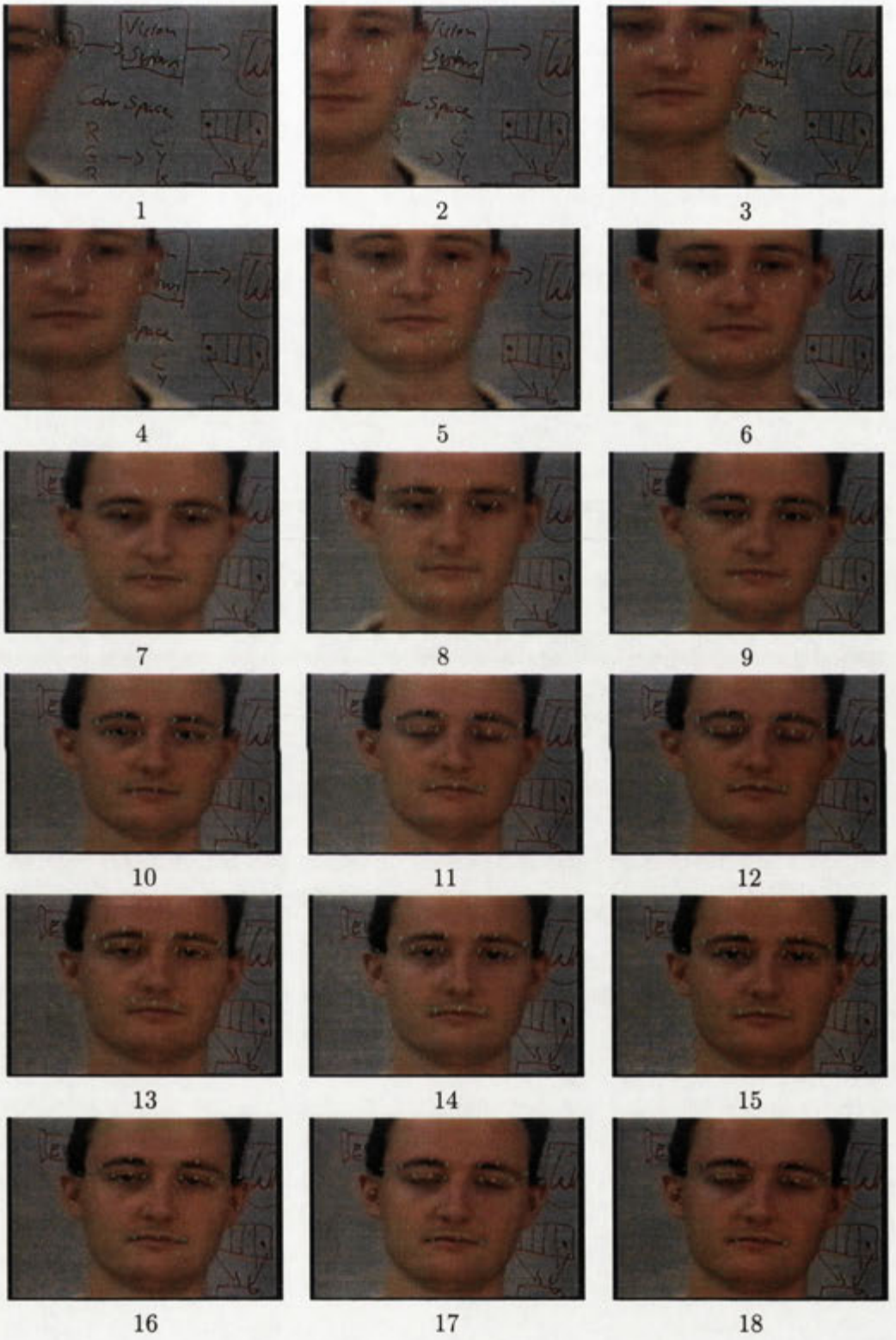


Figure 3.12: Progressive facial feature recovery image sequence

reduces this error and the features along the ear lines are finally recovered by the system in image 14. The pose of the head can be estimated much more precisely and the features in the centres of the ears are recovered swiftly (image 17). The recovery of all facial features took 1s from the time when the face was fully visible in the video image. This result was achieved by the well-balanced interaction of the various mechanisms of the second layer for robust feature recovery and tracking.

Figure 3.12 shows another image sequence from a typical tracking experiment extracted at a rate of one image per second. During the experiment the person rotates the head to the left and to the right more than 45° demonstrating the robustness of the online recovery from feature occlusion. Such large rotations also result in significant changes in the appearance of the features. The template network successfully compensates for these changes. Many of the facial features disappear completely during the sequence and must be recovered as soon as they reappear. This task is performed by the Kalman filter network and the strain integrators which ensure that reappearing features are localised correctly and their respective search windows do not get caught in local minima.

After frame 12 the person leaves the image and returns in frame 16. Within one second the system recovers the location of the facial features and continues tracking. The recovery after complete tracking failures is performed by the area search algorithm. Image 16 shows that after less than 10 frames almost all of the features have been recovered by the system except for the features on the side of the head. The sequence shows how effective the Kalman filter network and the strain integration are in recovering remaining features as soon as a few features are found by the area search windows.

3.5 Summary

This chapter presented the algorithms and techniques used for robust feature tracking in monocular video sequences. The system only uses natural features such as the eyes, eye brows and the mouth. It does not require any artificial markers nor any head gear and is therefore completely non-intrusive. The combination of real-time feature localisation in hardware, the improvements of the robustness in the feature localisation through inclusion of geometric constraints forms a robust real-time feature tracking system which can tolerate partial and complete tracking failures due to temporary occlusions or the person temporarily leaving the image.

The face tracking system is based on an image correlation hardware which forms the lowest layer of the architecture. This vision hardware allows the simultaneous tracking of up to 200 colour feature at frame rate. Most of these resources however are required to track changes in the appearance of the facial features and to enlarge the limited size of the hardware search windows. Although the simple SAD-correlation is not well suited to the requirements of the face tracking application the computational

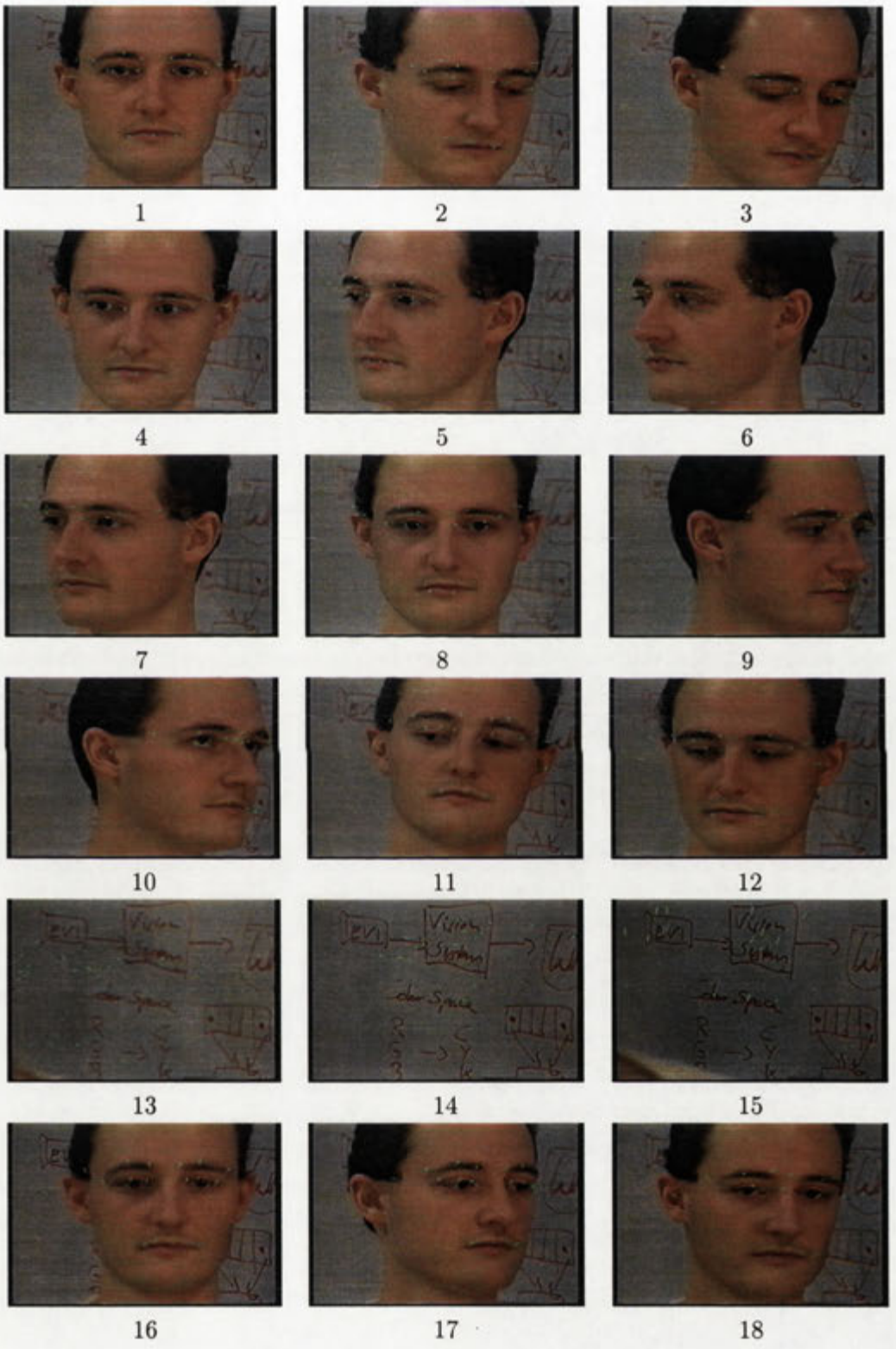


Figure 3.13: Feature tracking and recovery after partial and total occlusions

resources and the algorithms of the second layer allow for the compensation of the unreliable feature detection and tracking by the vision hardware.

The second layer of the system has the sole purpose of deriving robust image positions of the facial features and utilises a number of algorithms to achieve this goal. The Kalman filter network was reused from my Master's research and integrates the raw correlation results of the vision hardware with the geometric constraints. These constraints are derived from an apriori known 3D model of the facial features. This mechanism is integrated with Layer 3 which performs the 3D pose estimation and provides the geometric constraints between the features. In fact the Kalman filter network and the 3D pose estimation form a single measurement loop and Layer 2 can not function without the feedback provided by Layer 3. The problem of local minima is solved by the integration of strain in the 2D constraints between the features over time. This integration makes it possible to perform confident decisions about which feature trackers have lost their facial feature and are trapped in local minima. Once a poor feature tracker is identified with sufficient confidence it is relocated to the position suggested by the Kalman filter based on the position of well tracking neighbouring features.

Beside the inclusion of geometric constraints the second layer provides other functions aimed at the robust tracking of the facial features. The changes in the appearance of the facial features is tracked by exploiting the structure of the transitions between different appearances. This transition structure is represented by a network of templates which are interconnected with transition likelihoods. It is possible to find a small set of templates which cover the most probable transitions to a different appearance of a feature.

The initialisation of the feature trackers and the recovery after tracking failures is aided by a mechanism which artificially enlarges the fixed search window size of the vision hardware. The allocation of the computational resources and the distribution of the area search windows is based on the variance value of the Kalman filter position estimation. This variance value is an excellent indicator for how much of the available computational resources should be taken away from the appearance tracking. Tracking feature appearances only makes sense if the features are tracked well. Otherwise, the system uses more resources to recover the position of lost features.

The combination of all these algorithms provides rapid recovery times and robust position estimates of the facial features. Due to these algorithms the system is able to quickly recover the position of the facial features as soon as they appear in the image and resolve local minima situations. The experiments show that the progressive recovery of the features after a complete tracking failure requires approximately one second after person reappears in the image. After this time the feature positions and head pose are recovered and the system is fully operational. If a subset of the features are lost due to extensive head rotations the recovery using well tracking features is much quicker. Previously occluded features are recovered as soon as they

appear while other features disappear from the camera view. The mechanisms of the second layer allow for a continuous shift in the set of well tracking features and therefore allow the subject to move freely. This robust facial feature tracking system forms the basis of the 3D head pose estimation system implemented in the third layer.

Chapter 4

Head Pose Estimation

The recovery of the 3D pose of objects is one of the fundamental algorithms in computer vision. For rigid objects a variety of solutions have been proposed [Gee and Cipolla 1996], [Azarbayejani et al. 1993]. The spatial position of three features describes the pose of a rigid object in space and a variety of algorithms has been developed using feature triplets [Huttenlocher and Ullman 1990], [Shimizu et al. 1998]. Chapter 2 described a number of algorithms based on feature locations and other visual cues.

There are two major classes of solutions, those assuming affine projection and those assuming perspective projection of the object points. Although the perspective projection is the more accurate model, the affine projection has various advantages such as simpler calculations and is therefore better suited to real-time applications. Also, the focal length does not need to be known if only the rotational part of the pose is of interest. Instead of the four solutions produced by perspective projection formulations the affine projection has only two solutions. For these reasons the affine projection has been used [Maurer and von der Malsburg 1996], [Shakunaga et al. 1998], [Shimizu et al. 1998], [Malciu and Preteux 2000].

The use of the affine projection can be justified by the relatively large distance of the object from the camera compared to the small differences in depth of the features. In such cases the differences between the solutions using the affine projection model and the perspective model are negligible. As a rule of the thumb the ratio between the distances should be at least 10:1 [Thompson and Mundy 1987][Costall 1993][Gee and Cipolla 1994]. Some of the algorithms proposed to solve the three-point to 3D pose problem include [Ullman 1986], [Huttenlocher and Ullman 1990], [Grimson et al. 1992], [Alter 92] and [Cygnaski and Orr 1985]. The work of [Alter 92] also presents an analysis of the numerical stability of previously described algorithms. The work of [Grimson et al. 1992] presents an error analysis of the Huttenlocher and Ullman algorithm. Grimson derives bounds for the error in the final result bounded by an ϵ-circle around the correct position. The calculations of the error propagation through the algorithms are complex and only approximate overestimates

are derived. Grimson made no attempt to derive the systematic error introduced by the assumption of affine projection nor did he try to improve the accuracy of the algorithm.

The algorithm proposed by [Huttenlocher and Ullman 1990] was selected in this research to derive the 3D head pose since it produces the two possible solutions of the pose recovery problem, in contrast to the original formulation by [Ullman 1986] which produces a solution with a fourfold ambiguity. Nevertheless, the algorithm described by [Huttenlocher and Ullman 1990] suffers from a number of drawbacks:

- The algorithm requires precise image coordinates generated by an affine projection. The rotation matrix which is the output of the algorithm is not a proper rotation matrix if the feature positions are corrupted by noise and perspective deformations.

- The systematic error due to the perspective projection of real images is high. In a typical setup which complies to the rule of thumb the angular error can exceed 30°.

- The sensitivity of the 3D pose estimate is high for near-frontal views. Small errors in the detection of the feature positions can result in large pose estimation errors.

This chapter analyses the properties of the Huttenlocher algorithm and proposes a number of extensions which solve all of the above problems. The basic alignment algorithm is extended such that the resulting rotation matrix is always a proper matrix and the error systematic error is reduced by 75%.

Common to all pose detection algorithms based on monocular images and feature triplets is the high estimation sensitivity for configurations where the plane of features is close to parallel to the image plane. In control theory the term "sensitivity" is widely used to describe the change in a parameter a as result of changes in a parameter b. The sensitivity function $s = d(a(b), a(b + \Delta b))/\Delta b$ describes the fractional change in a according to a distance function d, due to changes in b, divided by the fractional change in b [Anand and Zmood 1995]. This chapter proposes a sensitivity model which quickly allows the calculation of the sensitivity of the pose estimate. The integration of multiple feature triplets allow the pose of the head to be determined with a low sensitivity irrespective of the current head pose.

4.1 Huttenlocher's alignment algorithm

The formulation of the Huttenlocher's alignment algorithm is stated here for completeness and can be found in [Huttenlocher and Ullman 1990]. The raw image

locations $\text{img}(\mathbf{p}_i)$ are shifted in the image plane such that the new image coordinates $\mathbf{q}_1$ of $\mathbf{p}_1$ coincide with the origin of the image plane.

$$\begin{aligned} \mathbf{q}_1 &= \text{img}(\mathbf{p}_1) - \text{img}(\mathbf{p}_1) = 0 \\ \mathbf{q}_2 &= \text{img}(\mathbf{p}_2) - \text{img}(\mathbf{p}_1) \\ \mathbf{q}_3 &= \text{img}(\mathbf{p}_3) - \text{img}(\mathbf{p}_1) \end{aligned}$$

The top left 2×2 submatrix L of the rotation matrix R which describes the orientation of the object conforms to the following equation.

$$L = Q \cdot M^{-1} \tag{4.1}$$

where $Q \in \mathbb{R}^{2\times 2}$ contains the image positions $\mathbf{q}_2$ and $\mathbf{q}_3$ in it's columns and $M \in \mathbb{R}^{2\times 2}$ contains the model points in it's columns. Choosing $\mathbf{e}_1, \mathbf{e}_2 \in \mathbb{R}^2$ to be $(1,0)$ and $(0,1)$, k_1 and k_2 become

$$k_1 = -\begin{pmatrix} l_{11} \\ l_{21} \end{pmatrix} \cdot \begin{pmatrix} l_{12} \\ l_{22} \end{pmatrix} \tag{4.2}$$

$$k_2 = \left\| \begin{pmatrix} l_{11} \\ l_{21} \end{pmatrix} \right\| \cdot \left\| \begin{pmatrix} l_{12} \\ l_{22} \end{pmatrix} \right\| \tag{4.3}$$

$$c_1 = \pm\sqrt{\frac{1}{2}\left(k_2 + \sqrt{k_2^2 + 4k_1^2}\right)} \tag{4.4}$$

$$c_2 = \begin{cases} \frac{k_1}{c_1} & c_1 \neq 0 \\ \pm\sqrt{-k_2} & c_1 = 0 \end{cases} \tag{4.5}$$

$$s = \sqrt{l_{11}^2 + l_{21}^2 + c_1^2} = \sqrt{l_{12}^2 + l_{22}^2 + c_2^2} \tag{4.6}$$

$$R = \frac{1}{s}\begin{bmatrix} l_{11} & l_{12} & \frac{1}{s}(c_2 l_{21} - c_1 l_{22}) \\ l_{21} & l_{22} & \frac{1}{s}(c_1 l_{12} - c_2 l_{11}) \\ c_1 & c_2 & \frac{1}{s}(l_{11} l_{22} - l_{12} l_{21}) \end{bmatrix} \tag{4.7}$$

$$\mathbf{d} = \frac{1}{s}\begin{bmatrix} \text{img}(\mathbf{p}_1).x \\ \text{img}(\mathbf{p}_1).y \\ f \end{bmatrix} \tag{4.8}$$

In the following discussion it is assumed that the correct sign of the solutions for c_1 and c_2 can be determined. Section 4.4 describes how this problem is solved in the actual implementation. The complete homogeneous transformation M from the

model triangle to the observed triangle is given by

$$M = \begin{bmatrix} & R & & \mathbf{d} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.9}$$

4.1.1 Analysis of the accuracy and sensitivity

The size of the overall error and sensitivity to non-precise feature tracking depends on a variety of parameters such as the focal length, the distance from the camera and the geometry of the feature triangle. Grimson's theoretical analysis provides only approximate overestimates of the sensitivity of the algorithm which can exceed the actual error up to three magnitudes [Grimson et al. 1992]. The application of the algorithm in a real system with noisy measurements requires an understanding of the relationship between the configuration of the model, the error and sensitivity of the pose estimate. By understanding the weak points of the algorithm an opportunity exists to develop improvements and compensation techniques. The improvements derived from the performance analysis are presented in Sections 4.2 and 4.3.1.

Section 4.2 presents an analysis to gain a realistic impression of the algorithm's performance for face tracking applications. Some of the system parameters are set to values similar to the face tracking case. The quality and quantity of changes in those parameters is discussed where necessary. If not stated otherwise, the following geometric parameter values are assumed:

- The distance of the plane containing all three features to the focal point equals 600mm.
- The focal length equals 45mm and the sensor contains 75^2 square pixels per mm^2.
- The geometry of the feature points is $\mathbf{p}_1 = (0,0,0)$, $\mathbf{p}_2 = (50,0,0)$ and $\mathbf{p}_3 = (50,50,0)$. In the projection the distance between $\mathbf{p}_1$ and $\mathbf{p}_2$ and $\mathbf{p}_2$ and $\mathbf{p}_3$ respectively is 281 pixels.

This geometric setup is illustrated in Figure 4.1. It should be noted that under these assumptions the ratio between the maximum depth of the feature points in 3D and the distance of the features to the focal point is $\frac{600}{50} = 12$. A minimum of 10 is assumed to be the lower bound such that the errors introduced by the assumption of affine projection become negligible.

The feature triplet's coordinate system origin is coincident with $\mathbf{p}_1$, the z-axis points towards the focal point and the x- and y-axis are parallel to the x- and y-direction in the image plane.

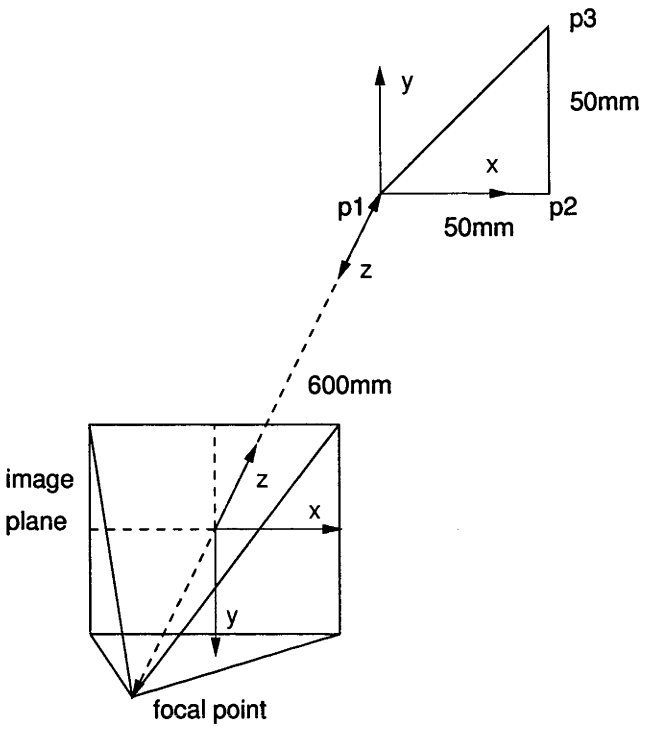


Figure 4.1: Geometry and dimensions of the setup for the case study

4.2 Systematic Errors

The pose estimation algorithm assumes an affine projection of the model points $\mathbf{p}_1$, $\mathbf{p}_2$ and $\mathbf{p}_3$ into the image plane. In reality the features are projected into the image plane under perspective projection. The systematic errors caused by this discrepancy are analysed in this section. This is based on the assumption that the feature positions in the image are known precisely. An improvement to Huttenlocher's alignment algorithm is presented which reduces the systematic angular error of the pose estimate in our case study by 75% and the maximum translational error by 90%.

As a metric for the angular error between two spatial orientations the angle in the equivalent angle-axis representation is used (as defined in [Craig 1986]). Two spatial orientations represented by their rotational matrices R_1 and R_2 can be aligned by one rotation about an appropriate axis $\mathbf{k}$. The norm $\Delta(R_1, R_2)$ between two spatial orientations R_1 and R_2 is defined as the angle of this rotation:

$$\Delta(R_1, R_2) = \arccos\left(\frac{(R_1 R_2^{-1})_{1,1} + (R_1 R_2^{-1})_{2,2} + (R_1 R_2^{-1})_{3,3} - 1}{2}\right); \qquad (4.10)$$

The columns of the rotation matrix generated by the original algorithm are guaranteed to be perpendicular because this is one of the properties used to synthesise the solution. However, if feature positions are used which were generated by a perspective projection the column vectors of the rotation matrix are in general not of

length 1. In that case the matrix is not a proper rotation matrix. The column vectors have to be normalised to length 1 to allow the use of the matrix in further calculations. This scaling operation preserves the orthogonality of the vectors.

In Appendix C.2 the closed form solution of the alignment algorithm is presented. Even thought the algorithm is straightforward, the closed form solution is complex and contains many parameters. The expressions for the systematic angular error $\Delta(R_1, R_2)$ are highly complex which makes it difficult to draw conclusions about the systematic error and limits the practical usability of the method for real-time object tracking. The approximate overestimates derived by Grimson exceed the true error up to three magnitudes and therefore are not suited for practical considerations.

The triplet $\mathbf{p_1 p_2 p_3}$ is considered in a range of spatial orientations and the systematic angular error is calculated. The rotation about the z-axis θ_z remains fixed to 0 since different values in θ_z correspond to different orientations of the same model. By redefining the orientation of the model θ_z can always be made 0. The angular error $\Delta(R_1, R_2)$ is plotted over the angles θ_x and θ_y. Since we are interested in rigid body tracking where the plane in which three features may be arranged is not visible from the rear only rotations between $-\frac{\pi}{2}$ and $\frac{\pi}{2}$ are considered.

The angular difference between the correct spatial orientation R_1 and the orientation R_2 derived with Huttenlocher's algorithm is plotted in Figure 4.2a) for $\theta_z = 0$. The angular error varies strongly over the visible range of the model plane. The four large humps are caused by errors in the estimation of the third elements of the first two column vectors $\mathbf{r}_1 = \frac{1}{s}(l_{11} l_{21} c_1)^t$ and $\mathbf{r}_2 = \frac{1}{s}(l_{12} l_{22} c_2)^t$ of the rotation matrix R_2, corresponding to the z-component of the new x- and y-axis. Also, the scaling of the projection which determines the displacement of the measured model position along the optical axis is directly derived from the length of these vectors. Errors in the z-component cause large errors in the depth estimate, for example with the geometry of the face tracking case the depth error exceeds 100%. However, the discrepancy in the length of the column vectors $\mathbf{r}_1$ and $\mathbf{r}_2$ is the key to the extension of the original algorithm which significantly reduces the angular and translational errors.

If the feature image coordinates are generated by a true affine projection, the first two column vectors have the same length. The depth parameter s in Equation 4.6 can be derived from either of the two column vectors. Thus, s scales both column vectors to length 1. However, if the algorithm is used with feature image coordinates generated by a perspective projection, the length of the first two column vectors will be different in general. The two columns correspond to two different depth estimates s_1 and s_2.

$$s_1 = \sqrt{l_{11}^2 + l_{21}^2 + c_1^2} = |\mathbf{r}_1| \tag{4.11}$$

$$s_2 = \sqrt{l_{12}^2 + l_{22}^2 + c_2^2} = |\mathbf{r}_2| \tag{4.12}$$

$$s_1 \neq s_2 \tag{4.13}$$

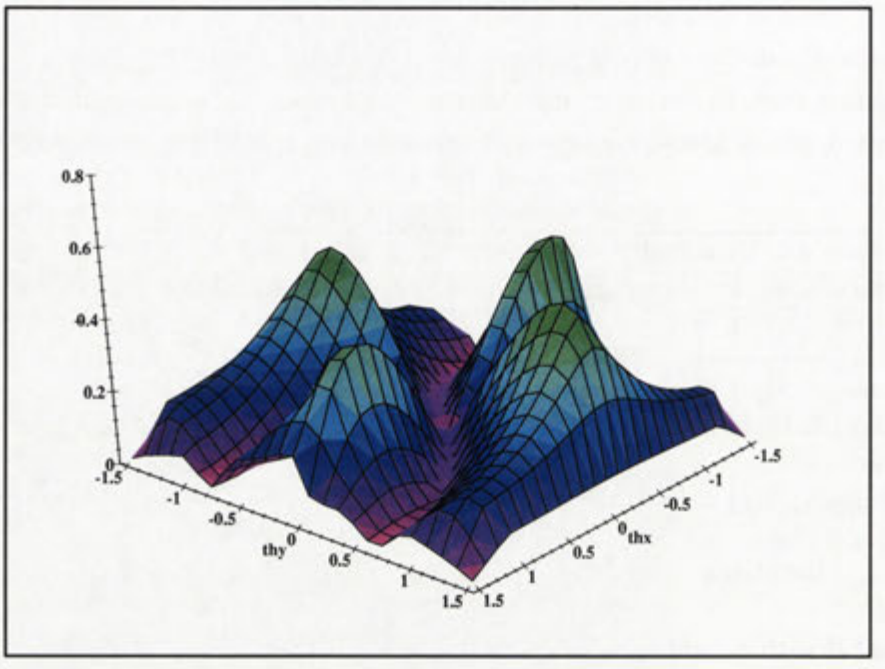
The correct scaling value $\tilde{s}$ can not be determined from perspective data, $\tilde{s}$ lies between s_1 and s_2. Simply choosing $s = s_n = \max(s_1, s_2)$ as the normalisation and depth estimation parameter in Equation 4.7 reduces the maximum error in the depth estimate in the example geometry to less than 10%. The other scaling value s_m obtained from the other column vector is ignored. With this modification the measured depth is always less than or equal to the true depth, and with a much reduced error.

Algorithm 4.1 Extended pose estimation : `pose(q2, q3)`

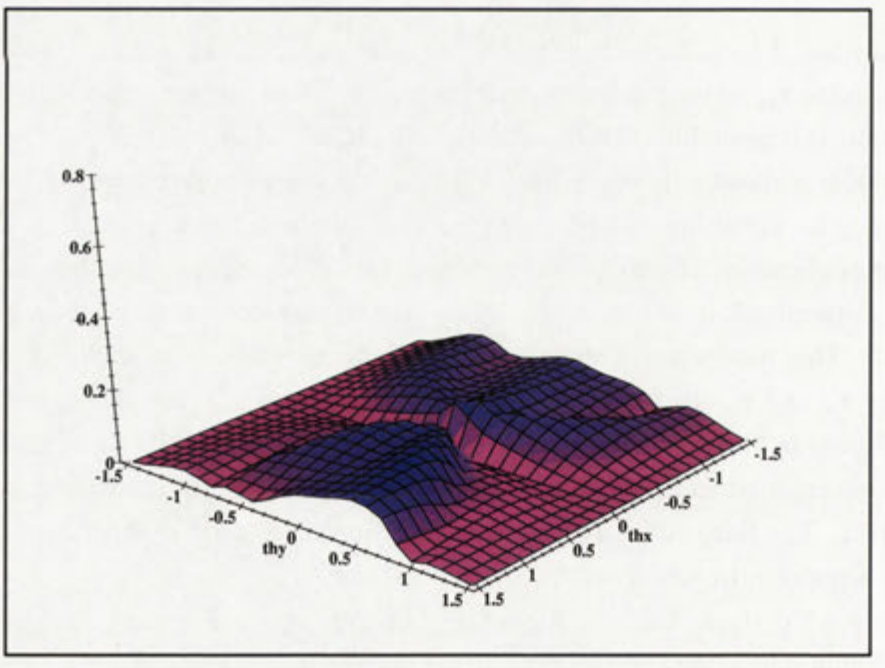
calculate L, k_1, k_2, c_1, c_2 and R according to formulas 4.1-4.5.
$s_1 = \sqrt{l_{11}^2 + l_{21}^2 + c_1^2}$
$s_2 = \sqrt{l_{12}^2 + l_{22}^2 + c_2^2}$
set$(m, n) \in \{1, 2\}$ such that $s_n \geq s_m$
$s = s_n$
$c_m = \text{sign}(c_m)\sqrt{1 - \frac{l_{m1}^2 + l_{m2}^2}{s^2}}$
if $c_m \neq 0$ **then**
  $c_n = -\frac{l_{11}l_{12} + l_{21}l_{22}}{c_m}$
end if
update R with c_n and c_m
$\mathbf{r}_n = \frac{\mathbf{r}_n}{|\mathbf{r}_n|}$
calculate $\mathbf{r}_3$ to complete right hand coordinate system
return R

Algorithm 4.1 outlines the extensions to the original algorithm. Scaling the unused vector $\mathbf{r}_m$ to length 1 will generate a valid rotation matrix while the spatial orientation corresponding to this rotation matrix is unchanged. However, a better scaling value s allows a more accurate estimation of the z-component $\frac{c_m}{s}$ of $\mathbf{r}_m$. This change to the algorithm reduces the systematic angular error by 75% in the face tracking application. Since the first two elements of the vector and the scaling are already determined, only the third element c_m can be recomputed such that $\mathbf{r}_m$ has length 1. This recalculation changes the orientation of the column vector $\mathbf{r}_m$, and therefore $\mathbf{r}_n$ and $\mathbf{r}_m$ are no longer necessarily orthogonal. Since the c_1 and c_2 are most affected by the distortion created by the perspective data, the third element $\frac{c_n}{s}$ of $\mathbf{r}_n$ is recomputed such that $\mathbf{r}_n$ and $\mathbf{r}_m$ are orthogonal again, and then $\mathbf{r}_n$ is scaled to length 1. The third column of the rotational matrix is generated according to the original algorithm to complete the rotation matrix R.

Figure 4.2a) shows the angular error of the *original* algorithm using s_1 as the scaling value and scaling $\mathbf{r}_2$ accordingly. Figure 4.3a) shows the error derived by the *improved* algorithm, but always setting $s = s_1$. The angular error significantly improved in the areas where $s_1 \geq s_2$ only. In the areas where $s_1 < s_2$ the error actually increased for some configurations. Figure 4.3b) shows the complementary result, obtained from always selecting s_2 as the scaling value. The importance of selecting the correct scaling value also for the spatial orientation estimate becomes

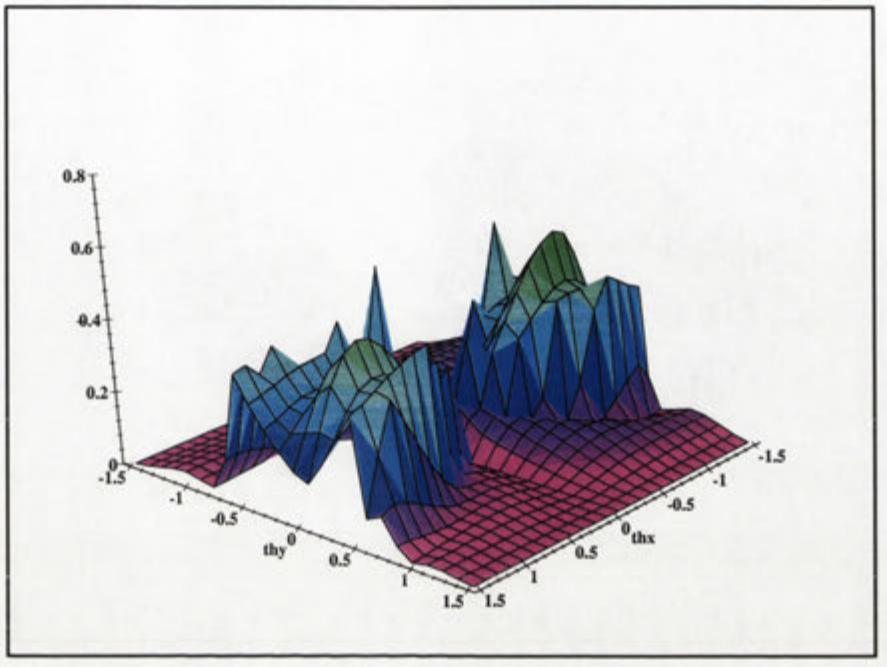


a)

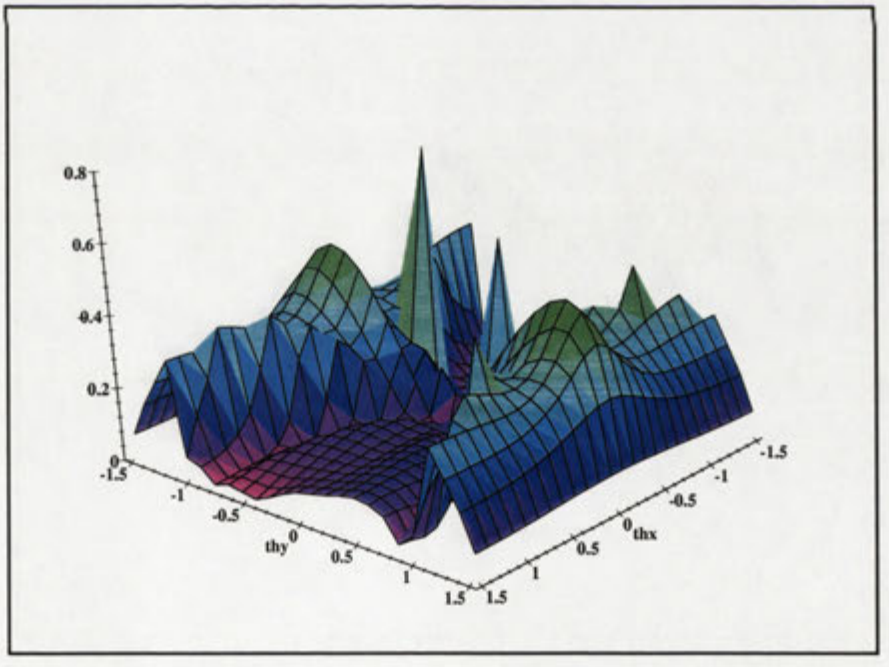


b)

Figure 4.2: Systematic angular error (a) for Huttenlocher's original algorithm and (b) for $s = \max(s_1, s_2)$.

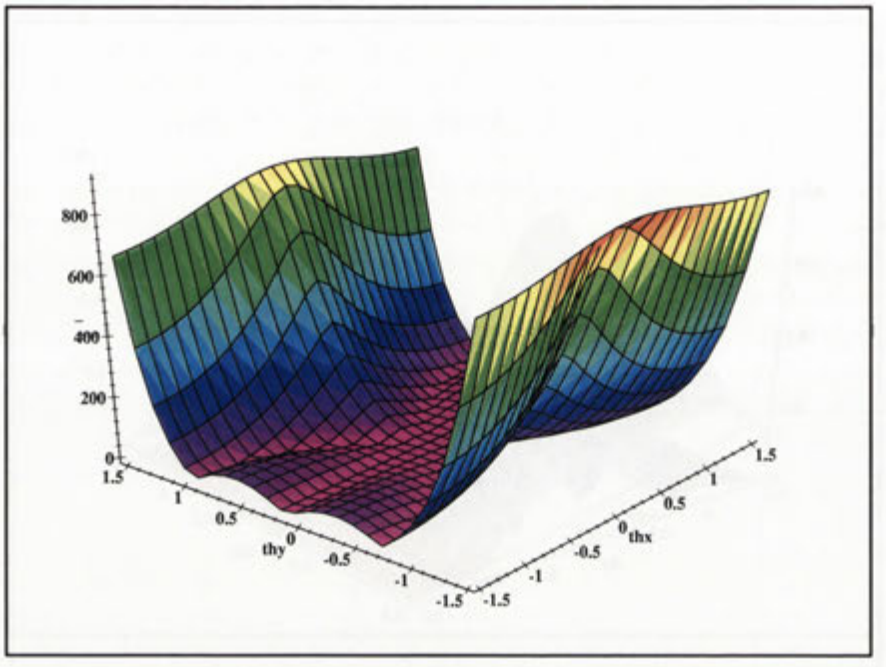


a)

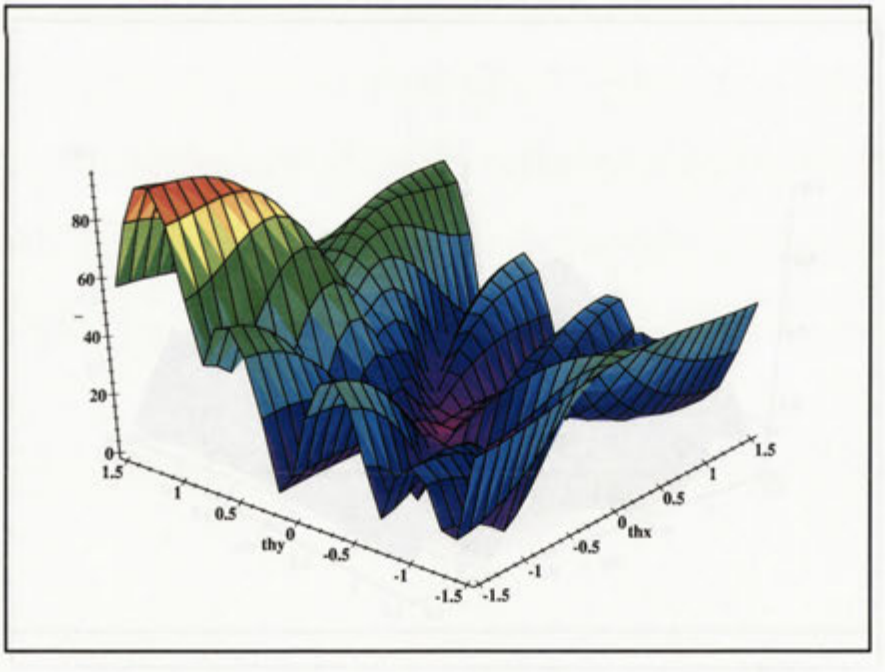


b)

Figure 4.3: Systematic error for (a) $s = s_1$ and (b) for $s = s_2$



a)



b)

Figure 4.4: Depth error for $s = s_1$ b) for $s = \max(s_1, s_2)$ (note the different scale).

obvious. The areas where $s_1 < s_2$ is improved in a similar way as the previous plot while the other configuration show increased errors. The two modifications yield improved results in complementary areas.

It should be noted that the difference in scaling itself does not influence the angular error, but it influences the following calculation steps. The angular error of the complete algorithm is plotted in Figure 4.2b). The areas of low angular error from the previous two plots are combined. The improvements to the algorithm reduces the angular error of the estimate of the spatial orientation to about 25% compared to Huttenlocher's original algorithm.

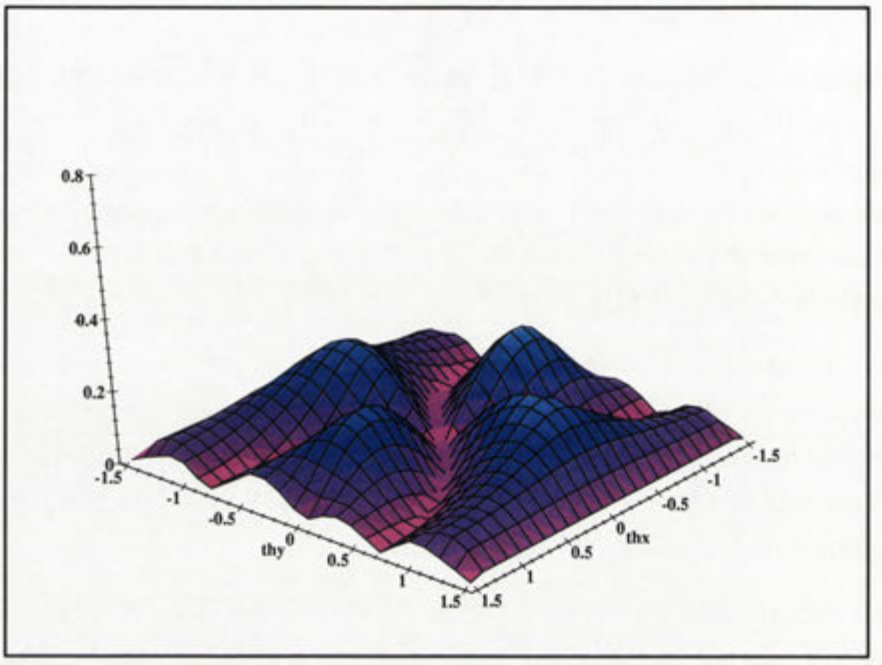
Figure 4.5 shows the angular error of the original and the improved alignment algorithm when the object is 2000mm away from the camera instead of 600mm. Since the difference between an affine and a perspective projection is small the systematic error is reduced. For both the original and the extended algorithm, the systematic error is approximately halved for all configurations.

At an infinite distance of the object from the camera the projection becomes affine. In this case the length of the first two columns of R generated by the original algorithm is the same and both vectors are orthogonal, and therefore the steps in the improved algorithm do not alter the result. The results of the original and the improved algorithm are the same and the systematic error equals 0 for all configurations[1].

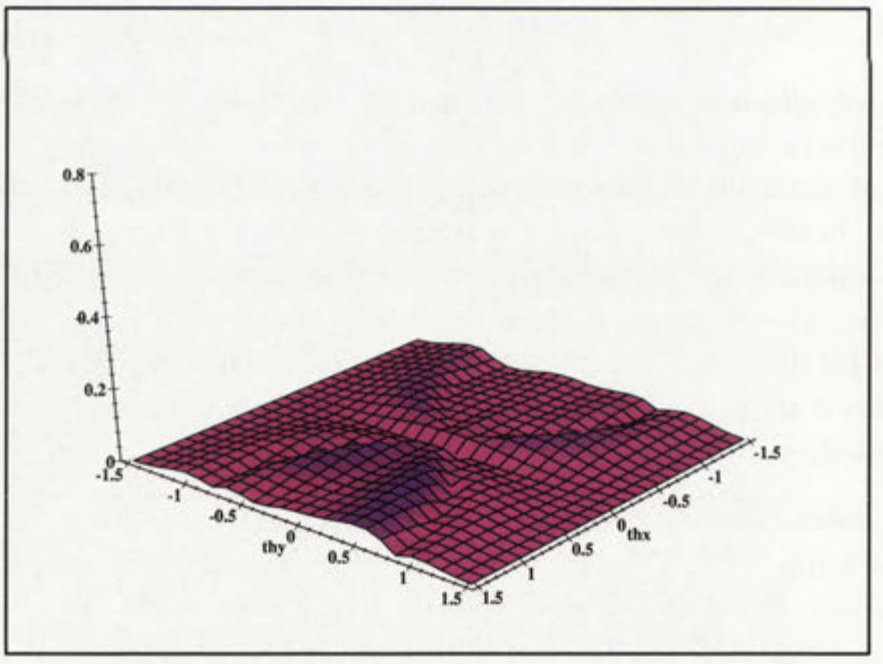
The only alteration in the improved algorithm which effects the calculation of the distance d of the object from the camera is the selection of s from the first or second column of R. The depth $d = \frac{f}{s}$ is calculated from the focal length f and the scaling factor s. In the geometry of the face tracking example the error in depth of the original algorithm reaches more than 800mm (133%). This is shown in Figure 4.4a) for $s = s_1$. The plot for $s = s_2$ is of similar appearance but it is rotated by 90°. Figure 4.4b) shows the depth error for $s = \max(s_1, s_2)$. Note the different scale in z-direction of the plots. The maximum error is reduced to about 10% of the error for the fixed selection algorithms.

The displacement of the feature triplet along the x- and y-axis is calculated from the image position $\text{img}(\mathbf{p}_1)$ of $\mathbf{p}_1$ and the scaling value s. These estimates are therefore affected by the selection of the scaling value by the same factor as the depth estimate. Thus, the error of the displacement along the x- and y-axis is decreased proportional to the error in d by the improvements to Huttenlocher's original algorithm.

[1]Except where the projection degenerates to a line as discussed by Grimson.



a)



b)

Figure 4.5: Plot a) shows the systematic error of the original algorithm at a distance of 2000mm over θ_x and θ_y. Plot b) shows the error of the extended algorithm.

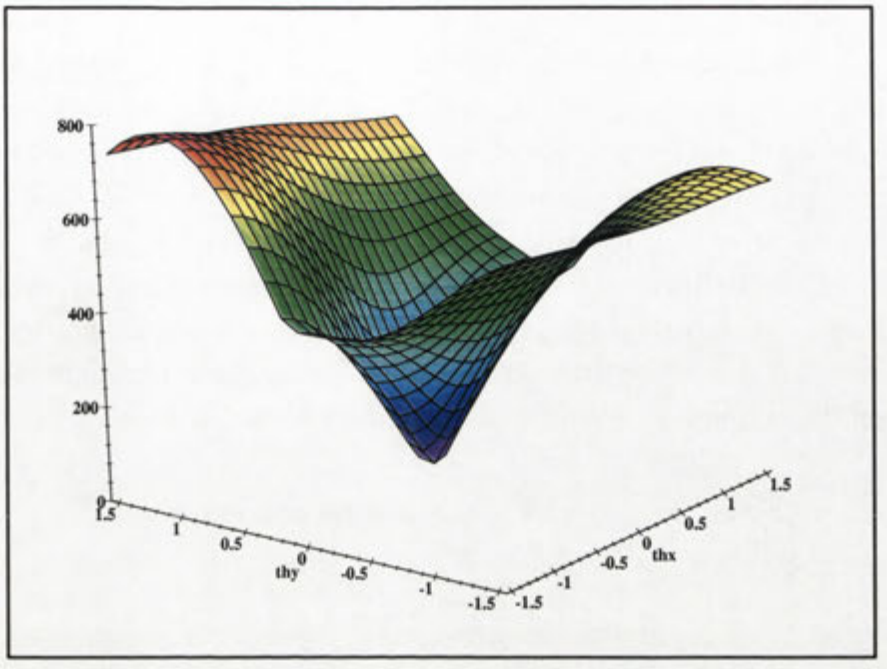


Figure 4.6: Feature velocity

4.3 Tracking Error Sensitivity

This section analyses the sensitivity of the pose estimate to tracking errors. All feature detection algorithms are subject to noise and most do not have sub-pixel accuracy, the sensitivity of the pose estimate to tracking noise is of major importance. In the previous section it was assumed that the feature positions ip_i in the image are precisely known with sub-pixel accuracy to isolate the systematic error from other artifacts. The sensitivity of the pose estimate to small tracking errors caused by image noise, pixel quantisation and the oversampling of the templates is analysed in different spatial orientations of the feature triplet.

A first glance of the situation is shown in Figure 4.6. It shows the derivative of the feature image positions over the rotations of the model around θ_x and θ_y in the face tracking example geometry as shown in Figure 4.1. For an orientation (θ_x, θ_y) the corresponding feature velocity v is calculated as

$$v(\theta_x, \theta_y) = \sum_{i=1}^{3} \left\| \left[\left\| \frac{\delta \mathbf{q}_i}{\delta \theta_x} \right\|, \left\| \frac{\delta \mathbf{q}_i}{\delta \theta_y} \right\| \right] \right\| \tag{4.14}$$

where $\mathbf{q}_i$ denotes the shifted feature image position of feature i for the the spatial orientation (θ_x, θ_y). For $(\theta_x, \theta_y) = (0, 0)$ the motion of the projections of all feature points equals 0, thus, if the feature plane is parallel to the image plane the projected feature positions do not move initially when the object is rotated around either the x-

or y-axis. With an increasing distance from $(0,0)$ the features move with increasing speed.

Therefore different orientations close to $(\theta_x, \theta_y) = (0,0)$ have almost identical projections in the image plane. In other words, small changes in the measured feature position of the image correspond to large changes in the spatial orientation. Configurations in which the plane spawned by the feature triplet is parallel to the image plane ($(\theta_x, \theta_y) \simeq (0,0)$) therefore result in high sensitivities of the pose estimate. In fact, all pose recovery algorithms using monocular images and feature points residing in a single plane suffer from this problem. The asymmetry of the plot is a result of the perspective projection which causes slower motion in the image plane of features which are further away from the camera.

To analyse the sensitivity of the pose estimate a tracking error vector $\mathbf{e}$ is considered which is added to the vector of the feature's image positions.

$$\mathbf{e} = (e_{x1}, e_{y1}, e_{x2}, e_{y2}, e_{x3}, e_{y3}) \tag{4.15}$$

Since the pose recovery algorithm shifts the image of $\mathbf{p}_1$ to the origin, the effective noise vector is $\mathbf{e}_f$.

$$\mathbf{e}_f = (0, 0, e_{x2} - e_{x1}, e_{y2} - e_{y1}, e_{x3} - e_{x1}, e_{y3} - e_{y1}) \tag{4.16}$$

Under the assumption of white noise in the close range of the correct position of a feature the shifting effectively doubles the standard deviation of the noise in the 3rd to 6th element. The sensitivity s_r of the spatial orientation derived by the alignment algorithm with respect to tracking errors for a pose W is defined as the length of the vector of the differential quotients of the orientation distance function Δ.

$$s_r(W) = \left\| \begin{bmatrix} \lim_{d\to 0} \frac{1}{d}\Delta(Pose(\mathbf{P}(W)), Pose(\mathbf{P}(W) + d(0,0,1,0,0,0)^t) \\ \lim_{d\to 0} \frac{1}{d}\Delta(Pose(\mathbf{P}(W)), Pose(\mathbf{P}(W) + d(0,0,0,1,0,0)^t) \\ \lim_{d\to 0} \frac{1}{d}\Delta(Pose(\mathbf{P}(W)), Pose(\mathbf{P}(W) + d(0,0,0,0,1,0)^t) \\ \lim_{d\to 0} \frac{1}{d}\Delta(Pose(\mathbf{P}(W)), Pose(\mathbf{P}(W) + d(0,0,0,0,0,1)^t) \end{bmatrix} \right\| \tag{4.17}$$

where $\mathbf{P}(W)$ is the vector of the image positions after the projection of the model into the image plane and $Pose(\mathbf{P}(W))$ is the rotation matrix R recovered from these image coordinates.

The rotational sensitivity s_r over (θ_x, θ_y) is plotted in Figure 4.7. Again, different values for θ_z do not change the sensitivity of the estimate. The sensitivity asymptotically approaches infinity when $(\theta_x, \theta_y) \to (0,0)$ since the derivative of the feature position with respect to θ_x and θ_y equals 0 at the origin. The plot shows where the pose estimate will be robust and where large errors based on artifacts are expected. At a distance of about 0.4rad from the origin the overall estimate error per one pixel measurement error settles to approximately 0.01radians/pixel. Again,

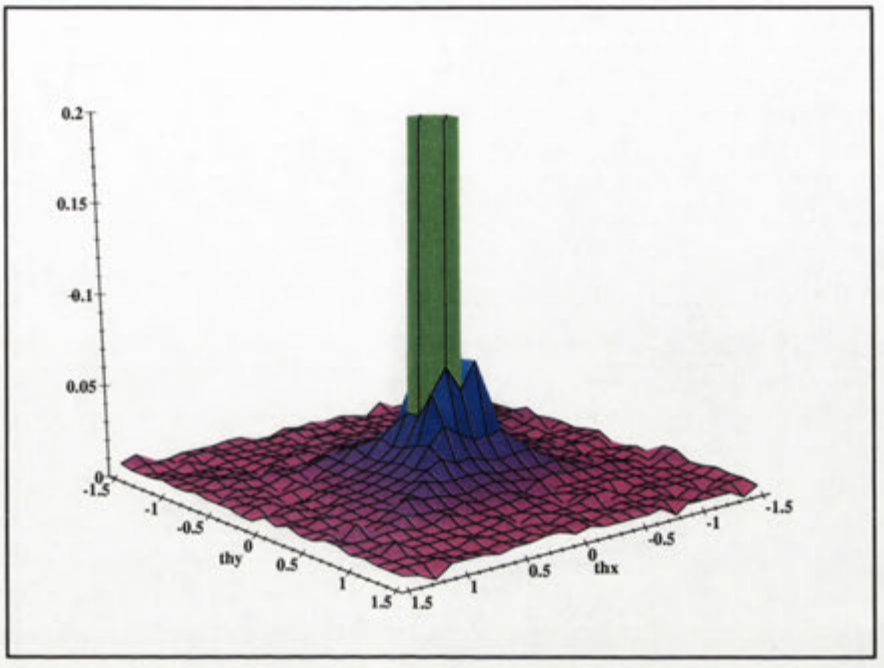
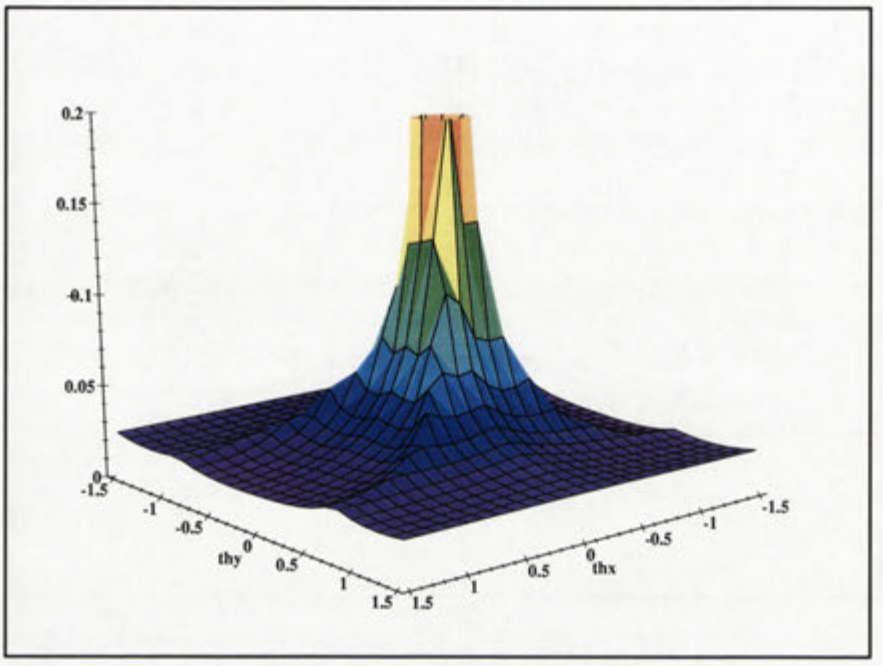


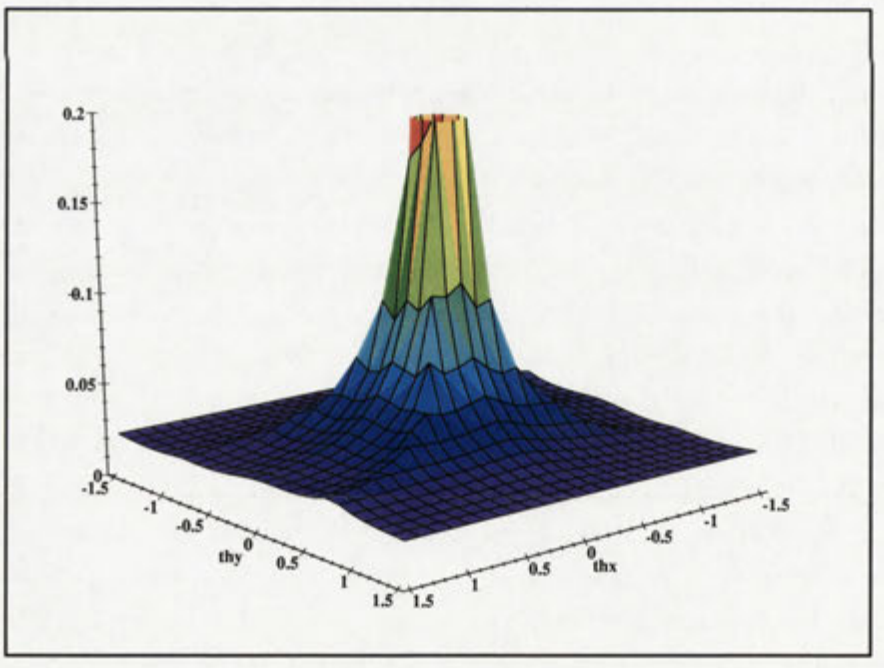
Figure 4.7: Rotational sensitivity s_r

the slight asymmetry is caused by the effects of the perspective projection.

Figure 4.8 shows how the rotational sensitivity changes with the size of the model triangle and the distance of the object from the camera. As expected the sensitivity is larger for a smaller model. Figure 4.8a) is produced with a smaller triangle with the corner points $(0,0)$, $(10,0)$ and $(10,50)$. Interestingly, even though the model triangle is reduced only horizontally, the sensitivity around the x- and y-axes is still similar. The region around the origin where a high sensitivity to tracking errors can be observed extends to a distance of about 0.8rad from the origin. Also the minimum sensitivity level is significantly raised, from 0.01 to 0.04. The plot in Figure 4.8b) shows the sensitivity function evaluated for the original feature triplet observed at an increased distance of 2000mm. This reduces the size of the projection and therefore increases the sensitivity. The effect is almost the same as in Figure 4.8a), both the area of high sensitivity is enlarged to a similar extent and the minimum level is raised by a factor of 4. At this distance the projection of the model parallel to the image plane has the same area as the model that was used in Figure 4.8a), however, the shapes of the two models are considerably different. These results show that the sensitivity is correlated as expected with the size of the projection of the triangle but it is only weakly correlated to the shape of the triangle. Even if one of the sides of the triangle is considerably shorter than the other sides the shape of the sensitivity retains the characteristic square cross section. Therefore, the pose estimate from model triangles with a high aspect ratio have a sensitivity with respect to θ_x and θ_y



a)



b)

Figure 4.8: Rotational sensitivity s_r for a smaller triangle (a) and for the original triangle at a distance of 2000mm

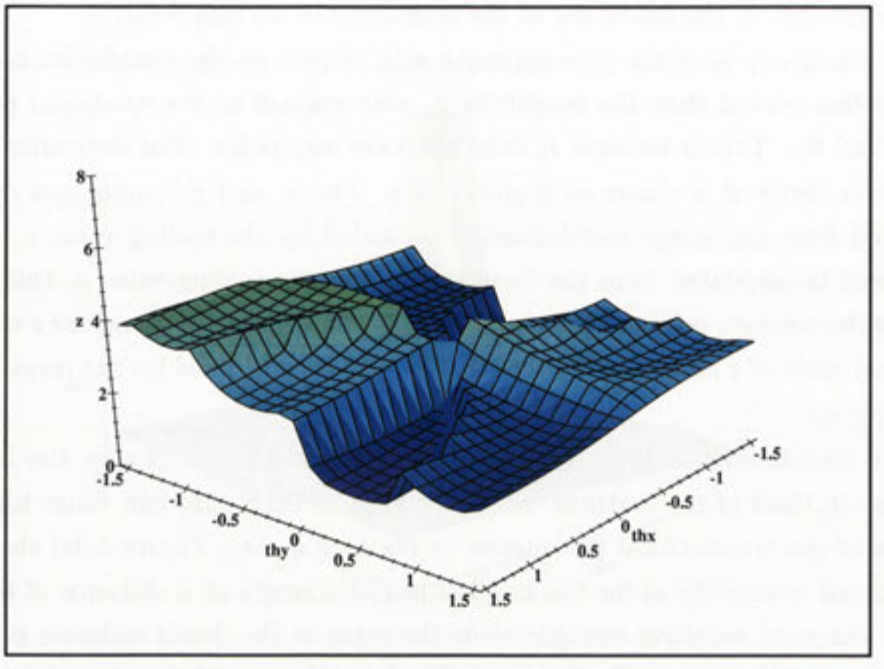
which does not depend on the location of the shorter side and which is comparable to a model triangle of similar size but which has a aspect ratio close to 1. This fact will be exploited in the modelling of the sensitivity in Section 4.3.1.

The sensitivity s_t of the pose estimate with respect to the translation measurements is less critical than the sensitivity s_r with respect to the rotational parameters θ_x and θ_y. This is because s_t does not have any poles. The derivation of the translation vector $\mathbf{d}$ is shown in Equation 4.8. The x- and y-components of $\mathbf{d}$ are calculated from the image coordinates of $\mathbf{p}_1$ scaled by the scaling value s. The z-component is calculated from the focal length and the scaling value s. Due to the common dependency on the scaling value s, only the translation along the z-axis, the reciprocal value of s multiplied by the focal length, is considered for the translational sensitivity s_r.

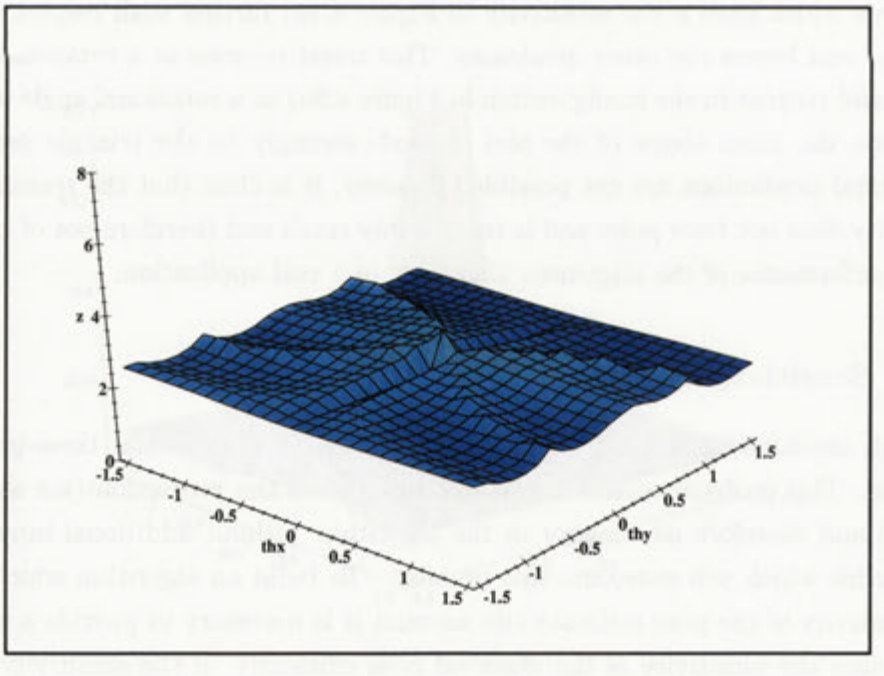
Since translations of the 3D-model along any spatial axis change the location of the projections of the features, no configurations exist that can cause high sensitivities of the translational parameters to tracking errors. Figure 4.9a) shows the translational sensitivity s_t for the original model triangle at a distance of 600mm. Even in the most sensitive configurations the error in the depth estimate per pixel tracking error is less than 1%. Figure 4.9b) shows the sensitivity plot of the model triangle rotated 45° around the z-axis. The sensitivity for the resulting triangle is almost constant with small values. Further rotation around the z-axes raises the quadrants which have a low sensitivity in Figure 4.9a) further with respect to Figure 4.9a) and lowers the other quadrants. This trend reverses at a rotational angle of 180° and returns to the configuration in Figure 4.9a) at a rotational angle of 360°. Therefore, the exact shape of the plot depends strongly on the triangle geometry and general predictions are not possible. However, it is clear that the translational sensitivity does not have poles and is comparably small and therefore not of concern to the performance of the alignment algorithm in a real application.

4.3.1 Sensitivity Model

The high sensitivity of frontal views is a problem for all monocular three-point algorithms. This problem results from the properties of the projection (see also Figure 4.6) and therefore no changes to the algorithm without additional input data are possible which will overcome this problem. To build an algorithm which takes the sensitivity of the pose estimate into account it is necessary to provide a method to calculate the sensitivity of the observed pose efficiently. If the sensitivity of the measured spatial orientation is known it is possible to weight the result to provide a more robust end result. As discussed previously the closed form solution of the algorithm is complex and it's derivative is not suited to real-time calculation. Even the numerical estimation of the differential quotient in Equation 4.17 requires the calculation of eight complete pose estimates. This section describes an approxima-



a)



b)

Figure 4.9: The sensitivity s_t of the depth estimate for the normal model triangle and a 45° rotated triangle

tion model for the rotational sensitivity s_r which allows for fast computation of the sensitivity.

The results in Figures 4.6 and 4.8 show that the sensitivity is almost invariant to 90° rotations in the (θ_x, θ_y) plane. Also, s_r is almost invariant for constant θ_x in the sections where $|\theta_y| < |\theta_x|$ and for constant θ_y in the sections where $|\theta_x| < |\theta_x|$. Therefore, one cross section along one of the major axes of the rotational sensitivity plot provides sufficient information to approximate the sensitivity of all other poses. This cross section is defined as the constrained sensitivity function

$$s_c(\theta_x) = s_r(R(\theta_x, \theta_y = 0, \theta_z = 0)) \tag{4.18}$$

If s_c is known, the sensitivity for other poses can be approximated by s_a according to the properties discussed above.

$$s_a(\theta_x, \theta_y, \theta_z) = s_c(\max(|\theta_x|, |\theta_y|)) \tag{4.19}$$

Figure 4.10a) shows the rotational sensitivity over θ_x and the distance d of the model to the camera for $\theta_y = 0$. The plots obtained from this and other triplet geometries resemble the class of functions

$$\sigma_c(\theta_x, d) = \frac{d^u}{|\theta_x|^v} \cdot w + c \tag{4.20}$$

The parameter set u, v, w, c of σ_c for a particular model triangle can be calculated from four points in the plot in Figure 4.10a). Figure 4.10b) shows the result of the approximation of s_c with σ_c.

The reduction of s_r to σ_c can be reversed using the same arguments. This yields the approximation function σ_a which allows a fast computation of the sensitivity of a model triangle in a measured configuration $(\theta_x, \theta_y, \theta_z, d)$.

$$\sigma_a(\theta_x, \theta_y, \theta_z, d) = \sigma_c(\max(|\theta_x|, |\theta_y|), d) \tag{4.21}$$

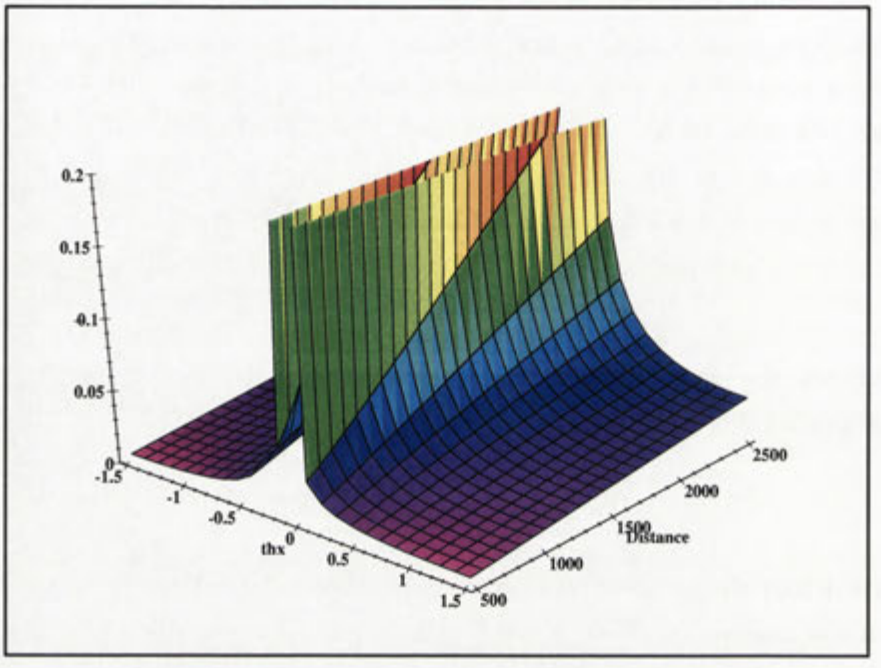
The general solution of u, v, w, c for a particular model triangle can be obtained from four sensitivity values $s_1 - s_4$ at the angles $\theta_x = \{\alpha_1, \alpha_2\}$ and the distances $d = \{d_1, d_2\}$ with the Formulas 4.22-4.25.

$$u = \ln\left(-\frac{s_2 - s_1}{s_3 - s_4}\right)\left(\ln\frac{d_1}{d_2}\right)^{-1} \tag{4.22}$$

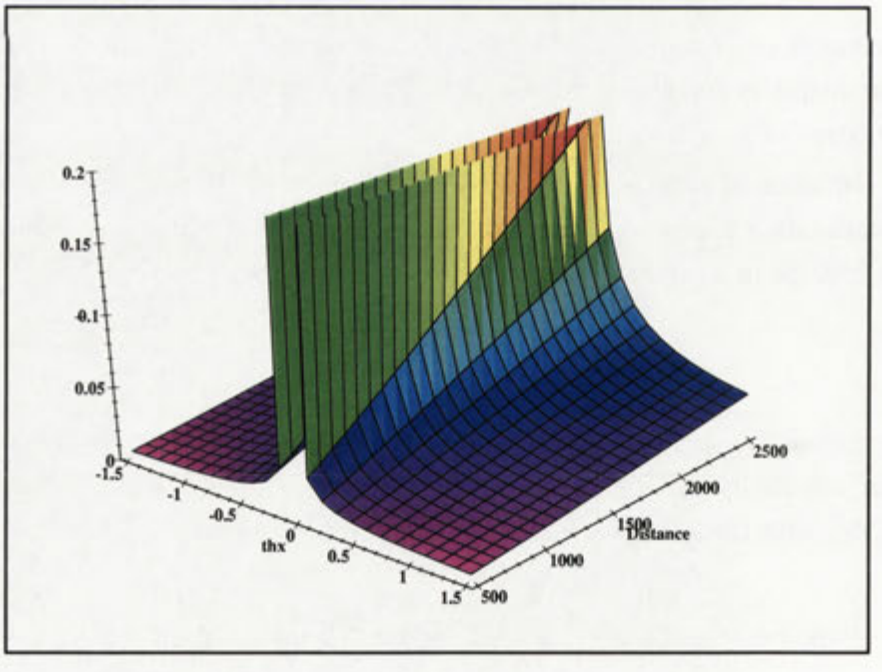
$$v = -\ln\left(-\frac{s_2 - s_4}{s_3 - s_1}\right)\left(\ln\frac{\alpha_2}{\alpha_1}\right)^{-1} \tag{4.23}$$

$$w = e^{q_3} \cdot \frac{s_1 s_2 - s_2 s_3 + s_1 s_3 - s_1^2}{-s_4 + s_3 - s_1 + s_2} \tag{4.24}$$

$$c = \frac{-s_1 s_4 + s_2 s_3}{-s_4 + s_3 - s_1 + s_2} \tag{4.25}$$



a)



b)

Figure 4.10: True sensitivity (a) and the approximation (b)

where

$$
\begin{aligned}
q_1 &= \frac{s_2 - s_4}{s_3 - s_1}, \qquad q_2 = \frac{s_2 - s_1}{s_3 - s_4}, \\
q_3 &= (\ln\alpha_1 \ln d_2 \ln(-q_1) - \ln\alpha_1 \ln d_1 \ln(-q_1) - \ln\alpha_2 \ln d_1 \ln(-q_2) + \\
&\quad \ln\alpha_1 \ln d_1 \ln(-q_2)) \left(\ln\frac{\alpha_2}{\alpha_1} \ln\frac{d_1}{d_2}\right)^{-1}
\end{aligned}
$$

This parameterisation of the approximation function σ_c for a particular model triangle has to be calculated only once when the model is generated. During tracking the simple function σ_a of the sensitivity model is used to provide estimates of the sensitivity in the real-time face tracking system.

Figure 4.11a) shows the plot of the true sensitivity of a modified model triangle with the corner points (0, 0), (100, 0) and (100, 25). The smaller model causes a faster increase of the sensitivity for increasing d and θ_x approaching 0. Figure 4.11d) shows the ratio $r_s = \frac{s_r}{\sigma_c}$ between the real rotational sensitivity s_r and its approximation σ_a. Except for large sensitivity values in s_r the error of the approximation is less than 15%. For θ_x close to 0 the error increases. The sensitivity values themselves in this area and their derivatives are so large that the uncertainty of the estimated pose is much more critical than a more precise approximation of the sensitivity. The asymmetry about positive and negative values of θ_x of the real sensitivity becomes obvious in this figure since the approximation is symmetric. It should be noted that the exact shape of the error ratio r_s for a given triangle also depends on the choice of $\alpha_{1,2}$ and $d_{1,2}$.

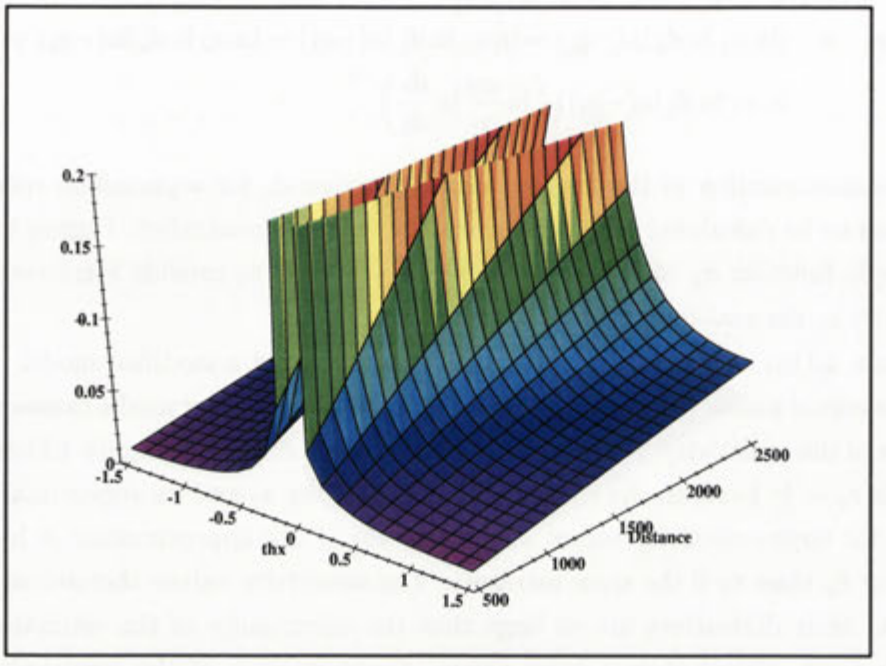
The transition area where the sensitivity starts to increase towards the pole at the origin is the most crucial aspect of the shape of s_r for pose tracking applications. In comparison the approximation accuracy for θ_x close to 0 is of minor importance. The discrepancy between s_a and σ_a in the important areas can be minimised by choosing angles for $\alpha_{1,2}$ which are significantly different from 0 as well as realistic distances $d_{1,2}$. The values used in the two examples were

$$
\begin{aligned}
\alpha_1 &= 0.5\text{rad} \qquad & d_1 &= 600\text{mm} \\
\alpha_2 &= 1.5\text{rad} \qquad & \mathrm{d}_2 &= 2000\text{mm}
\end{aligned}
$$

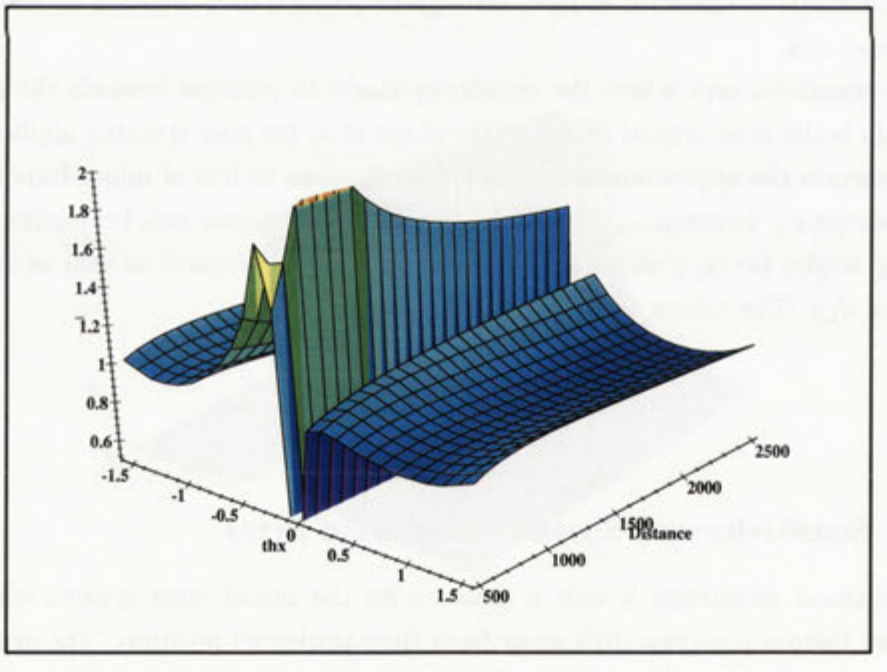
4.3.2 Sensitivity and Spatial Orientation Error

The rotational sensitivity is only a measure for the initial error growth when the measured feature positions drift away from their projected position. The growth of the error is not linear with the size of the error. In particular for near frontal views the second derivative of the angular error has high absolute values. It can be argued that the effective error in the spatial orientation is much smaller than suggested by the high sensitivity.

Figure 4.12 and 4.13 shows the rotational error $\Delta(W_1, W_2)$ as defined in Equa-



a)



b)

Figure 4.11: True sensitivity (a) and the error ratio (b)

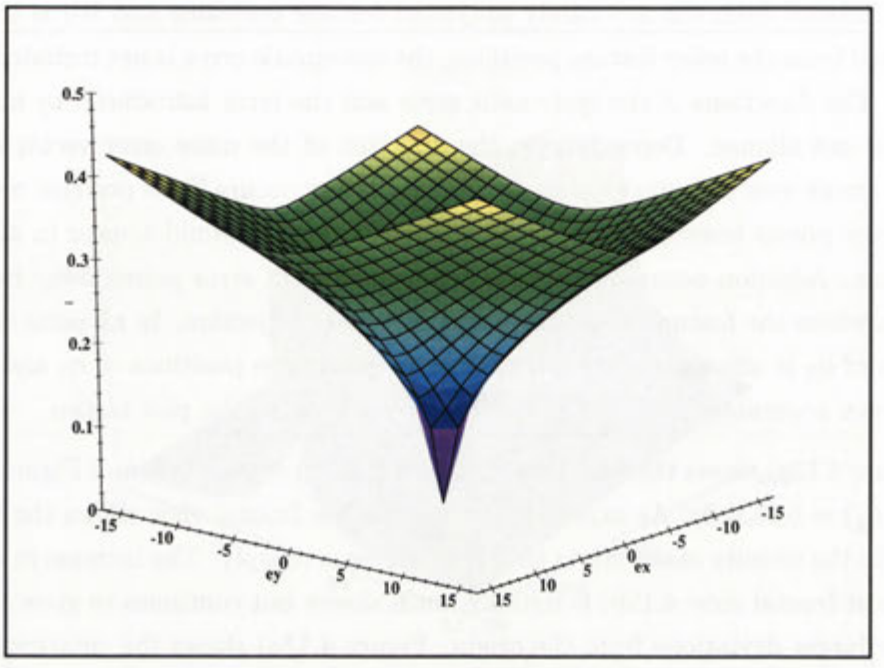
tion 4.10 over the feature position measurement errors in x- and y-direction for various spatial orientations of the example model triplet. Since W_1 is the spatial pose calculated from the accurately projected feature positions and W_2 is the pose calculated from the noisy feature positions, the systematic error is not included in the figures. The directions of the systematic error and the error introduced by noise are generally not aligned. Depending on the direction of the noise error vector (e_x, e_y) the two errors may add up or cancel out. Cancellation occurs if the position measurement error points towards the position where the feature would appear in an affine projection. Addition occurs if the position measurement error points away from the position where the feature would appear in an affine projection. In all plots only the position of $\mathbf{q}_3$ is affected by the tracking error, the image positions of $\mathbf{q}_1$ and $\mathbf{q}_2$ are still known accurately to limit the number of variables in the plot to two.

Figure 4.12a) shows the rotational error for $(\theta_x, \theta_y) = (0.0, 0.0)$ and Figure 4.12b) for $(\theta_x, \theta_y) = (0.0, 0.3)$. As expected the plot for the frontal view shows the highest errors. In the vicinity of the origin the error increases sharply. The increase in error in the almost frontal view 4.12b) is initially much slower but continues to grow steadily even for larger deviations from the origin. Figure 4.13a) shows the rotational error for $(\theta_x, \theta_y) = (-0.8, 0.6)$ and Figure 4.13b) for $(\theta_x, \theta_y) = (1.2, -1.4)$. For non-frontal views the error is in general much smaller than for the two frontal views. The speed of the increase is only a fraction of the increase for frontal views. Figure 4.13b) shows that the speed of increase can depend considerably on the direction of the error vector.

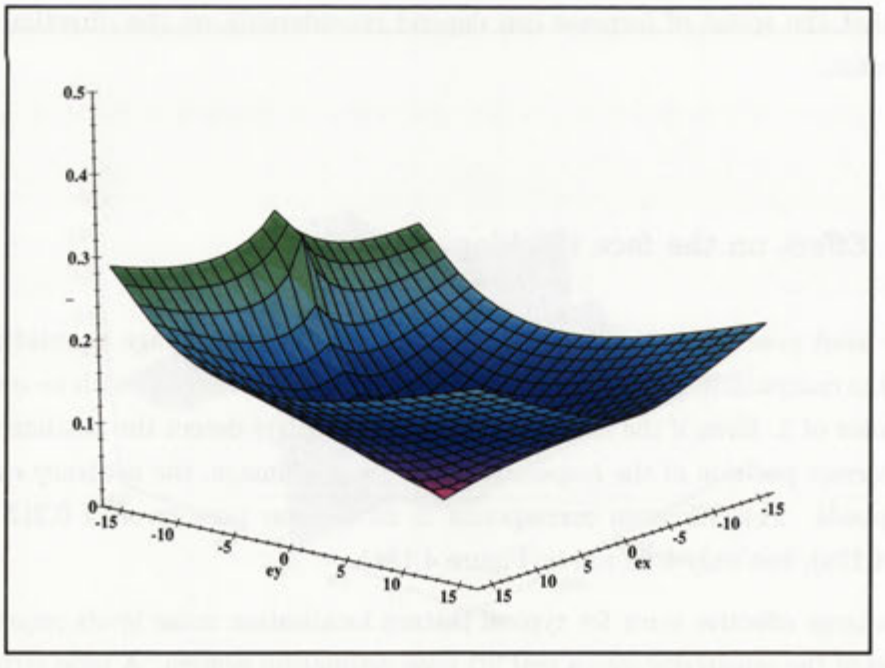
4.3.3 Effect on the face tracking system

For the head pose estimation the features are located using image correlation. To reduce the computational cost the feature templates are correlated with an oversampling factor of 3. Even if the image correlation can always detect the position closest to the correct position of the respective feature in the image, the accuracy can only be ± 2 pixels. This variation corresponds to an angular pose error of 0.215 rad in Figure 4.12a), but only 0.02 rad in Figure 4.13a).

This large effective error for typical feature localisation noise levels requires the inclusion of the sensitivity into a real 3D pose estimation system. A pose estimation system should never rely on the pose estimate derived from a feature triplet which is close to parallel to the image plane. Therefore a system must use multiple feature triplets in different planes and use the pose estimates from triplets which are currently non-parallel to the image plane. This insight led to the development of the multi-triplet estimation system described in the following section.

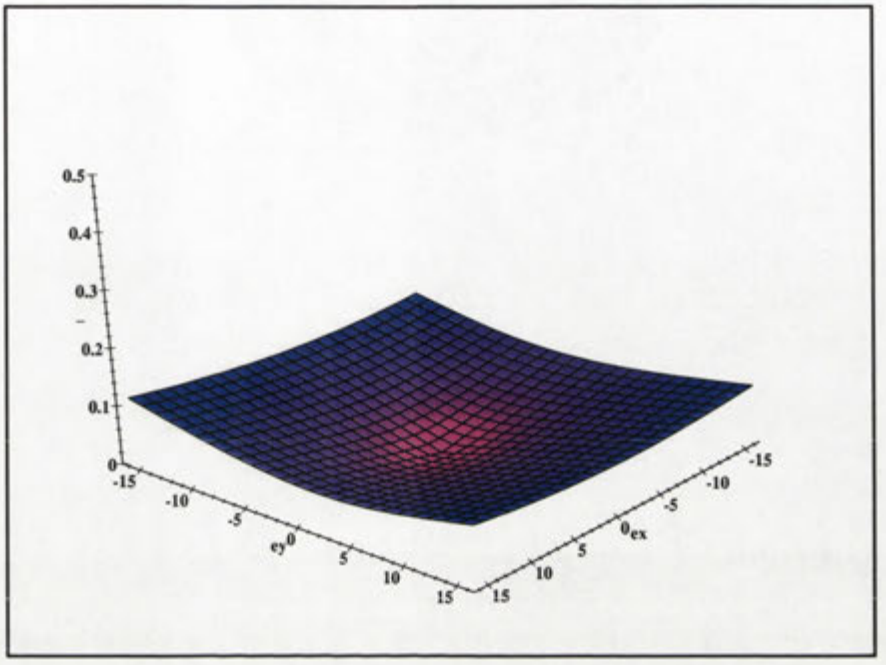


a)

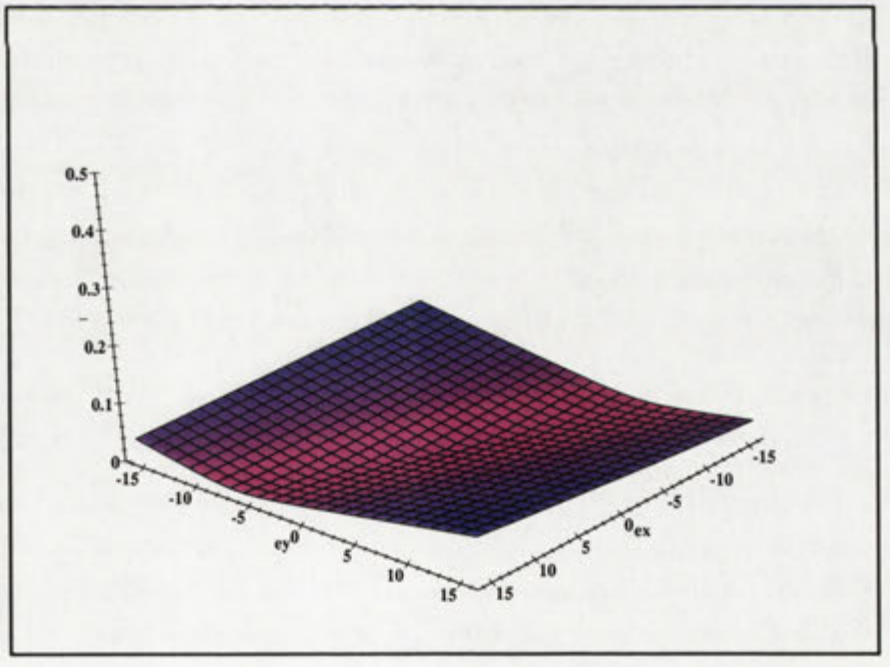


b)

Figure 4.12: Effective pose estimate errors (a) for $(\theta_x, \theta_y) = (0.0, 0.0)$ and (b) for $(\theta_x, \theta_y) = (0.0, 0.3)$



a)



b)

Figure 4.13: Effective pose estimate errors (a) for $(\theta_x, \theta_y) = (-0.8, 0.6)$ and (b) for $(\theta_x, \theta_y) = (1.2, -1.4)$

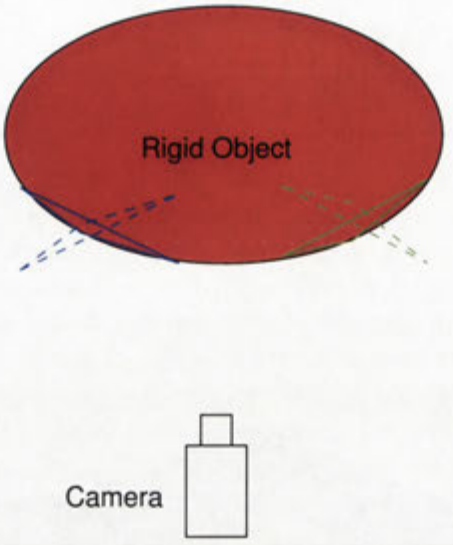


Figure 4.14: Multiple solutions

4.4 Multi-triangle pose estimation

The pose recovery of a feature triplet under affine projection has a twofold ambiguity. The two spatial orientations R_1 and R_2 that generate the same feature image positions under affine projection differ by a reflection about a plane parallel to the image plane. In the discussion of the systematic errors it was assumed that this ambiguity can be resolved and the correct solution $\widehat{R}$ for the pose could be determined. The key idea to solve this problem is to use at least two feature triplets which spawn non-parallel planes. Both triplets generate two solutions, of which only one solution is consistent with both triplets. This solution is the correct pose $\widehat{R}$ of the object. This idea is illustrated in Figure 4.14. The blue and green feature triplets are fixed within one coordinate frame and the six features belong to the same rigid object. The dashed triangles correspond to the second solution of the pose estimation. The two solution with respect to the object coordinate frame could be:

solution 1 $(\theta_x, \theta_y, \theta_z) = (0.2, 0.4, -0.1)$	solution 1 $(\theta_x, \theta_y, \theta_z) = (0.0, -0.3, -0.2)$
solution 2 $(\theta_x, \theta_y, \theta_z) = (0.6, -0.2, 0.1)$	solution 2 $(\theta_x, \theta_y, \theta_z) = (0.2, 0.4, -0.1)$

In this case only the first solution from the blue triplet and the second solution from the green triangle can be combined such that a consistent pose of both triplets results. However, the systematic error discussed in Section 4.2 prevents, even without measurement noise, the two solutions for the pose corresponding to the real pose to match perfectly. Instead, a combination of solutions from the two triplets with the smallest distance under the angular metric Δ defined in Equation 4.10 needs to be selected. The remaining problem is to determine a spatial orientation $\widetilde{R}$ from the two slightly different solutions which considers the sensitivity of the estimate of each triangle.

A straight forward solution to the problem is to weight the solutions according to their sensitivity. The equivalent angle-axis formulation allows interpolation between the two spatial orientations $\widehat{R}_1$ and $\widehat{R}_2$ from the two triplets which have the smallest angular distance. The interpolation $\widetilde{R}$ is generated by rotating $\widehat{R}_1$ only a fraction of the angle $\Delta(\widehat{R}_1, \widehat{R}_2)$ between $\widehat{R}_1$ and $\widehat{R}_2$ around the equivalent axis $\mathbf{k}$.

$$\widetilde{R} = \mathrm{EAR}\left(\widehat{R}_1, \widehat{R}_2, \frac{s_r(\widehat{R}_2)}{s_r(\widehat{R}_1) + s_r(\widehat{R}_2)} \Delta(\widehat{R}_1, \widehat{R}_2)\right) \cdot \widehat{R}_1 \tag{4.26}$$

However, this operation does not use the available information in an optimal way. The sensitivity is a good measure of the overall error that can be expected from a particular pose estimate. But different components of the pose estimates have different individual sensitivities. It is intuitively observable that rotations in the feature plane can be easily detected in the frontal view, whereas rotations about the x- and y-axis prove to be difficult. For this reason the rotation about the z-axis was ignored in the sensitivity analysis. The definition of individual sensitivities for each of the rotational components facilitates the synthesis of a better spatial orientation $\widetilde{R}$ from multiple feature triplets.

The metrics Δ_x, Δ_y and Δ_z for the rotations about the three individual axis are defined as the absolute angular distance between their respective spatial orientations R_1 and R_2.

$$\Delta_x(R_1, R_2) = |\mathrm{Rot}_x(\mathrm{EAR}(\widehat{R}_1, \widehat{R}_2, \Delta(\widehat{R}_1, \widehat{R}_2)))| \tag{4.27}$$

$$\Delta_y(R_1, R_2) = |\mathrm{Rot}_y(\mathrm{EAR}(\widehat{R}_1, \widehat{R}_2, \Delta(\widehat{R}_1, \widehat{R}_2)))| \tag{4.28}$$

$$\Delta_z(R_1, R_2) = |\mathrm{Rot}_z(\mathrm{EAR}(\widehat{R}_1, \widehat{R}_2, \Delta(\widehat{R}_1, \widehat{R}_2)))| \tag{4.29}$$

Similarly to these metrics, the sensitivity of the individual rotations is defined analogous to Equation 4.17 where the overall distance Δ is replaced with $\Delta_{x,y,z}$ respectively. Equation 4.30 defines the sensitivity s_x for the rotation about the x-axis, the sensitivities for the y- and z- axis are defined similarly.

$$s_x = \left\| \begin{bmatrix} \lim_{d\to 0} \frac{1}{d} \Delta_x(\mathrm{Pose}(\mathbf{P}(M), \mathrm{Pose}(\mathbf{P}(M) + d(0,0,1,0,0,0)^t) \\ \lim_{d\to 0} \frac{1}{d} \Delta_x(\mathrm{Pose}(\mathbf{P}(M), \mathrm{Pose}(\mathbf{P}(M) + d(0,0,0,1,0,0)^t) \\ \lim_{d\to 0} \frac{1}{d} \Delta_x(\mathrm{Pose}(\mathbf{P}(M), \mathrm{Pose}(\mathbf{P}(M) + d(0,0,0,0,1,0)^t) \\ \lim_{d\to 0} \frac{1}{d} \Delta_x(\mathrm{Pose}(\mathbf{P}(M), \mathrm{Pose}(\mathbf{P}(M) + d(0,0,0,0,0,1)^t) \end{bmatrix} \right\| \tag{4.30}$$

As expected the shape of the sensitivity of the rotation around the x- and the y-axis resembles the shape of the overall sensitivity and can be approximated in the same way. The sensitivity of the rotation around the z-axis is almost constant over the range where the feature plane is visible. The sensitivity only increases for orientations where the feature plane is almost perpendicular to the image plane.

Figure 4.15 shows the sensitivity of the x- and z-rotations.

The individual components of the spatial orientation $\widetilde{R}$ which describe the overall pose of the object can now be derived. Each rotation $\mathrm{Rot}_x(\widetilde{R})$, $\mathrm{Rot}_y(\widetilde{R})$ and $\mathrm{Rot}_z(\widetilde{R})$ about the three axis are defined as the weighted average of the two respective rotations in R_1 and R_2 with the sensitivity as weights. Equation 4.31, 4.32 and 4.33 define the rotation around the x-axis, y- and z-axis.

$$\mathrm{Rot}_x(\widetilde{R}) = \frac{\frac{\mathrm{Rot}_x(R_1)}{s_x(R_1)} + \frac{\mathrm{Rot}_x(R_2)}{s_x(R_2)}}{\frac{1}{s_x(R_1)} + \frac{1}{s_x(R_2)}} \tag{4.31}$$

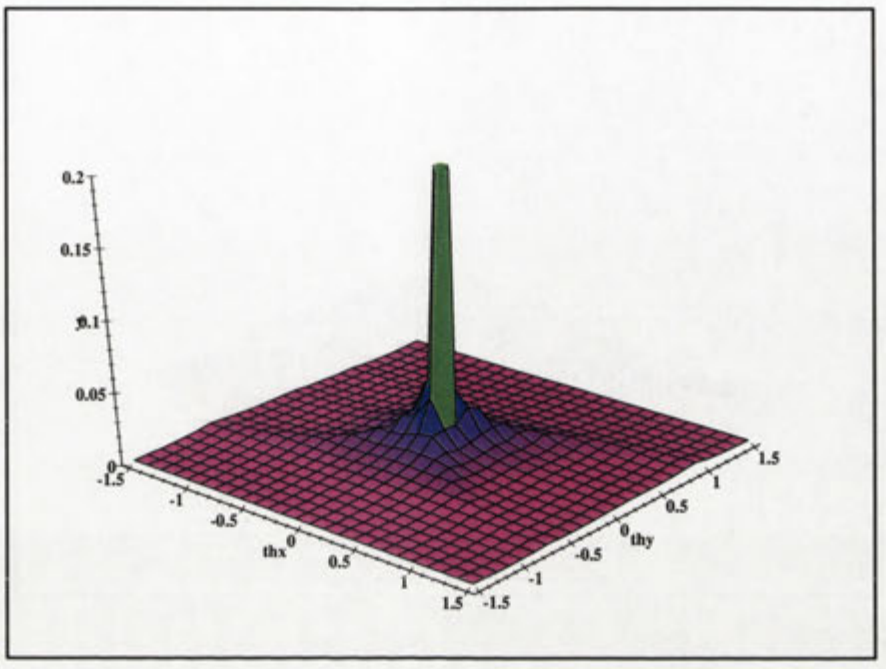
$$\mathrm{Rot}_y(\widetilde{R}) = \frac{\frac{\mathrm{Rot}_y(R_1)}{s_y(R_1)} + \frac{\mathrm{Rot}_y(R_2)}{s_y(R_2)}}{\frac{1}{s_y(R_1)} + \frac{1}{s_y(R_2)}} \tag{4.32}$$

$$\mathrm{Rot}_z(\widetilde{R}) = \frac{\frac{\mathrm{Rot}_z(R_1)}{s_z(R_1)} + \frac{\mathrm{Rot}_z(R_2)}{s_z(R_2)}}{\frac{1}{s_z(R_1)} + \frac{1}{s_z(R_2)}} \tag{4.33}$$

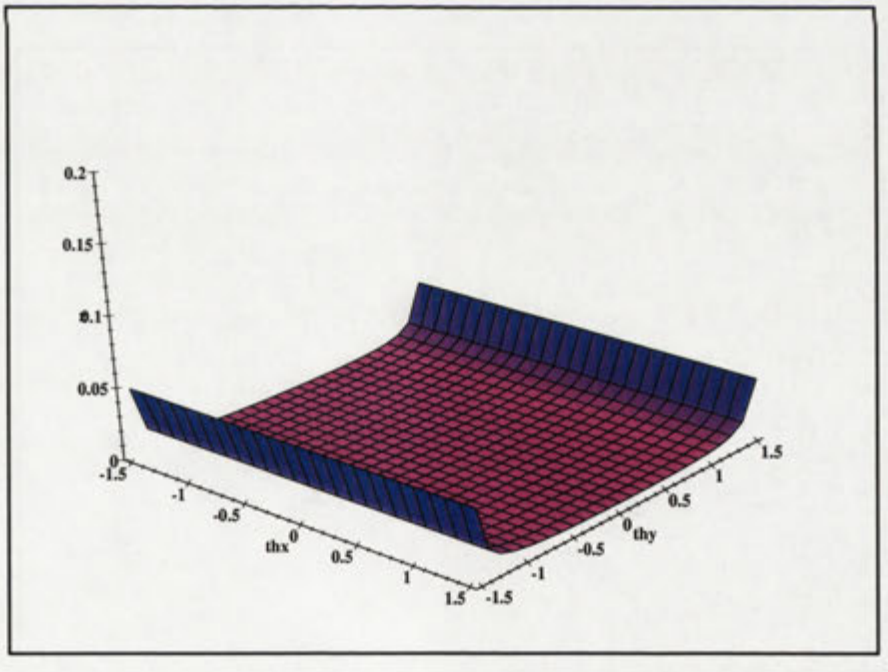
To evaluate the sensitivity improvement of this extension the original feature triplet and an additional triplet of the same shape rotated 45° around the y-axis are considered. This configuration is similar to the head pose application where a triplet in the facial plane and others on the side of the head can be used. Figure 4.16a) shows the overall rotational sensitivity of the pose estimate derived from the two combined triplets. The sensitivity pole in the origin has been eliminated. Instead, the sensitivity is almost uniform at a low level. The graph compares directly to Figure 4.6b) which shows the sensitivity of the single triplet solution. Only two small humps appear where the poles of the two triplets existed. The pole of the additional triplet is at 45° as expected. The angle between the triplet planes must be sufficiently large to achieve the eradication of the poles. For each spatial orientation of the object at least one of the triplets must provide a robust estimate.

Figure 4.16 shows the systematic angular error of the pose derived from the combined triplets. It shows a similar error level to those shown in Figure 4.2b). Note that the systematic error is not a criteria for the interpolation between the two spatial orientations. The goal of the combination of two estimates is the elimination of the sensitivity pole for frontal views. The systematic error can be used in the weighting of the merging process. However, since the systematic error is 0 for frontal views it would counter the weighting derived from the sensitivity and thus, would defeat the purpose of merging the two poses. Therefore a systematic error that does not increase through the combination of two triplets meets the expectations since a lower sensitivity is achieved at the same time.

The new algorithm can be extended to more than two feature triplets. For n feature triplets $\mathbf{T} = \{T_1 \dots T_n\}$ the set $\widehat{\mathbf{R}} = \{\widehat{R}_1 \dots \widehat{R}_n\}$ of solutions has to be determined such that the distances $\Delta(\widehat{R}_i, \widehat{R}_j)$ between the individual solutions is mini-



a)



b)

Figure 4.15: Sensitivity of the rotation around the x-axis (a) and the z-axes(b)

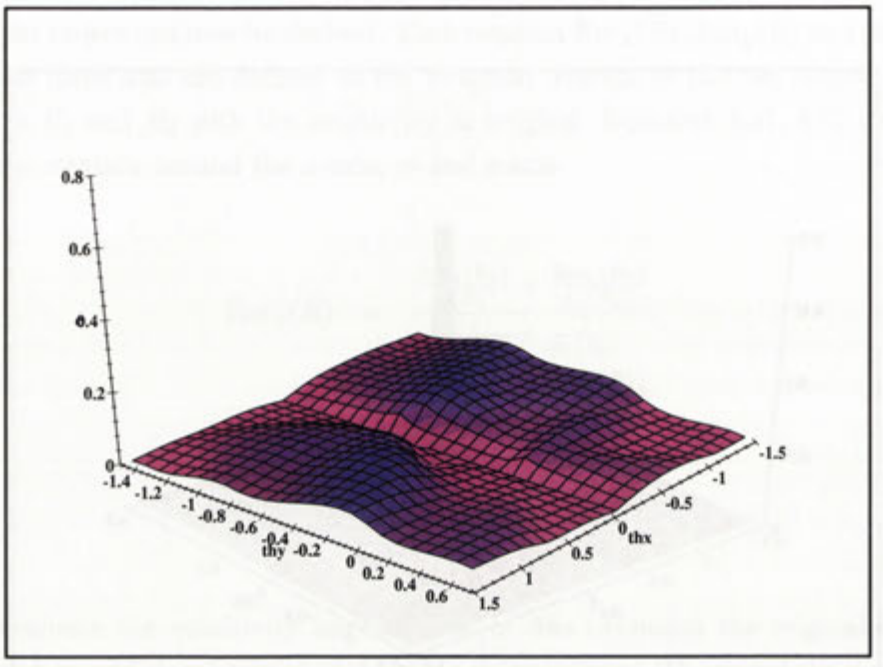


Figure 4.16: Overall pose estimate error

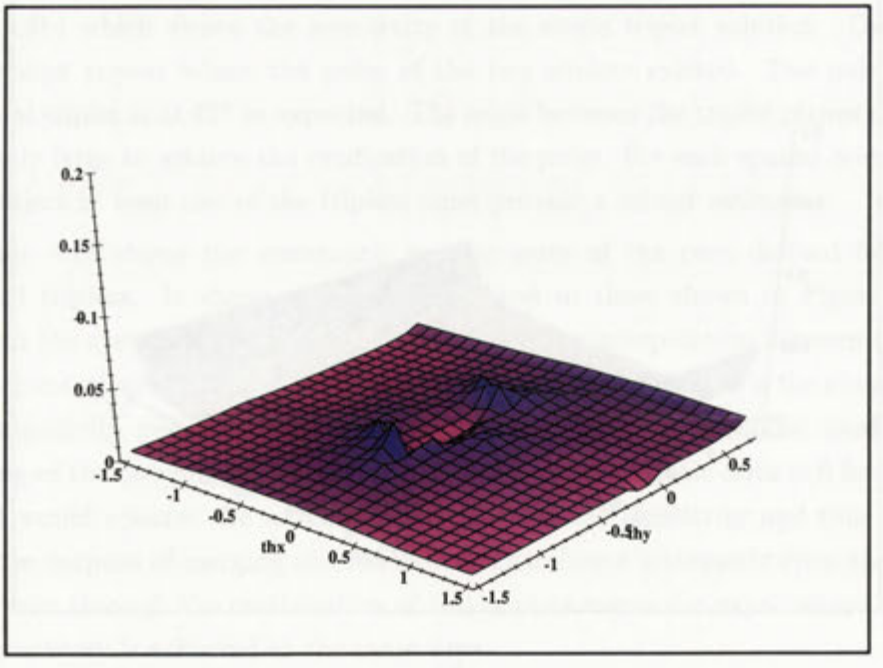


Figure 4.17: Overall sensitivity

mal. This is computationally expensive for large n. In the presence of strong noise a unique solution may not exist. In this case triplets need to be selected that provide the most reliable tracking results. Equations 4.31, 4.32 and 4.33 must be extended to interpolate between n measurements.

$$\mathrm{Rot}_x(\widetilde{R}) = \frac{\sum_{i=1}^{n} \frac{\mathrm{Rot}_x(\widehat{R}_i)}{s_x(\widehat{R}_i)}}{\sum_{i=1}^{n} \frac{1}{s_x(\widehat{R}_i)}} \tag{4.34}$$

$$\mathrm{Rot}_y(\widetilde{R}) = \frac{\sum_{i=1}^{n} \frac{\mathrm{Rot}_y(\widehat{R}_i)}{s_y(\widehat{R}_i)}}{\sum_{i=1}^{n} \frac{1}{s_y(\widehat{R}_i)}} \tag{4.35}$$

$$\mathrm{Rot}_z(\widetilde{R}) = \frac{\sum_{i=1}^{n} \frac{\mathrm{Rot}_z(\widehat{R}_i)}{s_z(\widehat{R}_i)}}{\sum_{i=1}^{n} \frac{1}{s_z(\widehat{R}_i)}} \tag{4.36}$$

The n-plane pose algorithm is outlined in Algorithm 4.2. It facilitates the improved triplet pose estimation algorithm, the sensitivity model for the individual rotation parameters and the sensitivity-weighted n-plane pose interpolation method.

Algorithm 4.2 Multi-plane pose estimation : `pose_n-plane(T)`

for all $T_i \in \mathbf{T}$ **do**
  Calculate orientations $R_i = \mathbf{pose}(T_i)$ according to Algorithm 4.1
  Calculate sensitivities $S_i = \begin{pmatrix} s_x & s_y & s_z \end{pmatrix}$ according to Equation 4.17.
end for
Calculate $\widetilde{R}$ according to Equation 4.34, 4.35 and 4.36
return $\widetilde{R}$

4.5 Experimental evaluation

The improvements of Huttenlocher's algorithm proposed in the previous sections were developed specifically to provide better performance in real implementations, in particular with the application to 3D head tracking. The improved pose estimation algorithm forms the third and top layer in the face tracking system. It uses an apriori known 3D feature point model with 19 facial features and three feature triplets to estimate the head pose robustly. The projection of the 3D feature point model in the image plane is used to update the geometric constraints in the second layer.

This section describes the experimental evaluation of the proposed system with real image sequences of a human head model.

4.5.1 Experimental setup

It is crucial in the evaluation of such techniques to use real data. Computer animated faces were used by [Basu et al. 1996] provide accurate knowledge of the head pose but

they do not provide realistic image noise. Measurement devices such as the Polhemos pose tracker used by [Azarbayejani et al. 1993] have accuracies in the order of the vision system and are therefore not suited for proper evaluation of the performance of head pose estimation systems.

The system described in this chapter was tested with the MARITA-system (Mannequin Robot for Investigation of Tracking Accuracy) and is shown in Figure 4.18a). MARITA consists of a mannequin head mounted onto a cable driven high performance pan-tilt-device. The device has two degrees of freedom which rotate the mannequin head around its vertical axes and tilt the head for- and backward. Rotations about the z-axis (facial normal) are not possible, however, the estimation error of this parameter can be measured nevertheless. In fact the transformation from the X-Y Euler angles of the mechanism to the Z-Y-X Euler angles used for the internal pose representation causes small rotations about the z-axis.

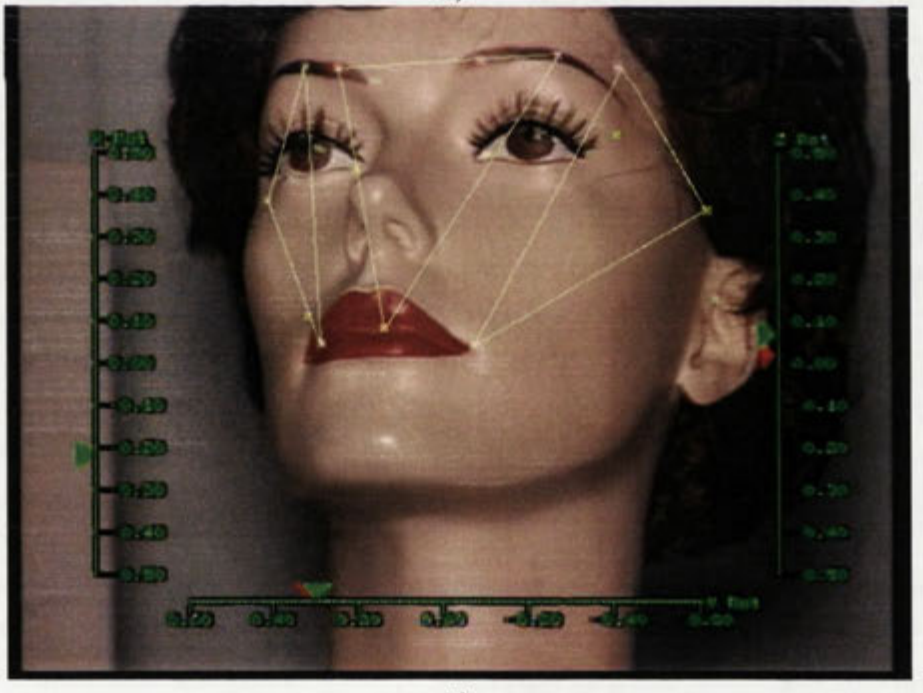
The pose of the MARITA system can be controlled with an accuracy greater than $0.1°$ and therefore allows to test the performance of the pose estimation algorithm under realistic conditions.

4.5.2 Features

The human face contains a number of features with different properties which make them more or less valuable for the face tracking application. Generally a suitable feature contains a high-contrast texture which is unique at least in the vicinity of the feature. If similar textures occur at all they should be spatially separated to avoid problems in the feature identification. A good feature also does not change its appearance drastically in different head orientations and it does not get occluded by other features such as the nose or hair.

Features with high-contrast textures are the eyes, the eye brows and the hairline. The eyes have the disadvantage that they change in appearance considerably when they are closed during blinking. The same is true for the corners of the eyes. The irises also move with respect to the head when the person looks around. If they are used in the 3D pose estimation this fact can cause significant errors and they are therefore not used for this purpose in this system. The hairline and the eye brows do not change their appearance even when the head is rotated. However, similar textures can usually be found in their close vicinity. Corners in the hairline and the ends of the eye brows do reduce this problem slightly. The nose is not a good feature although it is a prominent feature in the human face. The only high contrast texture can be found at the nostrils which become occluded when the person lowers the head slightly. The mouth also is not an easy feature to track. The contrast between the facial skin and the lips is small for many people, also the shape of the mouth can change considerably. When the mouth is opened the teeth and the tongue become visible which are occluded otherwise. In summary, the best features are the corners

a)



b)

Figure 4.18: MARITA-system (a) and view from the vision system (b)

of the eyes and the eye brows followed by features on the hairline and the mouth.

To track the pose of the mannequin head a set of 19 features were selected manually, including 6 features in each eye area, 3 features in the mouth area and 2 features in each ear area. The small white crosses in Figure 4.18b) mark the centre of the tracked features. Not all 19 features were used for pose estimation since only three feature triplets are used to derive the head pose. The other features provide more clues for the two lower layers to improve the overall tracking performance and make the system robust to noise and partial occlusions.

The lines in the figure indicate the feature triplets of which one is parallel to the facial plane and one triplet is located on each side of the head. Only the features at the corners of the triangles are actually used for the pose estimation. With this selection of features and model triangles the spatial orientation of the head can be measured by at least two triplets for a wide range of spatial orientation of the head.

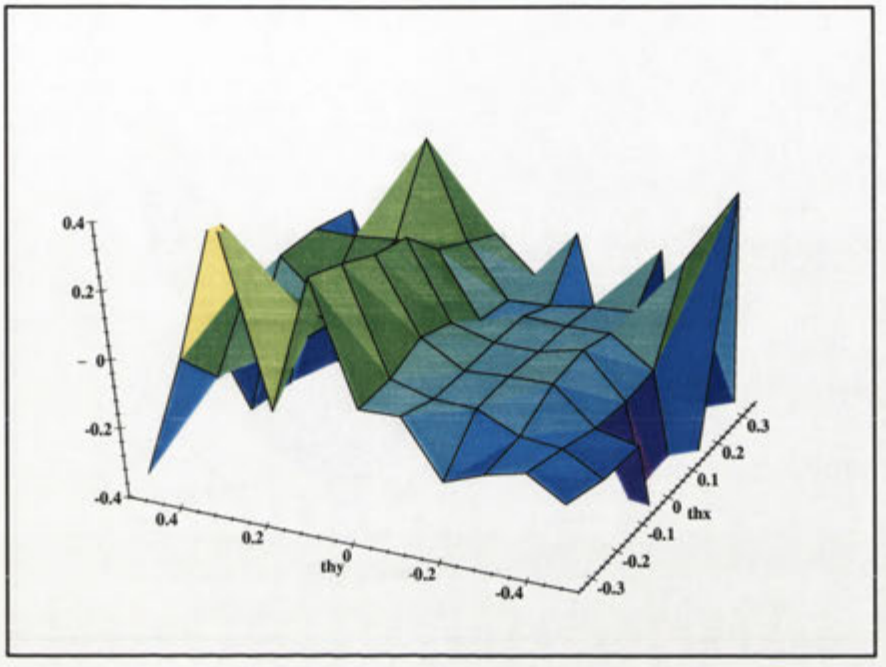
4.5.3 Mannequin tracking results

For the performance evaluation of the vision system, both the MARITA controller and the vision system were controlled by a supervisory program. The supervisory program moved the head into different positions and sampled the head pose measured by the vision system over a period of 500 frames for each head pose. During these measurements the mannequin head was motionless.

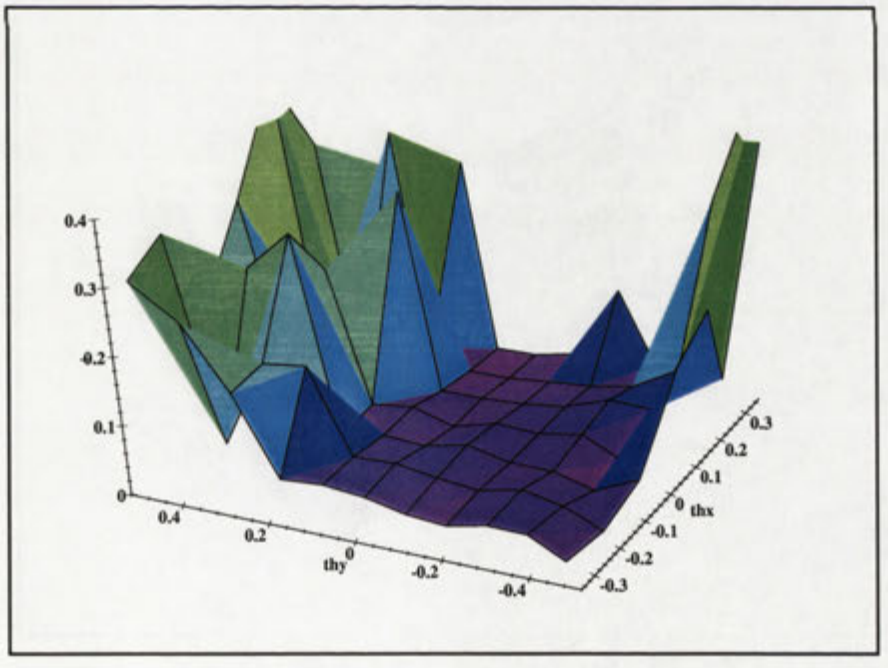
The parameters used to evaluate the performance are the bias (mean error) between the correct pose and the measured pose and the standard deviation of the head pose sample during the 500 measurements. These parameters closely relate to the systematic error and the sensitivity.

The spatial orientations used in the evaluation process are located on a grid in the (θ_x, θ_y) space evenly spaced with 0.1 rad, ranging from -0.5 to 0.5 rad for θ_y and -0.3 to 0.3 rad for θ_x. Overall 77 spatial orientations of the head were evaluated. All values are the raw pose estimation results. The values have not been scaled and biased to fit the correct values (such as [Azarbayejani et al. 1993]) and no filter was applied since the variance of the raw estimates is one of the performance parameters.

Figure 4.19a) shows the bias of the rotation about the x-axis and Figure 4.19b) the respective standard deviation. The area in the standard deviation plot where $\theta_y \in [-0.5 \ldots 0.2]$ the standard deviation is close to 0 which indicates reliable feature tracking and therefore accurate pose estimates. Only a few of the values in the -0.5 row contain large errors. The sections of the plot which contains larger errors correspond to configurations of the head where the feature tracking failed. The areas which show high values can be seen in Figure 4.19b). The tracking failures are mainly partial failures where some of the features on the hairline were lost. The estimate of the rotation around this axis is highly dependent on correct localisation of the feature triplets on the side of the head because the side triplets have a lower sensitivity for

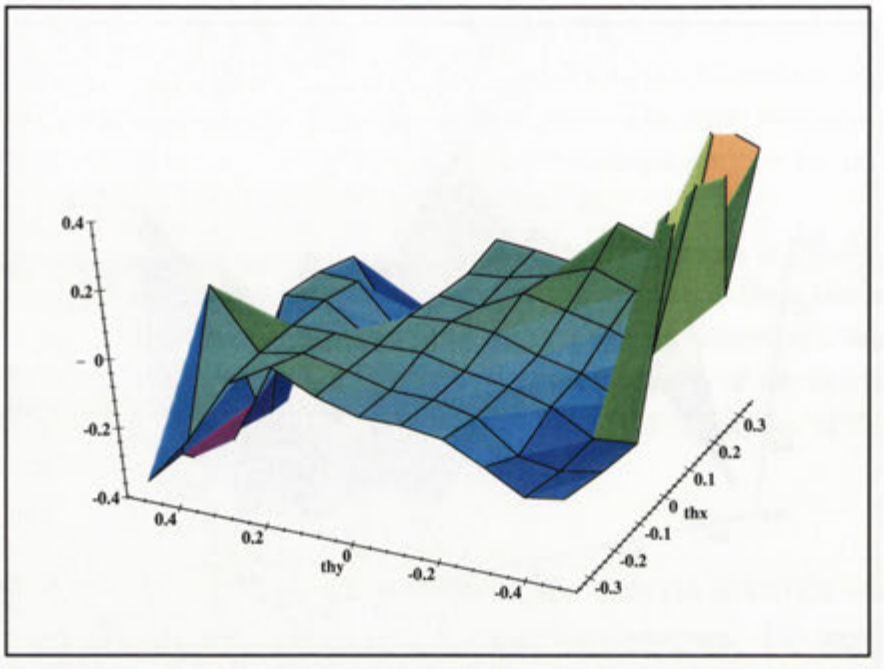


a)

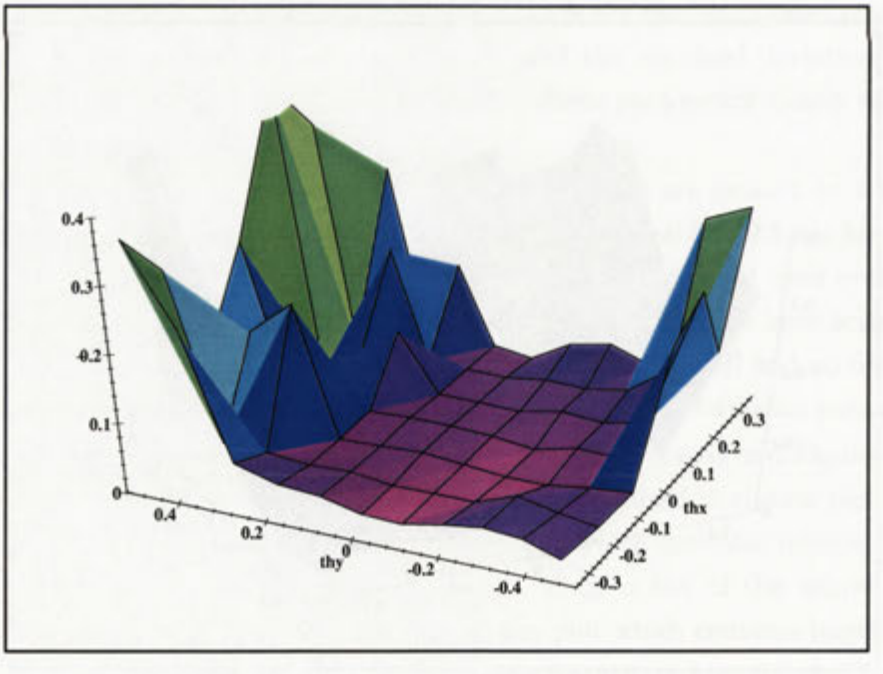


b)

Figure 4.19: Bias (a) and standard deviation (b) of Rot_x

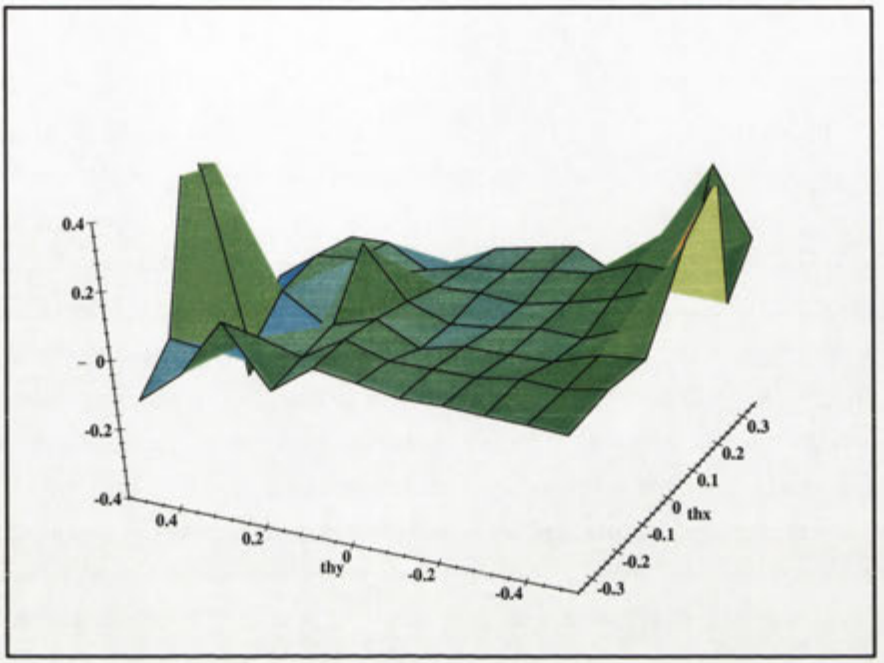


a)

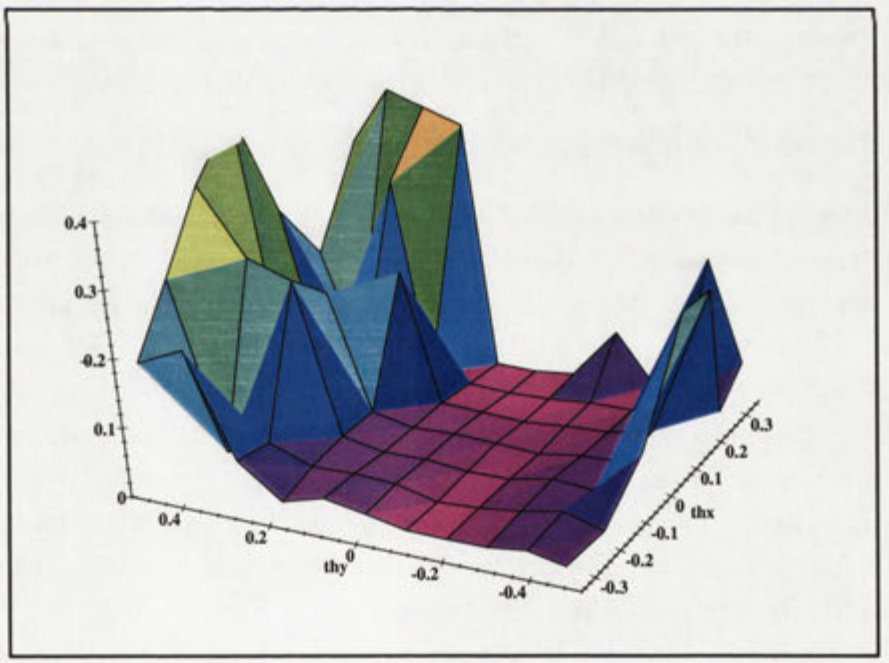


b)

Figure 4.20: Bias (a) and standard deviation (b) of Rot_y



a)



b)

Figure 4.21: Bias (a) and standard deviation (b) of Rot_z

	bias	standard deviation
Rot_x	1.691°	5.543°
Rot_y	−3.050°	3.794°
Rot_z	1.560°	2.969°

Table 4.1: Parameter estimation analysis

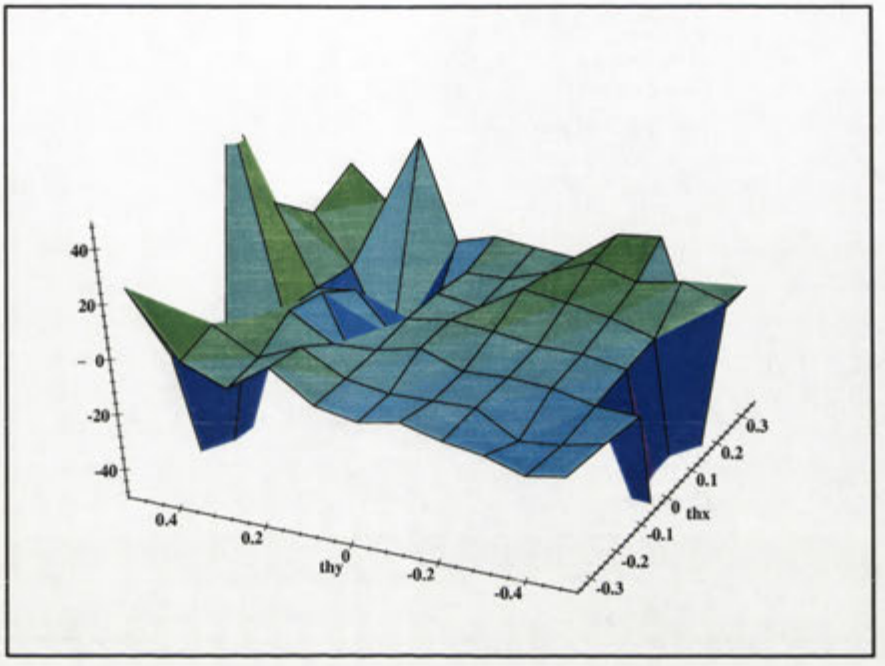
this axis. These features could not be localised properly for rotations around the y-axis larger than $\theta_y > 0.2$ due to the inhomogeneous lighting conditions. Hence, the rotation around the x-axis could not be determined reliably. In configurations in which the template tracking provided stable results over all 500 frames the standard deviation was 0.045 rad (2.57°) on average. The mean absolute bias of Rot_x for spatial orientations with successful feature localisation (68%) was 0.10479 rad (6.0°).

The results from the estimates of the rotation around the y- and z-axis are plotted in Figure 4.20 and Figure 4.21. These estimates show much smaller errors. The standard deviation is close to 0 in the same area as in Figure 4.19b) which shows that the different pose parameters are affected in a similar way by tracking errors. For large head rotations around the y-axis a (>30°) the feature localisation failed resulting in abrupt changes in the magnitude of the error. The mean absolute bias of Rot_y is 0.06699 rad (3.8°) for 75% of configurations and 0.04075 rad (2.3°) for 74% in Rot_z. The standard deviation plots for Rot_y in Figure 4.20b) and Rot_z in Figure 4.21b) are almost identical to the plot of the standard deviation of Rot_x in Figure 4.19b).

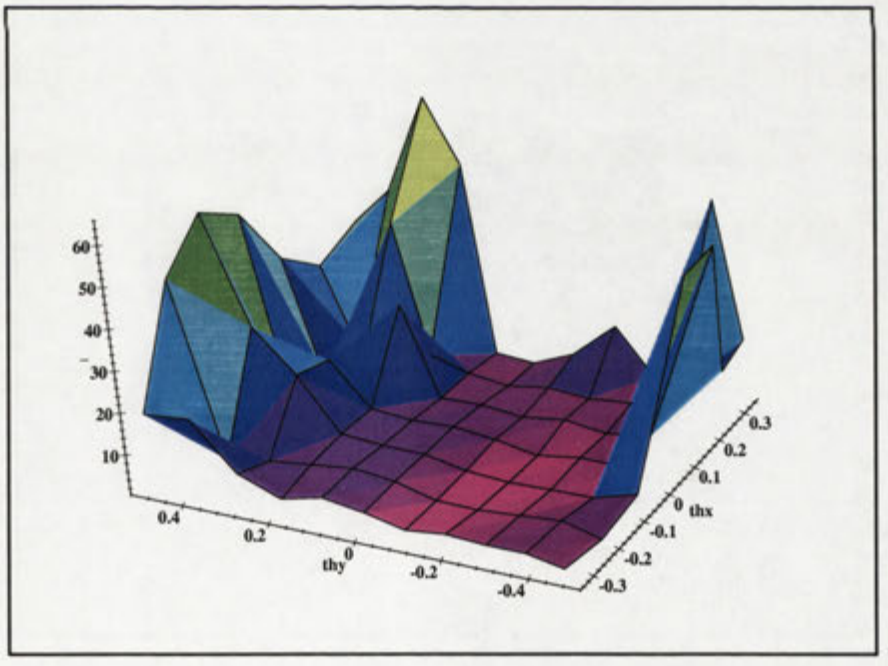
The overall angular distance Δ between the correct and the measured spatial orientation is 0.14098 rad (8.1°) for the 68% of the configurations when feature localisation was successful.

The Figures 4.22, 4.23 and 4.24 show the bias and standard deviation of the estimates for the translation in the direction of the x-, y- and z-axes. The plots of the standard deviations allow the same conclusions as the standard deviation plots of the estimates of the rotational parameters of the head pose. The feature tracking worked robustly for $\theta_y \in [-0.5 \ldots 0.2]$ while the high values on the left side of the plots indicate tracking failures. Both the x- and y-translations were well estimated as expected, the mean absolute bias was 6.97mm and 14.08mm respectively in the x- and y-direction. As expected from a monocular system, the error for the depth measurement was one magnitude higher. On average the z-translation contains an error of 60.04mm in the well tracking areas which corresponds to 10% of the overall distance (600mm) of the head from the camera.

Figure 4.25 shows the trace plot for an experiment where the mannequin head was continuously rotated into random orientations. The plots show both the correct parameter value marked as a continuous line and the estimates in each frame as dots. The estimates of the three rotational parameters have different characteristics as can be seen from Table 4.1. It should be noted that all values represent the error of the

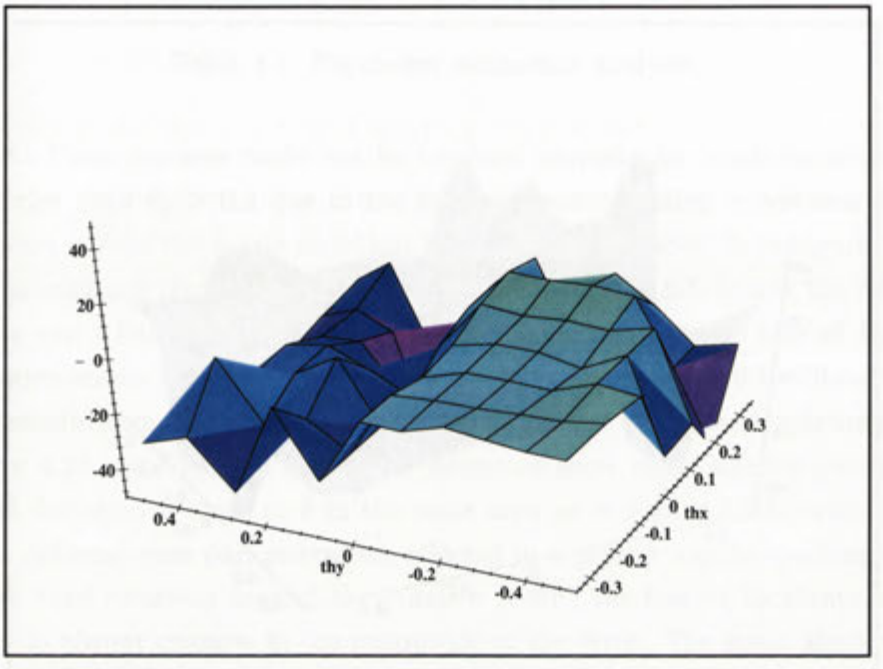


a)

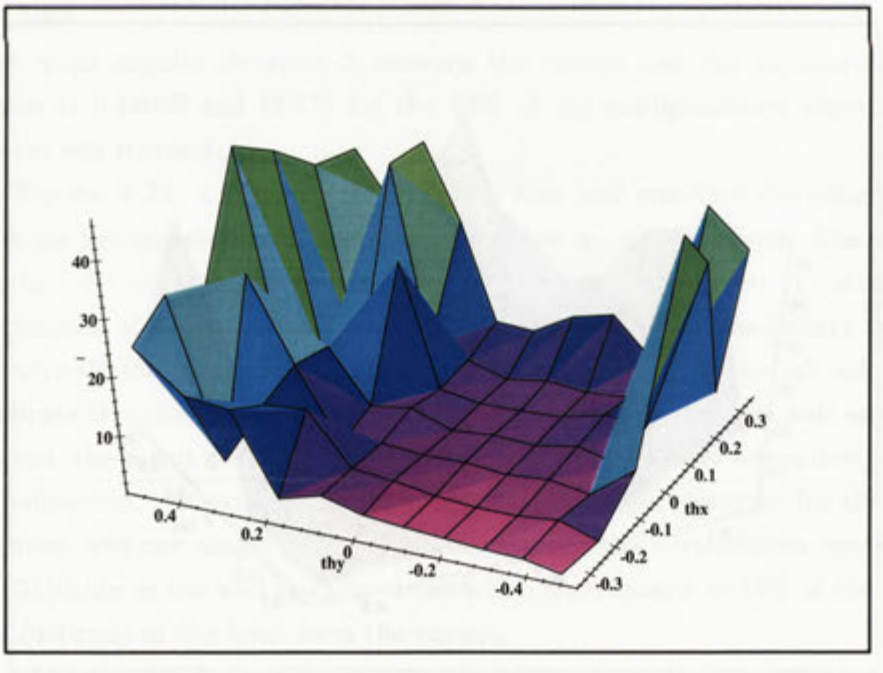


b)

Figure 4.22: Bias a) and standard deviation (b) of the translation in x-direction

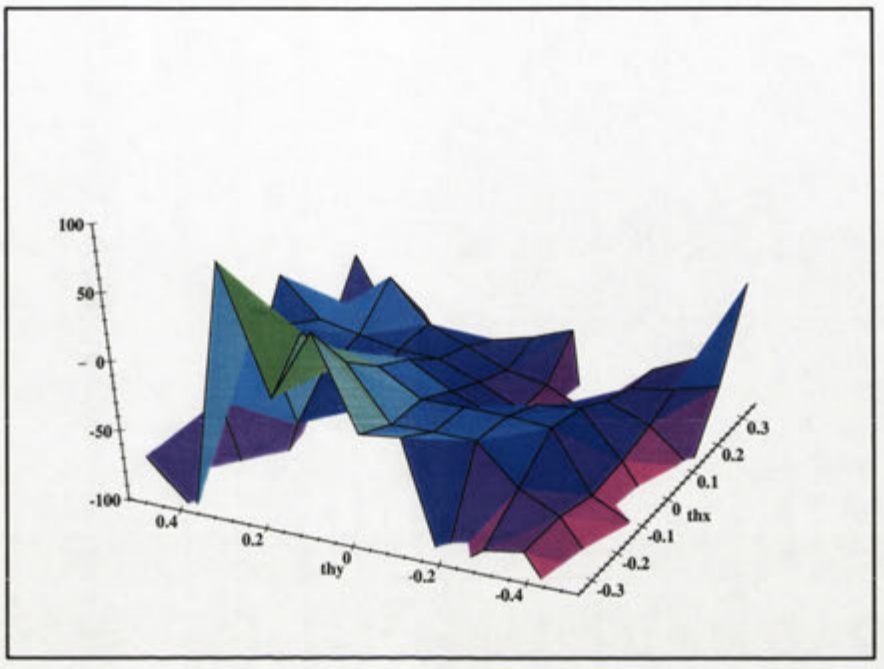


a)

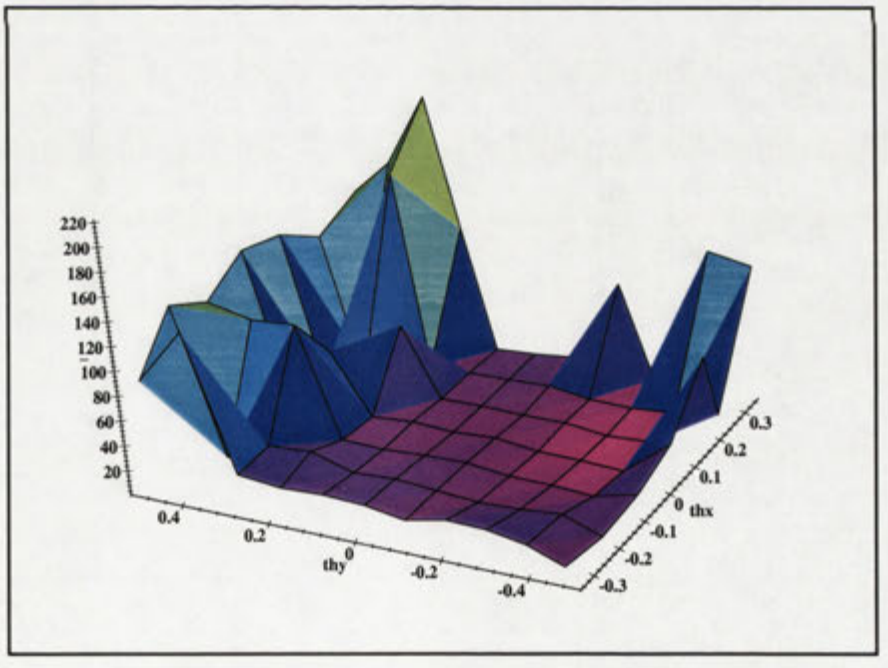


b)

Figure 4.23: Bias a) and standard deviation (b) of the translation in y-direction



a)



b)

Figure 4.24: Bias a) and standard deviation (b) of the translation in z-direction

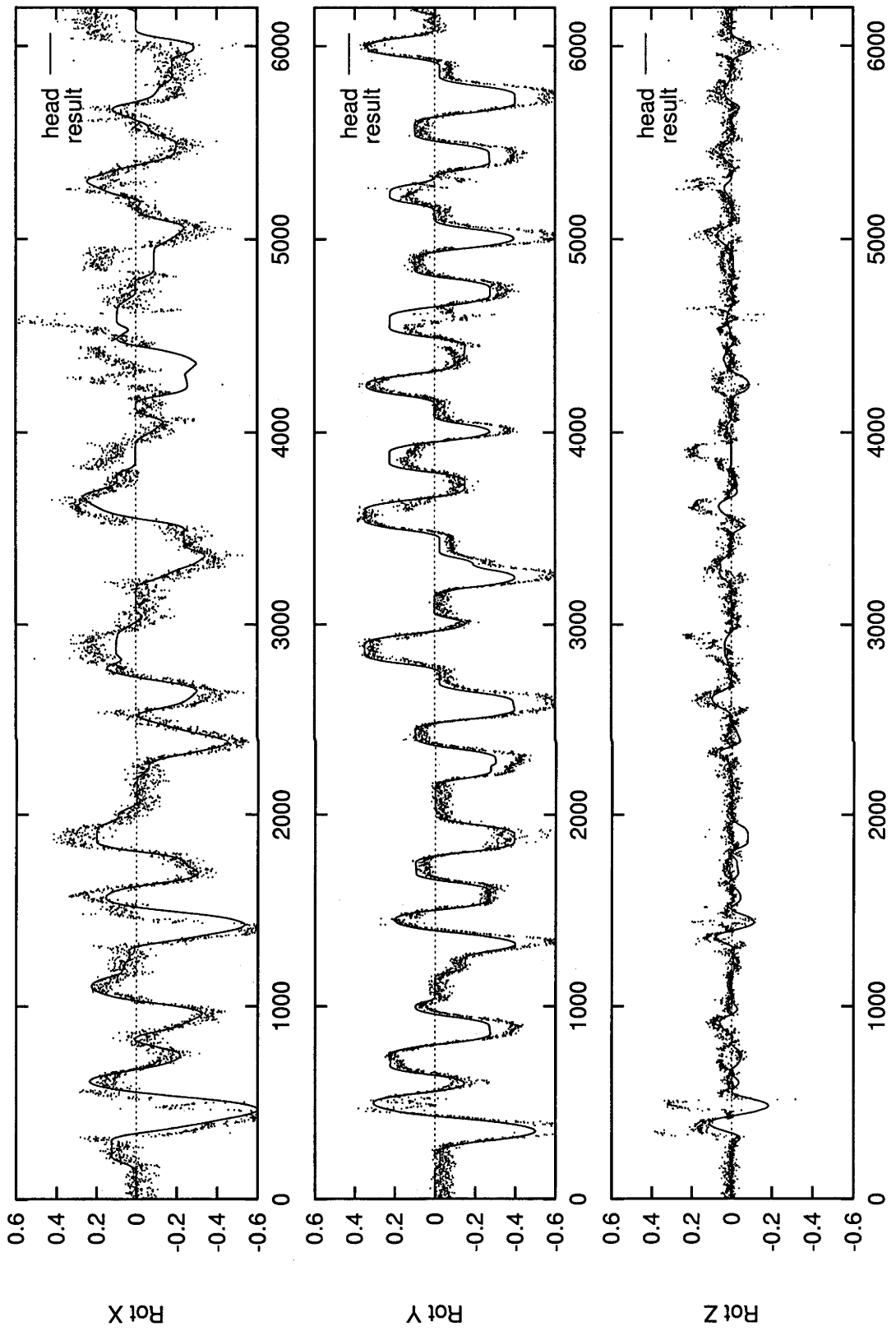


Figure 4.25: Rot_x, Rot_y and Rot_z over 6000 frames

complete sequence in this experiment.

The bias of the error in Rot_x is small with 1.691° while the standard deviation is high with 5.543°. It is visibly larger than the standard deviation of the other two parameters. The large standard deviation is a result of the difficult localisation of features on the side of the head which are crucial for the calculation of Rot_x. The feature on the hair/skin boundary "slips" up and down along the hair line which causes the strong variation in the estimate of Rot_x.

However, these vertical oscillations of this particular feature only weakly affects the estimate of Rot_y which has opposite characteristics than the estimate of Rot_x. The bias of Rot_y is large with $-3.050°$ which is visible in the plot. The standard deviation is small with 3.794°. The high bias is a result of exaggerated negative rotation estimates which were caused by the systematic feature localisation. Due to the inhomogeneous illumination of the scene the frontal features were falsely localised for large negative head rotations causing the systematic exaggeration of negative Rot_y.

As stated previously, the transformation from the mechanical system to the internal representation of the spatial orientation causes small rotations in Rot_z. Similar to the results of the static pose estimation experiment this parameter was estimated robustly. With a bias of 1.560° and a standard deviation of 2.969° the rotation around the z-axes has the lowest error.

The results of the dynamic motion experiment confirm the accuracy results of the static pose estimation experiments. Unless the feature localisation fails due to the limitations of the fragile SAD template correlation of the vision hardware the system provides accurate estimates even during rapid head motion.

4.5.4 Real-head tracking experiments

Figures 4.26 and 4.27 show an image sequence of a face tracking experiment with a person displayed at 2 frames/sec. The green and red overlay triangles of the vision system shows the three feature triplets used to derive the 3D head pose. Green triangles indicate feature triplet which are visible and therefore used to some extent in the estimation of the head pose while red triangles currently are not visible and are ignored in the pose estimation. The centre triangle is spawned by the two eye brows and the centre of the mouth while the two triangles on the side of the head are spawned by an eyebrow, the edge of the hear line close to the ear and a corner of the mouth. Note that the triangles are arranged in a way such that for the most common head motions, rotations to the left and right, at all times at least two of the triangles are visible to the camera. This is necessary to resolve the twofold ambiguity of the alignment algorithm.

During this sequence the subject rotates his head more than ±45° to the left and right. For example, in images 8-12 the head is rotated to the right side and

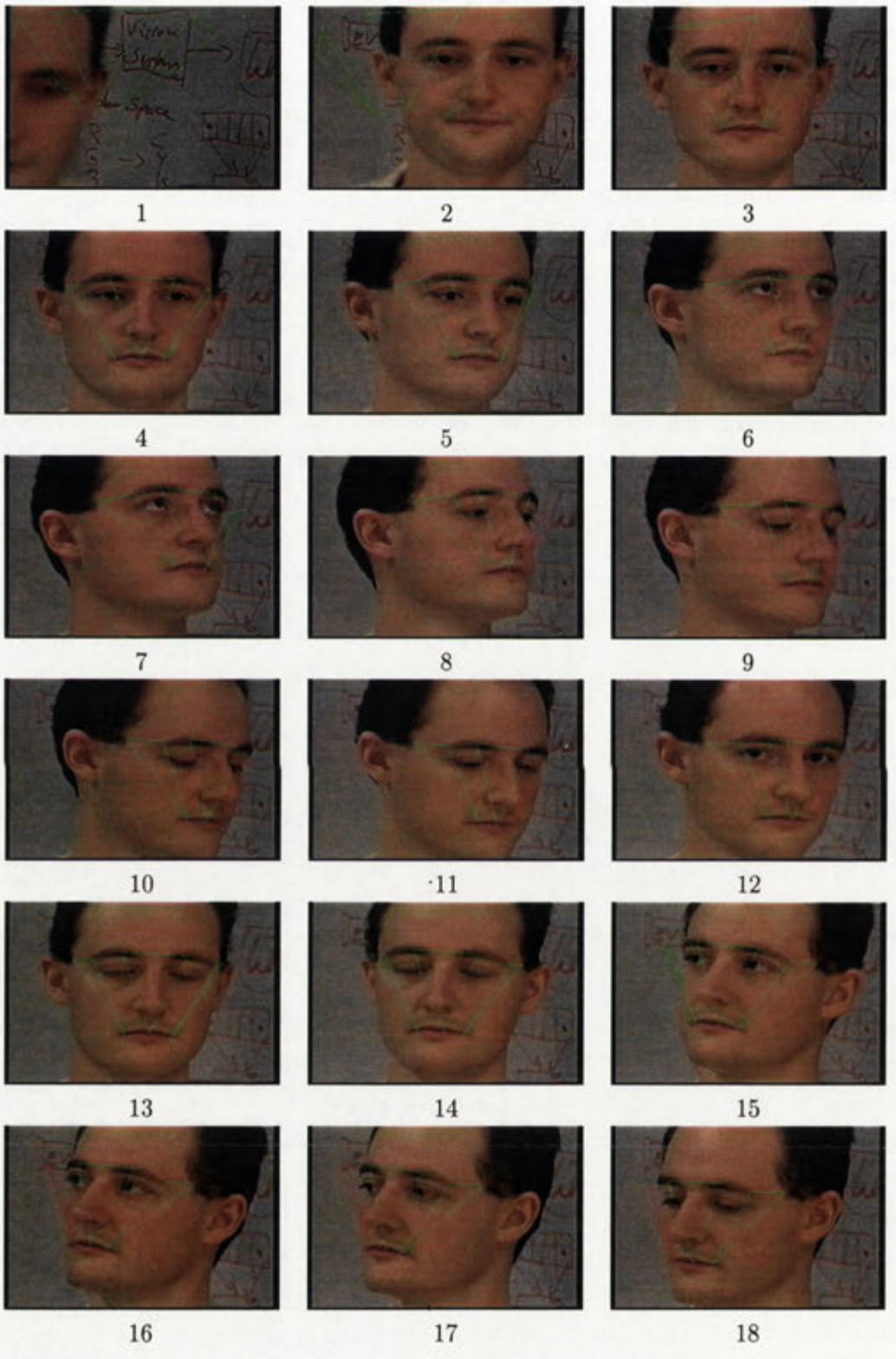


Figure 4.26: Tracking image sequence with pose triplets overlay (part 1)

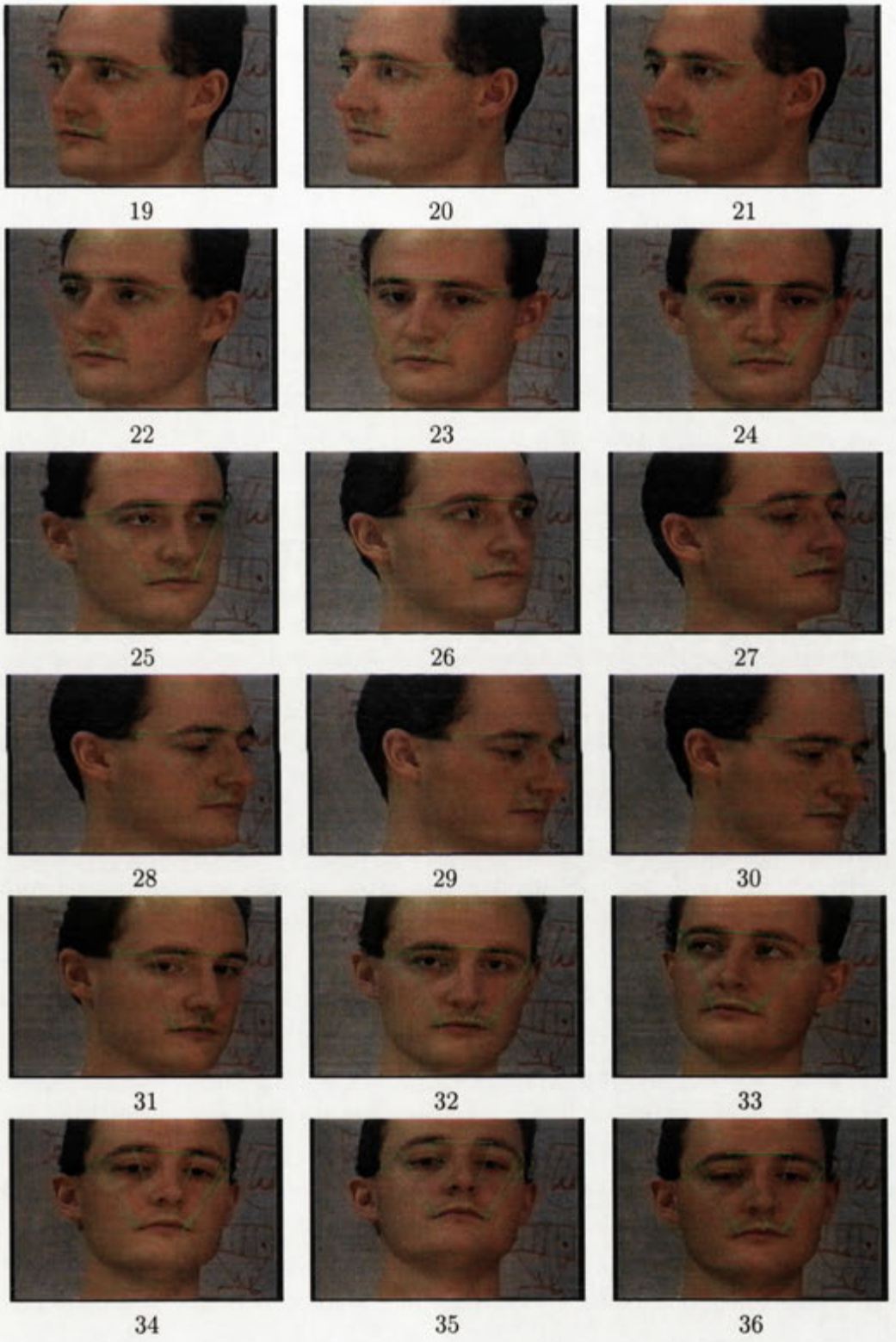


Figure 4.27: Tracking image sequence with pose triplets overlay (part 2)

the right triangle turns red. The remaining two triangles are sufficient to continue the 3D pose estimation which allows to recover the features of the hidden triangle as soon as they reappear in image 13. In the following images the subject turns his head to the left occluding the left side triangle which is marked red from image 16 to 22. Image 23 shows the transition between occlusion and recovery. The 3D head pose indicates that the features on the left side of the head should have reappeared and the triangle is marked green again. However, the respective features other than the mouth corner have not been recovered yet. The top side of the triangle is visibly offset from the head. In Image 24, only a fraction of a second later, the system has recovered the temporarily occluded features of left feature triplet. The remaining images show similar scenarios with large head rotations, for example image 29, and with the subject looking up.

The experiment shows how the multi-triplet approach not only allows to resolve the ambiguity of the pose estimation algorithm but also to continuously track and estimate the 3D head pose during large head rotations. In frontal head views the estimates of the centre feature triplet is subject to an increased sensitivity. This problem is resolved by incorporating the estimates from the side triplets and therefore the head pose can be estimated robustly for all head poses.

4.6 Summary

This chapter presented the algorithms and techniques used in the third layer of the face tracking and 3D head pose estimation system. The purpose of the third layer is to derive the 3D pose of the head from the 2D feature positions robustly determined by the second layer. The pose estimation is based on the 3-point alignment algorithm of Huttenlocher and Ullman which provides a closed form solution with only a twofold ambiguity. To address the problem of the systematic error caused by the discrepancies between a perfectly affine projection and the noisy real perspective projection a number of improvements to the original algorithm were presented which reduce the systematic angular error by 75% and the translational error by 90% in the geometry of the face tracking application. The varying sensitivity of the estimate is modelled by an approximative model which provides real-time estimates of the sensitivity. By incorporating multiple triangles into the head pose estimate not only the twofold ambiguity of the pose is resolved but also the resulting pose estimate has a low sensitivity for all spatial orientations of the head.

The proposed algorithms were implemented in the face tracking application for performance evaluation. Over a wide range of spatial orientations the system provides robust and accurate estimates. The limitations of the system's performance for larger rotations are a result of the failed feature localisation in the hardware layer. Overall the face tracking system represents a reliable visual interface which is capable to provide robust measurements for additional modules such as eye gaze estimation

and facial gesture recognition.

Chapter 5

Gaze Tracking and Gesture Recognition

Chapter 1 described a wide range of application areas for visual head pose and facial feature tracking systems. Chapter 3 described implementation and employed algorithms of such a system. This system forms the basis for a human-machine interface based on facial gestures such as nodding and shaking the head and the detection of the person's eye gaze. This chapter is concerned with the modelling of the eye geometry and the derivation of the 3D eye gaze direction from a monocular face view based on the results of the 3D head pose estimation system described in the previous chapter.

5.1 Underlying eye model

The estimation of the 3D eye gaze direction of a person requires the determination of the offset point in space and the spatial direction of the gaze vector. The offset point is derived by the 3D head tracking system described in Chapter 3. To derive the direction of the eye gaze the spatial orientation of the eye ball with respect to a head-independent world coordinate system is required. As described in Chapter 2 most commercial systems use corneal reflections to calculate the spatial orientation of the eye ball(s) since this method does not require an estimate of the 3D head pose. The passive methods detect the location of the iris in the image and derive the spatial orientation either from the distance relative to surrounding features [Baluja and Pomerleau 1994a] or by intersecting the ray from the camera centre through the image location of the iris with the eye ball as detected by a 3D head pose estimation system [Matsumoto and Zelinsky 2000]. Due to the high deformability of the skin surrounding the eyes the first approach has limited accuracy, particularly if the 3D head pose is not derived. For the second approach an accurate 3D head pose estimate is crucial since a ray from the camera must intersect the eye ball sphere at a distance of 1-2m. This approach has only been implemented on a stereo camera

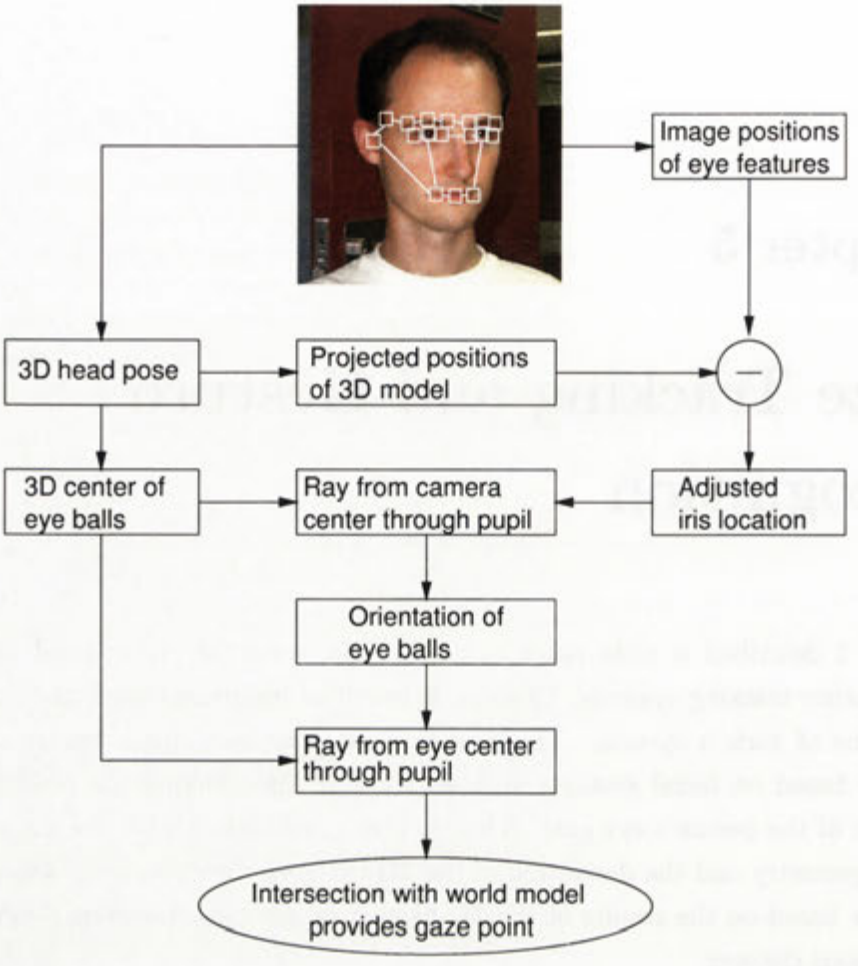


Figure 5.1: Schematic of the gaze vector calculation

system which provides highly reliable 3D position measurements of the location of facial features [Matsumoto and Zelinsky 2000]. The system described in this thesis incorporates the strong points of both passive eye gaze direction measurement methods.

5.1.1 General architecture

The architecture of the eye gaze estimation system is illustrated in Figure 5.1. The face image represents the functions of the visual face tracking systems. It provides both the 3D head pose and the image positions of the features, in particular the iris and the corners of the eyes. The following calculations involve both 2D and 3D results of the face tracking system.

The 3D model of the head also includes the centre of the eye balls. The 3D pose estimation of the head provides the current 3D position of the eye balls in space. The model points of the corners of the eye are projected into the image plane to compare the model position with the 2D location of the feature in the 2nd level of the face tracking system. This discrepancy is considered when the image position of

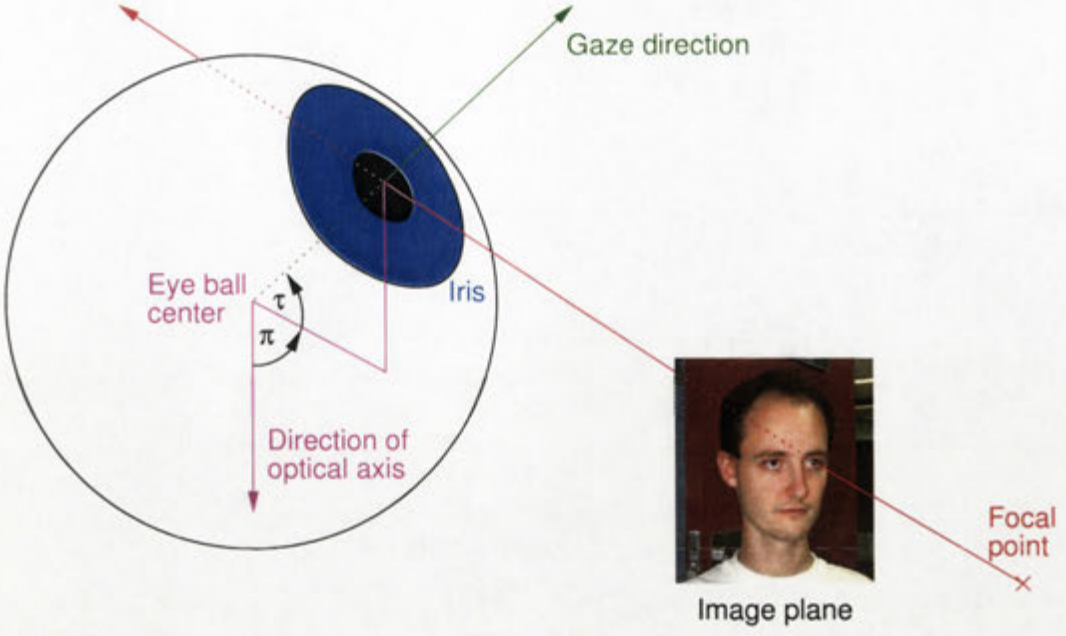


Figure 5.2: The underlying model of the eye gaze direction calculation

the iris is used to derive the orientation of the eye ball in space. The orientation is calculated by intersecting a ray from the optical centre of the camera through the corrected iris position in the image plane with the eye ball. The resulting gaze vector is offset by the already calculated 3D position of the eye ball to derive the spatial eye gaze vector. The gaze point of the person is calculated by intersecting the eye gaze vector with a 3D model of the environment.

5.1.2 Eye gaze direction calculation

The underlying model of the gaze direction estimator is shown in Figure 5.2. For each eye three features are located by the vision system, the iris and the inner and outer corners of the eye. The localisation of all three features is based on template matching. The following calculations are performed for each eye individually.

Let $\mathbf{i}', \mathbf{c}_r', \mathbf{c}_l' \in \mathbb{R}^2$ denote the image positions of the left and right corner and the iris of the eye measured by the vision system, $\mathbf{b} \in \mathbb{R}^4$the 3D centre of the eye ball in the 3D face model in face coordinates, and $P, C \in \mathbb{R}^{4\times4}$ the homogeneous transformation which describe the pose of the head and camera respectively in world coordinates. The centre of the eye ball in camera coordinates $\mathbf{b}_c$ is then given by

$$\mathbf{b}_c = C^{-1}P\mathbf{b} \tag{5.1}$$

The next step is to calculate the ray from the focal point of the camera through the iris. This 3D line is shown in red in Figure 5.2. It could be directly derived from $\mathbf{b}_c$ and the image location $\mathbf{i}'$ of the iris similar to the purely model based method described by [Matsumoto and Zelinsky 2000] which was discussed in Chapter 2.

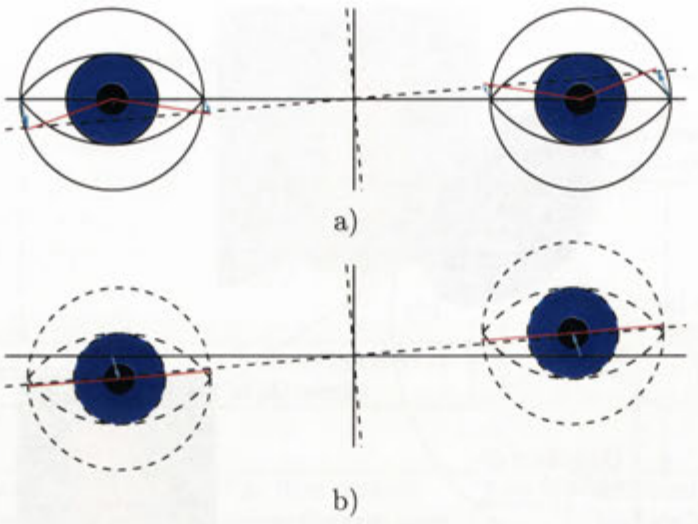


Figure 5.3: Iris location correction

Figure 5.3 shows how a more accurate estimate of the eye gaze vector can be derived. Figure 5.3a) shows the true position of the eyes in the image and the true axis of the face coordinate system as solid black lines. Both the head and the eyes point straight into the camera. The dashed lines indicate the orientation of the measured head pose which shows small errors in the rotation about the y- and z-axis. The turquoise arrows indicate the offset of the corner features caused by this small error in the head pose estimate. The result with a purely model based gaze estimation would be that the left eye points to the top left corner and the right eye points to the bottom right corner.

Figure 5.3b) shows how the robustness of the gaze vector estimate is improved. The underlying assumption is that in well stable tracking situations the image positions of the eye corner features derived by the image correlation system are more accurate than the projected positions derived from the 3D model. This assumption is well justified in situations in which most visible features are tracked well. In these situations the search window for the respective corner features are located above the actual feature and because the corners of the eye are relatively robust features the image correlation is able to locate the feature robustly. The difference between the more accurate correlation results $\mathbf{c}'$ and the projected positions $\mathbf{a}'$ is used to adjust $\mathbf{b}_c$. Figure 5.3b) shows the relocation of the assumed eye ball positions. The location of the eye ball $\mathbf{b}_c$ that is used in the following steps is calculated according to

$$\mathbf{b}_c = C^{-1}P\mathbf{b} + \begin{bmatrix} \frac{f_l\left(\mathbf{c}'_r+\mathbf{c}'_l-\mathbf{a}'_r-\mathbf{a}'_l\right)}{2b_z} \left(\in \mathbb{R}^2\right) \\ 0 \\ 1 \end{bmatrix} \tag{5.2}$$

where f_l is the focal length of the camera.

This adjustment step could be omitted if perfect feature extraction results can be

achieved with the vision system where the measured positions would be equal to the projected positions of the model. In case of feature tracking failures the overall pose of the head may be estimated inaccurately. However, small errors in the location of the eye ball lead to large errors in the orientation estimate of the eye as the radius of a human eye is ~12mm as opposed to a distance of the eye to the camera of ~1000mm. Therefore, the local adjustment using the corner features as references improves the gaze direction estimate's robustness considerably under most tracking conditions encountered with real video imagery.

Given an estimate of the eye ball location the line which intersects the pupil is calculated with the focal centre $\mathbf{f}$ as the origin and $\mathbf{e} \in \mathbb{R}^4$ connecting the focal point with $\mathbf{e}'$ in the image plane.

$$\mathbf{e} = \begin{pmatrix} e'_x, & e'_y, & f_l, & 1 \end{pmatrix}^t \tag{5.3}$$

The intersection of this ray with the eye ball yields the location of the centre of the iris $\mathbf{i} \in \mathbb{R}^4$ as shown in Figure 5.2. Therefore, we require $x \in \mathbb{R}$ such that $\mathbf{i} = 0 + x\mathbf{e}$. The expressions for the calculation of x can be simplified by translating the origin of the considered base from from the focal centre to the centre of the eye ball. Therefore, $\mathbf{f}$ becomes $-\mathbf{b}_c$ and the origin is coincident with the centre of the eye. The equation to solve for x can now be simplified to

$$r^2 = \|\mathbf{f} + x\mathbf{e}\|_2 = (f_x + xe_x)^2 + (f_y + xe_y)^2 + (f_z + xe_z)^2 \tag{5.4}$$

where r is the radius of the eye ball. This yields a quadratic equation of the form

$$(\mathbf{e} \cdot \mathbf{e})\, x^2 + 2\,(\mathbf{e} \cdot \mathbf{f})\, x + \mathbf{f} \cdot \mathbf{f} - r^2 = 0 \tag{5.5}$$

which provides the two solutions x_1 and x_2 if the ray intersects the eye ball. If the ray is a tangent to the eye ball $x_1 = x_2$ and if the ray does not have any points in common with the eye ball Equation 5.5 does not have a solution. The correct solution for the iris location is $x = \min(x_1, x_2)$ since it can be assumed that the iris is on the side of the eye ball visible to the camera[1]. The location of the iris $\mathbf{i}_c$ in camera coordinates is easily derived with the coordinate transformations eye/head which is known and head/camera which is provided by the head pose measurements. This yields directly the eye gaze vector $\mathbf{g}_c$ in camera coordinates and the pan and tilt angles π and τ between $\mathbf{g}_c$ and the facial normal.

$$\begin{aligned} \mathbf{g}_c &= \mathbf{i}_c - \mathbf{b}_c \\ \pi &= -\arcsin \mathbf{i}[1] \\ \tau &= \arcsin \mathbf{i}[2] \end{aligned} \tag{5.6}$$

[1]Otherwise, if the iris is on the hidden side of the eye ball, the gaze can not be determined.

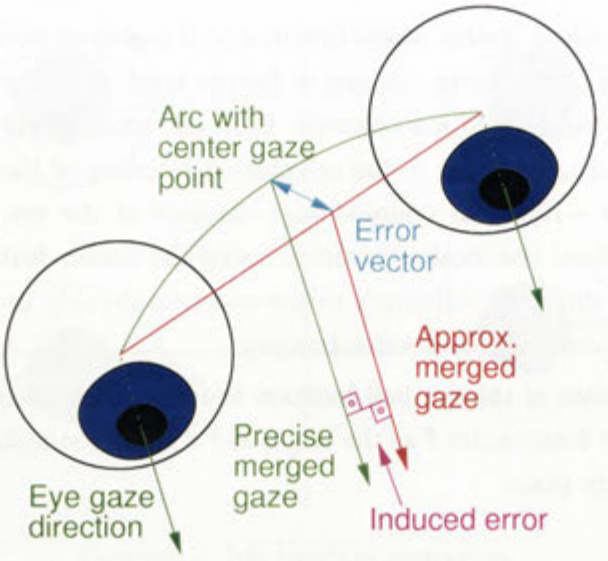


Figure 5.4: Systematic error introduced by the approximation when merging the two gaze vectors.

5.1.3 Merging results from two eyes

The proposed method provides two independent measurements for the gaze direction of the person in each instance. It can be assumed that the person is looking with both eyes at the same point in space. Therefore, merging the two gaze directions provides (a) a more robust estimate of the gaze direction and the gaze point and (b) a unique result. To maximise the robustness for difficult tracking situations, for example when the head is rotated to the side and the location of the eye corners of one eye only can be measured, the gaze angles are merged weighted according to the variance of the feature locations of the corner of the respective eye. The merged gaze angles are calculated as

$$\begin{bmatrix} \pi \\ \tau \end{bmatrix} = \frac{1}{\frac{1}{v_{ll}+v_{lr}} + \frac{1}{v_{rl}+v_{rr}}} \begin{bmatrix} \frac{\pi_l}{v_{ll}+v_{lr}} + \frac{\pi_r}{v_{rl}+v_{rr}} \\ \frac{\tau_l}{v_{ll}+v_{lr}} + \frac{\tau_r}{v_{rl}+v_{rr}} \end{bmatrix} \tag{5.7}$$

where v_{xy} is the y-corner of the x-eye. The combined eye gaze vector $\mathbf{g}$ is calculated based on the 3D head pose and the merged gaze angles π and τ.

The origin $\mathbf{o}_g$of the combined eye gaze vector $\mathbf{g}$ is on the line which connects the the centres of the two eye balls and is weighted along this line in a similar way as the gaze angles.

$$\mathbf{o}_g = \frac{\frac{\mathbf{b}_l}{v_{ll}+v_{lr}} + \frac{\mathbf{b}_r}{v_{rl}+v_{rr}}}{\frac{1}{v_{ll}+v_{lr}} + \frac{1}{v_{rl}+v_{rr}}} \tag{5.8}$$

The weighting of the origin reduces the offset error caused by the different origins of the eye gaze lines of the two eyes. For similar variances of the two eyes the gaze angles are merged with similar weights and therefore the resulting gaze line should

have an origin in between the two eye balls. In this case the calculated gaze line intersects the true gaze point accurately given that the system derived correct gaze angles. One of the eyes can be lost when the user turns the head to the side. Then one of the eyes can not be found by the vision system and tracks with high variances. In this case the gaze angles of the well tracking eye are weighted heavily and the calculated gaze line must originate in the centre of the well tracking eye ball to intersect the true gaze point accurately.

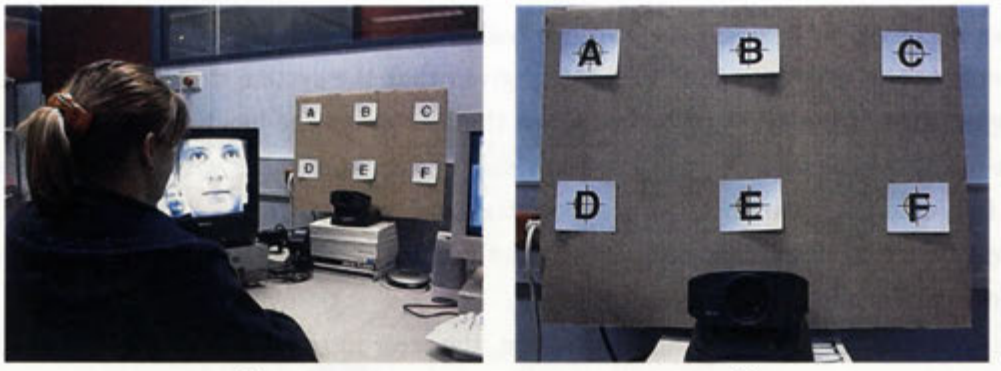
Although this is only an approximation, the error introduced is only small for the geometric configurations that are typical for face tracking applications. Figure 5.4 shows where the error occurs in the merging process. To derive the precise origin of the merged gaze direction ray, the origin $\mathbf{o}_g$ would need to be weighted along the circle around the true gaze point $\mathbf{s}$ in space that intersects both centres of the eyes. In this case, the change of the gaze angle is linear with the length of the arc between $\mathbf{o}_g$ and $\mathbf{b}_l$ and $\mathbf{b}_r$ respectively. However, since the distance of the true gaze point to the eyes is not known, this arc can not be parametrised. Considering that the distance between the eye balls $\|\mathbf{b}_l - \mathbf{b}_r\|_2$ is small compared to the radius r_c of the circle $\|\mathbf{b}_l - \mathbf{b}_r\|_2 \ll r_c$, the difference between the arc and the straight connection line is small. Moreover, the part of the error vector of $\mathbf{o}_g$ that actually takes effect later in intersections with the world model is the projection of this error vector onto a plane perpendicular to $\mathbf{g}$. For approximately frontal views of the face and small gaze angles π and τ the error vector are almost aligned with $\mathbf{g}$ and therefore its projection is considerably smaller than the error vector itself.

5.1.4 Summary

For reasonably frontal views of the face the proposed algorithms produce good results with low errors and high robustness. With increased rotation angles of the head, the errors increase, as the exact location of the facial features becomes more imprecise. Since the algorithm relies on robust tracking results from at least one eye, large head rotations are not tolerated. Failure to locate the corners of the eyes reliably directly causes failure in the calculation of the gaze direction. For rotations of more than 45° of the head relative to the camera it becomes difficult to locate features for pose and gaze estimation.

5.2 Experimental results

The eye gaze estimation system has been extensively tested with people. The experimental setup to evaluate the accuracy of the system is shown in Figure 5.5a). For testing a person looks at a board with the six letters A to F arranged as shown in Figure 5.5b). The letters are augmented with a central cross hair to allow the person to fixate the gaze accurately. The camera which observes the persons face is located



a) b)

Figure 5.5: Experimental setup for the measurements of the gaze estimators accuracy

directly underneath the letter board. The distance between the face and the board in this setup is 980mm, the spacing between the letters is 210mm both horizontally and vertically. This corresponds to $\sim 12^{\circ}$ divergence in eye gaze direction between the individual letters.

In the experiment the person looks at the letters A to F consecutively while the system estimates the gaze direction. Figure 5.6 shows the scatter plot of one performance of the experiment which lasted about 30 seconds. In each video frame the intersection between the eye gaze vector and the letter board is calculated and marked as a black cross in the plot. The accumulation of crosses around the letters is clearly visible. Also, the transitions of the gaze point from one letter to the next can be identified easily.

The recording was started when the person initially looked at the centre of the board and the transition to the letter A can be seen in the plot. The gaze point estimate for the letter A was corrupted by poor tracking results of the corners of the eyes or the irises. The feature tracking system was unable to recover from this particular error and produced an inaccurate but stable result. In the other five cases the feature tracking produced sufficiently accurate results. During the tracking of the letters B, C and D the person blinked once in each instance, and a small deviation of the gaze point from each letter is visible.

A set of images from the vision system during this experiment are shown in Figure 5.7. In the first image the subject is looking at the centre of the board. In the following images the person looks at the six letters respectively and in the last image the gaze returns to the centre of the board. The blue overlay of the vision system represents the letters on the letter board. A small blue cross-hair marks the current measured gaze point. If the system is able to match the gaze point to a letter, the letter is highlighted with a blue box. Note that the person changes both the head pose and the orientation of the eyes with respect to the head during this experiment. Simple appearance based systems such as described by [Baluja and Pomerleau 1994a]

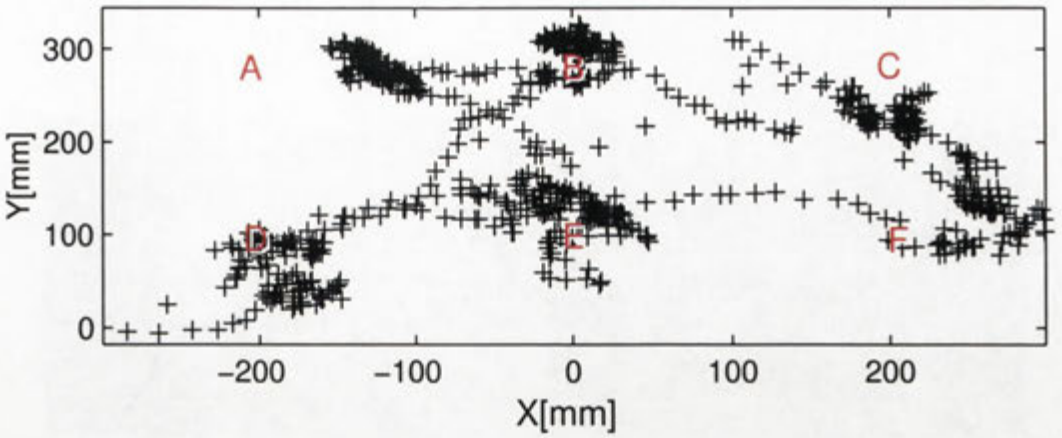


Figure 5.6: Trace plot of the eye gaze vector intersected with the target plane

Target	Average error	Standard deviation
A	80.5mm, 4.7°	12.4mm, 0.7°
B	22.6mm, 1.3°	11.4mm, 0.6°
C	57.4mm, 3.3°	24.6mm, 1.4°
D	41.7mm, 2.4°	21.0mm, 1.2°
E	47.3mm, 2.7°	19.4mm, 1.1°
F	55.7mm, 3.2°	15.6mm, 0.9°
Total	50.9mm, 2.9°	17.4mm, 1.0°

Table 5.1: Error analysis of the eye gaze tracking

are unable to track the gaze point when a user moves his/her head freely[2].

The same experiment was repeated eight times, the results are shown in Table 5.1. The average error and standard deviation from the correct target locations are shown in the target plane and angle in the gaze point estimate for each letter individually and in total for the whole experiment. The average error is ~5mm in the target plane corresponding to just under 3° in angular error in the gaze direction estimate. For a frontal view of the face and the eye in the given setup, 3° correspond to a movement of the iris of 1.3 pixels in the image. Note that the iris location is measured using template correlation with an oversampling factor of 3 and therefore can only be determined up to an accuracy of ±1.5 pixels. The average result of 1.3 pixels was achieved by low pass filtering the correlation results and merging the results from both eyes.

Note that the standard deviation is subject to the amount of filtering applied to the data. The system uses first order low pass filters for all visual measurement which are configured not to inhibit the tracking of the fast eye movements but rather to filter out the noise caused by the oversampled correlation calculations.

[2]The video sequence of this experiment is included on the CD-ROM included in this thesis. Please refer to Appendix E for details.

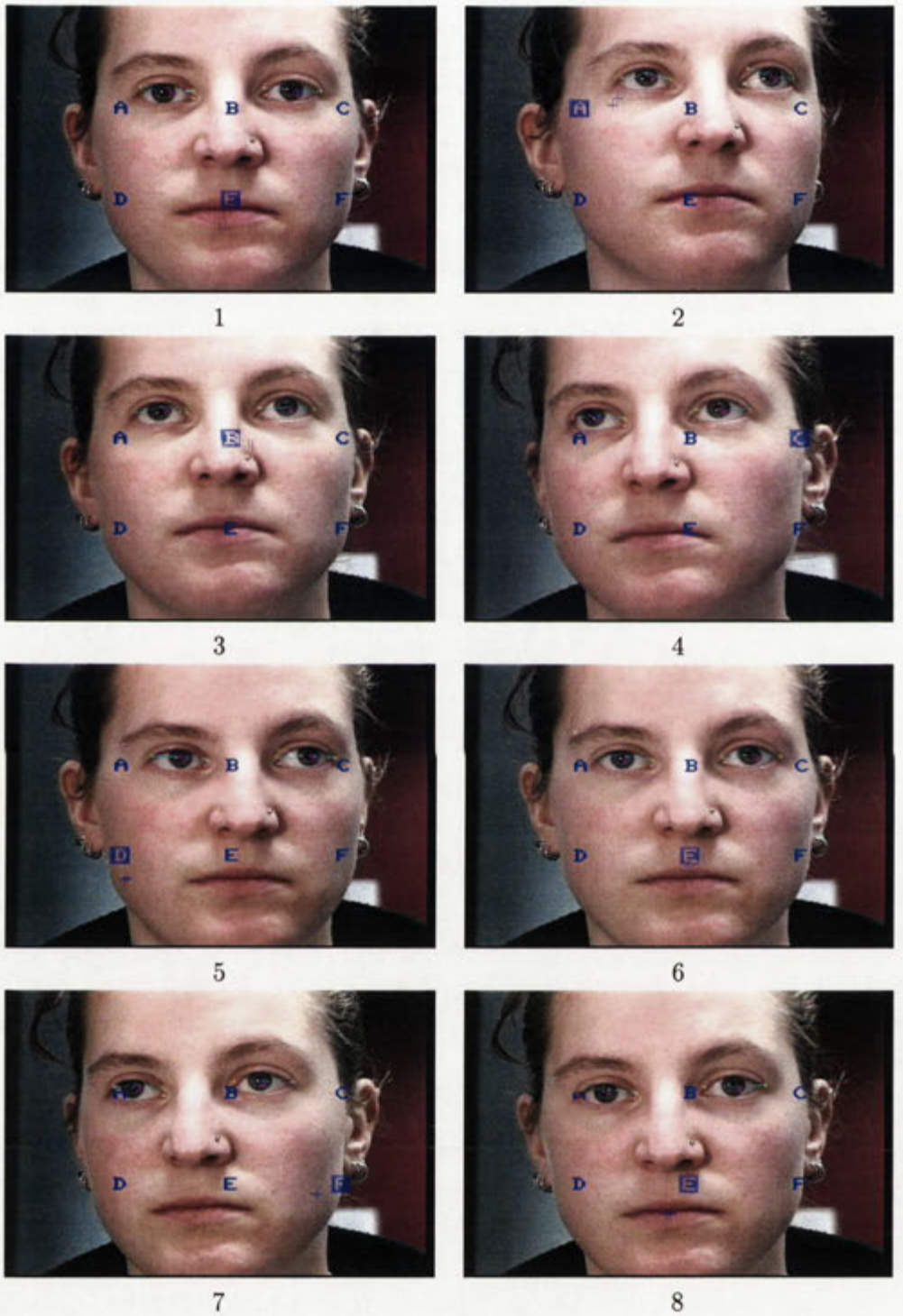


Figure 5.7: Tracking sequence

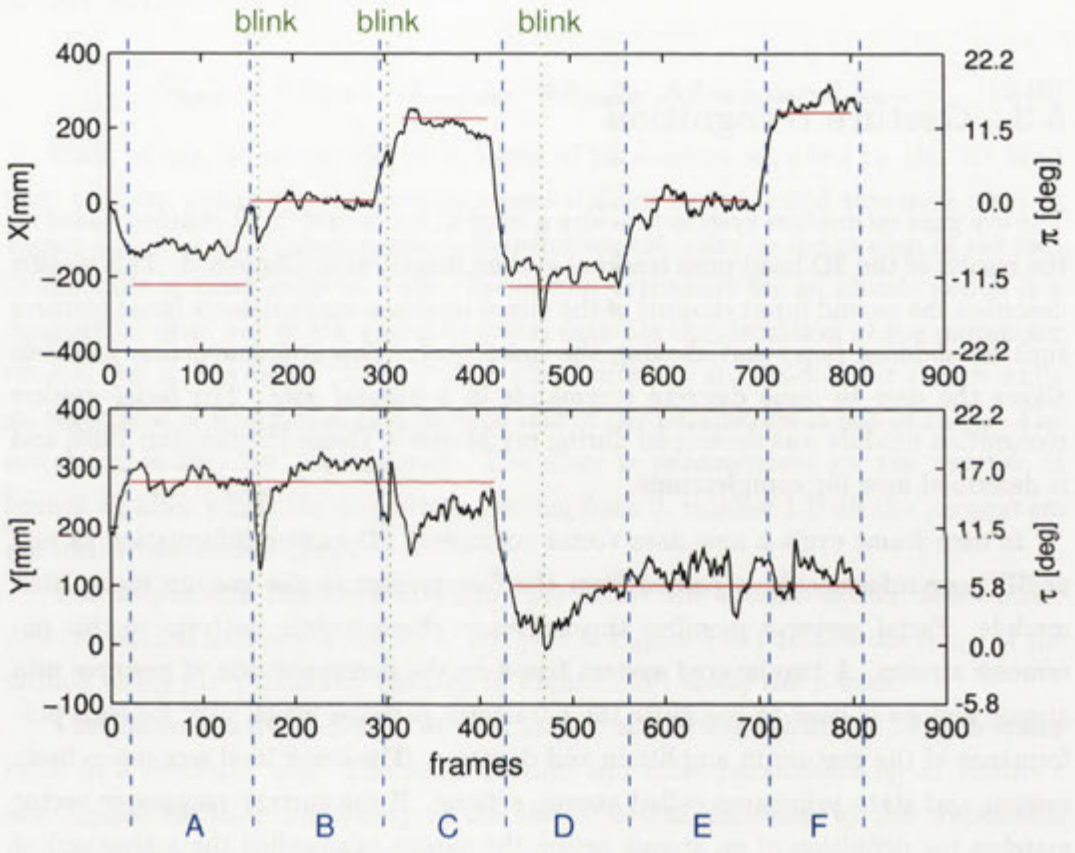


Figure 5.8: Eye gaze trace plot for the pan and tilt of the gaze direction, measured in mm on the left side and in degrees on the right side.

Figure 5.8 shows the trace of the gaze point parameters in the experiment described in Figure 5.6. The units along the y-axes are shown in milli-meter on the letter board on the left side of the plot and in degrees of the gaze angle relative to the optical axis of the camera on the right side. The blue dashed lines indicate the time intervals where the subject was looking at the letter denoted in the bottom line of the graph. The red lines show the correct values for the targets. The green lines indicate the instances when the person blinked. In all three cases a strong deviation of the gaze point (and direction) was the result of the blink. The affected results could be marked as invalid by a process that monitors the state of the eyes.

5.3 Gesture recognition

The eye gaze estimation system provides a continuous deictic input channel based on the results of the 3D head pose tracking system described in Chapter 3. This section describes the second input channel of the visual interface which detects facial gestures such as nodding (*yes*) and shaking the head (*no*). This communication channels allows the user to issue discrete commands in a natural way. The facial gesture recognition module was developed during my Master's Thesis [Heinzmann 1996] and is described here for completeness.

In each frame cycle a new data vector containing 2D motion information as well as 3D pose information is passed from the face tracker to the gesture recognition module. Facial gestures manifest themselves as characteristic patterns in this parameter stream. A two layered system based on the decomposition of gestures into atomic actions is used to recognise the parameter patterns which vary for each performance of the gesture in amplitude and duration. The lower level recognises basic motion and state primitives called atomic actions. If the current parameter vector matches the definition of an atomic action the output (also called the activation) of the atomic action is increased. A pattern classification mechanism for time varying vector patterns classifies the activation patterns of the atomic actions into different gestures.

This method differs significantly from earlier proposed gesture recognition methods such as [Darrell and Pentland 1993] as described in Chapter 2. The main focus in the development of this gesture recognition system was on the real-time capability for a large range of gestures. The goal was to develop a system which allows the online recognition of gestures without the necessity to look back at previous motion data. The processing of a particular input parameter vector must be performed in real-time. The following sections describe the mechanisms employed to achieve this goal.

5.3.1 Gesture decomposition

The basic idea of the gesture recognition module is to decompose the gestures in sequences of uniform movements or states called atomic actions. For example the *nod* gesture can be defined as a sequence of accelerations and deaccelerations in the y-direction:

$$G_{nod} = AA_{up} \cdot AA_{down} \cdot AA_{up} \cdot AA_{down} \tag{5.9}$$

A gesture that uses states instead of motion such as the *blink* gesture can be defined similar to the *nod* gesture:

$$G_{wink} = AA_{open} \cdot AA_{halfopen} \cdot AA_{closed} \cdot AA_{halfopen} \cdot AA_{open} \tag{5.10}$$

Each atomic action is defined in terms of parameters supplied by the 3D head pose tracking system and respective ranges defined by trapezoid functions such as Figure 5.9. The parameters used in the definition can refer to the motion of the face or the state of features or to both. The output activation for an atomic action is a measure to what extent the current motion data fits the definition of the parameter ranges. An activation of 1 occurs when all parameters are inside their ranges while an activation of 0 indicates that at least one of the parameters is out of range. The activation is also low pass filtered. The filter is parametrised by the number of frames Φ_i after which the activation, starting from 0, reaches 1 if all the parameters are inside the defined range.

The trapezoidal functions in Figure 5.9 define the atomic action *down* which detects vertical motions of the face. The plot in Figure 5.9a) defines the range of the motion along the x-axes and the plot in Figure 5.9b) along the y-axes.

The internal activation $a_i(t)$ of each atomic action i is calculated in each frame cycle in a recursive way. The facial motion and state parameters p_{ij} of feature i are checked for their conformity to the atomic action definition by the trapezoidal functions Tr_{ij}. The product of the trapezoid function values works like a logical *and* operator for the different range definitions for the atomic action.

$$a_i(t) = \min\left(a_i(t-1) + \frac{1}{\Phi_i}, 1\right) \prod_j Tr_{ij}(p_{ij}(t)) \tag{5.11}$$

The internal activation increases linearly when all input parameters are within the predefined ranges. To achieve a better motion pattern discrimination the square value of the internal activation is used as externally visible activation.

$$e_i(t) = (a_i(t))^2 \tag{5.12}$$

Figure 5.10 shows the graph of the external output activation pattern of an atomic action. Initially the activation is 0 and increases over the time Φ during

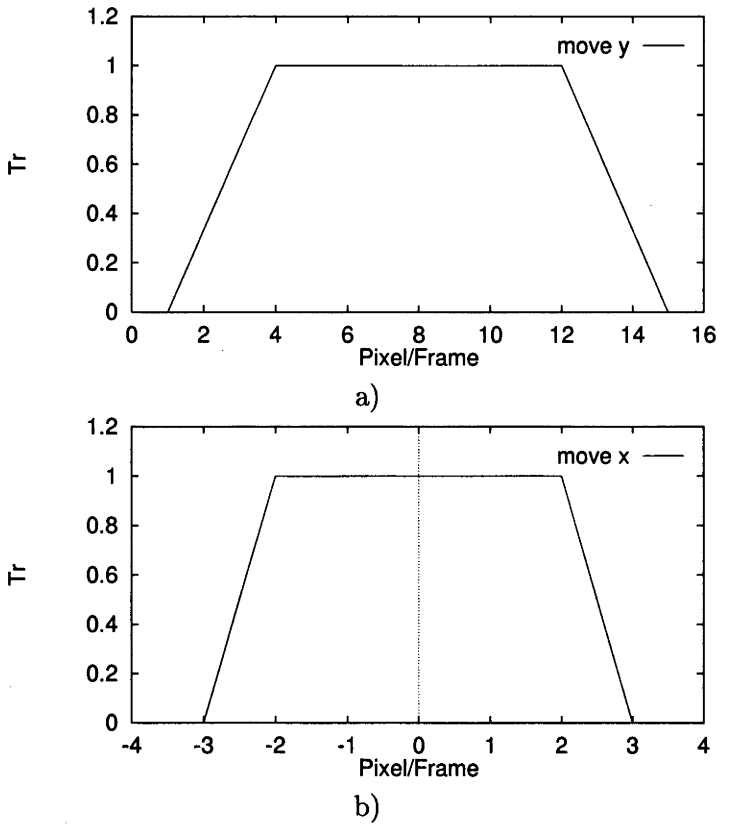


Figure 5.9: Trapezoid functions which define the *down* action.

which the parameters of the input parameters are within the inner range of the trapezoid functions. When the activation reaches 1 the activation can increase no further and returns to 0 as soon as one of the input parameters leaves the predefined range.

5.3.2 Gesture definition and recognition

Particular gestures produce characteristic patterns in the motion parameters derived by the face tracker and in the activation of the atomic actions. A flexible recognition system is required to match the atomic action patterns and ignore the minor differences in the performance.

The gestures are defined as a sequence of atomic action activations and the time constraints between the individual activations. This sequence is observed by a finite state machine which is instantiated when the first atomic action in the sequence is activated. The state machine observes the activation of the consecutive atomic actions in the sequence. It performs the state transitions when the next atomic action in the sequence is activated and the time constraints for the transition are met. The time constraints for the transitions are defined by trapezoid functions similar to the ranges of the motion parameters. The variable of the transition trapezoid function is the time t_s in frames since the state machine entered the current state. Therefore

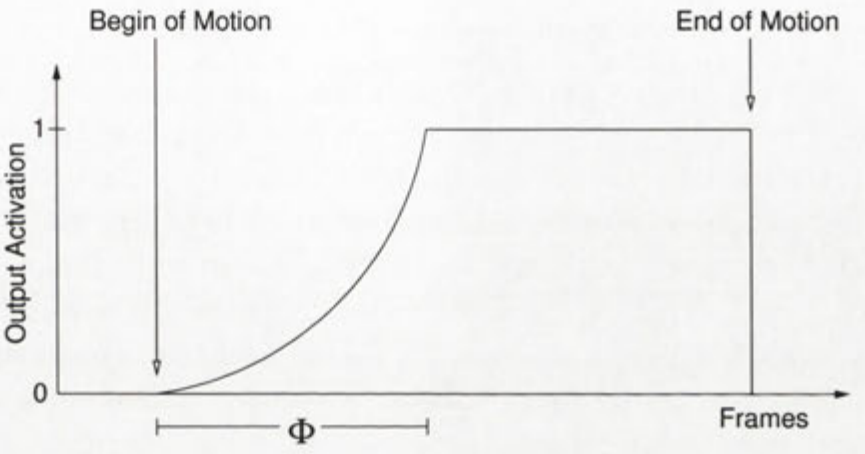


Figure 5.10: Typical activation pattern

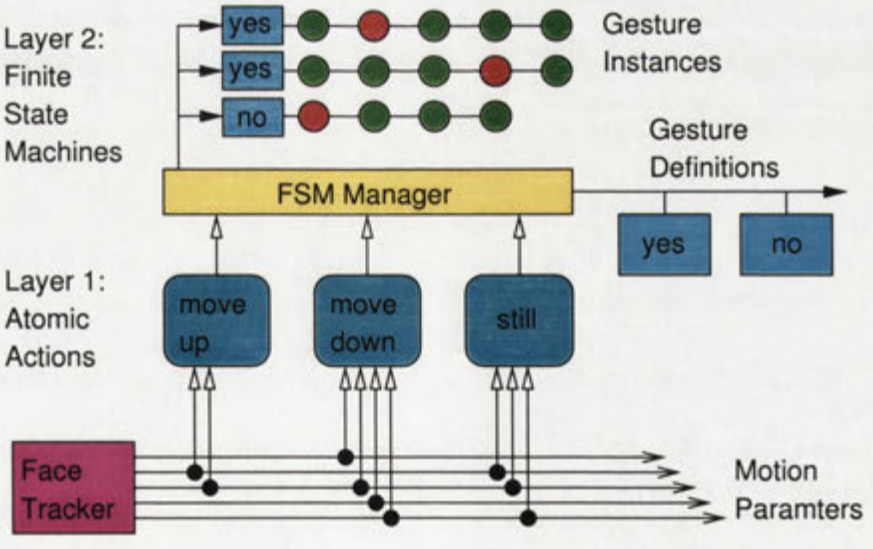


Figure 5.11: Architecture of the gesture recognition system

the transition trapezoid function defines a time window in which the next transition can occur. To make the transition to the next state $l+1$ the function $S_{kl}(t_s)$ of gesture k must have a falling edge from a 1-level.

$$S_{kl}(t) = e_{kl}(t) \cdot Tr_{kl}(t_s) \tag{5.13}$$

The multiplication of e_{kl} and Tr_{kl} requires that the next atomic action must be activated inside the time frame defined by the transition trapezoid function. The transition is made when the trapezoid reaches its falling edge or if the activation of the atomic action e_{kl} decreases while Tr_{kl} is still active. If no valid falling edge is detected before Tr_{kl} returns to zero the activation pattern of the gesture did not match the predefined gesture pattern and the finite state machine is deleted by the FSM manager. If the state machine reaches the last transition in the sequence all atomic actions have been activated inside the specified time constraints and the gesture is recognised.

At the same time there can be more than one instance of the state machine for a specific gesture. Especially cyclic gestures like nodding or shaking the head can have more than one instance. This is necessary because a gesture, especially if it is represented by a long sequence of atomic actions, can be missed if it starts while the state machine is not in its initial state.

Figure 5.11 shows the architecture of the gesture recognition system. At the lower level the atomic actions pre-process the motion data and produce their activation patterns. The activations are monitored by the FSM manager in the second layer. Every time an atomic action is activated one instance of the finite state machine is created for each gesture that starts with the activated atomic action. The FSM manager keeps track of the different instances and reports completed gesture recognitions with their respective accuracy.

5.3.3 Gesture vocabulary

The system is able to detect a large number of motion and state gestures in real-time. The following gestures are defined in the system and can be recognised:

- *yes* and *no*: Multiple up-and-down and left-and-right motion respectively
- *up*, *down*, *left* and *right*: Only one short motion in the respective direction and a return to the neutral position.
- *turn left* and *turn right*: Rolling the head clockwise and anti clockwise respectively.
- *maybe*: Tilting the head to the left and right multiple times.
- *blink*: Blinking with both eyes simultaneously.

- *wink left* and *wink right*: Winking once with the left and right eye respectively.

Because of the efficient processing of the data with no look-back requirements many more gestures could be defined and recognised in real-time.

Figure 5.12 shows examples of motion parameter plots from a test run which are then processed by the atomic actions. For the image velocity parameters in x- and y-direction the unit is pixels per frame, the 3D tilt angle is scaled to 30°/ unit.

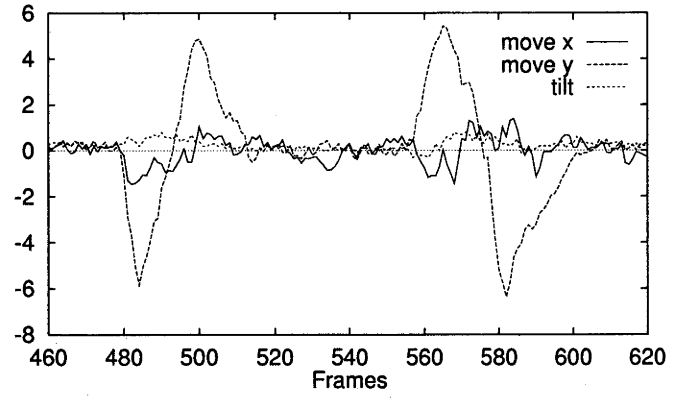
Figure 5.12a) shows the output generated by a nodding gesture. The tilt angle and the movement in x-direction are small, while the movement in y-direction oscillates with an amplitude of about 6 pixels per frame. First, the head is raised generating a negative peak. Then the oscillation of the y-movement follows. Figure 5.12b) shows the *left* and *right* gestures. They are similar to the nodding and shaking, but only one cycle is performed. Figure 5.12c) shows the *turn left* gesture which uses both the x- and y-motion. When the person rotates his/her head clockwise or anti clockwise the x- and y-motions have one cycle shifted by 90°. Figure 5.12d) shows the *maybe* gesture. The person tilts his/her head to the left and to the right generating a significant oscillation of the tilt angle. The x- and y-motion also show small oscillations caused by the displacement of the face during tilting.

Figure 5.13 shows an image sequence from a gesture recognition experiment.

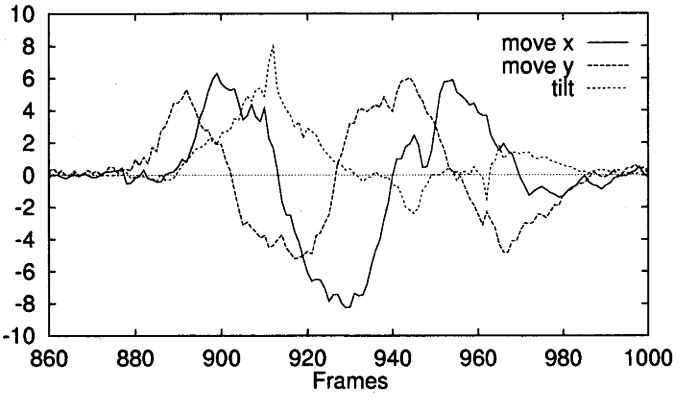
5.4 Summary

This chapter described two independent modules for human-machine interfaces. The gaze point estimation system allows the detection of the orientation of the eye balls in space, and therefore the calculation of the gaze point in combination with a 3D model of the environment. The system incorporates of the two predominant passive methods that have been proposed previously for gaze direction estimation. The gaze direction is derived from the spatial position of the eye ball and the image location of the iris. This method provides a high accuracy and robustness to different facial expressions *if* the head pose can be calculated accurately. Because of the geometric relationships small errors in the spatial position and orientation of the head cause large errors in the estimation of the eye gaze direction. This problem is solved by including two features which are largely unaffected by facial expressions into the calculations. The inner and outer corners of the eyes provide landmarks that are used to correct the eye ball position and the resulting eye gaze direction is more robust to head pose estimation errors.

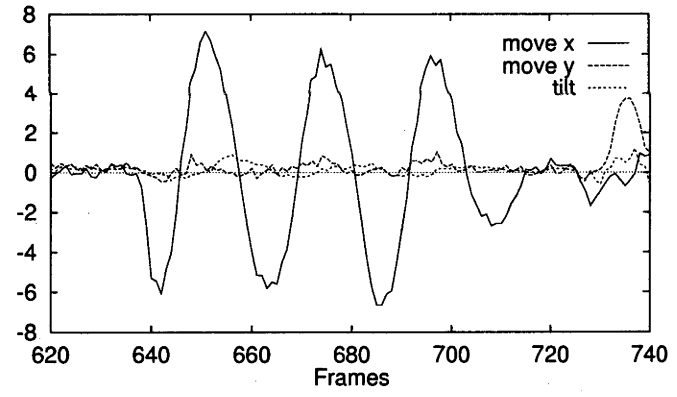
The presented gesture recognition system is a flexible pattern recognition system which allows the online detection of characteristic patterns in a vector stream. It's main advantage is the low computational cost of the algorithm which does not require to look at the history of the input data. The modelling accuracy and discrimination ability of the method is well suited for a proposed set of 12 gestures.



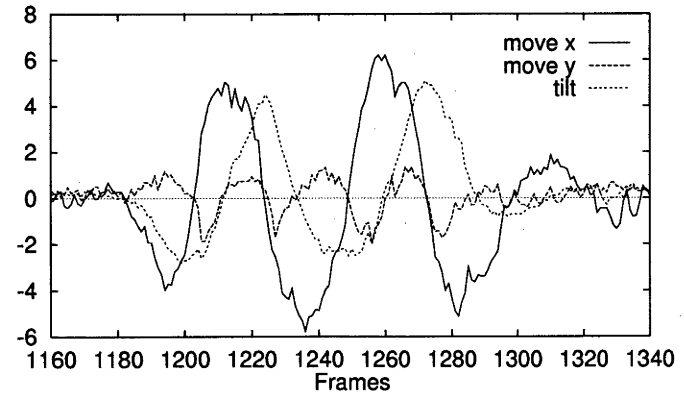
a) nod



c) turn left



b) left and right



d) maybe

Figure 5.12: Tracking Information passed to the gesture recognition module

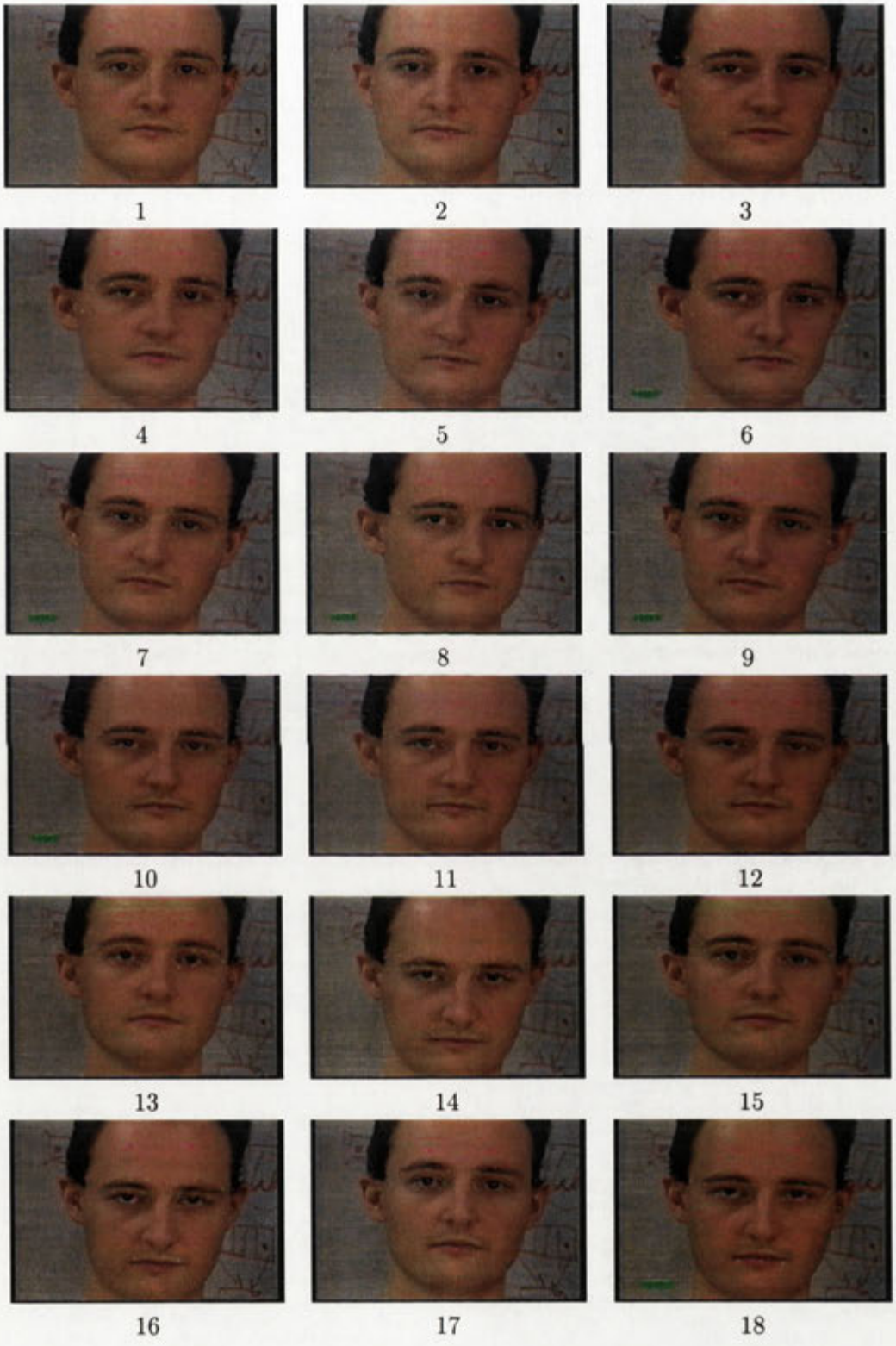


Figure 5.13: Gesture recognition image sequence

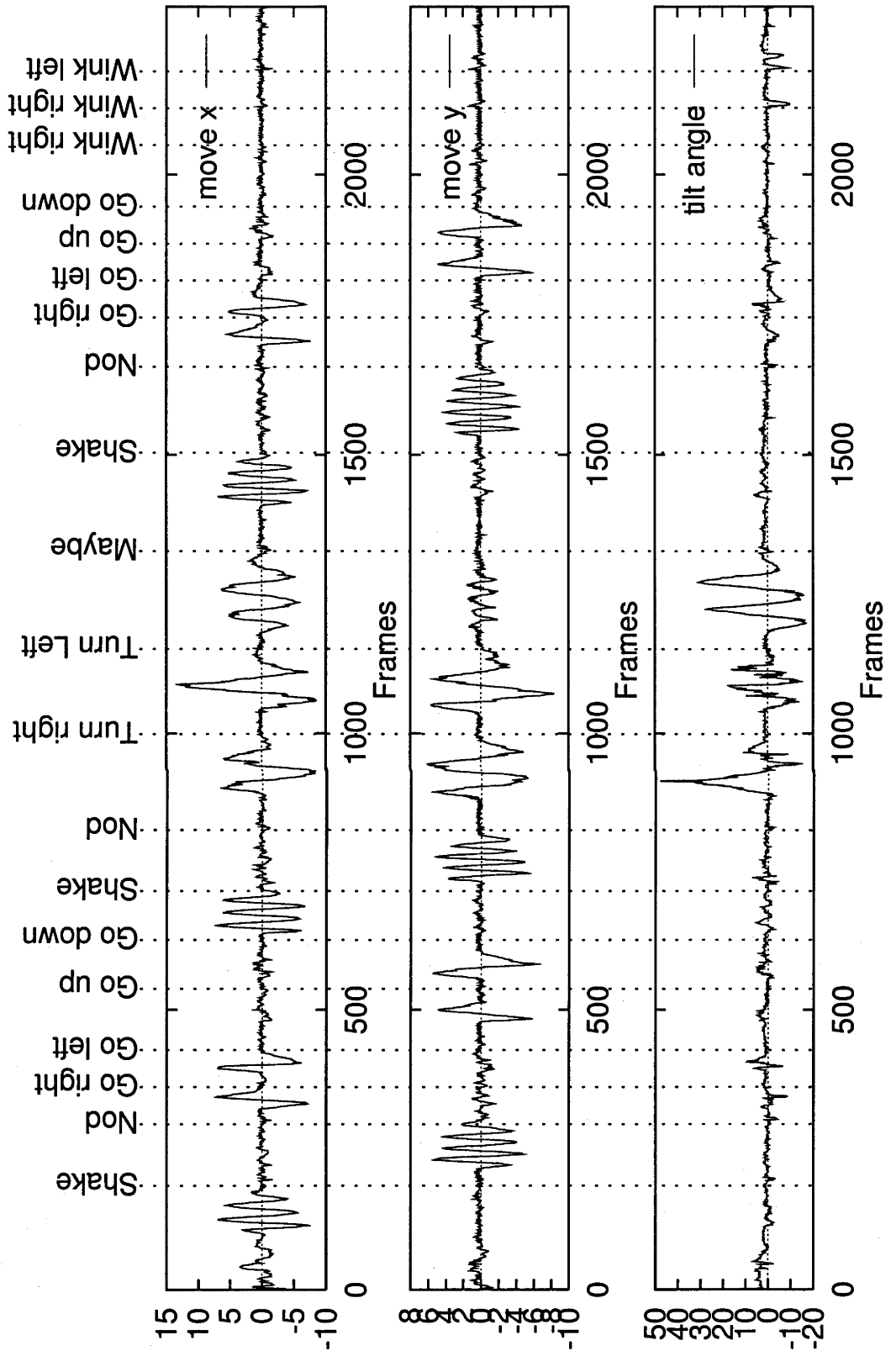


Figure 5.14: Measurements during gesture recognition

The combination of the gaze point estimation module and the facial gesture recognition module forms a powerful visual interaction system that allows an untrained user to interact with a computerised system in a natural and intuitive way. Both modules are based on the 3D head pose and feature tracking system described in Chapter 3. The gesture recognition module allows the user to issue discrete commands and the gaze point estimation system serves as a deictic pointing device for the user to reference objects and locations in the environment. Chapter 1 described a range of applications for the such visual interfaces including human-computer and human-robot interaction.

Chapter 6

Human-Robot Coexistence

Visual interfaces such as the one described in the previous chapters have a wide range of possible applications, including the control of human-friendly robots. This particular application has been pursued in this research. However, as stated in Chapter 2, human-friendly interfaces are only one of the two major issues in human-friendly manipulator robots. The second important issue that needs to be addressed is the safety of the operator in systems where robots and humans share the work space.

6.1 The vision

If robotics technology is to be introduced into the everyday human world, the technology must not only operate efficiently while executing complex tasks such as house cleaning or putting out the garbage, the technology must be safe and easy for people to use. Executing complex tasks in unstructured and dynamic worlds is an immensely challenging problem. Researchers have realized that to create robotic technology for the human world, the most efficient form is the humanoid robot form, as people have surrounded themselves with artifacts and structures that suit dual-armed bipeds, for example staircases, tables and doors. However, merely giving a robot the human shape does not make it friendly to humans. Present day humanoid robots have little understanding of how their actions could harm a person in their vicinity. These robots possess nothing but the crudest hardware and software features to reduce their potential danger to people.

One way to solve the human-safety problem is to build robots that could never harm people. This could be achieved by making the mechanisms small, light-weight, padded and weakly actuated. However, this will result in systems that can't work in the real world since they won't be able to lift or carry objects of any significance. The only alternative is to build manipulators that are strong and fast, but which are also human-friendly.

Current commercial technology for robot manipulators is human-unfriendly. This

is because the robots are position-based controlled. Position based control is quite dangerous in dynamic environments where situations and conditions can unexpectedly change. For example, if a person unexpectedly blocks a robot's planned path, considerable injuries could arise from the impact.

Clearly, compliant force based control is needed in such situations. However, the technology for force-based control of robots is not yet readily available. The current practice is to add force sensors to existing robot manipulators, usually at the tool plate. This allows force control of only the robot's end point. This means the robot could still easily collide with unexpected objects with other parts of the manipulator. To ensure safety a *whole arm manipulation* (WAM) approach must be adopted, in which all the links considered for manipulation and collision situations.

The following sections describe the safety requirements for the control of a human-friendly robot. The outline of the design for a control strategy that fulfils these requirements is presented.

6.2 A safety philosophy for human-friendly robots

Firstly, it is necessary to define the desirable behaviours that a human-friendly robot should posses. In recently reported human-friendly manipulator projects [Yamada 1997], [Morita et al. 1999], [Sakaguchi et al. 2000] and [Lim and Tanie 2000] the behaviour of the robot with respect to safety considerations is not clearly defined. Instead, a number of mostly mechanical features such as padding and mechanical compliance are described. Such measures partially reduce the threat potential of the robot. To develop a consistent safety strategy, the behaviour of the robot, safety limitations and the assignment of responsibility for safe human-robot interaction between the operator and the robot must be formulated.

6.2.1 Safety requirement

Our goal is to construct robotic manipulators that are able to interact and cooperate with humans. The most basic requirement for such system is that a person must be able to safely share a common workspace with a robot.

This of course is orthogonal to the current philosophy of safety in terms of robot applications. The most common solution to human safety, for good reasons, is to separate the workspace of robots and humans. Industrial robots usually work inside cages that prevent people from entering the robots workspace. Most accidents with robots are caused by people overriding the safety mechanisms and entering the work cell while the robot is in operation. The robot's position based controller will consider the resistance caused by a person in the workspace as a disturbance and will only push harder to reach the set position!

Possible collisions caused by a robot are the main threat to people, and therefore

should be the focus of safety considerations. Basically the separation of humans and robots must be overcome while not compromising the safety of the humans. This requirement for human-friendly robots is expressed in the following statement.

> **SAFETY REQUIREMENT 1:** *A human-friendly robot must be controlled in such a way that humans and robots are able to safely share a common workspace.*

6.2.2 The perception problem

Industrial work cells are highly structured since all objects are located in precisely defined positions. This is the only way to ensure that the robot's internal model of the environment does not diverge from reality. The introduction of people into the workspace makes the environment unstructured. People may bring in unknown objects or remove objects from the environment. In addition, the people themselves are dynamic objects that can influence the environment beyond the robots perception.

In such unstructured environments a perfectly safe robot must respond in all situations in a manner that minimises the possible harm to any person. This not only requires the robot to acquire all relevant information about it's environment, but also to draw the correct inferences from this data. This is an impossible task with present and foreseeable technology. Currently the perception capabilities of robot systems are limited. In particular the information that can be acquired *reliably* is essentially restricted to the internal state information of the robot. Information about the robot's environment is usually limited to data from laser, ultrasonic, and vision sensors. This information is not sufficiently reliable to make critical safety decisions. Errors such as double reflections or absorption with ultrasonic sensors or false results from vision algorithms can easily occur.

Considering this lack of perception and understanding of high level concepts, it is not sensible to define the safety goals for a human-friendly robot in high level terms. Instead the safety goals should be defined at a level that a control system is able to deal with, and which are easily understood by the user. Only when the user understands the behaviour of the robot is it possible to share the responsibility for a safe interaction between the two parties.

6.2.3 Assignment of responsibility

The responsibilities between a human operator and a robot control system must be distributed with the low perception capabilities of robotic systems in mind. Many technical and potentially dangerous systems are controlled exclusively by humans by using the support of simple, easy to understand and predictable control systems. For example people are safely able to control cars, aircrafts and construction site machinery. The autonomy of such systems is limited to the extent that such sys-

tems can achieve a set task flawlessly while the overall behaviour of the system is understandable and controllable by the human operator.

Motor vehicles are potentially dangerous for both the driver and other people in the environment. However, most people feel comfortable and safe while driving cars or walking along the side of streets. While the driver is controlling the overall behaviour of the car, various electronic and mechanical systems are providing specialised functions such as cruise control, automatic gear changes, and anti locking brake systems. These mechanisms work autonomous without requiring the driver to know how they are implemented or the internal state of these systems. The driver can specify the overall behaviour of the system with the steering wheel, the brake and the throttle. Recently systems for autonomous highway driving have been developed [Thomanek and Dickmanns 1996], [Jochem et al. 1993]. These systems have been designed as extensions to existing mass-produced motor vehicle technology.

A similar approach is appropriate for human-friendly robots. Before high level autonomous behaviours of robots can be developed, the basic safety characteristics for human-friendly robots must be investigated and practical strategies to control such systems must be found. Building on these fundamental strategies higher levels of autonomy can be implemented. Similar to other technical systems a human-friendly robot should have the following characteristics:

- Ease of control by a human operator
- Provide a certain degree of autonomy which allows easy operation of the system
- Autonomous actions that are predictable and understandable to the human operator
- Non threatening to the human operator

In motor vehicles the central coordinator in the system is the human operator. Instead of trying to build a perfectly safe car, it is the drivers responsibility to operate the car in a safe manner. Various safety measures, such as safety belts and air bags, are built into cars to protect people from the results of human error.

A similar philosophy should apply to human-friendly robots. Instead of trying to build a system that is 100% fail safe, the system should be controllable and predictable to the human operator. Similar to motor vehicles autonomy should be added in order to make it easier for the human operator to use the system. However, all autonomous actions taken by the system must be safe in all situations, given the perceptual limitations. Consider anti locking systems in motor vehicles. The driver does not control the braking force in a vehicle, rather only the desired rate of deacceleration of the car is controlled. This makes the control of a car much easier in emergency stop situations.

Human-friendly robots can be compared to motor vehicles in another respect. A major problem with the introduction of autonomous driving devices into motor vehicles is the liability problem. Car manufactures can not afford to be liable for accidents that were caused by a device which takes responsibility for the steering of the car but has a reliability of only 99%. The same problem can arise with personal assistant robots. It is therefore crucial for the real-world applicability of such technology that the operator has sufficient control over the system and is fully responsible for all actions of the robot.

Once a basic personal assistant robot is available which is safe to operate with people, higher levels of autonomy can be added that increase the capabilities of the system.

6.2.4 Defining human-friendly robots

As discussed in Chapter 1 the basic idea of cooperative human-robot systems is to build teams of humans and robots in which each party performs the actions for which they are best suited. People are better suited to high level decision making while robots are the best performers of manipulation tasks. In terms of safety this means that the operator is responsible for providing the robot with the appropriate commands which do not compromise human safety.

To allow an operator to fulfil this responsibility it is necessary for the robot to work in a way that is predictable and understandable. This predictability manifests itself in high level action procedures which are logical, but on a lower level in a continuity of motion adapted to the bandwidth of the perception of the user. Robot manipulators are able to accelerate and deaccelerate quickly in order to perform rapid manipulation operations. However, to a human operator rapid motions of this kind are unpredictable. Therefore, it is necessary to limit the bandwidth of the motion to allow a human to react appropriately, for example stopping or evading the robot.

> **SAFETY REQUIREMENT 2:** *The bandwidth of operations by a human-friendly robot must be restricted to allow a human operator to fully understand and predict the motion of the robot.*

The robot has the responsibility of performing the manipulation tasks in a safe manner. Particularly collisions with people must be considered by the controller of the robot. As stated earlier the current lack of perception capabilities in robots makes it impossible to detect humans in a reliable way. It can not be assumed that humans can always be detected and the collision problem can be solved by path planning. If collisions with people can not be avoided with sufficient reliability, then the actions of the robot must be planned under the assumption that collisions during motion are possible. Since the user only specifies high level commands and not every motion of the robot, the autonomously generated actions of the robot must conform to the following safety requirement.

SAFETY REQUIREMENT 3: *The collision of a human-friendly robot with a stationary person must not result in any serious injury to the person.*

This is a basic safety restriction. It is also desirable to equip the robot with sensors and algorithms that prevent collisions and use this restriction as a fall-back option if the sensors fail to detect a collision in time.

It is necessary to restrict this requirement to stationary persons since a collision with a moving person can inflict arbitrary collision forces on the person. However, since such situations are not within the control or responsibility of the robot and therefore should not be considered in the formulation of the safety requirements for autonomous operations of the robot. It is reasonable to expect that people in the near vicinity will not deliberately collide with the robot in a dangerous manner.

There is a clear assignment of responsibilities. The robot is responsible for its autonomous planned motions including possible collisions with people in its work space. The user is responsible to provide the robot with commands that consider the limited perception of the robot and the high level context of the situation. This division of responsibilities facilitates the definition of a control strategy for human-friendly robots that fulfils the stated safety requirements.

6.2.5 A basic safe behaviour

The safety requirements that have been defined suggest that autonomous actions of the robot are the main safety hazard. Therefore, according to this definition a passive robot is safe. A passive behaviour is well suited as the basic idle behaviour of a robot. There are three approaches to how a passive behaviour could be defined.

One solution is to deactivate the motors of the robot. This would result in a passive behaviour as soon as the robot is in a stable configuration. The disadvantage of such an approach is that usually at the time of a task completion if the system is not in a stable configuration, the manipulator will fall down. To prevent this the robot would need to be parked in a stable configuration during idle times which is inefficient and this will not be intuitive to a user.

A second solution is to freeze the position of the robot either by engaging brakes or by using position control. This behaviour would be safe and stable. The disadvantage of this solution is that it is unsuitable as a basic behaviour which is always active and would need to be "switched off" during task execution. Also, this approach does not allow the user to change the configuration of the robot if the need arises.

One solution that incorporates few disadvantages is the following: If the robot only compensates for gravitational forces it can be moved by the user manually. When no external forces are present the robot will remain in any given configuration. In the case of a collision the robot is compliant. This reduces the risk of damage and injuries. The configuration of the robot can also be modified by the operator.

For example the operator could bring the gripper to a particular position to grasp an object. Gravity compensation is also suitable as a basic behaviour which is continuously active. When the robot is active, autonomous motion is achieved by applying additional joint torques. If feed forward control is used, floating and ballistic motion of the robot is achievable. This requires less energy than tight closed loop control and also appears natural, intuitive and therefore non-threatening to people. Because of these advantages gravity compensation has been chosen as the basic behaviour for our human-friendly robot system. A robot controller based on gravity compensation has been implemented and is designated *Zero-G* controller in the following.

The compensation of the gravitational forces is a well known problem and is achieved by calculating the joint torques required to balance the gravitational pull on the links of the robot [Craig 1986]. The required torques only depend on the joint positions and can be calculated in a feed-forward fashion inside the control loop. The derivation of the gravity compensation torques τ_g for a serial manipulator is described in Appendix D.

Since the gravity compensation forces do not affect the environment they can be considered to be fail safe and are not subject to safety limitations. This independence from the safety restrictions facilitates the formation of a safety core. The Zero-G controller can be encapsulated inside a safety envelope which guarantees that additional joint torques generated by controllers outside of this safety core comply with the safety restrictions. Figure 6.1 illustrates this concept. The safety envelope filters the torque commands from external controllers according to the defined safety requirements. The combination of the safety envelope with the Zero-G behaviour forms the safety core. The basic gravity compensation behaviour is an integral part of the safety philosophy for human-friendly robots. The safety core represents the bottom layer with inherent safety restrictions that can not be violated by the higher levels. This structure is important for the development of higher level control algorithms that are safe.

It should be noted that this basic behaviour differs significantly from systems that use force-torque (FT) sensors in the wrist. Such systems can not sense forces that are applied to the robot away from the FT sensor, thereby presenting safety risks when collisions with the body of the robot occur.

When working with robot under Zero-G behaviour, the human operator only feels inertial and frictional effects. When the robot is pushed it floats until the inertial energy is consumed by the friction or the robot collides with an obstacle. The CD-ROM at the end of this thesis contains a video sequence which demonstrates the physical interaction between an operator and a robot under Zero-G behaviour[1].

[1]Please refer to Appendix E for details on the contents of the CD-ROM.

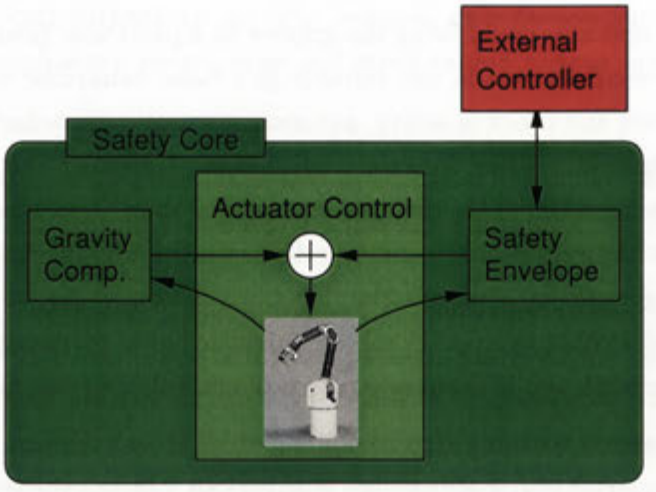


Figure 6.1: Safety core and safety envelope

6.3 Summary

The objective of this chapter was to formulate a concept for the safe control of a human-friendly robot. Three key safety requirements which represent control constraints that allow safe coexistence of humans and robots in a common work space were presented. The definition of the basic rules of the behaviour for a robot with respect to the safety of humans are required. It is important to avoid the pitfalls that can easily render this technology unusable in real applications. A comparison of applying the approach to driving autonomous motor vehicles highlighted the desirable features of a cooperative robot system. The combination of a passive basic behaviour and a safety envelope which restricts the control algorithm to commands that comply with the safety requirements forms a safety core which is well suited for the development of a real human-robot interaction system.

Chapters 7 and 8 present the derivation and implementation of such a controller for an actual robot manipulator. Simulations and real experiments verify the validity of the proposed approach.

Chapter 7

Safe Robot Control

The safety of humans is a central aspect of human-friendly robotics. The safety considerations can be split into two distinct phases, pre-collision safety and post-collision safety [Morita et al. 1999]. Pre-collision safety is concerned with collision forces exerted at the time of impact while post collision safety is concerned with forces exerted by a robot after the impact. In a truly human-friendly robot system both safety aspects should be covered. The experiments using a real manipulator reported [Lim and Tanie 2000] clearly show that passive compliance can only improve post-collision safety but not pre-collision safety. Compliance on its own can not be the solution to achieve safety guarantees for the interaction between humans and robots. A control method which is suitable to combine pre- and post-collision safety is required.

Pre-collision safety can be improved by mechanical measures such as padding of the robot and by a light weight design of the robot. However, such measures do not allow the *limitation* of the impact force, only a reduction. Although such features are desirable for a human-friendly robot, it is necessary to provide *quantifiable* limits for the threat potential posed by a collision of the manipulator with a person.

Chapter 6 developed a framework of behavioural constraints for the autonomous actions of a human-friendly robot and a software architecture called the *safety core.* This chapter presents the mathematical formulation of the modules required for the safety core. Section 7.1 describes the functional blocks that are required within the safety core and their interaction. The mechanics of the safety core define the input and output of the various functional blocks. Section 7.2 derives the collision models required for the safety core. The behaviour of the collision model with respect to the kinematic and geometric structure of the robot is discussed and suggestions for the design of human-friendly robots are presented. A physical parameter we call the *"impact potential"* of the robot is defined to set control constraints for the robot.

Section 7.3 transforms the control constraints from the state space of the robot to the space of robot control vectors. A solution to the problem of clipping unsafe control vectors is presented. The safety core must replace unsafe vectors with a

suitable vector in order to ensure continuous control of the robot.

A simulation study of the new methods and algorithms is presented in Section 7.4. The study demonstrates the feasibility of the safety core architecture and the effectiveness of the control constraints to guarantee quantifiable limits of the robot's impact potential. Finally, Section 7.5 summarises the insights into the properties, limitations and possibilities of robot control schemes based on the concept of impact potential.

7.1 Safe robot control

The control constraints which are required to achieve the safe behaviour of a robot manipulator are based in the SAFETY REQUIREMENTS presented in Chapter 6. The requirements stated that collisions of the robot during autonomous motion with a stationary person should not cause harm to the person and that the bandwidth of the robot's motions must be limited such that a person who observes the robot is able to predict the actions of the robot. Both requirements must be enforced for all actuator commands that are generated by any controller outside the safety core.

It is therefore necessary for the safety core to be able to classify actuator commands as safe or unsafe. Safe actuator commands should be passed through to the robot while unsafe commands must be rejected.

Effectively the envelope of the safety core has the function of a command filter. If unsafe actuator commands are generated by the outside controller, the envelope detects this condition and rectifies the problem. The required operations must be performed online and in real-time at the frequency of the control loop of the outside controller. Although it is possible for the safety core to simply reject an unsafe actuator command and inform the external controller of this problem, the safety core is also left to decide what actuator command must be sent to the robot if the external controller is unable to provide a timely safe command. It is therefore sensible to provide the safety core with the ability to derive a safe actuator command that is *closest* to the unsafe command, which can be used in the place of the unsafe command from the external controller. It is also sensible to provide feedback to the external controller when unsafe actuator commands have been modified.

The first step towards the definition of a filter for the safety envelope is to derive a physical parameter that corresponds to the safety restrictions. The threat potential posed by collisions of the robot with persons must be quantified. Since active and passive compliance can only improve the post-collision safety [Lim and Tanie 2000], therefore the pre-collision safety should be considered for this physical parameter. The *impulsive* impact force is a logical starting point for the development of the safety envelope.

The key components of the safety envelope and the interaction between components are shown in Figure 7.1. A physical control parameter for the safe region in

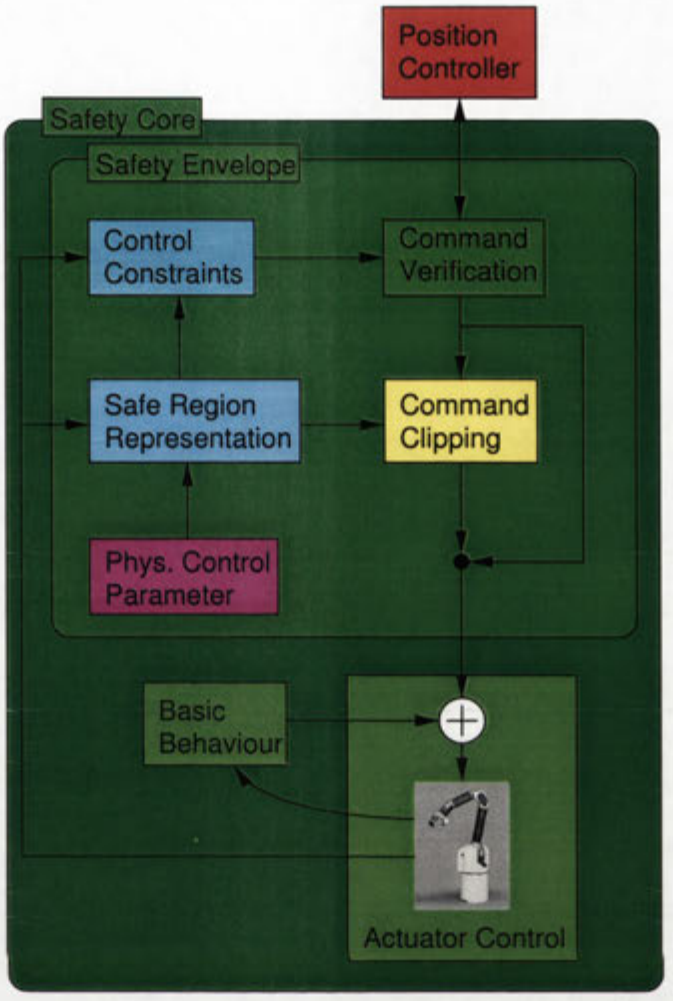


Figure 7.1: Components of the safety envelope

the command space is derived. The control constraints are used to verify commands from the external controller. If a command is safe it does not need to be clipped. Otherwise the clipping algorithm finds the closest safe vector to replace the original unsafe vector for the robot to execute.

7.2 Impact Potential

In the following sections the collision of an articulated mechanical system with a stationary rigid object is considered. The mechanical system is assumed to be a serial linkage of rigid bodies. This is a sufficient assumption for most robot systems. For parallel and flexible systems similar considerations are possible. For the robot manipulator used as the implementation platform in this research the assumption of a serial rigid mechanical system is sufficient.

7.2.1 Impulsive impact force

The collision between a mechanical system and an obstacle can occur at any point $\mathbf{p}$ on the surface $\mathbf{P}$ of the mechanical system. The Cartesian position $\mathbf{p}_c$ of a point $\mathbf{p}$ depends on the vector of joint angles Θ and the mapping function $f_{\mathbf{p}} : \Theta \mapsto \mathbf{p}_c$.

$$\mathbf{p}_c = f(\Theta) \tag{7.1}$$

Differentiating of Equation 7.1 gives the Cartesian velocity $\mathbf{v}$

$$\mathbf{v} = \dot{\Theta} = J(\Theta)\dot{\Theta} \tag{7.2}$$

where $J(\Theta) \in \mathbb{R}^{3\times n}$ is the Jacobian matrix of point $\mathbf{p}$ which relates the joint velocities $\dot{\theta}$ to the respective Cartesian velocity vector $\mathbf{v}$. The dynamic equation of an n-degree of freedom manipulator without friction is

$$\tau = M(\theta)\dot{\theta} + N(\theta, \dot{\theta}) + J_{\mathbf{p}}(\Theta)^t\mathbf{f} + g(\Theta) \tag{7.3}$$

where $\tau \in \mathbb{R}^n$ is the joint torque vector, $M(\Theta) \in \mathbb{R}^{n\times n}$ is the inertia matrix of the manipulator, $N(\Theta, \dot{\Theta}) \in \mathbb{R}^n$ is the vector of Coriolis and centrifugal forces and $\mathbf{f}$ is the force applied at point $\mathbf{p}$ of the manipulator.

If the manipulator moves in free space then $\mathbf{f} = 0$ for all points $\mathbf{p}$. The gravitational term $g(\Theta)$ can be disregarded since it is compensated for by the Zero-G behaviour.

As stated previously the impulsive force at the time of impact poses the highest safety risk, in particular when the collision is assumed to be between two rigid objects. This is a valid approximation for rigid parts of the body such as the head, which is also the most sensitive part of the body. For other parts of the body that are covered by thicker tissue, the rigid object assumption is a conservative over estimate. The worst case scenario must be considered.

The derivation of the impact force $\hat{\mathbf{f}}$ of a particular point $\mathbf{p}$ on the surface $\mathbf{P}$ of a serial manipulator with a stationary obstacle shown in Equation 7.13 is well known [Walker 1994]. For completeness, the derivation is included and the naming convention has been adapted to fit the conventions in this thesis.

The collision of two rigid bodies with velocities $\mathbf{v}_1$ and $\mathbf{v}_2 \in \mathbb{R}^3$ can be modelled as

$$[(\mathbf{v}_1 + \Delta\mathbf{v}_1) - (\mathbf{v}_2 + \Delta\mathbf{v}_2)]^t\,\mathbf{n} = -e\,(\mathbf{v}_1 - \mathbf{v}_2)^t\,\mathbf{n} \tag{7.4}$$

where $\mathbf{n} \in \mathbb{R}^3$, $\|\mathbf{n}\| = 1$ is the surface normal vector, $\Delta\mathbf{v}_1$ and $\Delta\mathbf{v}_2$ are the change in $\mathbf{v}_1$ and $\mathbf{v}_2$ respectively and $0 \leq e \leq 1$ is the elasticity constant. A configuration with $e = 0$ models a purely plastic collision and $e = 1$ models a purely elastic collision. Values between 0 and 1 describe intermediate cases.

We assume that the impact between the two bodies occurs during a small time Δt

and in the model the idealisation that $\Delta t \to 0$ is made. Assuming that all velocities are finite, the position and orientation of all bodies in the system remains constant during the time of impact. The impulsive force $\hat{\mathbf{f}}$ created at the collision point $\mathbf{p}$ is

$$\lim_{\Delta t \to 0} \int_t^{t+\Delta t} \mathbf{f}(s)ds = \hat{\mathbf{f}} \tag{7.5}$$

where $\left\|\hat{\mathbf{f}}\right\|$ is a finite quantity. The relation between $\hat{\mathbf{f}}$ and the changes in velocity is determined by integration of Equation 7.3.

$$\int_t^{t+\Delta t} \tau ds = \int_t^{t+\Delta t} M(\Theta)\ddot{\Theta} ds + \int_t^{t+\Delta t} N(\Theta, \dot{\Theta}) ds - \int_t^{t+\Delta t} J_{\mathbf{p}}(\Theta)^t \mathbf{f}(s) ds \tag{7.6}$$

Since the positions and and velocities are finite as $\Delta t \to 0$, therefore the integrals containing N and τ become 0. The remaining two terms after resolving the integrals are

$$M(\Theta)\dot{\Theta}(t + \Delta t) - \dot{\Theta}(t) = J_{\mathbf{p}}(\Theta)^t \hat{\mathbf{f}} \tag{7.7}$$

Defining $\Delta\dot{\Theta}$

$$\Delta\dot{\Theta} = \dot{\Theta}(t + \Delta t) - \dot{\Theta}(t) \tag{7.8}$$

Substituting into Equation 7.7 yields

$$\Delta\dot{\Theta} = M(\Theta)^{-1} J_{\mathbf{p}}(\Theta)^t \hat{\mathbf{f}} \tag{7.9}$$

Combining Equations 7.4 and 7.9 forms the basis of the impact model. It relates the impulsive force $\hat{\mathbf{f}}$ of the collision with the respective change in velocity. The velocity relationship of Equation 7.2 still holds at the instant of the impact. Combining with Equation 7.9 yields

$$\Delta\mathbf{v} = J_{\mathbf{p}}(\Theta) M(\Theta)^{-1} J_{\mathbf{p}}(\Theta)^t \hat{\mathbf{f}} \tag{7.10}$$

For the robot's safety controller it is assumed that the person is at rest at the time of impact. This is a sensible assumption since the philosophy of the safety core is based on the idea that the robot can not reliably detect the position (and velocity) of a person in it's workspace. Therefore, the robot is only responsible for the consequences of it's own motions. If the person performs dangerously fast motions and purposely collides with the robot, the robot can not be expected to sense this type of behaviour and compensate for it. It is therefore assumed that the velocity of the obstacle equals 0 ($\mathbf{v}_2 = 0$).

Usually a person is positioned in free space and the human body is able to comply when the impact occurs. However, this may not always be the case as the person may be in contact with a rigid object which prevents the body from complying with the impact. For example, a person could be standing next to a wall or manipulating an object on a bench-top. One of the most likely medium term applications for human

friendly robots is as helping hands for the disabled. In this scenario the body of the person could be fully supported by a wheelchair or a special seat. Therefore, the worst case must be assumed. Meaning that the human body can not be considered to be compliant, and the velocity of the obstacle remains unchanged ($\Delta\mathbf{v}_2 = 0$).

An argument has been made for

$$\mathbf{v}_2 = \Delta\mathbf{v}_2 = 0 \tag{7.11}$$

Substituting Equation 7.11 into Equation 7.4 yields

$$\Delta\mathbf{v}_1 = \Delta\mathbf{v} = J_\mathbf{p}(\Theta)M(\Theta)^{-1}J_\mathbf{p}(\Theta)^t\widehat{\mathbf{f}} \tag{7.12}$$

Solving Equation 7.12 for the magnitude $\widehat{f}$ of the impulse force, given that $\widehat{f} = \widehat{\mathbf{f}}\mathbf{n}$ where $\mathbf{n}$ is the surface normal at point $\mathbf{p}$ on the robot's surface, results in Equation 7.13. This equation is the general expression for the magnitude of the impulse force resulting from the collision of a single serial manipulator with a rigid non-compliant obstacle.

$$\widehat{f} = \frac{-(1+e)\mathbf{v}^t\mathbf{n}}{\mathbf{n}^t J_\mathbf{p}(\Theta)M^{-1}(\Theta)J_\mathbf{p}^t(\Theta)\mathbf{n}} \tag{7.13}$$

Here, $\mathbf{v} \in \mathbb{R}^3$ is the velocity of the mechanical system at the point $\mathbf{p}$ of impact, $\Theta \in \mathbb{R}^n$ are the joint angles, $J_\mathbf{p}(\Theta)$ is the manipulator Jacobian at the surface point $\mathbf{p}$ and $M(\Theta)$ is the n×n inertia matrix [Walker 1994]. This result is examined in more detail in the following section in order to derive robust control constraints.

7.2.2 Poles of the impulsive impact force

It is crucial to understand the properties of the impact force with respect to the kinematic properties and the surface geometry of the mechanism to avoid pitfalls which lead to impractical and unusable control constraints. In particular the conditions for poles in $\widehat{f}$ must be understood in order to avoid their appearance.

Poles occur where the denominator of Equation 7.13 equals 0. A n-DOF serial manipulator is considered with the impact point $\mathbf{p}$ on the m-th link with $m \leq n$. Hence, the last $n - m$ columns of the Jacobian $J_\mathbf{p}(\Theta)$ are zero. Since M is positive definite M^{-1} is also positive definite and therefore $J_\mathbf{p}(\Theta)\mathbf{n}^t$ must equal 0 for the denominator to equal 0. Therefore for each of the first m columns $\mathbf{j}_i, i \leq m$ of $J_\mathbf{p}(\Theta)$ on of the following two conditions must be true for $J_\mathbf{p}(\Theta)\mathbf{n}^t$ to equal 0:

- The surface point $\mathbf{p}$ is on joint axis i. In this case $\mathbf{j}_i = 0$ because a rotation of joint i does not result in translational motion of the point $\mathbf{p}$ in Cartesian space. Hence, $\mathbf{j}_i^t\mathbf{n} = 0$.

- The surface normal $\mathbf{n}$ at point $\mathbf{p}$ is orthogonal to $\mathbf{j}_i$ and therefore $\mathbf{j}_i^t\mathbf{n} = 0$.

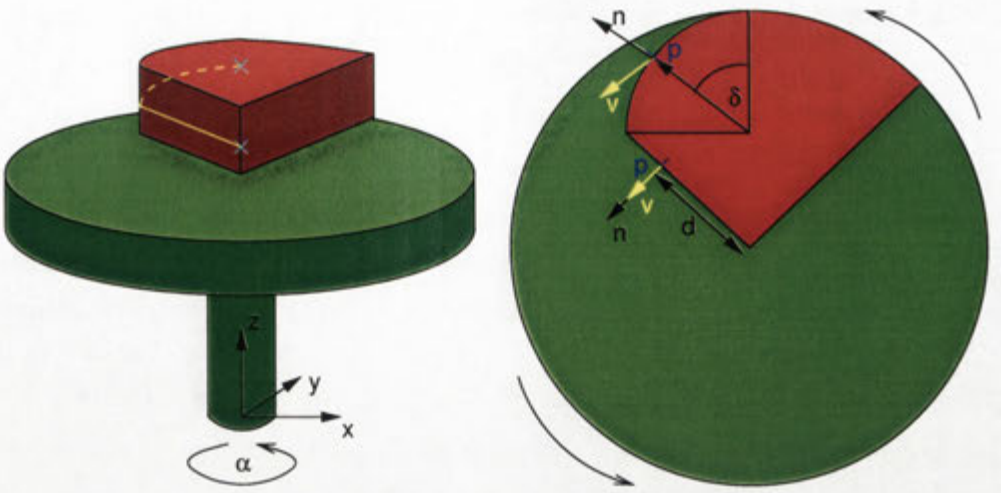


Figure 7.2: 1 DOF example

If and only if one of these conditions is true for each of the first m columns $\mathbf{j}_i$ of $J_{\mathbf{p}}(\Theta)$, for $i \leq m$ the denominator of Equation 7.13 becomes zero and $\hat{f}$ has a pole.

The two types of poles in the impact force $\hat{f}$ are illustrated in Figure 7.2. Figure 7.2a) shows the geometry of a 1 DOF mechanism which can be rotated about a drive shaft. The red shaded impact block is fixed to the disc. Points $\mathbf{p}$ shown on the yellow line along the side of the impact block are considered in the discussion. This surface consists of a straight part coincident with the rotation axis and a circular part with half the radius (0.5 units) of the green disc (1 unit). Figure 7.2b) shows the top view on the impact block and the disc. The collision is assumed to occur at an angular velocity of 1 radians per second. The mass matrix $M \in \mathbb{R}^{1\times 1}$ of the mechanism is assumed to equal 1.

Type 1: p is on the joint axis

Surface points $\mathbf{p}$ along the straight part of the impact block with a distance $0 < d < 0.5$ from the rotation axis are considered. For convenience the position of the mechanism in Figure 7.2b) is defined as $\theta = 135°$. In this case the variables take the following values:

$$\mathbf{p} = (d\cos\theta, d\sin\theta, z) \tag{7.14}$$

$$\mathbf{n} = (\sin\theta, -\cos\theta, 0) \tag{7.15}$$

$$J = \begin{pmatrix} -d\sin\theta, & d\cos\theta, & 0 \end{pmatrix}^t \tag{7.16}$$

$$\mathbf{v} = J \tag{7.17}$$

The impact force $\widehat{f}$ at $\mathbf{p}$ is

$$\widehat{f} = \frac{-(1+e)(-d\sin^2\theta - d\cos^2\theta)}{(-d\sin^2\theta - d\cos^2\theta)[1](-d\sin^2\theta - d\cos^2\theta)} \tag{7.18}$$

$$= \frac{(1+e)d}{d^2} \tag{7.19}$$

and therefore $\widehat{f} \to \infty$ as $d \to \infty$. The pole at the joint axis is marked with a turquoise cross in Figure 7.2a). At the pole itself the impact force is undefined. The intuitive explanation of the first class of poles along the axis of a joint is the increasing leverage of the moving mass at points closer to the axis of rotation.

Type 2 : The surface normal at p is orthogonal to $\mathbf{j}_i$

Surface points $\mathbf{p}$ on the circular part are considered for $0 < \delta < \frac{\pi}{2}$. For convenience the position of the mechanism in Figure 7.2b) is defined as $\theta = 90°$. In this case the variables take the following values:surface of the

$$\mathbf{p}_1 = \frac{1}{2}\begin{pmatrix} 1+\cos\delta \\ \sin\delta \\ z \end{pmatrix} \tag{7.20}$$

$$\mathbf{n}_1 = \begin{pmatrix} \cos\delta \\ \sin\delta \\ 0 \end{pmatrix} \tag{7.21}$$

$$T_{01} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{7.22}$$

$$\mathbf{p}_0 = T_{01}\mathbf{p}_1 \tag{7.23}$$

$$\mathbf{v}_0 = J_0 = \frac{\partial \mathbf{p}_0}{\partial\theta} \tag{7.24}$$

$$\mathbf{n}_0 = T_{01}\mathbf{n}_1 \tag{7.25}$$

The impact force at $\mathbf{p}$ is

$$\widehat{f} = \frac{-(1+e)\frac{1}{2}\mathbf{v}_0^t\mathbf{n}_0}{\mathbf{n}_0^t J_0\,[1]\,J_0^t\mathbf{n}_0} \tag{7.26}$$

$$= \frac{(1+e)\frac{1}{2}\sin\delta}{\sin^2\delta} \tag{7.27}$$

Therefore, $\widehat{f} \to \infty$ as $\delta \to 0$. This second pole of the impact force is marked with a turquoise cross in Figure 7.2a). Intuitively this class of poles occurs through a *wedge effect*. The mechanism is shaped such that it drives itself between the rigid axis of rotation and the rigid obstacle. The closer $\mathbf{n}$ is to being orthogonal to the direction of motion $\mathbf{v}$ the larger the wedge effect. However, the underlying assumption is that

the contact between the mechanism and the obstacle is frictionless. If the contact between the two objects is not frictionless the force vector at the contact point is not aligned with the surface normal and therefore, the resulting impact force will have no pole.

In the following discussion the poles of $\widehat{f}$ are of major importance. It is crucial to understand where poles occur and under what conditions. This section showed two different classes of poles which occurred in two different geometric constellations of the kinematic and surface geometry.

7.2.3 Collisions while considering friction

Obviously the assumption of a frictionless collision is unrealistic and results in unrealistic high collision forces through the wedge effect described in the previous section. It is therefore assumed that at the moment of contact there is sufficient friction to align the impact force vector $\widehat{\mathbf{f}}$ with the velocity vector $\mathbf{v}$ of the surface point $\mathbf{p}$. This situation can be modelled by assuming a surface normal vector $\mathbf{n}$ which is aligned with the velocity vector $\mathbf{v}$ such that the vectors point in opposite directions $\left(\mathbf{n} = -\frac{\mathbf{v}}{s_\mathbf{v}}, s_\mathbf{v} = |\mathbf{v}|\right)$. This is possible because the impact force vector $\widehat{\mathbf{f}}$ is always aligned with the surface normal $\mathbf{n}$. This assumption yields the effective impact force $\widehat{f}_e$ based on Equation 7.13

$$\widehat{f}_e = \frac{-(1+e)\mathbf{v}^t(-\frac{1}{s_\mathbf{v}}\mathbf{v})}{(-\frac{1}{s_\mathbf{v}}\mathbf{v}^t)J_\mathbf{p}(\Theta)M^{-1}(\Theta)J_\mathbf{p}^t(\Theta)(-\frac{1}{s_\mathbf{v}}\mathbf{v})} \tag{7.28}$$

$$\overset{s_v \neq 0}{=} \frac{(1+e)s_\mathbf{v}\mathbf{v}^t\mathbf{v}}{\mathbf{v}^t J_\mathbf{p}(\Theta)M^{-1}(\Theta)J_\mathbf{p}^t(\Theta)\mathbf{v}} \tag{7.29}$$

$$= \frac{(1+e)s_\mathbf{v}^3}{\mathbf{v}^t J_\mathbf{p}(\Theta)M^{-1}(\Theta)J_\mathbf{p}^t(\Theta)\mathbf{v}} \tag{7.30}$$

This modification removes the wedge effect and the poles of the second type and yields a more realistic behaviour of the effective impact force $\widehat{f}_e$. It should be noted that this modification also removes the problem of surface points on edges where the surface normal can not be specified. However, points on edges are the most likely surfaces to collide with obstacles. The impact force for stationary points where the Cartesian velocity $\mathbf{v} = 0$ is undefined under this assumption since $\mathbf{v}$ takes the place of $\mathbf{n}$ in the denominator.

The only unspecified parameter left in the formulation is the elasticity constant e of the impact. In the worst case the impact is fully elastic ($e = 1$) and under this worst case assumption Equation 7.30 further simplifies to:

$$\widehat{f}_e = \frac{2s_\mathbf{v}^3}{\mathbf{v}^t J_\mathbf{p}(\Theta)M^{-1}(\Theta)J_\mathbf{p}^t(\Theta)\mathbf{v}} \tag{7.31}$$

7.2.4 Stationary points

For the derivation of control constraints for the manipulator it is necessary to define the impact force $\widehat{f}_e$ for all surface points including stationary points. The Cartesian velocity $\mathbf{v} = J_{\mathbf{p}}(\Theta)\dot{\Theta}$ of a particular surface point $\mathbf{p}$ may equal 0 in two different configurations:

- The link to which $\mathbf{p}$ belongs is motionless. In this case $\widehat{f}_e$ is not defined for any of the surface points which belong to this link. Since the link is motionless it can not collide with a stationary obstacle and therefore the impact force of all surface points $\mathbf{p}$ on this link is defined to be zero $\widehat{f}_e(\mathbf{p}) \stackrel{def}{=} 0$.

- The link is in motion but the translation and rotation of the link compensate each other at point $\mathbf{p}$ or the translational velocity is 0 and the point $\mathbf{p}$ is on the axis of rotation. Since the subspace of stationary points is 1-dimensional at most while the surface is 2-dimensional, a set of non-stationary surface points $\mathbf{p}'(\varepsilon)$ exist such that $\mathbf{p}' \to \mathbf{p}$, as $\varepsilon \to 0$ and $\mathbf{p}' \neq \mathbf{p}$. In this case the impact force $\widehat{f}_e$ is defined as the supremum of the limits for $\varepsilon \to 0$ of all sets $\mathbf{p}'(\varepsilon)$.

In summary, the impact force $\widehat{f}_e$ at a stationary point $\mathbf{p}$ is defined as

$$\widehat{f}_e(\mathbf{p}) \stackrel{def}{=} \begin{cases} 0 & \text{for a stationary link} \\ \sup_{\mathbf{p}'(\varepsilon)} \left(\lim_{\varepsilon \to 0} \widehat{f}_e(\mathbf{p}') \right) & \text{for a moving link} \end{cases} \tag{7.32}$$

Therefore, if $\mathbf{p}$ is a pole of the first type as described in Section 7.2.2 the supremum is ∞, and hence, $\widehat{f}_e$ is undefined. The following section discusses the conditions for the appearance of such poles of $\widehat{f}_e$.

7.2.5 Poles of the generalised impact force

The substitution of the surface normal $\mathbf{n}$ with the normalised Cartesian velocity vector $-\mathbf{v}/s_{\mathbf{v}}$ removes the poles caused by the wedge effect and therefore yields a more realistic model for impacts in the real world. However, it introduces new poles at the stationary points where the Cartesian velocity $\mathbf{v} = 0$.

For non-stationary points the following argument is true. If $\mathbf{v} \neq 0$ there is a non-empty set K of indices for which $\dot{\theta}_k \neq 0 \forall k \in K$ and $\mathbf{v} = \sum_K \mathbf{j}_k \dot{\theta}_k$ where $\mathbf{j}_k \in \mathbb{R}^{3\times 1}$ is the k-th column of $J_{\mathbf{p}}(\Theta)$. Therefore $\mathbf{v}$ is part of the subspace spawned by the $\mathbf{j}_k$. However, for the denominator to approach 0, $J^t_{\mathbf{p}'}(\Theta)\mathbf{v}$ must approach 0. Therefore, it is necessary that $\mathbf{v} \perp \mathbf{j}_k \forall k \in K$ since the $\mathbf{j}_k$ do not approach 0. However, $\mathbf{v}$ can not be an element of the subspace spawned by the $\mathbf{j}_k$. Therefore for non-stationary points $\mathbf{p}$ where $\mathbf{v} \neq 0$ the denominator $\mathbf{v}^t J_{\mathbf{p}}(\Theta) M^{-1}(\Theta) J^t_{\mathbf{p}}(\Theta)\mathbf{v} > 0$ since M^{-1} is positive definite. If the denominator is > 0 then the surface point can not be a pole of $\widehat{f}_e$. Hence, a necessary condition for a pole in $\widehat{f}_e$ is that $\mathbf{p}$ is a stationary point.

In a n-DOF serial manipulator with the surface point $\mathbf{p}$ on the m-th link with $m \leq n$, the last $n-m$ columns of the Jacobian $J_{\mathbf{p}}^t(\Theta)$ are zero. Points $\mathbf{p}' = \mathbf{p} + \varepsilon\mathbf{u}$, $|\mathbf{u}| = 1$ in the vicinity of a stationary point $\mathbf{p}$ are considered. Since M and M^{-1} are positive definite, therefore for the denominator to approach 0, $J_{\mathbf{p}'}^t(\Theta)\mathbf{v}$ must approach 0 for $\varepsilon \to 0$. Moreover, since $s_{\mathbf{v}} \to 0$ linearly, the numerator approaches 0 at the order of 3 for $\varepsilon \to 0$. Also, since M^{-1} has a linear mapping, $J_{\mathbf{p}'}^t(\Theta)\mathbf{v}$ must approach 0 at an order higher than 1.5 for $\widehat{f}_e \to \infty$. The surface point $\mathbf{p}$ is a pole of f_e if and only if there exists an unit vector $\mathbf{u}$ such that for each of the first m columns $\mathbf{j}_i, i \leq m$ of $J_{\mathbf{p}'}(\Theta)$ one of the following two conditions is true:

- $\mathbf{j}_i \to 0$ for $\varepsilon \to 0$ which is equivalent to $\mathbf{p}$ being an element of joint axis i. In this case $\mathbf{j}_i \to 0$ linearly and therefore $\mathbf{j}_i^t\mathbf{v}_{\mathbf{p}'} \to 0$ quadratically.
- $\mathbf{j}_i \frac{\mathbf{v}_{\mathbf{p}'}}{s_{\mathbf{v}}} \to 0$ for $\varepsilon \to 0$ which is equivalent to the direction of the Cartesian motion becomes orthogonal to column $\mathbf{j}_i$. In this case $\mathbf{j}_i^t\mathbf{v}_{\mathbf{p}'} \to 0$ quadratically.

If one of the conditions is met for each of the first m columns $\mathbf{j}_{1-m}$ of $J_{\mathbf{p}}(\Theta)$, the denominator of Equation 7.31 approaches 0 at the order of 4 and therefore $f_{max} \to \infty$ for $\varepsilon \to 0$. If for one index i none of the two conditions are met then $\mathbf{j}_i^t\mathbf{v}_{\mathbf{p}'} \to 0$ only linearly and therefore the denominator does approach 0 only quadratically and $\widehat{f}_e$ has no pole. With this necessary and sufficient condition poles in $\widehat{f}_e$ can be determined from the system state $(\Theta, \dot{\Theta})$.

The conditions for the appearance of singularity points are intuitively understandable. A pole must be coincident with one or more joint axes and the direction of motion of points in the vicinity of the singularity point must be orthogonal to the motion which would result from motion of all other joint axes. Otherwise, the impact could be partly compensated for by a motion around those joints. Figures 7.3 and 7.4 present a simple example to illustrate this situation. Figure 7.3 shows the geometry of a 1 DOF robot. The single link is modelled as a hollow pipe with length $l = 0.58$ m and a mass of $m = 2.682$ kg which is equivalent to the upper arm link of the WAM arm[1]. The pipe has a diameter of $d = 9$ cm and rotates around the green joint axis at its lower end which is perpendicular to the dotted symmetry axis of the link.

The plot in Figure 7.4 shows the potential impact force $\widehat{f}_e$ of the points $\mathbf{p}(l, \alpha)$ on the surface of the pipe over the distance l from the bottom end along the symmetry axis and the angle α around the symmetry axis. In this example the joint rotates at an angular velocity of 1 rad/s. Generally the impact force decreases with increasing l because the leverage of the mass of the link on the impact point increases as the point of impact moves closer to the joint axis. This effect also causes a sinoseudal ripple with respect to α which becomes more obvious closer to the joint axis. Points in the

[1]The WAM robot is the hardware platform for the experimental validation of the proposed control method. It is described in detail in Chapter 8.

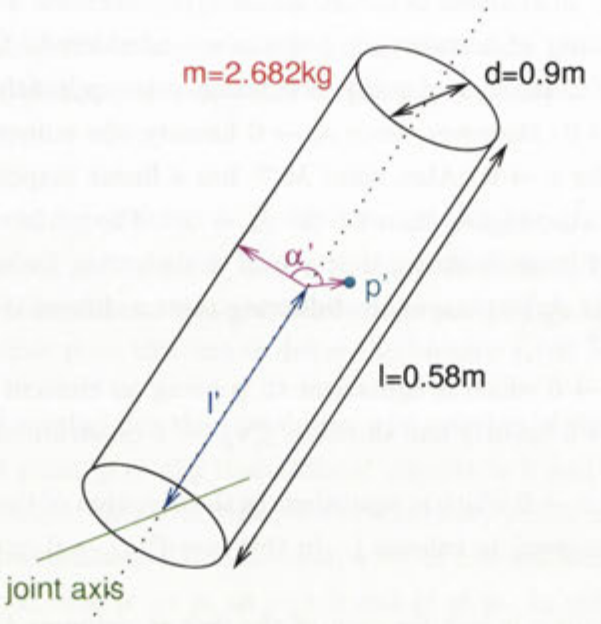


Figure 7.3: 1 DOF manipulator configuration

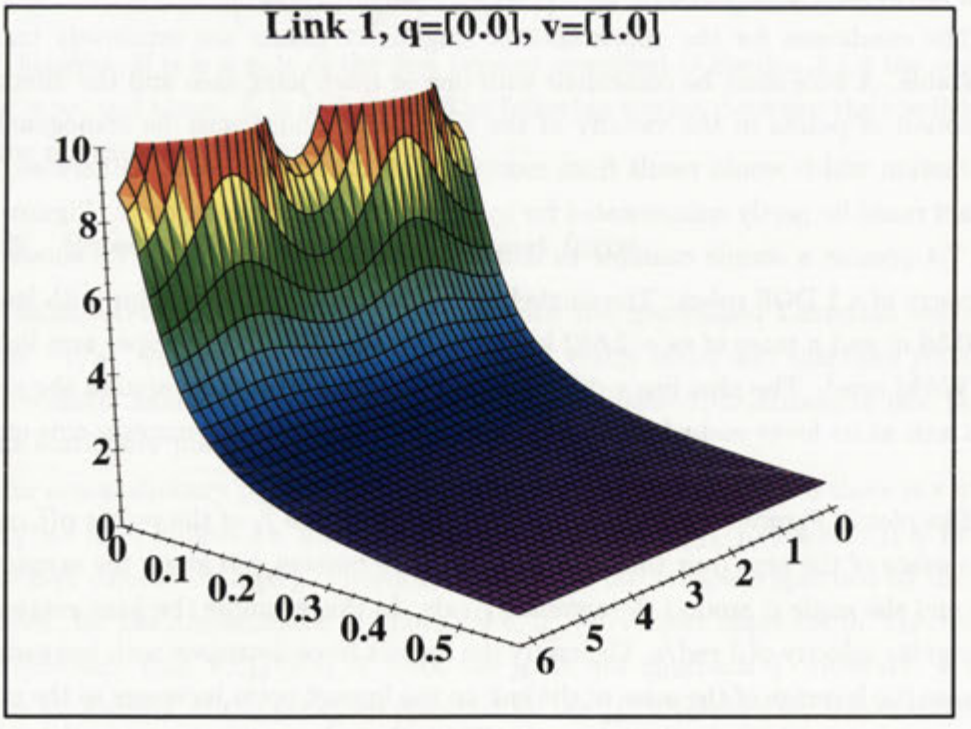


Figure 7.4: Impact force distribution on the 1 DOF manipulator

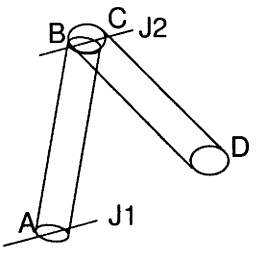
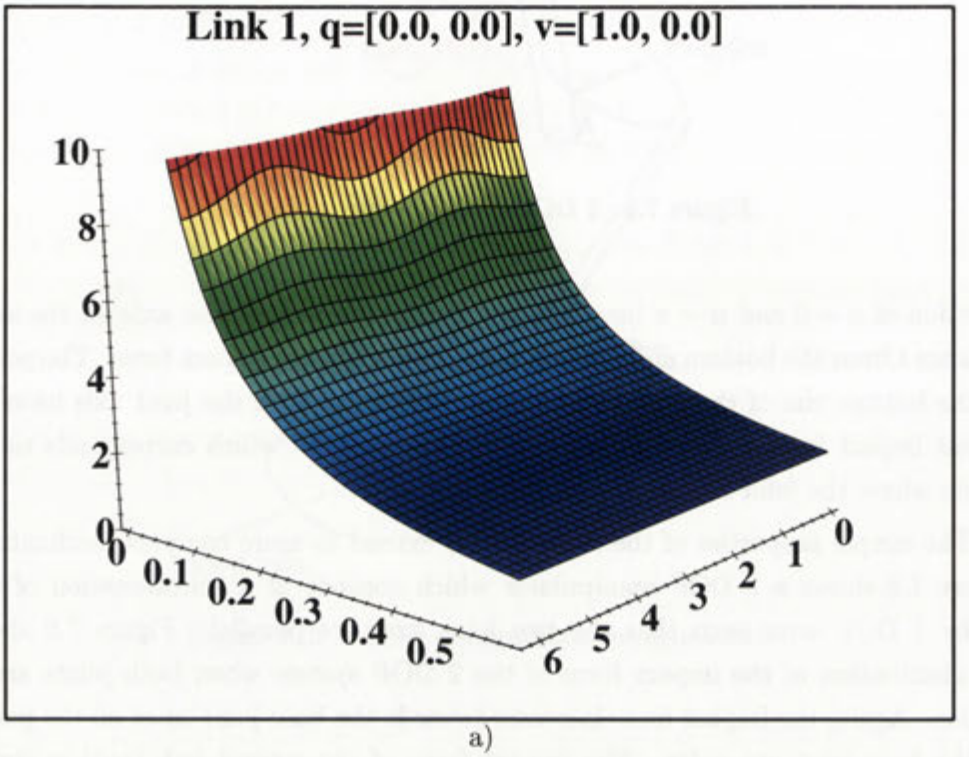


Figure 7.5: 2 DOF example manipulator

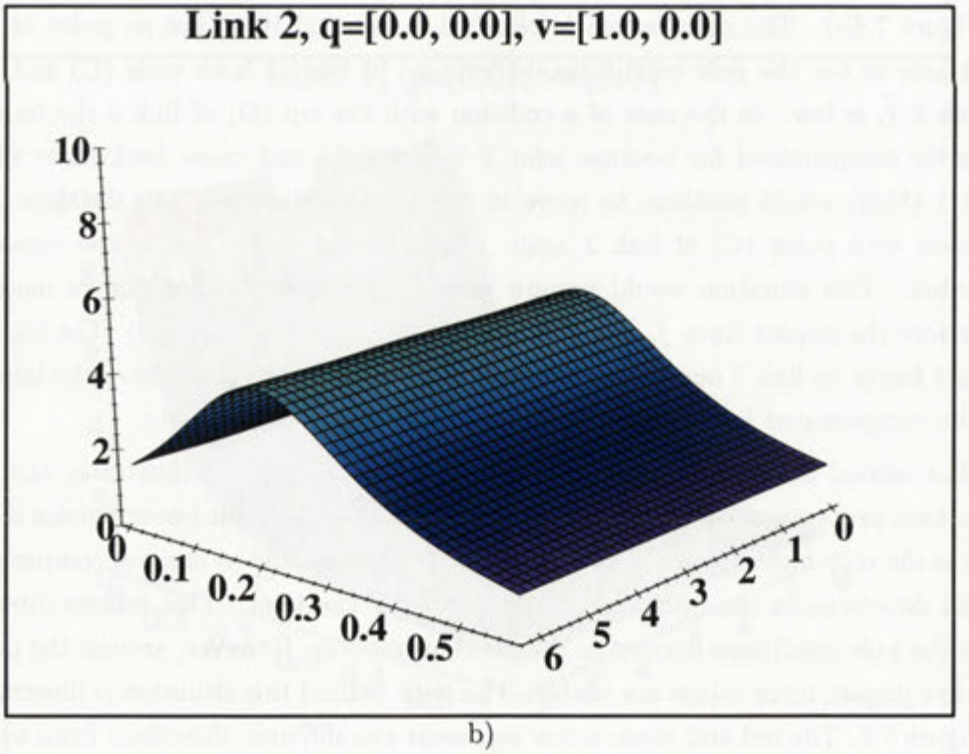
direction of $\alpha = 0$ and $\alpha = \pi$ have a larger distance from the joint axis for the same distance l from the bottom end. Hence, they have a smaller impact force. The points on the bottom rim of the pipe $\mathbf{p}(0, \alpha)$ which intersects with the joint axis have the largest impact forces. $\widehat{f}_e \to \infty$ as $\alpha \to \frac{1}{2}\pi$ and $\alpha \to \frac{3}{2}\pi$ which corresponds to the points where the joint axis intersects the rim.

The simple properties of the impact force extend to more complex mechanisms. Figure 7.5 shows a 2 DOF manipulator which consists of a concatenation of two of the 1 DOF arms such that the two joint axes are parallel. Figure 7.6 shows the distribution of the impact force of the 2 DOF system when both joints are in motion. Again, the impact force increases towards the base joint since all the points on the base joint are poles. The impact force of the second link (red) is shown in Figure 7.6b). The distribution is relatively even and there are no poles at the joint axis as per the pole conditions derivation. In fact at both ends (C) and (D) of link 2 $\widehat{f}_e$ is low. In the case of a collision with the tip (D) of link 2 the impact is partly compensated for because joint 2 would yield and move backwards while joint 1 (blue) would continue to move in the positive direction. In the case of a collision with point (C) of link 2 again joint 2 would yield, but in the opposite direction. This situation would require joint 1 to change its direction of motion. Therefore the impact force $\widehat{f}_e$ is higher at point (C) than at point (D). The highest impact forces on link 2 occur somewhere in the middle of the link where the impact can be compensated for the least by a compliant motion of the robot.

The second configuration of this robot shown in Figure 7.8 illustrates another important property of the impact force. The points on any joint become poles if the joint is the only moving joint in the system *and* the joints below can not compensate for all directions in the cyclic motion field around the joint. This follows directly from the pole conditions derived in the previous section. However, around the poles also low impact force values are visible. The logic behind this situation is illustrated in Figure 7.7. The red and green arrow represent two different directions from which the pole can be approached. For the green direction the purple velocity vectors are always aligned with the turquoise Jacobian vector of the joint 1 and therefore $\widehat{f}_e$ does not approach ∞. However, by approaching the pole from the red direction the velocity vectors become orthogonal to the Jacobian vector of joint 1 and therefore



a)



b)

Figure 7.6: Impact force $\widehat{f}_e$ on a 2 DOF manipulator arm

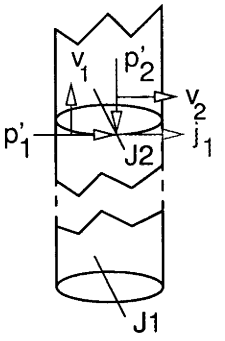
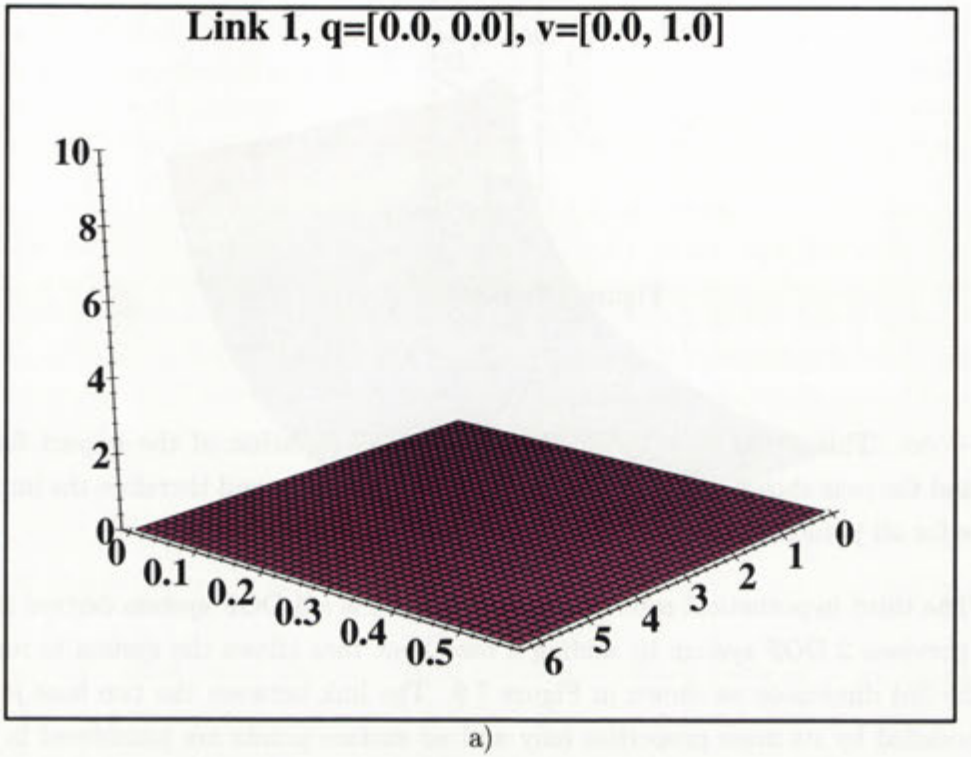


Figure 7.7: Poles at joint 2

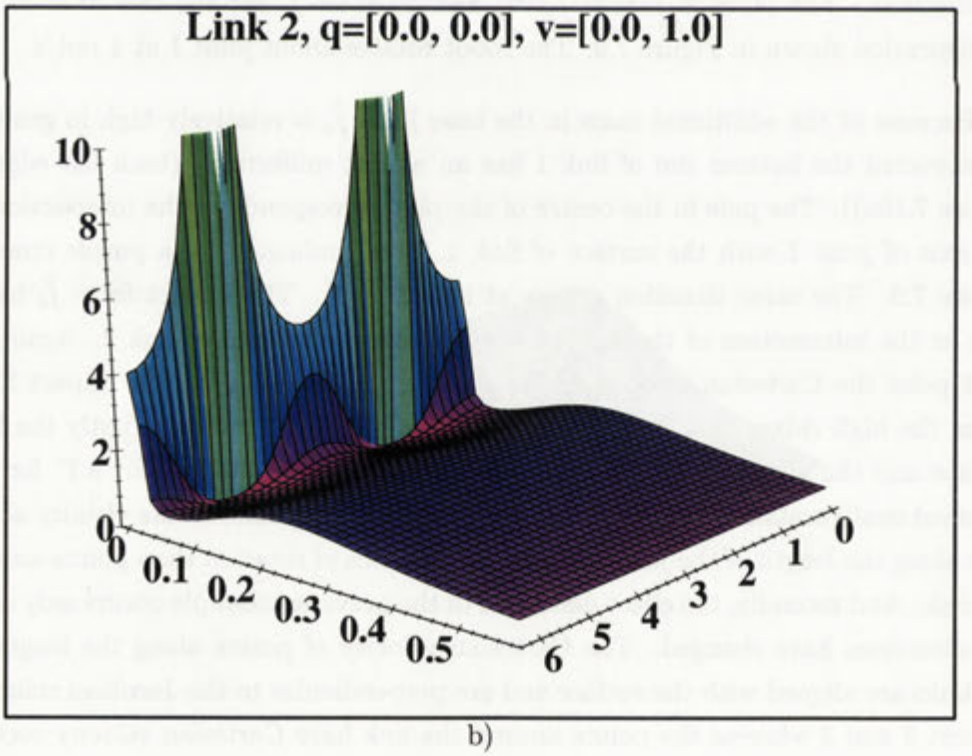
$\widehat{f}_e \to \infty$. This situation is reflected in the strong variation of the impact forces around the pole shown in Figure 7.8b). Link 1 is stationary and therefore the impact force for all points in Figure 7.8a) on this link equals 0.

The third hypothetical robot that is considered is a 3 DOF system derived from the previous 2 DOF system by adding a base joint that allows the system to rotate in the 3rd dimension as shown in Figure 7.9. The link between the two base joints is modelled by its mass properties only and no surface points are considered in the calculations. The plots in Figure 7.10a) and b) show $\widehat{f}_e$ for the two links in the configuration shown in Figure 7.9. The robot rotates about joint 1 at 1 rad/s.

Because of the additional mass in the base joint $\widehat{f}_e$ is relatively high in general. As expected the bottom rim of link 1 has an almost uniform $\widehat{f}_e$ (back left edge in Figure 7.10a)). The pole in the centre of the plot corresponds to the intersection of the axis of joint 1 with the surface of link 1 (blue) indicated by a purple cross in Figure 7.9. The same situation occurs at link 2 (red). The impact force $\widehat{f}_e$ has a pole at the intersection of the axis of joint 1 with the surface of link 2. Again, at both poles the Cartesian velocity of the surface points equals 0. The impact force forms the high ridges along the length of the link for two reasons. Firstly the link surface and the joint axis 1 intersect at a shallow angle of only about 9.7° for the assumed configuration $\theta = (0, 0.17, -0.34)$ and therefore points in the vicinity of the pole along the length of the joint are closer to the axis of rotation than points around the link. And secondly, the effect described in the previous example occurs only after the directions have changed. The Cartesian velocity of points along the length of the links are aligned with the surface and are perpendicular to the Jacobian columns of joint 2 and 3 whereas the points around the link have Cartesian velocity vectors which are almost aligned with the Jacobian columns of joint 2 and 3. These two effects are responsible for the high ridges along the length of the link. This example shows clearly that singularities can not only occur at the joint between two links, but at any point where the joint axis of a joint intersects the surface of a link.



a)



b)

Figure 7.8: Impact force $\widehat{f}_e$ on a 2 DOF manipulator arm

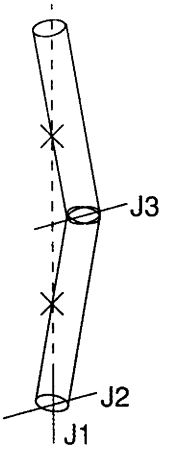


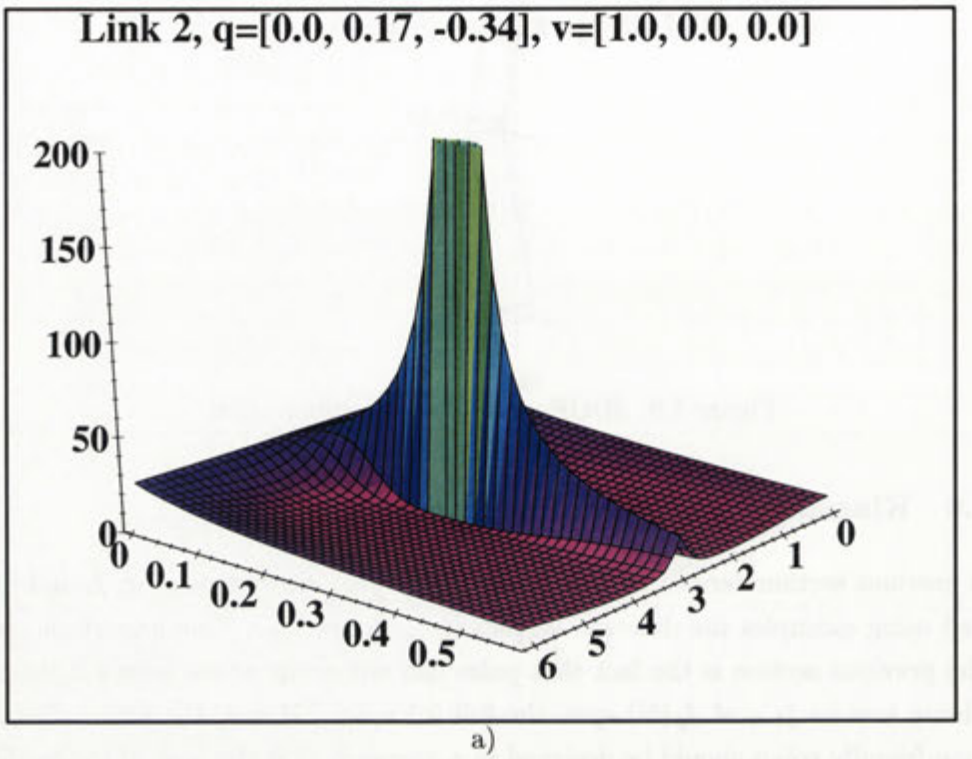
Figure 7.9: 3DOF manipulator configuration

7.2.6 Kinematics of human-friendly robots

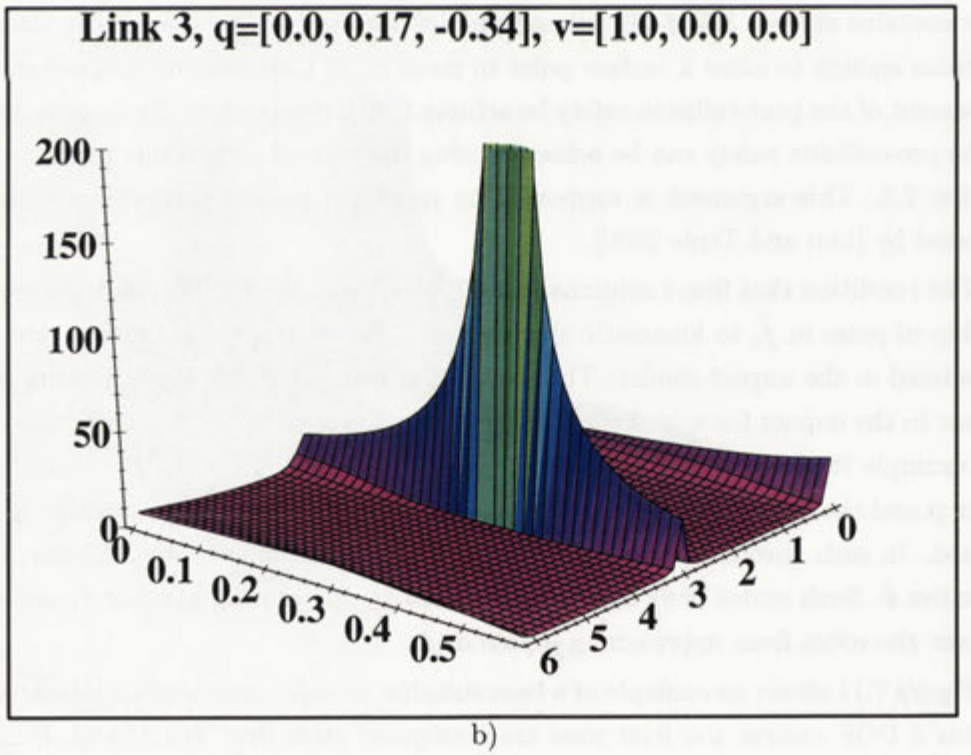
The previous section derived conditions for the appearances of poles in $\widehat{f}_e$ and illustrated using examples the different kinematic configurations. One important result of the previous section is the fact that poles can not occur above joint i if the first i column vectors $\mathbf{j}_{1-i}$ of $J_{\mathbf{p}}(\Theta)$ span the full 3D space. Hence, the kinematics of a human-friendly robot should be designed in a way such that the base of the manipulator contains at least 3 non-aligned and non-intersecting joints. Only if the robot is dextrous enough to allow a surface point to move in all Cartesian directions can improvement of the post-collision safety be achieved with compliance. An improvement of the pre-collision safety can be achieved using the control constraints presented in Section 7.3. This argument is supported by results of passive compliance systems reported by [Lim and Tanie 2000].

The condition that first i columns $\mathbf{j}_{1-i}$ of $J_{\mathbf{p}}(\Theta)$ span the full 3D space shows the kinship of poles in $\widehat{f}_e$ to kinematic singularities. However, spatial rotations are not considered in the impact model. The joint configurations θ under which poles may appear in the impact force are often the same in which kinematic singularities occur. For example in the case that two or more joints are coincident or the considered point $\mathbf{p}$ and the two or more joint axes lie within a plane then the respective $\mathbf{j}_k$ are aligned. In such special joint configurations θ poles can appear for particular joint velocities $\dot{\theta}$. Such states $(\theta\,\dot{\theta})$ must be prevented by the motion planner in order to prevent the robot from approaching a pole in $\widehat{f}_e$.

Figure 7.11 shows an example of a base suitable for safe human-robot interaction. In this 3 DOF system the joint axes are configured such that $J_1 \perp J_2$ and $J_2 \perp J_3$. In a general configuration, points on links above J_3 are not poles of $\widehat{f}_e$ if they are not coincident with any of the first three joint axes. The spherical shell around the base is part of the cover of the link above J_3 and therefore the surface of the base is free of poles in $\widehat{f}_e$ except for the special configuration of $\theta_2 = n\pi$. In this special



a)



b)

Figure 7.10: Singularities in $\widehat{f}_e$ for a 3 DOF manipulator

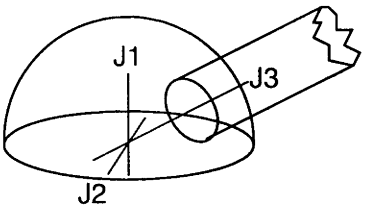


Figure 7.11: Example of a base design for a human-friendly robot

configuration J_1 and J_3 are coincident and therefore poles appear at the intersection of the sphere with J_2 if $\dot{\theta}_1 = \dot{\theta}_3 = 0 \neq \dot{\theta}_2$. This special case must be prevented by the motion planner which is desirable since this configuration is a kinematic singularity. The proposed robot base design is well suited for human-robot interaction systems.

Robots which provide insufficient dexterity therefore pose a higher risk in the event of a collision. If $\widehat{f}_e$ has a pole at any surface point $\mathbf{p}$ of the robot it is not possible to provide a quantitative limit for the contact force in the case of a collision and therefore, the system is unsuited for human-machine interaction.

7.2.7 Definition of Impact Potential

One goal of this research is to derive control constraints for a human-friendly manipulator which allow a person to physically interact with a robot in a safe way. Considering the possible legal liability problems[2] of human-friendly robots it is crucial to identify ways to provide quantitative safety guarantees for such systems. The potential impact force of the manipulator is an appropriate control parameter to guarantee safe interaction between a robot and a person in the real world. In this section the basic control parameter based on impact force f_e is derived.

With $\mathbf{v} = J_\mathbf{p}(\Theta)\dot{\Theta}$ and $s_v = |\mathbf{v}| = |J_\mathbf{p}(\Theta)\dot{\Theta}|$, $\widehat{f}_e$ can be expressed as a function of the state $(\Theta, \dot{\Theta})$ of the mechanical system.

$$\widehat{f}_e = \frac{2|J_\mathbf{p}(\Theta)\dot{\Theta}|^3}{\dot{\Theta}^t J_\mathbf{p}^t(\Theta) J_\mathbf{p}(\Theta) M^{-1}(\Theta) J_\mathbf{p}^t(\Theta) J_\mathbf{p}(\Theta)\dot{\Theta}} \tag{7.33}$$

To describe the ability of a robot to cause impact the term *impact potential* is introduced. Impact potential is defined as the maximum impact force that a moving mechanical system can create in a collision with a static obstacle. In this case all points $\mathbf{p} \in \mathbf{P}$ on the surface of the robot have to be considered. $\widehat{f}_e : \mathbf{P} \times \mathbb{R}^n \times \mathbb{R}^n \mapsto \mathbb{R}$ is a function of the point $\mathbf{p}$ and the dynamic state $(\Theta, \dot{\Theta})$ of the system. We can now define the impact potential π of the system as

$$\pi = \sup_{\mathbf{p} \in \mathbf{P}} \pi_\mathbf{p}, \quad \text{where } \pi_\mathbf{p} = \widehat{f}_e(\mathbf{p}, \Theta, \dot{\Theta}) \tag{7.34}$$

The impact potential is a well-defined scalar value that provides an upper limit

[2]Compared to the situation of autonomous highway driving described in Chapter 6.

for any impact of the system with a static object. Since it assumes elastic impact and aligned surface normals, the limit is rather conservative.

With the impact potential π a controllable parameter is available. To limit the impact potential of the robot to a desired level a limit π_{max} is set to an appropriate level. This maximum impact potential for a robot defines the *safe region* in the state space $(\Theta, \dot{\Theta})$ where the following inequality holds

$$\pi_{\mathbf{p}} \leq \pi_{max} \ \forall \mathbf{p} \in \mathbf{P} \tag{7.35}$$

7.2.8 Controlling the impact force

A novel *Impact Potential Control* scheme (IPC) is proposed in this research to ensure that the impact force of all surface points does not exceed a preset limit and the robot can not leave the safe region in the state space by autonomous actions. The safety envelope which encapsulates the safety core acts as a saturating filter between the external motion control algorithm and the robot. The safety envelope passes the torque vector generated by the motion control algorithm unchanged to the robot as long as it complies with the safety constraints. If the desired torque vector does not satisfy these safety constraints, a clipping function must be applied to the torque vector. The resulting safe torque vector guarantees that the robot does not leave the safe region in state space. This satisfies the SAFETY REQUIREMENT 3 defined in Chapter 6.

SAFETY REQUIREMENT 2 is related to the bandwidth of the robot's motion and it must be transformed into a control parameter. The impact potential π can also be used to achieve this requirement. The robot under this safe control scheme operates in the interval $\pi \in [0, \pi_{max}]$ where 0 corresponds to a stationary robot and impact abilities close to π_{max} correspond to a rapid velocity (of course this formulation is not a velocity limitation but it is velocity dependent). Therefore limiting the bandwidth of π effectively limits the bandwidth of the velocity of the robot.

A suitable state-dependent control constraint which conforms to both safety requirements is defined in the following inequality.

$$\frac{d}{dt}\pi_{\mathbf{p}} \leq \frac{1}{t_c}(\pi_{max} - \pi_{\mathbf{p}}) \ \forall \mathbf{p} \in \mathbf{P} \tag{7.36}$$

This constraint limits the increase in impact potential $\frac{d}{dt}\pi_{\mathbf{p}}$, to the difference between the current impact potential and the maximum impact potential. The time constant t_c defines how fast the robot may gain impact potential. The value is set such that the motion of the robot is sufficiently predictable to the human operator.

The control constraint defined in Equation 7.36 differs significantly from the following constraint which could also be used.

$$\frac{d}{dt}\pi \leq \frac{1}{t_c}(\pi_{max} - \pi) \tag{7.37}$$

In Equation 7.36 since π is the supremum over the impact potential of all points on the surface $\mathbf{P}$ this constraint will guarantee SAFETY REQUIREMENT 3. However, SAFETY REQUIREMENT 2 can be compromised within this constraint. Only the point with the highest $\pi_{\mathbf{p}}$ is restricted tightly by the constraint. Other points $\mathbf{p}$ with lower $\pi_{\mathbf{p}}$ are allowed to increase their impact potential at any rate as long as $\pi_{\mathbf{p}} < \pi$. In practice a subset of the links of the robot are able to accelerate sharply. Therefore the control constraint in Equation 7.36 provides the robot with a superior behaviour.

7.3 Implementing Impact Potential Control

To implement the proposed IPC-scheme Equation 7.36 must be transformed into motor torque space. The transformed control constraint allows verification of whether a given torque vector τ conforms to the safety restrictions. As stated earlier, this classification is a integral part of the safety envelope. This section derives the necessary mathematical tools for the implementation of the IPC-scheme on a real robot manipulator.

7.3.1 Transformation to torque space

Considering that $\pi_{\mathbf{p}}(t) = \pi_{\mathbf{p}}(\Theta(t), \dot{\Theta}(t))$, Equation 7.36 can be expanded to the following representation.

$$\frac{\partial \pi_{\mathbf{p}}(\Theta, \dot{\Theta})}{\partial \dot{\Theta}}\ddot{\Theta} + \frac{\partial \pi_{\mathbf{p}}(\Theta, \dot{\Theta})}{\partial \Theta}\dot{\Theta} \leq \frac{1}{t_c}(\pi_{max} - \pi_{\mathbf{p}}(\Theta, \dot{\Theta})) \;\; \forall \mathbf{p} \in \mathbf{P} \tag{7.38}$$

Inequality 7.38 represents a system of state dependent constraints on the accelerations $\ddot{\theta}$ of the joints. Using the dynamic model $\ddot{\theta} = M^{-1}(\Theta)(\tau - N(\Theta, \dot{\Theta})$ for the manipulator the Inequality 7.38 can be expanded further.

$$\frac{\partial \pi_{\mathbf{p}}}{\partial \dot{\Theta}}\left[M^{-1}(\Theta)(\tau - N(\Theta, \dot{\Theta})\right] + \frac{\partial \pi_{\mathbf{p}}}{\partial \Theta}\dot{\Theta} \leq \frac{1}{t_c}(\pi_{max} - \pi_{\mathbf{p}}) \;\; \forall \mathbf{p} \in \mathbf{P} \tag{7.39}$$

Reordering of Inequality 7.39 shows the nature of the constraints more clearly.

$$\frac{\partial \pi_{\mathbf{p}}}{\partial \dot{\Theta}}M^{-1}\tau \leq \frac{1}{t_c}(\pi_{max} - \pi_{\mathbf{p}}) - \frac{\partial \pi_{\mathbf{p}}}{\partial \Theta}\dot{\Theta} + \frac{\partial \pi_{\mathbf{p}}}{\partial \dot{\Theta}}M^{-1}N \;\; \forall \mathbf{p} \in \mathbf{P} \tag{7.40}$$

The torque constraint vector $\mathbf{c_p} \in \mathbb{R}^{1\times n}$ and a scalar limitation value $l_{\mathbf{p}}$ are defined as follows.

$$\mathbf{c_p} = \frac{\partial \pi_{\mathbf{p}}}{\partial \dot{\Theta}}M^{-1} \tag{7.41}$$

$$l_{\mathbf{p}} = \frac{1}{t_c}(\pi_{max} - \pi_{\mathbf{p}}) - \frac{\partial \pi_{\mathbf{p}}}{\partial \Theta}\dot{\Theta} + \frac{\partial \pi_{\mathbf{p}}}{\partial \dot{\Theta}} M^{-1} N \tag{7.42}$$

Substitution into Equation 7.40 yields the simplified form of the torque constraints.

$$\mathbf{c_p}\tau \leq l_{\mathbf{p}} \ \forall \mathbf{p} \in \mathbf{P} \tag{7.43}$$

The inequalities in Equation 7.43 represent a closed halfspace in torque space. The hyperplanes which separate the torque vectors that comply with the constraint in Inequality 7.43 are defined by the equation $\mathbf{c_p}\tau = l_{\mathbf{p}}$. The safe region in torque space that contains all torque vectors that comply to the safety constraints for the robot is given by the intersection of all the halfspaces.

7.3.2 Torque verification

The Inequality 7.43 expresses the control constraints which are required for the torque vector verification module in the safety envelope (refer to Figure 7.1). Each torque vector τ_e generated by external motion control algorithms must satisfy the inequalities for all points $\mathbf{p}$ on the surface $\mathbf{P}$ of the robot. If all inequalities are satisfied τ_e is allowed to pass the safety envelope unchanged and becomes the safe actuator command torque vector τ_s for the robot.

The torque verification module also provides simple feedback to the position controller. Any external motion controller that seeks to operate the robot at the boundaries of the safe region in torque space requires feedback. It is necessary to know where a command torque vector is located with respect to the control constraints, in other words, how far from the boundaries of the safe region the torque vector is located. This minimum distance d is calculated by the torque verification module *if* the torque vector was safe. Since the intersection of halfspaces is convex, the closest point on the boundary from a point τ *within* the intersection is simply the minimum of the distances from τ to its orthogonal projection into each of the hyperplanes which define the boundaries of the halfspaces.

$$d = \min_{\mathbf{p} \in \mathbf{P}} \text{OrthProj}(\tau, (\mathbf{c_p}, l_{\mathbf{p}}))$$

If the command torque vector τ_e is not safe, the distance d between τ_e and the envelope acts as feed back to the external controller to indicate the magnitude of the constraint violation. This allows the external controller to take appropriate measures. The details of the calculation of the orthogonal projection of a point into a subspace are presented in Appendix F.

7.3.3 Torque clipping

As discussed in Section 7.1 the real-time requirement of the safety core requires a decision within the core if the external controller is not able to provide a safe torque

vector within the execution cycle. The safety core must be inherently safe and it is not feasible to rely in the core design for all external controllers to provide safe torque vectors in time. The safety core must have the ability to generate a safe torque vector from an unsafe torque vector.

This process is called *clipping*. The expression refers to the effect that unsafely large torque vectors are reduced in length while it is desirable to retain the general orientation of the torque vector in order to provide the external controller with a degree of controllability. The clipping is complicated by the fact that a zero torque vector is not safe in all situations. The scaling of τ_e does not necessarily produce a safe torque vector. The solution we propose in this research is to find the closest safe torque vector to τ_e. The solution generated by this clipping function is as close to the desired torque vector as possible. However, it does not provide any guarantee about the orientation of the generated torque vector.

The clipping function is calculated for a set $\mathbf{P}$ of surface points $\mathbf{p}$ with the following algorithm. The input parameters for the algorithm are the desired joint torque vector $\tau_e \in \mathbb{R}^n, n < \infty$ and the set of constraints defined in Inequality 7.43. The basic idea of the algorithm is to calculate the orthogonal projection of τ_e into all constraint hyperplanes *and* into all pairwise intersections of the hyperplanes. The process of forming a new set of subspaces through pairwise intersections is continued until the result of all intersections is empty. The safe torque vector τ_s is found by finding the projection which is closest to τ_e and which is valid with respect to the control constraints. Algorithm 7.1 summarises the calculations.

Algorithm 7.1 Torque vector clipping function: `clip (`τ_e`, Constraints)`

```
use Sets Subspaces, ProjPoints, ValidPoints=empty
Subspaces = Hyperplanes(Constraints)
repeat
  ProjPoints = Orthogonal_Projection(τe, Subspaces)
  ValidPoints = ValidPoints ∪ select (ProjPoints, satisfy(Constraints))
  Subspaces = All_Binary_Intersections(Subspaces)
until (Subspaces = empty)
τs= select(τ ∈ValidPoints, min |τ − τe|))
return τs
```

The details of the calculation of the intersection of affine subspaces are given in Appendix F.

In the case of an unsafe command torque vector τ_e the distance d of the command torque vector τ_e to the boundary of the safe region in command torque space is obviously $|\tau_e - \tau_s|$. To indicate to the external position controller that the command torque was *outside* the safe region the feedback value d is set to the negative distance $d = -|\tau_e - \tau_s|$.

Preconditions

The set $\mathbf{P}$ of surface points is not empty for any manipulator. Therefore the set of constraints is also not empty. It requires the mechanical system to have a finite number of joints $n < \infty$ and therefore the joint torque vectors are of finite dimension. The algorithm assumes that at least one point in the torque vector space complies to all constraints. In terms of the IPC-scheme this means that a safe torque vector exists. If all constraints are derived from impact force limits as proposed this is a reasonable assumption.

However, when an impact potential controller is implemented on a actual robot system, the torque limits of the motors form an additional hypercube of constraints in the joint torque space. If the robot is pushed hard enough by external forces, none of the physically possible torque vectors may be safe in terms of the given impact force constraints.

Termination

The set of subspaces can not contain the same subspace twice. The intersection of non-identical subspaces reduces the dimension of the resulting subspace by one. Therefore the dimension of the subspaces is reduced by 1 in each iteration. Thus, in the n-th iteration the set of subspaces considered are 0-dimensional. The intersection between two non-identical 0-dimensional subspaces is empty, and thus, the algorithm terminates after at most n iterations.

Correctness

The algorithm calculates the orthogonal projections of τ into the hyperplanes derived from the safety constraints and intersections for all combinations of those hyperplanes and finds the closest projection to τ within the safe region. We can prove that the closest point in the safe region is the orthogonal projection into one of those subspaces.

The closest point to τ_e in the safe region is found by growing a n-dimensional sphere around τ_e. The first point $\mathbf{c}$ of contact between the sphere and the safe region satisfies corresponds to the safe torque vector closest to τ_e.

The intersection of half spaces is always convex and therefore $\mathbf{c}$ is unique. Because the safe region is bordered by constraint hyperplanes $\mathbf{c}$ is part of m, $1 \leq m \leq |\mathbf{P}|$ hyperplanes. If $m = 1$ then $\mathbf{c}$ is part of a single hyperplane. Since the connection between τ_e and $\mathbf{c}$ is the radius of a sphere around τ_e which touches the hyperplane in $\mathbf{c}$, $\mathbf{c}$ is the orthogonal projection of τ_e into this hyperplane. It is therefore member of the *ValidPoints*-set and is returned by the algorithm. If $m > 1$ let S be the subspace created by the intersection of all those hyperplanes. The dimension of S is constrained to $\max(0, n - m) \leq \dim(S) \leq n - 1$. If $\dim(S) = 0$ then $\mathbf{c}$ is the orthogonal projection into a 0-dimensional subspace which contains a single point

with equals $\mathbf{c}$. Thus, $\mathbf{c}$ is part of the *ValidPoints*-set when the loop terminates and is returned as the closest point. If $\dim(S) \geq 1$ then $\mathbf{c}$ is the orthogonal projection of τ_e. Otherwise the orthogonal projection $\mathbf{c}' \neq \mathbf{c}$ of τ_e can not be in the safe region since the sphere around τ_e touches S first in the orthogonal projection of τ_e into S. If $\mathbf{c}$ is safe and $\mathbf{c}'$ is unsafe another hyperplane must intersect the connection line in S between $\mathbf{c}$ and $\mathbf{c}'$ at a third point $\mathbf{d} \neq \mathbf{c} \neq \mathbf{c}'$ which is still safe. However, the distance of points along the connection line in S between $\mathbf{c}$ and $\mathbf{c}'$ increases strictly monotonously and therefore the sphere must have intersected $\mathbf{d}$ before $\mathbf{c}$. This implies $\mathbf{c}$ must equal $\mathbf{d}$. Therefore, $\mathbf{c}$ must have been the orthogonal projection of τ_e. Hence, $\mathbf{c}$ is part of the *ValidPoints*-set when the loop terminates and is returned by the algorithm.

7.3.4 Stationary robot

The formulation of the control constraints in Section 7.3.1 is defined for a robot in motion ($\dot{\Theta} \neq 0$). For a stationary robot ($\dot{\Theta} = 0$) Equation 7.38 is undefined since the partial derivative of $\pi_{\mathbf{p}}$ with respect to $\dot{\Theta}$ does not exist. The impact force for points on stationary links was defined in Equation 7.32 to equal 0. Also if $\mathbf{P}$ does not contain a pole then $\pi \to 0$ as $\dot{\Theta} \to 0$. However, the increase of the impact potential depends on the direction in which $\dot{\Theta}$ moves away from 0. Hence, the directional derivative of $\pi_{\mathbf{p}}$ in the direction of a unit vector $\mathbf{u}$,$\mathbf{u} \in \mathbb{R}^n$, $|\mathbf{u}| = 1$ must be considered for a stationary robot ($\dot{\Theta} \neq 0$).

$$\frac{\partial \pi_{\mathbf{p}}}{\partial \mathbf{u}} = \lim_{\varepsilon \to 0} \frac{\pi_{\mathbf{p}}(\Theta, \dot{\Theta} + \varepsilon \mathbf{u}) - \pi_{\mathbf{p}}(\Theta, \dot{\Theta})}{\varepsilon} \tag{7.44}$$

For any given external command torque τ_e the joint accelerations $\ddot{\Theta}_e$ that result from this torque vector can be derived from the dynamic model of the robot. Note; that $N(\Theta, \dot{\Theta}) = 0$ for a stationary robot without gravitational effects. Therefore,

$$\ddot{\Theta}_e = M^{-1}(\Theta)\tau_e \tag{7.45}$$

If $\tau_e = 0$ the joint acceleration $\ddot{\Theta}_e$ equals 0 and the robot remains stationary. Therefore, no further calculations are required since the zero command torque vector satisfies the constraints in Equation 7.38 for a stationary robot.

For a command torque $\tau_e \neq 0$ Equations 7.44 and 7.45 are used to calculate the impact force constraint. Since the robot is stationary, $\dot{\Theta}$ and $\ddot{\Theta}$ are initially aligned[3]. Thus, the partial derivative of $\pi_{\mathbf{p}}$ with respect to $\dot{\Theta}$ is replaced by the directional derivative of $\pi_{\mathbf{p}}$ in direction of $\ddot{\Theta}_e$. Let $\mathbf{w}_e = s\ddot{\Theta}_c$, $s > 0$ such that $|s\ddot{\Theta}_c| = 1$. For a given torque vector $\tau_e \neq 0$ and a stationary robot the safety constraint in

[3]The orientation of the two vectors is the same at an infinitesimal time t after the acceleration started. Mathematically speaking, if the accelleration $\ddot{\Theta}$ starts at time $t = 0$ then the unit vectors $\dot{\Theta}_1 = \frac{\dot{\Theta}}{\|\dot{\Theta}\|}$, $\ddot{\Theta}_1 = \frac{\ddot{\Theta}}{\|\ddot{\Theta}\|}$ are identical for $t \to 0$: $\lim_{t \to 0} \dot{\Theta}_1(t) = \lim_{t \to 0} \ddot{\Theta}_1(t)$.

Equation 7.40 is replaced by the following:

$$\frac{\partial \pi_{\mathbf{p}}}{\partial \mathbf{w}_e} M^{-1} \tau_e \leq \frac{\pi_{max}}{t_c} \tag{7.46}$$

given that $N(0,0) = 0$ and $\dot{\Theta} = 0$. If the command torque τ_e does not satisfy inequality 7.46, it can be scaled down such that it satisfies Equation 7.46 rewritten as an equality. In the special case of a stationary robot scaling the command torque vector τ_e is possible since a zero torque is always safe. Unfortunately, the closest torque vector to τ_e within the safe region is analytically difficult to determine because the value of $\frac{\partial \pi_{\mathbf{p}}}{\partial \mathbf{w}_e}$ changes with the direction of τ_e and therefore the boundaries of the safe region are curved (refer to Figure 7.14).

It is questionable if it is necessary to consider the case of the stationary robot in such detail. The alternative is to allow any torque vector for $\dot{\Theta} = 0$. In the continuous-time case the safety restrictions can only be violated at the beginning of an acceleration with a duration of 0 which certainly does not violate the spirit of the safety requirements. In the discrete-time case of an actual implementation arbitrary joint torques applied to a stationary robot can cause twitching movements that can make the robot appear to be unsafe and unpredictable. The formulation of the limits for a stationary robot permits smooth commencement of motions. This approach is used in the implementation of the safety scheme.

7.3.5 Unsafe robot state

Limiting the joint torques prevents the robot from leaving the safe region in the state space. However, a robot with backdriveable joints may still experience external forces such as a person pushing or pulling the arm which can add to the actuation forces and push the robot out of the safe region. In this case the safety core should provide a mechanism which brings the robot back into a safe region provided that the external forces allow this to occur.

If the robot leaves the safe region and $\pi > \pi_{max}$ then the right side of Equation 7.36 becomes negative and the control constraint will enforce a reduction of the impact potential according to the time constant t_c without depending on the output of the external controller.

7.4 Simulation of a 2 DOF robot

The effectiveness of the new safety paradigm is illustrated in a simulation of a simple 2 DOF planar robot. Figure 7.12 shows the structure of the robot with two point masses m_1 and m_2 at the end of the upper and lower arm respectively of the manipulator. The simulation assumes compensation of the gravitational effects. The mass

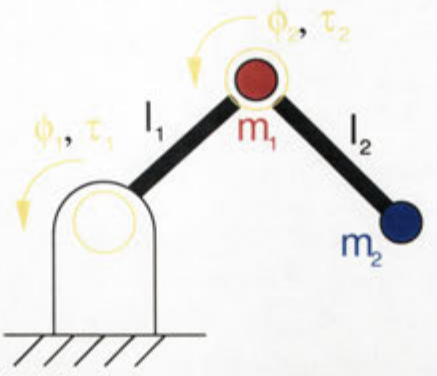


Figure 7.12: Two link robot with two point masses at the end of the links

matrix of the robot equals

$$M(\Theta) = \begin{bmatrix} l_2^2 m_2 + 2 l_1 l_2 m_2 \cos\theta_2 + l_1^2 (m_1 + m_2) & l_2^2 m_2 + l_1 l_2 m_2 \cos\theta_2 \\ l_2^2 m_2 + l_1 l_2 m_2 \cos\theta_2 & l_2^2 m_2 \end{bmatrix} \tag{7.47}$$

and the velocity term equals

$$N(\Theta, \dot{\Theta}) = \begin{bmatrix} -m_2 l_1 l_2 \dot{\theta}_2^2 \sin\theta_2 - 2 m_2 l_1 l_2 \dot{\theta}_1 \dot{\theta}_2 \sin\theta_2 \\ m_2 l_1 l_2 \dot{\theta}_1^2 \sin\theta_2 \end{bmatrix} \tag{7.48}$$

In the simulation both link lengths are set to $1m$ and both point masses are set to $1kg$. The time constant t_c is set to 1 and the maximum impact potential π_{max} in the safety core is set to $1N$.

7.4.1 Impact control point selection

The implementation of the IPC scheme requires a finite subset $\tilde{\mathbf{P}}$ of control points from the set $\mathbf{P}$. The set of surface points $\mathbf{P}$ for a real manipulator is infinite, and hence, the numerical evaluation of the control constraints for surface point is impossible. Our aim is to select a suitable finite subset of surface points $\tilde{\mathbf{P}} \subseteq \mathbf{P}$ which approximates the true impact potential of the manipulator as accurately as possible. Elaborate sets $\tilde{\mathbf{P}}$ that consider more detailed robot geometry may yield more precise results, but the IPC scheme remains applicable as long as $\tilde{\mathbf{P}}$ is finite and the IPC scheme provides correct limits of the impact force for all points in $\tilde{\mathbf{P}}$.

In an implementation on an actual robot where the consideration of the available computational resources and the required update frequency is important it is possible to consider only a few impact potential control points. In a simulation an arbitrary number of IPC points can be considered, even though such a simulation does not reflect the behaviour of a real system with real-time constraints. Hence, the subset $\tilde{\mathbf{P}}$ of the simulated robot contains only two points[4] $\mathbf{p}_1$ and $\mathbf{p}_2$ which are coincident

[4]The same number of points was used in the implementation of the IPC scheme on the Barrett

with the location of the two point masses m_1 and m_2. This small subset allows visualisation of the control constraints, the clipping and demonstrates the feasibility of a safety core with only a few control points.

7.4.2 Static simulation

Figure 7.14 shows the control constraints for stationary robot and the clipping by the scaling of the torque vectors that exceed the safe regions. The control constraints are plotted in the joint torque space (τ_1, τ_2). The position of joint 1 is not relevant since it is assumed that no gravitational forces act on the robot. Figure 7.14a) shows the constraints for $\theta_2 = 0$, when the arm is extended as shown in Figure 7.13a). The red line in Figure 7.14a) shows the constraint derived from $\mathbf{p}_1$ and the blue line the constraint from $\mathbf{p}_2$. Note that any torque vector where τ_1 and τ_2 have a 3:1 ratio are valid under the constraint from $\mathbf{p}_1$. Such torque vectors would only accelerate the lower arm and m_2, and thus, the impact potential of $\mathbf{p}_1$ remains 0 as indicated in Figure 7.13b). A similar effect occurs with the torque vectors when $\tau_2 = 0$. With respect to the constraint from $\mathbf{p}_2$ the torque vectors can have any length. These torque vectors do only accelerate m_1 while m_2 only starts to be pulled towards joint 1 as indicated in Figure 7.13b), while the initial acceleration is 0. An example torque vector of $\tau = (-2, -2)$ is plotted in the graph and scaled such that it complies to both the red and the blue constraint.

Figure 7.14b) shows a similar plot for $\theta_2 = 0.5$ for the configuration shown in Figure 7.13c). The constraint from $\mathbf{p}_2$ (in blue) has changed its shape and size while torque vectors with a certain ratio always fulfil the constraint from $\mathbf{p}_1$. The example torque vector of $\tau = (2.5, 1.5)$ complies to the constraint from $\mathbf{p}_1$ but not the constraint from $\mathbf{p}_2$ and is therefore scaled appropriately.

The plot in Figure 7.14c) shows the constraints for $\theta_2 = \frac{\pi}{2}$ for the configuration shown in Figure 7.13d). In this configuration the lower arm is perpendicular to the upper arm. In this configuration the torque vectors $\tau_1 = \tau_2$ only accelerate m_2 and are therefore valid under the constraint from $\mathbf{p}_1$. The constraint from $\mathbf{p}_2$ is reduced in size and forms a circle like region around the origin. The explanation for the smaller size of the constraint along the major axis is because in this configuration torques in joint 1 require m_2 to accelerate at the same angle acceleration as m_1 around the axis of joint 1, but only at a larger distance. The example torque of $\tau = (0.4, 0.4)$ is well within both constraints in this case.

7.4.3 Dynamic simulation

Various simulations of situations where the robot is initially inside and outside the safe region in the state space have been conducted. Figure 7.15 shows the results of an experiment where the robot is initially at rest with $\Theta = (-0.7, 0.8)$. The external

WAM robot manipulator.

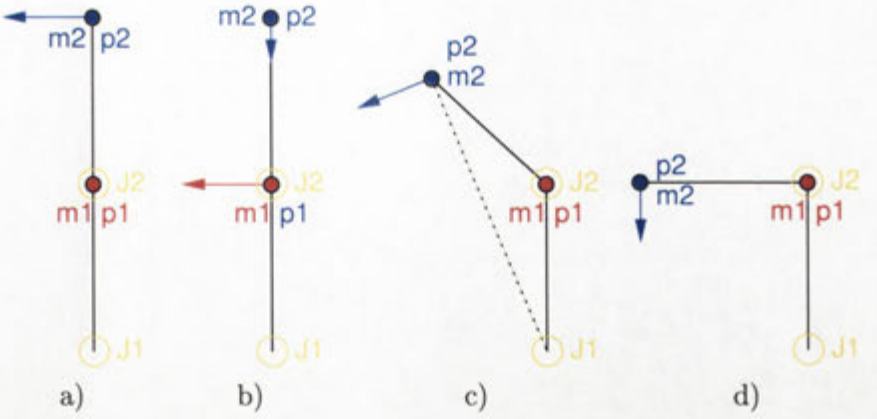


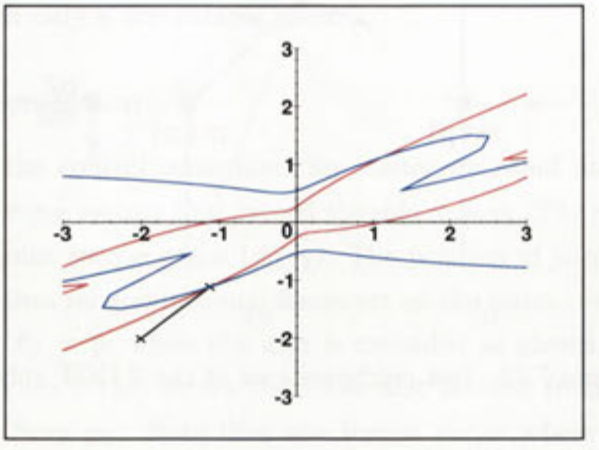
Figure 7.13: Test configurations of the 2 DOF robot

controller produces a constant torque vector $\tau_e = (0.2, 0.5)$, a control strategy which would result in high speed motion without the safety core. This torque vector is initially safe but soon after the start of the motion the command torque vector becomes unsafe and must be clipped by the safety envelope. The bottom row of the figure shows the plots of the safe region in torque space. The row above shows the configuration of the robot. The two joints of the robot are marked with red circles and the point masses with black dots. The dotted green lines indicate the point in time when the snapshot shown in the two bottom rows was taken. The top plot shows the joint velocities $\dot{\theta}_1$ in red and $\dot{\theta}_2$ in blue. Initially $\dot{\theta}_1$ becomes negative because of the moments created by τ_2 and m_2 (τ_2 is more than double τ_1). The second plot shows the potential impact force $\widehat{f}_e$ of the two control points. The point located at m_2 quickly reaches the limit of $\pi_{max} = 1$ where it remains for the rest of the simulation.

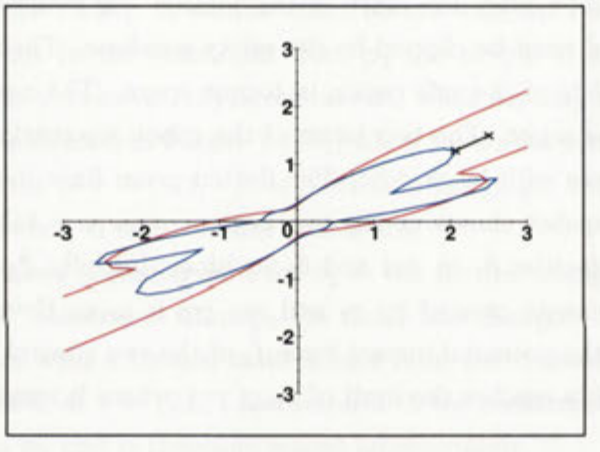
The torque space plots show how quickly the control torque τ_e becomes unsafe and must be clipped by the safety envelope. The lower arm quickly gains angular velocity and performs a greater than 360° rotation by the end of the simulation. The plots of $\dot{\theta}_1$ and π_1 clearly show that the velocity of m_1 and therefore the potential impact force at this point reaches a short maximum when m_2 becomes coincident with the joint axis of joint 1. During the approach to this configuration $\dot{\theta}_2$ picks up speed because the point mass m_2 comes close the joint axis 1. After m_2 passes joint 1 the angular velocity of both joints is reduced by the safety core in order to keep the impact potential below the limit which is set at 1.

Figure 7.16 shows another experiment in which the robot is initially moving with a joint velocity vector $\dot{\Theta} = (-1.5, -1.5)$. At this speed $\pi_2 > 7\pi_{max}$ and the robot is obviously not in a safe state. The external controller continuously outputs a command torque vector of $\tau_e = (-0.4, -0.3)$. The IPC core reduces the impact potential π of the system to 2 N after 2 s and later to 1 N after 4 s.

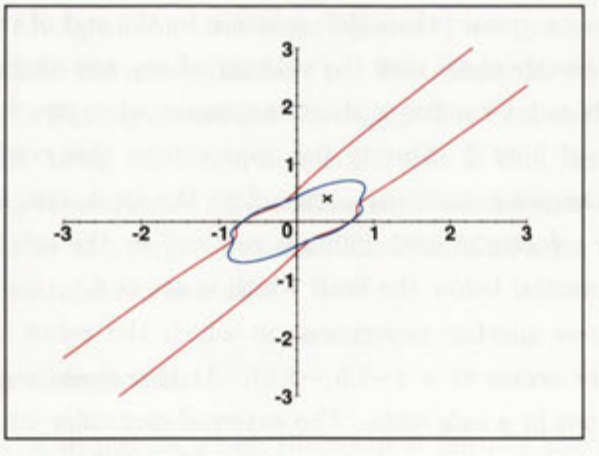
The simulation results demonstrate that the IPC-scheme is sound and limits the impact potential of a manipulator provided no external forces act on the robot. If



a)



b)



c)

Figure 7.14: Safe region in the torque space

Figure 7.15: Simulation of a 2DOF robot with IPC core

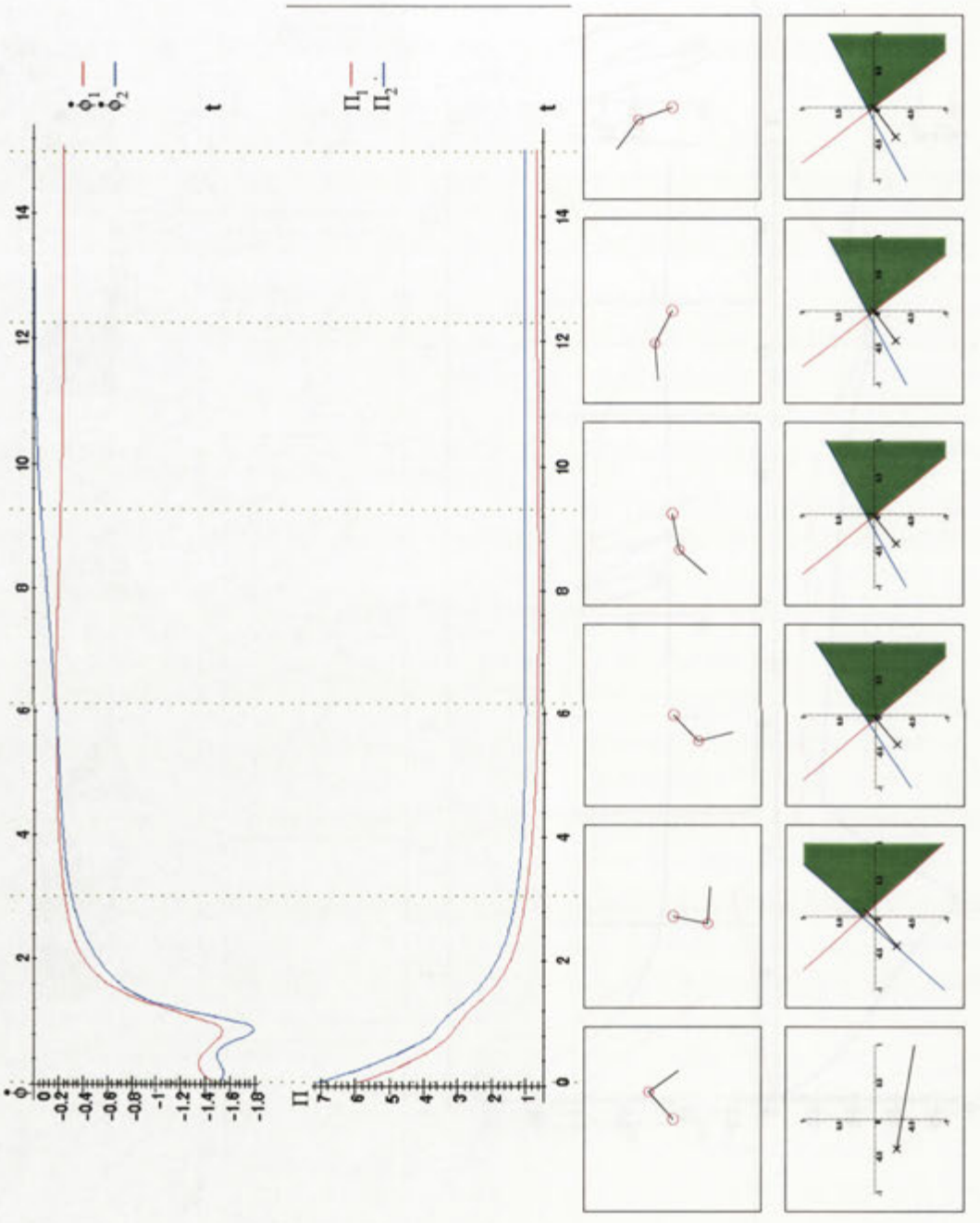


Figure 7.16: Simulation of the robot initially in an unsafe state

the robot leaves the safe region the IPC scheme enforces a continuous reduction of the impact potential until the system returns to the safe region. The time constant t_c influences both the ability of the robot to accelerate within the safe region and the minimum impact potential reduction rate outside the safe region. It is also possible to decouple these two effects by defining two different time constants for operation inside and outside the safe region. This does not cause temporal discontinuities in the control constraint at transitions between the two values since the transition always occurs for $\pi = \pi_{max}$ and therefore the right side of Equation 7.38 will equal 0.

7.5 Summary

This chapter has shown how quantitative limits of the impact potential as a measure of the pre-collision threat posed by a serial manipulator to a person can be achieved. The basic component of the scheme is a state dependent physical parameter that can be limited. The transformation of the respective control constraints into the command space combined with a verification module and a clipping module which allows the verification of commands and projects them into the safe region of the command space.

The scheme derived in this chapter uses the impact force of all surface points of the robot as the limit parameter. The impulsive impact force is an intuitive choice for pre-collision safety guarantees. The approach models the actual threat potential of the robot in a superior manner compared with the kinetic energy of the robot [Li and Horowitz 1995]. This is because the new approach takes into account the kinematic configuration of the robot, and the resulting compliance of the mechanism with respect to collisions with particular parts of the robot. The original impact model proposed by [Walker 1994] has been augmented to account for friction. This collision model is more realistic than the friction-free original model. This extension also eliminates the wedge effect that produces infinitesimally high impact forces at surface points where the surface normal is directed towards the joint axis. The wedge effect represents a serious limitation of the original formulation for safety considerations. For disadvantageous surface geometries even the smallest motion of a joint may produce arbitrary high potential impact forces. On the other hand, poles of the impact force close to joint axes reflect the true injury potential of the mechanical system. This mechanical effect can generate high impact forces and therefore must be considered in a safety control strategy. Simple solutions such as limiting the kinetic energy of a robot [Li and Horowitz 1995] without poles is a simpler strategy but this does not provide the same level of protection for the human operator.

The conclusions drawn from the various effects that may occur in collisions suggest certain mechanical and kinematic design features for human-friendly robots. To reduce occurrences of poles at surface points which are coincident with a joint axis

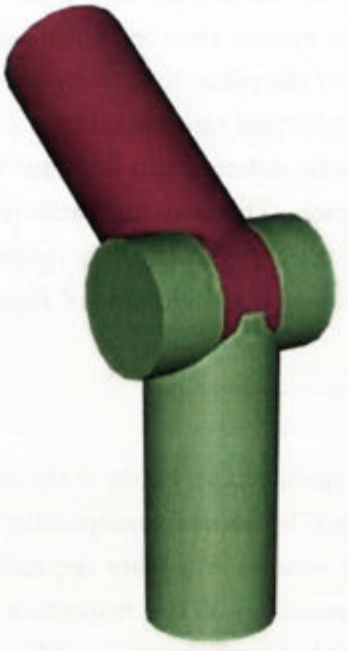


Figure 7.17: Safe link design

the joints should be designed such that the axis intersects only with surfaces belonging the link below the joint axis while the link above the joint forms a circular coverage around the joint axis. This concept is illustrated in Figure 7.17 where the green lower link covers the intersection of the joint axis and the red link has no surface points on the joint axis. The base of the robot should contain at least three non-parallel joints to prevent poles in higher serial links. Some special states $(\Theta, \dot{\Theta})$ in which poles of the impact force still appear must be prevented by the motion planner.

The formulation of the impact potential of the robot defines the safe region in the state space $(\Theta, \dot{\Theta})$ and yields the definition of state dependent control constraints. These control constraints are then transformed into the command torque space of the robot that define the safe region for the command torque verification. The boundaries of the safe region in command torque space are the hyperplanes that correspond to the control points. A real-time safety core must be able to derive safe commands from unsafe commands supplied by an external controller. A clipping algorithm was presented which calculated the closest safe command vector within the safe region for a given unsafe command vector. The number of control points must be restricted to the smallest number possible in order to achieve real-time capability of the algorithm.

A successful simulation of a 2 DOF manipulator with a safety core control demonstrated the feasibility of the approach. With only two control points, one on each link, the robot was restricted to safe motions while the external "controller" produced

constant motor torques which would normally continuously accelerate the robot into states that are considered unsafe. In the second experiment the robot was initially in an unsafe state. The experiment demonstrated how the safety envelope continuously and quickly reduced the impact potential of the robot, until the robot reached a safe region in the state space.

The chapter presented a complete and sound solution to achieve a quantitative safety guarantees for the motions of a robot manipulator for any position controller. The impact potential criteria allows quantitative limits for pre-collision safety. The new scheme considers the kinematic properties of the robot and maximum speed performance is achieved within the safety limits. Implicitly the impact potential control constraints provide a post-collision limitation of the forces exerted onto a person. The successful simulation of a manipulator under the IPC-scheme demonstrates the feasibility of the approach and is a prerequisite for a successful implementation on a real robot.

Chapter 8 presents the application of the proposed control methods on an actual robot system, the Barrett WAM robot manipulator. The implementation of the impact potential control scheme is interfaced with the visual face and eye gaze tracking system to implement an interactive pick-and-place system.

Chapter 8

Impact Potential Control Implementation

The next step after a successful simulation is an implementation of a real system. In robotics it is well known that a successful simulation of an algorithm does not necessarily imply that a successful implementation on a real robot is possible. This chapter consists of two major sections, the first section presents the details of the implementation, the robot, its suitability for human-robot interaction, auxiliary software modules such as external force sensing and position control, the software architecture and its distribution on a network of computers. The second section reports on experiments that demonstrate various human-friendly aspects of the system. Particularly showing that an Impact Potential Control scheme using a safety core can be implemented successfully on a real robot.

Chapter 7 discussed how the kinematic design of a robot and also the mechanical design of the outer shell influence the results and the applicability of the Impact Potential Control (IPC) scheme. In particular the occurrences of poles in the impact force $\widehat{f}_e$ must be examined and considered for the implementation on a particular robot. In this chapter Section 8.1 describes the robot on which the IPC scheme is implemented. The kinematic configuration, the robot's hand and a number of auxiliary hardware safety features and their relevance to the safe development of a human-robot interaction system are described.

Section 8.2 analyses the kinematics of the robot and discovers all possible locations where poles in the impact force $\widehat{f}_e$ could appear. The following discussion on the geometry of the robot's surface and its influence on the appearance of possible poles emphasises how important the mechanical design of a robot is for successful implementation of the IPC scheme. Although the kinematic configuration of the robot allows a large number of poles, the design of the robot's surface eliminates almost all of the poles.

Section 8.3 describes a software module for the detection of external forces acting on the robot. In the area of human-friendly robots a sensing method which allows the

detection of such external forces *anywhere* on all relevant links of the robot which are accessible to the user is required. A unique method is used in this research, it uses the dynamic model of the robot to sense joint torques which are a result of external forces acting on the robot, instead of the usual approach of mounting a force sensor at the wrist.

The software architecture of the complete implementation is presented in Section 8.5. The implementation of the IPC scheme requires considerable computational resources and is therefore distributed over multiple processors which perform the various tasks within the system. The architecture is centred around the architecture of the safety core which is illustrated in Figure 7.1. Software modules with tight real-time constraints are located on the central robot control PC while other modules are located on other machines. The section presents an overview of the interfaces which connect the different software modules.

The second part of this chapter starts with the presentation of two experiments using the basic function of the safety core in Section 8.6. The experiments demonstrate the basic behaviour "gravity compensation" and the influence of the safety envelope on this behaviour in the absence of an external controller. Section 8.7 presents the results of moving the robot in free space under the control of an adapted PID position controller. The experiments show that the PID controller achieves stable goal-seeking behaviour while maintaining the impact potential limits set in the safety envelope. Satisfying this criteria is crucial for the implementation of a human-robot interaction system.

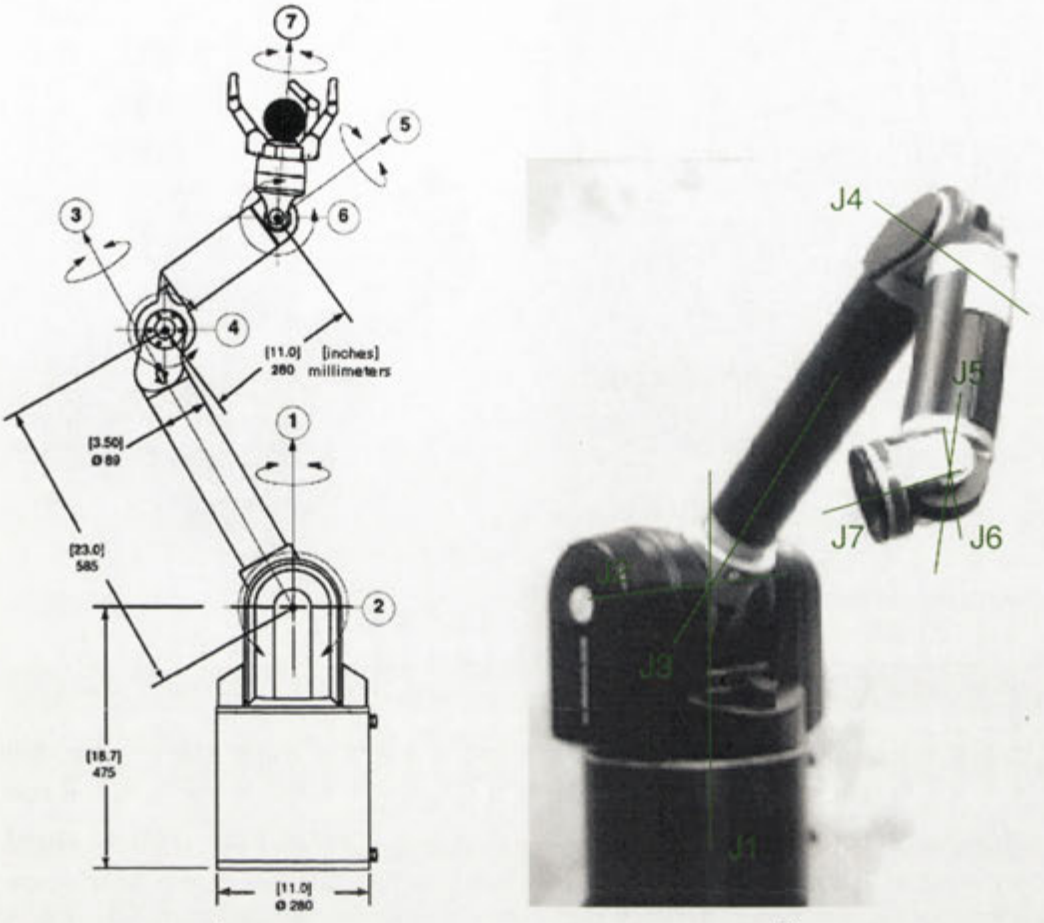
Section 8.8 provides physical evidence that the implemented the IPC scheme actually limits the impact potential of the robot. The robot is propelled into a static plasticine block with different impact potential limits. The deformation results of the plasticine block are analysed. Although this experiment does not provide a quantitative evaluation of the impact potential it does give an indication of the proper function of the safety core.

A summary of the important aspects of the implementation of the IPC scheme is presented in Section 8.9.

8.1 Hardware platform

The hardware platform for the implementation of the IPC scheme is a Barrett Whole Arm Manipulator (WAM) robot, a commercial version of an MIT arm [Townsend and Salisbury 1993]. The reach of the robot is approximately 1 m, it has a 4 DOF and a 7 DOF[1] configuration and all joints are equipped with cable transmissions. The cable drives allow for low friction, therefore, the robot can be easily backdriven by a person. Also, the transmission is almost lossless. This allows the joint torques to

[1]Depending on the configuration of the wrist.



a) b)

Figure 8.1: The WAM robot (Figures courtesy of Barrett Technology)

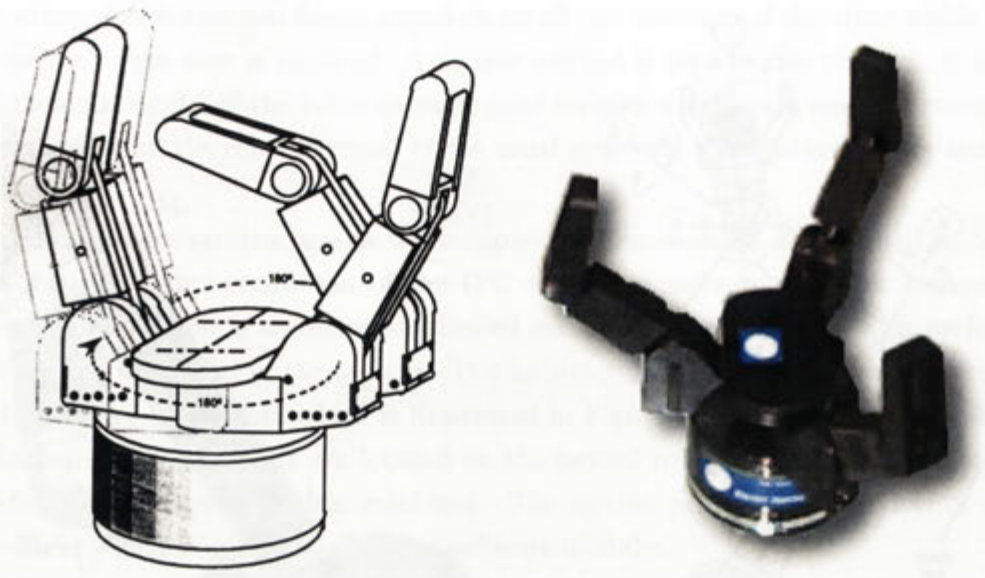
be estimated from the motor torques. The WAM is not equipped with any additional force/torque sensors. Force sensing is performed using motor torques.

8.1.1 The WAM robot

Figure 8.1 shows a technical drawing and a photograph of the WAM without the 4DOF gripper. The system consists of a 3 DOF shoulder, a 1 DOF elbow and a 3 DOF wrist. Both the shoulder and the wrist are kinematically identical. In the reduced 4 DOF configuration of the system the 3 wrist joints are replaced by a rigid connector.

Due to technical problems with the WAM hardware it was not possible to use the articulated wrist in the experiments described in the following sections and chapters. However, the reduced 4DOF configuration was sufficient to demonstrate the efficiency and practicality of the IPC scheme.

The system requires large amplifiers since it uses brushless motors in all seven joints. The advantage of brushless motors is their good torque to weight and size ratio, this allows a light weight design and a compact shape of the robot. The



a) b)

Figure 8.2: 3 fingered Hand of the WAM (courtesy of Barrett Technology)

disadvantage of brushless motors is the significant torque ripple effect. The term torque ripple describes the phenomenon of a changing output torque for a constant control input depending on the orientation of the rotor and the rotation speed. Torque ripple can cause problems in open loop control approaches such as the basic gravity compensation behaviour of the safety core and the detection of contact with obstacles.

Our system incorporated a torque ripple compensation algorithm which uses pre-recorded lookup tables. For every motor shaft position a compensation value was measured in a robot configuration with average motor load. However, this method is insufficient. The correct compensation value depends not only on the motor position, but also on the motor velocity and the desired output torque. A satisfactory torque ripple compensation scheme requires sophisticated motor models and is beyond the scope of this thesis.

The WAM is controlled by a standard 500MHz Pentium III PC with a simple D/A converter board which also provides hardware quadrature encoders for the motor positions. The implementation of the IPC core and the external controller implementation described in Section 8.4 achieve a control rate of 1KHz. The robot control PC and all other computers in the system run the VxWorks[2] operating system.

8.1.2 The WAM hand

The gripper of the WAM robot is a self-contained 3 fingered hand. Each of the 3 fingers has two joints to adapt to the shape of the grasped object. Each of the

[2]VxWorks is a trademark of Wind River Systems `http://www.windriver.com/`.

fingers is actuated with a single motor only and the two joints are connected by a clutch mechanism. When the finger closes both joints move together. When the upper finger makes contact with the object the clutch disconnects and the finger tips keeps closing until it also make contact with the object. This mechanism allows the grasping of objects of various shapes without the requirement to control the configuration of the hand. The simple *close*-command is sufficient for grasping a wide range of shapes.

To make the hand even more adaptable the two outer finger can be rotated around the palm. Objects can be grasped from one side with the fingers side by side, from three sides with the two outer fingers rotated at some angle $0° < \alpha < 180°$ or from two sides with the two outer fingers located opposite of the middle finger (compare Figure 8.2). The hand has therefore 4 degrees of freedom plus the clutch operated extra joints in the fingers.

The system connects to a power supply and a serial port of the WAM control PC. The firmware runs on a 68HC11 controller and the amplifiers are all contained within the unit. Figure 8.2 shows the hand with the outer finger rotated about 30° away from the middle finger.

8.1.3 Hardware safety features

The WAM robot is equipped with a number of hardware safety features. These features are designed to limit the effects of software and hardware failures and therefore differ significantly from the IPC-scheme proposed in the Chapter 7. The IPC-scheme allows a quantitative limitation of the impact force, the hardware safety features are meant to detect exceptional conditions which should not occur during normal operation of the robot. The clipping function of IPC-scheme is designed to be used during normal operation as an efficient control algorithm to move the robot within the safe region of the command torque space. The limits and fault conditions of the hardware safety features are activated only if a fault in the software or hardware occurs, triggering the emergency shutdown of the robot which cuts the power to all actuators.

Joint torque limiter

The brushless joint motors are driven by high power digital amplifiers. The maximum motor torques are far beyond the torques required for human-robot interaction systems. Such a powerful robot poses a safety risk to people. The current output is proportional to the joint torque. To limit the joint torques generated by the robot even in the case of a hardware or a software failure the amplifiers are configured such that the motor current is limited. The current is limited to a value which does not restrict the motions of the robot during the interaction experiments but reduces the potential that a person may be injured in the case of a fault. A threshold limit set

to 10% of the maximum motor torque is used. It should be noted that limiting the joint torque is the only hardware safety feature that does not automatically trigger an emergency shutdown of the robot.

Joint velocity limiter

Using joint torques that have been limited the robot is still capable of reaching dangerous velocities, in particular if gravity supports the motion of the robot. It is therefore necessary in addition to limiting the joint torques to also limit the velocity of each of the joints using a hardware circuit. As opposed to joint torque, excessive joint velocities trigger the emergency shutdown of the robot. A non-terminating limitation of the joint velocities controlled via the command torques would require a joint torque vector filter similar to the architecture used in the IPC-scheme because at this level the robot is torque controlled. However, such a command filter would also require computational resources and would not yield the same safety constraints and would also depend on correct software. It is therefore advantageous to use the IPC-scheme in the control of the system and use a hardware limiter for the joint velocities. This setup combines the high safety constraints of the IPC-scheme with the robustness of a hardware solution. The velocity limit for each joint can be set in the hardware safety circuit individually. Limits set in the order of $100°/s$ have been shown to provide a reasonable level of protection from software and hardware faults.

Hardware watchdog

The control loop of the WAM operates at 1KHz. The regular execution of the control loop is crucial to provide the safety guarantees of the safety core and to control the position and velocity of the robot. If the control program is not executed regularly or stops completely due to a hardware or a software fault no safety guarantees or predictions about the resulting motions of the robot can be made. For example, if the control computer stops, sending constant joint torque commands will most likely result in a constantly accelerated motion until the velocity limiter is activated or the joint limits are reached. Therefore it is advantageous to make an early detection of irregularities in the control loop execution and shut the robot down if necessary. This is done using a watchdog timer which is reset within the control loop and which triggers the emergency shutdown of the robot when the timer expires. The hardware watchdog is located on the A/D conversion board which generates the control signals.

Safety switch

The safety switch enables the user to shutdown the system at any time. The hand-held switch must be continuously depressed to enable the robot. A release of the switch immediately triggers an emergency shutdown. Although the switch has the functional similarity to "dead man" switches in safety-sensitive applications such as

train driving, the safety switch in this research is used as a fast shut-down switch. It has the advantage over conventional emergency switches that it can be triggered by a person quickly and with little conscious effort. This is important in cases when the system may appear to be out of control or performs an unexpected fast motion. Since the switch allows the user to shut down the system at any time it gives the user peace of mind and helps unexperienced users to feel comfortable with the robot.

8.1.4 Suitability for human-robot interaction

Overall the WAM system is well suited for human-machine interaction experiments and was specifically chosen for its unique design characteristics. It provides the following key elements which are an important prerequisite for the implementation of safe physical human-robot interaction experiments:

- A dextrous base with three joints which allows limiting the impact force using the IPC-scheme.
- Easily backdriveable joints which allow a person to physically influence the motion and configuration of the robot.
- A closed surface without sharp edges which improves the passive safety of the robot and allows the selection of higher values for the impact force limit π_{max} without compromising the safety of the user.
- A light weight design due to extensive use of light weight materials such as aluminium and magnesium. The small mass of the manipulator allows fast motions while maintaining low potential impact forces.
- Hardware safety features which cover all aspects of hardware or software failure and allow automatic shut down of the robot.
- Open access torque control of the robot which is essential for the implementation of a human-friendly control using the IPC-scheme.

These characteristics of the WAM robot manipulator were essential for the successful implementation of a human-friendly robot. Our experiments demonstrate how the specific design of the WAM robot supports the realisation of the concept of quantitative safety in situations where humans and robots share the same work space and physically interact with each other.

8.2 Control point selection

Chapter 7 presented algorithms to validate and, if necessary, clip command torque vectors by projecting them into the safe region of the command torque space. Since

the projection algorithm is computationally expensive only a small subset $\widetilde{\mathbf{P}}$ of control points can be considered if real-time performance is to be realised. Hence, the selection of control points to approximate the true impact potential of the robot is important.

Besides the approximation aspect the avoidance of poles in $\widehat{f_e}$ is crucial to the successful implementation of the IPC-scheme.The manipulator's kinematics and geometry can be designed such that poles do not occur irrespective of the selection of the control points. However, in the case of joints and links close to the base of the robot this is not always possible due to the insufficient number of joints further along the kinematic chain.

The selection criteria for control points can work against each other since a pole in $\widehat{f_e}$ anywhere on the manipulator will cause a pole in the impact potential π. Therefore the selection of control points should reflect this fact. However, in a specific application practical considerations based on the location of the poles and the accessibility of these locations to a user can justify the selection of the control points which disregards inaccessible poles. In the WAM robot all poles occur either in inaccessible places or only when the robot is in a particular position and the joint velocity vector has particular values. *Occurrences* of *accessible* poles are eliminated using a motion planning heuristic which avoids the critical constellations.

8.2.1 Poles of $\widehat{f_e}$ on the WAM

Firstly the appearance of poles on the surface of the WAM is examined. The kinematic configuration and surface geometry shown in Figures 8.3-8.5 is similar to the example of an almost pole-free base presented in Chapter 7. The key difference is that the surface of link 3 above J_3 does not cover the complete base. This difference is responsible for the appearance of additional poles in the impact force. The following sections discuss where and under what conditions poles appear on the surfaces of the links above each of the four joints.

Links 1 and 2

The WAM robot has poles in $\widehat{f_e}$ for surface points on the links 1 and 2 are shown in Figure 8.1. Obviously all points on link 1 on the axis of joint 1 are poles for $\dot{\theta}_1 \neq 0$ which is the case for every serial manipulator. On link 2 again all points on J_2 are poles for $\dot{\theta}_1 = 0 \neq \dot{\theta}_2$. In addition, the points on link 2 that are on the axis of joint 1 are poles for $\dot{\theta}_1 \neq 0 = \dot{\theta}_2$ since at least two further joints are required to compensate for the motion vectors in the cylindrical motion field. Figure 8.3 shows the lines along which the potential poles are highlighted in red.

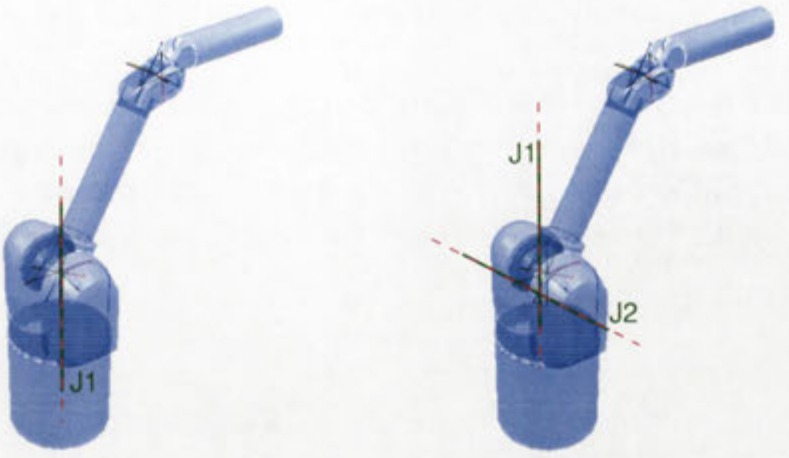


Figure 8.3: Potential poles on the first two links

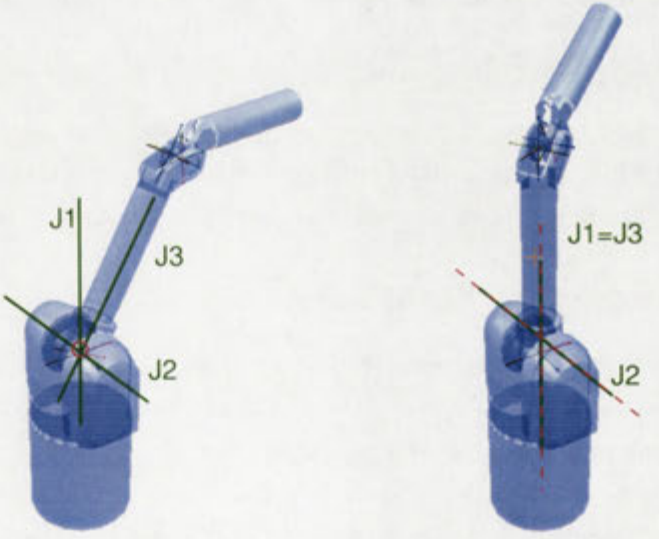


Figure 8.4: Potential poles on link 3

Link 3

If $\theta_2 \neq n\pi$, $n \in \mathbb{N}$ the first three joints are in general orientations with respect to each other. In this case only the intersection of all three axes is a potential pole. All other points on link 3 can not be poles of $\widehat{f}_e$. Figure 8.4a) shows a small red circle at the intersection point.

Additional poles appear if θ_2 is in the kinematically singular position $n\pi$ where the axes of J_1 and J_3 are coincident. In this case points on link 3 which are on either axes 1 or 3 are poles for $\dot{\theta}_1 + \dot{\theta}_3 \neq 0 = \dot{\theta}_2$ and points on joint axis 2 are poles for $\dot{\theta}_1 = \dot{\theta}_3 = 0 \neq \dot{\theta}_2$. Figure 8.4b) shows the lines along which the potential poles are in red.

Link 4

In a general configuration with $\theta_2 \neq n\pi$, $n \in \mathbb{N}$ no poles appear on the first three joint axes except for their intersection where the first three columns $\mathbf{j}_1..\mathbf{j}_3$ of the Jacobian equal 0. The pole appears for all non-zero joint velocity vectors with $\dot{\theta}_4 = 0$. One other point which is located on the joint axis J_4 at the intersection with the plane spawned by J_1 and J_3 a pole occurs for $\dot{\theta}_1 = \dot{\theta}_2 = \dot{\theta}_3 = 0 \neq \dot{\theta}_4$ as shown in Figure 8.5a). Here the columns $\mathbf{j}_1$ and $\mathbf{j}_3$ are aligned and approaching from the appropriate direction, the Cartesian velocity $\mathbf{v}$ becomes orthogonal to $\mathbf{j}_k$, $k = 1..3$.

Again, in the kinematic singular configuration $\theta_2 = n_1\pi$, $n_1 \in \mathbb{N}$ more poles appear as indicated in Figure 8.5b). If $\theta_3 \neq n\pi$ and joint axes 2 and 4 are not parallel only the following points are potential poles: The intersection point of $J_1 = J_3$ with J_2 for $\dot{\theta}_1 \neq 0 = \dot{\theta}_2 = \dot{\theta}_3 = \dot{\theta}_4$ and $\dot{\theta}_3 \neq 0 = \dot{\theta}_1 = \dot{\theta}_2 = \dot{\theta}_4$ and the points on J_2 and J_4 at the shortest connection between J_2 and J_4 for $\dot{\theta}_2 \neq 0 = \dot{\theta}_1 = \dot{\theta}_3 = \dot{\theta}_4$ and $\dot{\theta}_4 \neq 0 = \dot{\theta}_1 = \dot{\theta}_2 = \dot{\theta}_3$ respectively.

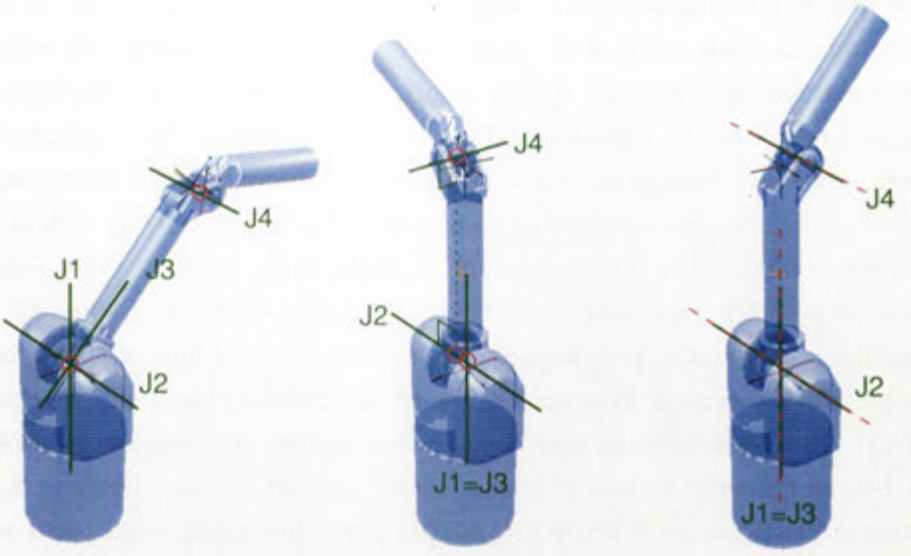
If additionally $\theta_3 = n_2\pi$, $n_2 \in \mathbb{N}$ the joint axes 2 and 4 are parallel as shown in Figure 8.5c). In this case all points on link 4 which are coincident with any joint axis i, $1 \leq i \leq 4$ are poles for $\dot{\theta}_i \neq 0$ and the three other joints are still $\dot{\theta}_{k \neq i} = 0$.

8.2.2 Geometry of the WAM robot

The previous section discussed the various poles in $\widehat{f}_e$ for the kinematics of the WAM robot. The mechanical design of the WAM and the resulting surface geometry eliminate these poles in almost all configurations and the surface of the manipulator is free of poles in $\widehat{f}_e$.

On link 1 none of the possible poles are part of the surface. The base plate covers the intersection of joint axis 1 with the surface and the intersection on the top side is covered by link 2. Therefore all poles on link 1 are covered by other parts and therefore no poles need to be considered.

On link 2 poles can appear on both joint axes 1 and 2. The surface points which are coincident with joint 2 are covered by link 1. However, on the top side of the link



a) b) c)

Figure 8.5: Potential poles on link 4

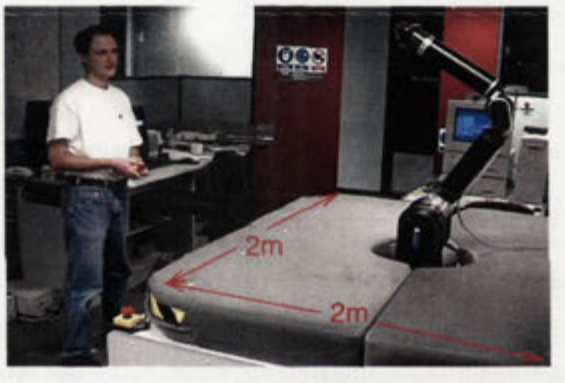


Figure 8.6: Experimental setup of the WAM robot

a pole can appear where axis 1 intersects the surface of link 2 if $\dot{\theta}_1 \neq 0$ and $\dot{\theta}_2 = 0$. This pole is disregarded since the base of the robot is inaccessible to the user in the setup of the WAM. The experimental setup consists of a 2m×2m table top with the robot mounted in the centre as shown in Figure 8.6. Hence, people interacting with the robot do not come in contact with the base of the robot. Due to this limitation in the accessibility it is sufficient to only consider the links 3 and 4 in the IPC-scheme.

Figure 8.4 shows the poles on link 3. In the general configuration only the intersection of the joint axes 1 and 3 is a potential pole. This location is inaccessible since it is covered completely by link 2. In the singular configuration $\theta_2 = n\pi$ all points along the joint axes are potential poles. Again, the intersections of the poles are covered by other links. The two intersections of joint axis 2 are within link 2 and so is the lower intersection of axis 3 with link 3. The upper intersection of axis 3 with link 3 is covered by link 4. Hence, the surface geometry eliminates all possible poles for link 3.

The individual possible poles on link 4 in the Figures 8.5a) and b) are all embedded within links. Only a few of the poles in the singular configuration $\theta_2 = n_1\pi$ and $\theta_3 = n_2\pi$ actually appear on the surface of link 4. The intersections of the joint axes 2 and 4 are covered by the links 1 and 3 respectively. However, the intersection with the joint axes 1 and 3 which are coincident with the surface of link 4 are exposed. As described in the previous section this pole appears for $\dot{\theta}_1 + \dot{\theta}_3 \neq 0 \wedge \dot{\theta}_2 = \dot{\theta}_4 = 0$. For a particular configuration of the robot only a single surface point can be a pole. The locations of possible poles form two lines on two sides of link 4. Figure 8.7 shows the two lines of potential pole locations along link 4 as red lines and the location of the pole in the current location as a small red circle. This is the only pole on the WAM robot that must be considered in the motion planning for the WAM. A simple heuristic for the motion planning would be never to stop joint 2 at $\theta_2 = n\pi$ but other heuristics exist with more sophisticated rules which yield fewer restrictions. This configuration is a kinematic singularity and should be avoided by a path planning algorithm that is independent of the IPC-scheme.

The following plots illustrate these properties of the WAM robot. The plots show the distribution of the impact force $\widehat{f}_e$ on the surface of the cylindrical links 3 and 4 of the WAM robot. In the plots the 4 DOF configuration of the WAM robot is modelled as two cylinders with the three shoulder joints below the first link and one elbow joint in between the tubes as illustrated in Figure 8.7. The mass matrix is derived from inertia tensors generated from a CAD model of the WAM which also contains the material properties such as the density. The two base links between joint 1 and joint 3 are only represented in the mass matrix and are not modelled in the surface geometry.

Figure 8.8 shows the plots of the impact force $\widehat{f}_e$ in the singular configuration where the pole on link 4 appears. The impact force on link 3 in Figure 8.8a) is uniform since link 3 rotates about its axis of symmetry. The variations close to joint 4 are caused by the offset of the joint 4 from the plane spawned by joint 2 and 3. The compliance effect provided by joint 3 depends on location of the impact point on link 4. Figure 8.8b) shows the plot of the impact force $\widehat{f}_e$ on link 4. The pole clearly visible in the centre of the graph corresponds to the surface point where the joint axes 1 and 3 intersect the cylinder.

Figure 8.10 shows the plots of the impact force $\widehat{f}_e$ for the general configuration $\Theta = [1.2, -0.5, 0.7, 0.3]$ illustrated in Figure 8.9 where all joints are in motion $\dot{\Theta} = [0.3, 1.2, -0.6, -0.2]$. In Figure 8.10a) the poles inside the mechanism at the intersection of joint axis 2 with link 3 are visible, but as mentioned before, can be disregarded because these points are not surface points. The plot of the impact force $\widehat{f}_e$ in Figure 8.10b) shows the non-uniformity and unpredictability of the distribution of the impact force on link 3. Although it is impossible to find a small number of surface points which are likely to cover the maximum impact forces in all configurations, the plot shows that the variation between the surface points is moderate and

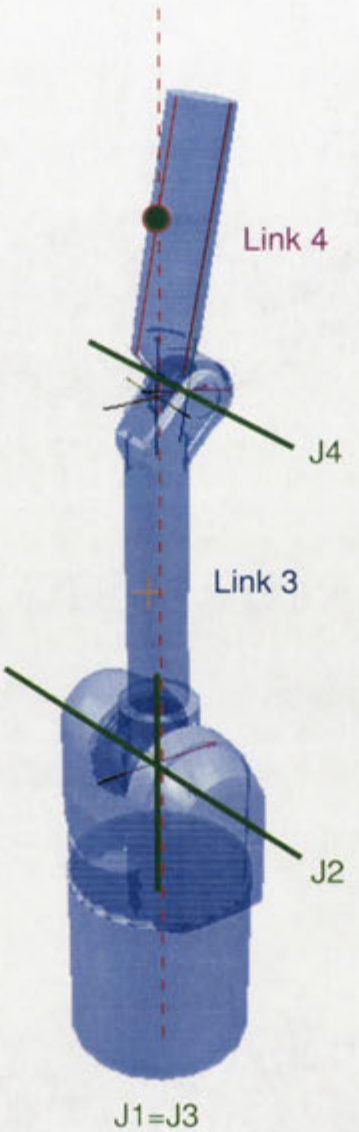
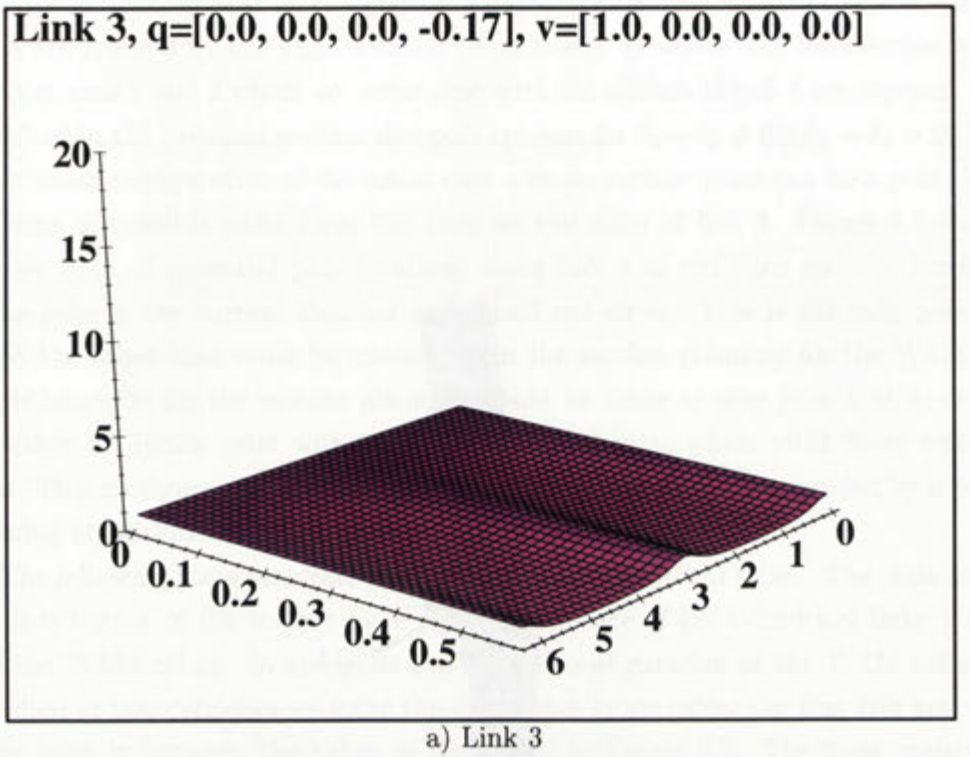
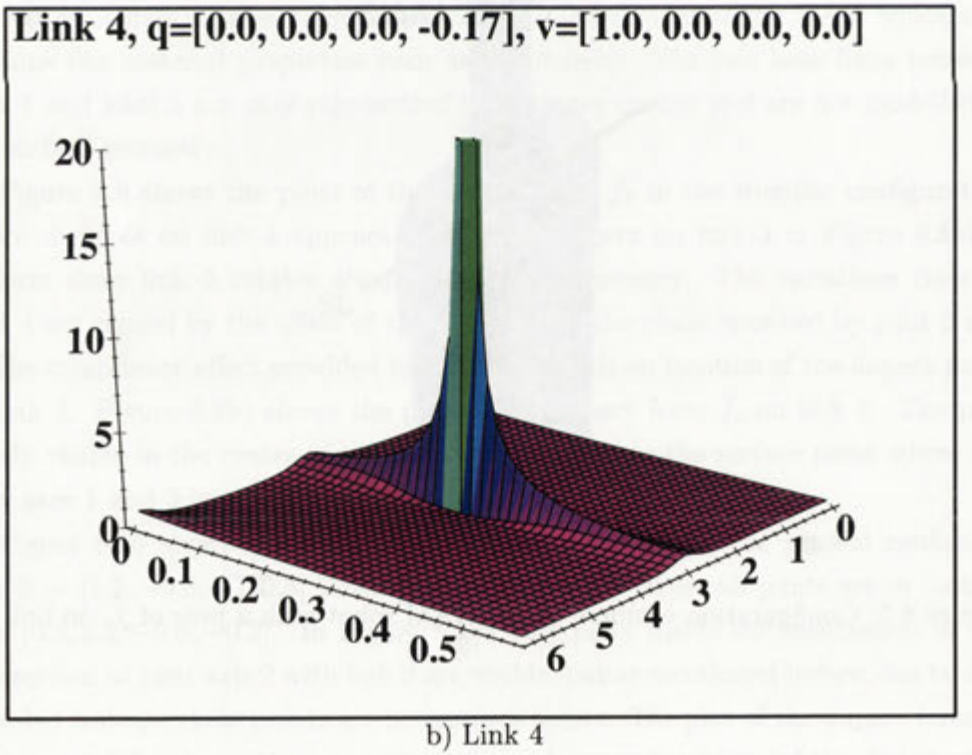


Figure 8.7: Configuration example of the WAM robot with a pole of $\widehat{f}_e$ on link 4



a) Link 3



b) Link 4

Figure 8.8: The pole in $\hat{f}_e$ on link 4

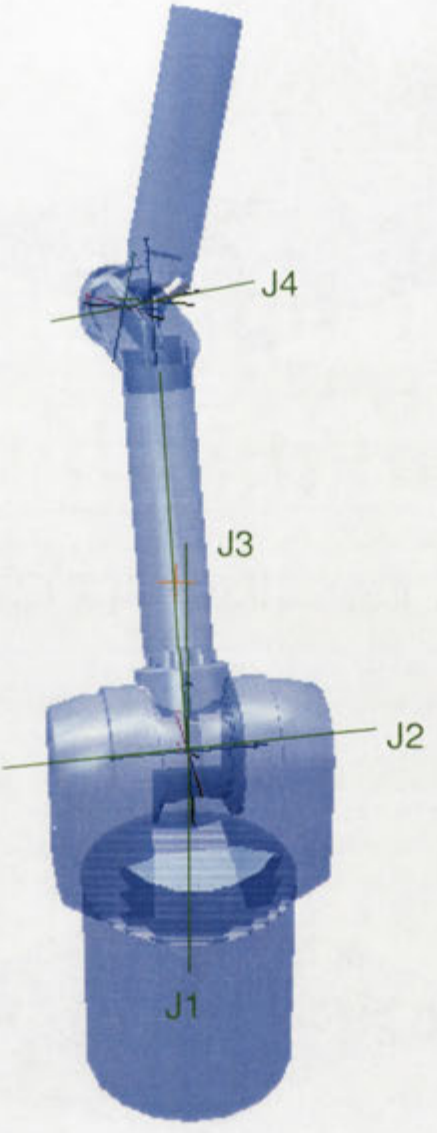
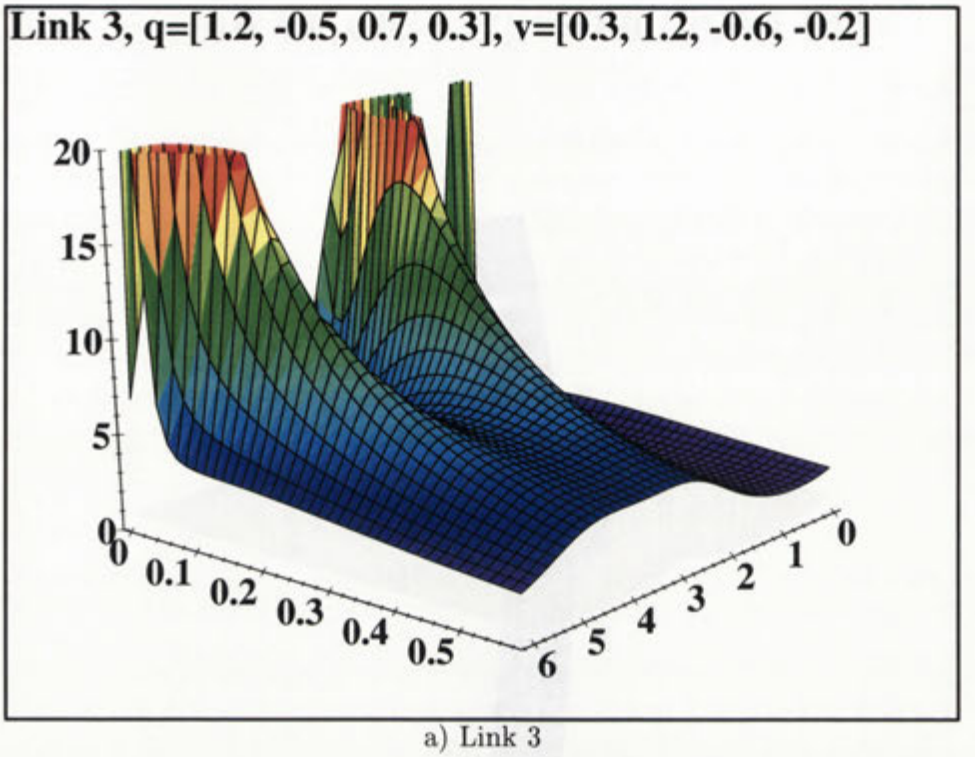
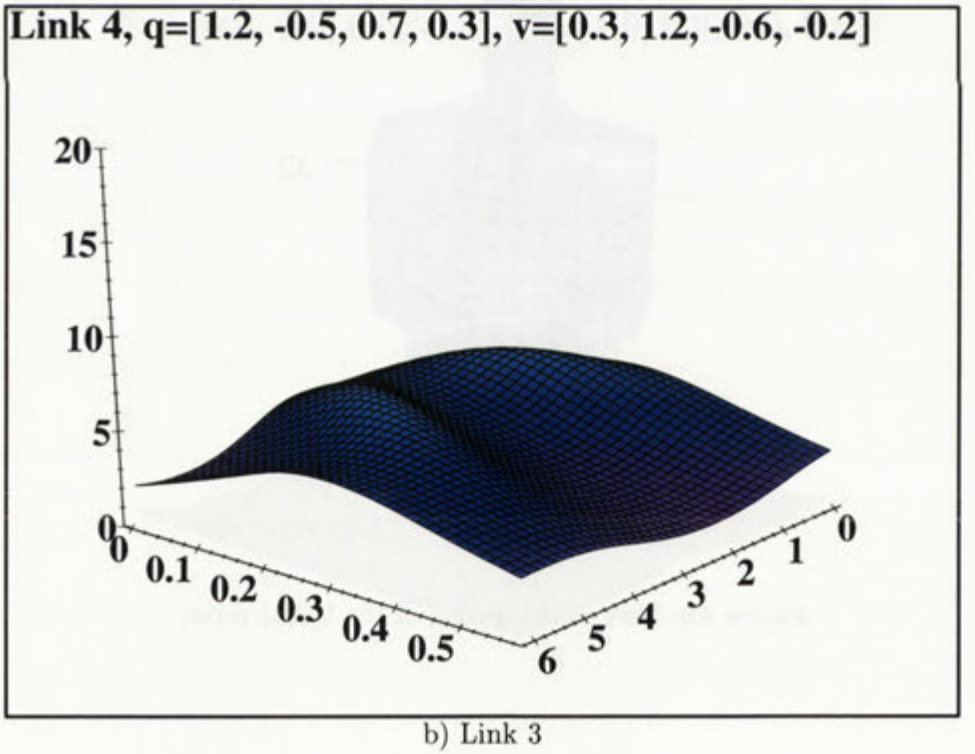


Figure 8.9: Test configuration of the WAM robot



a) Link 3



b) Link 3

Figure 8.10: Impact force $\widehat{f}_e$ on link 3 and 4 in a general configuration

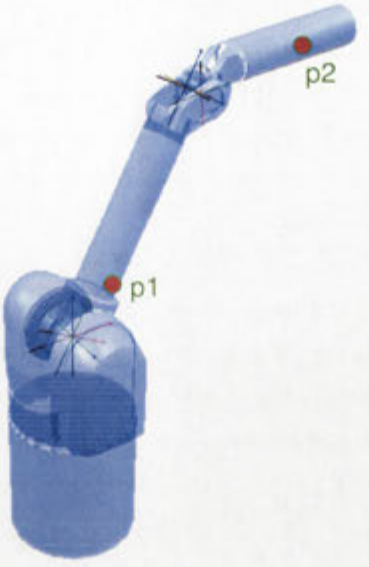


Figure 8.11: Control points on the WAM robot

a reasonable approximation can be achieved even with a small number of control points.

8.2.3 Implementation

To achieve a cycle frequency of 1KHz for the safety envelope calculations only a small set $\widetilde{\mathbf{P}}$ can be considered. In fact only two control points are feasible without compromising the control frequency on the given hardware. Any number of control points could be used if sufficient computational resources are available. Although this may seem to be a poor approximation of the original claim to consider the complete surface of the robot, the simulations in Chapter 7 have shown that even a single control point on each link provides effective limits of the robot's motion.

The control point on link 3 must be close to the base since $\widehat{f}_e$ increases nearer toward the base. The lower 10cm can be disregarded due to the kinematic configuration and geometry of the base. Hence, the control point can be 10cm from joint axis 2. Preferences for the position of the control point on the link can not be made without making assumptions about the most likely configuration of the robot. Therefore the first control point chosen for the implementation of the IPC-scheme is $\mathbf{p}_1 = (0.045, 0, -0.48)$ in frame 3 coordinates[3] as illustrated in Figure 8.11 (see also Appendix D, Figure D.1).

For the control point on link 4 there is no preference unless assumptions about the configurations and motions of the robot are made. However, the points along the two lines of possible poles have the greatest possibility of approaching a pole if the motion planner fails to avoid this problem. During physical interaction between the

[3]It should be noted here that frame 3 is located at the ellbow end of link 3, and hence, the z-coordinate is negative.

robot and a person a singular configuration could easily be caused by the user who enforces a particular motion. As the control point approaches a pole the safety core generates joint torques which are suitable to limit the potential impact force. The torque vectors increase in magnitude quickly as the control point approaches a pole, which may result in sudden and forceful motions of the manipulator. The possibility of this happening can be ruled out by not selecting any control points along the two lines of possible poles. Since these points are not advantageous compared to other points on the surface of link 4, the control point for the implementation of the IPC-scheme on the WAM is selected in between the two ends of link 4 and on the side of this link such that the control point is in the centre between the lines of possible poles measured around the circumference. The frame 4 coordinate vector of the control point on link 4 is $\mathbf{p}_2 = (-0.047, 0.25, 0.045)$. The resulting set $\widetilde{\mathbf{P}} = \{\mathbf{p}_1, \mathbf{p}_2\}$ of control points for the WAM is used for the implementation of the IPC-scheme.

This implementation is meant to be a proof of concept that the IPC-scheme is applicable to real robots. It is also shown that the implementation on the WAM robot actually does allow the safe interaction between a user and the robot. Due to the real-time constraints the set of control points $\widetilde{\mathbf{P}}$ must be reduced to two control points. The remainder of the chapter shows that even such a small set is sufficient to achieve a behaviour which is in accordance to the SAFETY REQUIREMENTS stated in Chapter 6. The human-robot interaction experiments presented in the following chapter support this claim.

8.3 Contact sensing without sensors

The importance of the detection of physical contact for human-machine interaction has been discussed in Chapter 1 and Chapter 2. In particular in collaborative object manipulation and passing it is necessary for the robot to "feel" the grip of the person.

The unique characteristics of the cable transmissions of the robot which have almost no friction nor backlash allows the detection of physical contact[4] of the robot without contact sensors. Contact detection is based on the comparison of the dynamic model of the manipulator moving in free space and the actual motion of the robot. Therefore the only sensors necessary for this new contact detection scheme are the position encoders of the motors. As opposed to F/T-sensors mounted at the wrist of the robot this novel technique allows sensing of physical contact with the environment not only at the robot hand but at any surface point of the robot.

[4]The term "contact" is used here in the sense of "a force created by a physical contact". Similar to most other techniques such as F/T wrist sensors and artificial skin, the proposed method requires that in order to sense the contact a force must be exerted onto the robot.

8.3.1 Methodology

A dynamic model of the robot has already been used in the formulation of the safety core. The basic equations of the model were presented in Chapter 7. The gravitational term is disregarded in the following sections. The joint torque vector τ is known because it is the output of the safety core. The joint positions, velocities and accelerations $\Theta, \dot{\Theta}$ and $\ddot{\Theta}$ are known from the encoder data from the motors of the robot. The only unknown is the vector of joint torques $J_{\mathbf{p}}(\Theta)^t\mathbf{f}$ generated due to external contact(s[5]). This torque discrepancy vector τ' is an indicator of contacts with the environment.

$$\tau' = J_{\mathbf{p}}(\Theta)^t\mathbf{f} = M(\Theta)\ddot{\Theta} + N(\Theta, \dot{\Theta}) - \tau \tag{8.1}$$

Similar to the situation with a F/T wrist sensor the surface point $\mathbf{p}$ can not be derived from the given information, in particular if multiple simultaneous contacts are possible. Therefore the contact point $\mathbf{p}$ and the contact force vector $\mathbf{f}$ remain unknown. Moreover, a contact with the environment, in particular with a link close to the base of the robot and in singular configurations of the robot, may not result in a non-zero torque discrepancy vector. If $\mathbf{f}$ is in the core of $J_{\mathbf{p}}(\Theta)^t$ then $\tau' = \mathbf{0}$ and the contact can not be detected with the torque discrepancy method. However, a non-zero torque discrepancy vector is always the result of a contact with the environment.

This situation does not pose a problem for the contact sensing applications in the context of a human-friendly robot system, where the main concern is the detection of persons pulling or pushing the manipulator. In general people have an idea which joint of the robot they want to push or pull and therefore apply suitable forces $\mathbf{f}$ which are detectable by the system.

It should be noted that the contact detection system is not a safety function. Collisions of the robot with its environment are made safe by the safety envelope. The detection of external contacts is rather a communication channel which allows a natural interaction between the robot and the operator.

8.3.2 Implementation

A contact detection system has been implemented on the WAM robot. The dynamic simulation module of the safety core is reused in the contact detection system to derive the torque discrepancy vector τ'. The calculations are performed on the robot control PC which has tight real-time constraints for robot control. Since the dynamic simulation requires considerable computational resources it is impossible to perform the contact detection at the control frequency of 1KHz. Instead, the cycle time of the contact detection calculations is adapted to the external interface of the WAM

[5]If the robot is in contact with more than one surface point multiple terms are required to model this effect.

software. This interface provides a network interface which updates the system state at 10Hz for high level controllers which plan trajectories and goal configurations of the robot. The calculation of $\dot{\Theta}$ and $\ddot{\Theta}$ is based on the changes of the joint angles Θ in 0.1s intervals. The effective joint torque vectors are averaged over this interval.

The resulting torque discrepancy vectors are effectively low pass filtered. However, the torque ripple effect which is not modelled in the dynamic simulation can cause large phantom torque discrepancy vectors. Chapter 9 shows plots of the magnitude of the torque discrepancy vector τ'. On average the magnitude of τ' is approximately 5Nm for free space motion. The phantom torques limit the sensitivity of the system to torques that significantly exceed the noise floor that is created by the torque ripple. Nevertheless the system is sufficiently sensitive for the physical human-robot interaction experiments described in Chapter 9.

8.4 External position control

The safety core acts as a filter at the torque-level interface between the motion controller and the actuators. This allows the use of the IPC scheme with virtually any position, velocity or compliance control method. Independent of the control laws of the external controller the safety core ensures safe operation of the robot according to the combination of π_{max} and t_c. However, the external controller must be able to cope with the saturation effect that the safety core exhibits which might require adaptation of the implementation of a particular control method. The saturation effect can be reduced by feeding back the current impact potential π of the robot and the distance d of the current torque vector to the constraint boundaries to the motion control algorithm. This information could be used as an indicator of how close the safety core is to saturation and the velocity along the trajectory can be adjusted accordingly.

8.4.1 Example: PID control

To demonstrate the effectiveness of this methodology an adapted PID controller has been implemented on the WAM which allows the robot to move into a goal configuration. The controller moves the robot along a straight line in configuration space towards a given goal configuration while the speed of the motion is controlled by the restrictions of the safety envelope.

It is possible to implement such a controller by setting the command positions of the PID controller to the given goal configuration and let the safety core limit the velocity of the robot. However, the saturation effects of the safety envelope can severely limit the performance of such a system. If the goal configuration is far from the current position, the desired torques of the PID controllers may exceed the safe torque limits. Generally the closest safe torque vector to such a command torque

vector has a significantly different orientation. The trajectory taken by the robot may deviate significantly from the desired straight line in configuration space and large overshoots can be the result. To reduce this undesired behaviour the saturation effect of the safety core must be taken into account.

The basic idea is to shift the command position of the PID controller towards the goal configuration. The shifting velocity depends on the current impact potential π of the robot. At the beginning of a motion the shifting velocity is increased steadily. As soon as the distance d of the generated command torques to the constraint boundaries approaches zero the shifting velocity is steadily maintained. Should unsafe torque vectors be generated which result in $d < 0$ the shifting velocity is decreased. Algorithm 8.1 shows the calculations of the control loop that generate the command positions for the PID position controller.

Algorithm 8.1 Command Position Generation

$v_{scale} = 0$
loop
  $\mathbf{e} = \Theta_g - \Theta_c$
  $\mathbf{b} = -\dot{\Theta}\frac{\|\mathbf{e}\|\cdot\pi}{\|\dot{\Theta}\|\cdot\pi_{max}}$
  $v_{scale} += \min(1, \max(0, s_d \cdot d))$
  $\mathbf{p} = v_{scale}(\mathbf{e} + \mathbf{b})$
  $\Theta_p = \Theta_c + \mathbf{p}$
  Set Θ_p as command positions
end loop

The first steps in the control loop calculate the vectors along which the command position is shifted. The goal configuration engagement vector $\mathbf{e}$ is set to the distance between the current configuration Θ_c and the goal configuration Θ_g. The braking motion vector $\mathbf{b}$ is calculated in the next step. The orientation of the vector is opposite to the current joint velocity vector $\dot{\Theta}$ and it's magnitude is scaled to the ratio between the maximum impact potential π and the maximum safe impact force π_{max} times the magnitude of $\mathbf{e}$. Therefore the braking vector is of magnitude 0 if $\pi = 0$ and equal to the magnitude of the engagement vector $\mathbf{e}$ if $\pi = \pi_{max}$. The braking vector has two functions, firstly to dampen motions which are not aligned with the desired trajectory and secondly to counterbalance the acceleration of the command position shifting as the robot approaches the constraint boundaries. Note that if $\dot{\Theta}$ is aligned with $\mathbf{e}$ and $\pi = \pi_{max}$ then $\mathbf{b} = -\mathbf{e}$ which makes the command position identical to the current position. In the next step the shift speed scaling value v_{scale} is calculated. The scalar d is the smallest distance of the current robot state to the safety constraints. If the robot state is within the safe area d is positive, otherwise negative. The coefficient s_d is an appropriate positive scaling value[6]. Therefore, if the robot is within the safe area, v_{scale} increases proportional to the remaining

[6]This value is determined empirically and is set to $3 \cdot 10^{-6}$ in the implementation on the WAM. Higher values result in a more rapid increase of the shifting velocity.

distance to the safety constraints. This causes the robot to accelerate further. If the robot is in an unsafe state, v_{scale} decreases, causing the robot to slow down. The velocity scale v_{scale} is always limited to the interval $[0, 1]$ to ensure that the direction of the shift can not be inverted and the command configuration Θ_p can not be set beyond the goal configuration Θ_g. Finally, the command position shift vector $\mathbf{p}$ is calculated and the new command position Θ_p is determined relative to the current configuration Θ_c.

Figure 8.12 illustrates the mechanism of the adapted PID controller in three 2 DOF examples. Figure 8.12a) shows a general configuration of the parameters. The goal configuration Θ_g in the top right corner is connected with the current robot configuration Θ_c on the left with the green engagement vector $\mathbf{e}$. The current joint velocity vector $\dot{\Theta}$ in red points in a different direction and is counteracted by the turquoise braking vector $\mathbf{b}$. The blue shifting vector $\mathbf{p}$ is the sum of $\mathbf{e}$ and $\mathbf{b}$ and is scaled by v_{scale} such that the resulting torque vectors are close to the constraint boundaries. The direction of $\mathbf{p}$ is determined to align the joint velocity vector $\dot{\Theta}$ with $\mathbf{e}$ and approach the goal configuration in a straight line from the current configuration. It should be noted that if during motion external forces push the robot off the straight path to Θ_g the controller will not try to return to the original path. Rather it suppresses the non-goal-directed motion and then continues to approach Θ_g along a straight path from the current position. This behaviour is deliberately designed to suit physical interaction between the robot and a person. For example a person could decide during the motion to alter the path of the robot by pushing the robot to prevent a collision with an object.

Figure 8.12b) shows how the braking vector can counteract the engagement of the goal configuration. In most cases the velocity $\dot{\Theta}$ is aligned with $\mathbf{e}$ and $\mathbf{b}$ simply counterbalances $\mathbf{e}$ according to the current impact potential π. As long as $\pi < \pi_{max}$ the resulting shifting vector $\mathbf{p}$ points towards the goal configuration Θ_g. Figure 8.12c) shows the case when the manipulator leaves the safe region in state space and $\pi > \pi_{max}$. In this case $|\mathbf{b}| > |\mathbf{e}|$ and therefore $\mathbf{p}$ is directed away from Θ_g. The PID controller actively supports the reduction of the velocity of the robot which is enforced by the safety core to ensure the state $(\Theta, \dot{\Theta})$ returns to the safe region.

8.5 Software architecture

The software architecture of the human-robot interaction system is illustrated in Figure 8.13. The system is distributed over three computers, the vision PC executes the application server and performs the image processing, the robot control PC executes the safety core, and the Impact Force Constraint (IPC) server PC calculates the control constraints for the safety envelope.

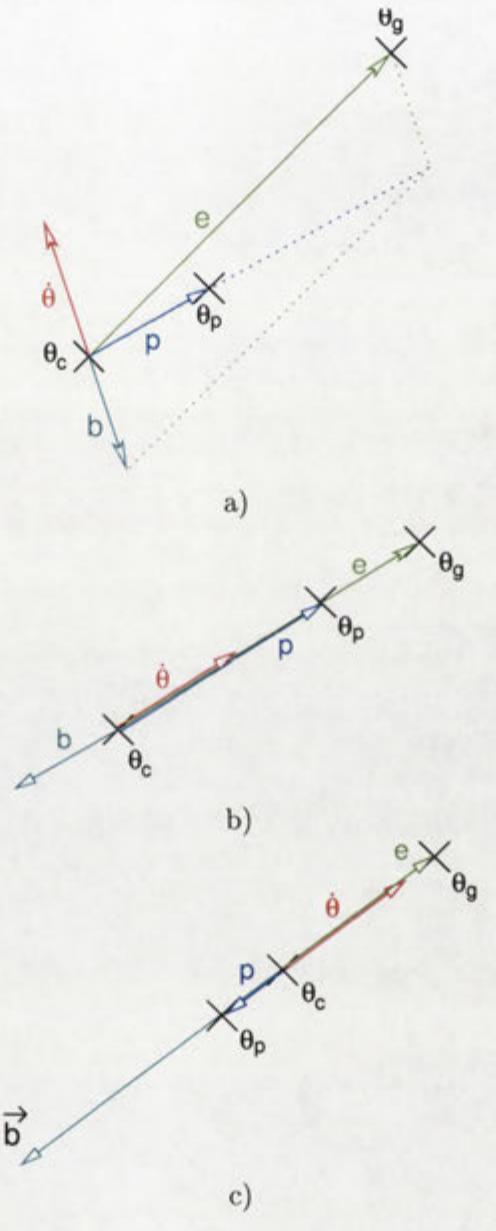


Figure 8.12: Example configuration of the PID controller

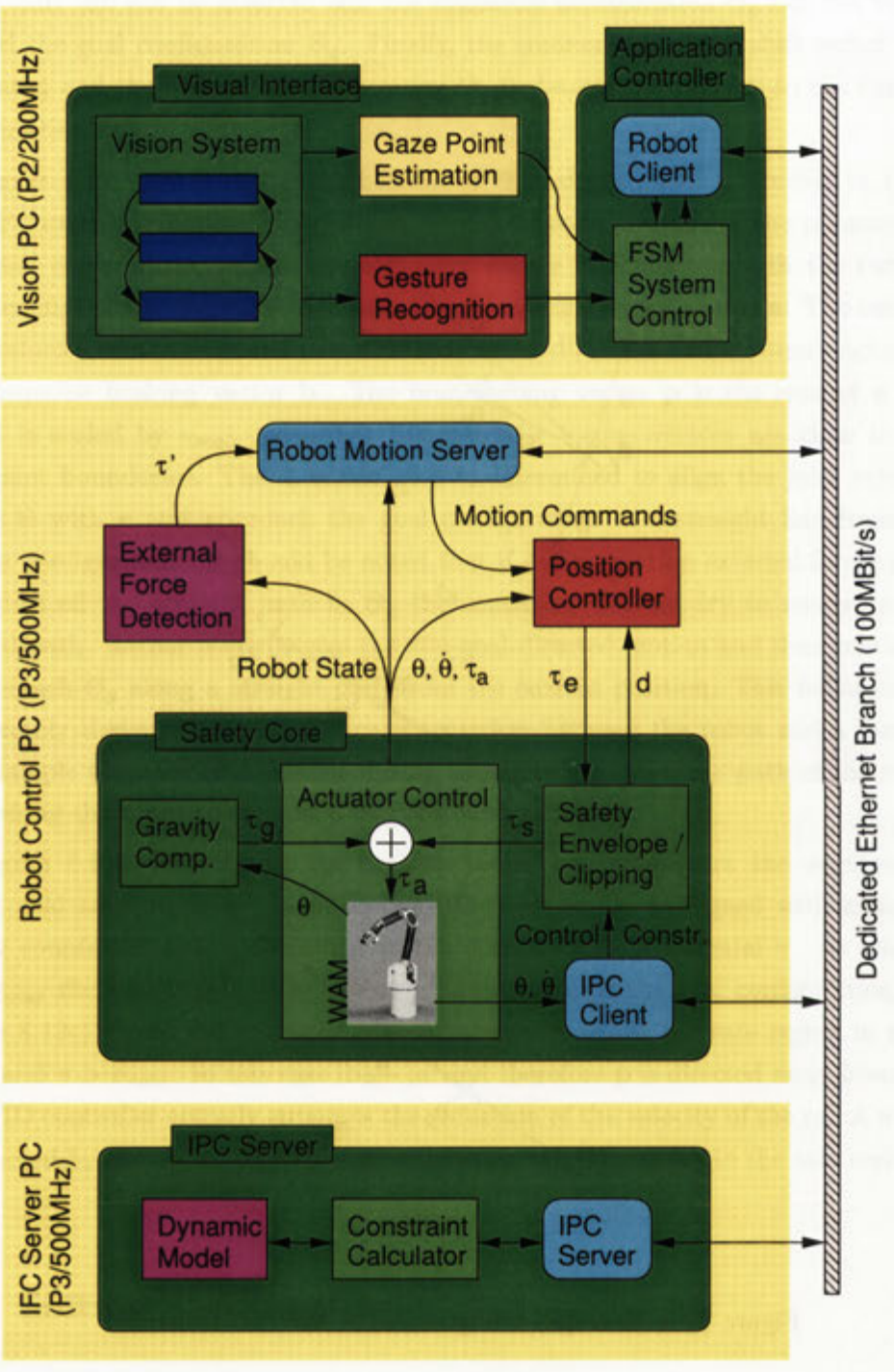


Figure 8.13: Software architecture of the WAM control scheme.

8.5.1 Visual interface and application controller

The visual interface described in Chapters 3 and 5 is implemented on the vision PC. This computer also runs the application server which processes the information from both the visual interface such as the gaze point and gestures and from the robot such as the current configuration and the measured external forces and derives motion commands for the robot. The selection of robot commands is performed by a finite state machine (FSM). The details of the FSM and the application controller are described in Chapter 9.

8.5.2 Safety core implementation

The robot control PC runs the safety core and the external position controller, the two modules with hard real-time constraints. All modules within the safety core execute at a cycle frequency of 1KHz. The external position controller and the input part of the external force detection module are synchronised with the safety core. The main control loop is closed between the position controller, the safety envelope and the robot itself. The position controller produces command torque vectors τ_e which are forwarded to the safety envelope. The compliance of the torque vectors to the control constraints is verified, and if necessary the torque vector is clipped. The resulting safe torque vector τ_s is forwarded to the robot's actuators and added to the Zero-G torque vector τ_g. For feedback the robot's position is forwarded to all modules that require this information, including the position controller. The safety envelope also provides the distance measurement d between the command vector τ_e and the boundary of the safe region to adjust the velocity of the motion generated by the position controller.

The external force detection module samples the robot parameter sets $(\Theta, \dot{\Theta}, \tau_s)$ at the control frequency of 1KHz. These parameter sets are accumulated until the next robot state information packet is sent to the application controller by the robot network interface. The robot network interface sends information packets at 10Hz to keep the application controller updated on the robots position and velocity, external forces acting on the robot, and the state of the robot with respect to the safe region in the state space.

8.5.3 Robot motion server interface

The robot motion server on the robot control PC offers a network interface to the safety core, the external force detector and the position controller. Any application controller with a robot motion client can connect to the server and can begin controlling the robot.

The server allows an arbitrary number of clients to connect and receive updates of the robot's configuration, current impact potential and external forces. These

information packets are provided by the server for all clients at a frequency of 10Hz via an UDP[7] connection. Only one client at a time is allowed to take control of the robot. In such cases the commanding client streams command packets to the server via an UDP connection. Since the connection is unreliable each packet contains the command position for all joints of the robot. The server receives the command packets and forwards the command position to the position controller after a sanity check which filters out obvious transmission errors such as impossible joint angles. Since the PID controller shifts the command position based on the impact potential the client does not need to provide any path planning. Instead, it simply provides the goal configuration and the robot moves in a straight line in configuration space at the highest possible velocity. The control client is also allowed to disable the position controller to bring the robot into Zero-G which is desirable for some physical interaction applications where the user needs to manipulate the robot easily.

The state information packets provided by the server contain the joint positions and velocities, the joint torques, the discrepancy joint torque vector τ' from the external force detection module and the impact potential and distance d of the command torque vector from the boundary of the control constraints. Due to the lower update frequency some of the parameters such as the joint position and velocity are always current values while the joint torques, the external force and the distance d are averaged over the time between consecutive state information packets.

8.5.4 IPC server

The update of the constraints is asynchronous to the control cycle. The IPC constraints are generated by the server shown in Figure 8.13. The calculation is initiated through the IPC network interface which sends the robot state $(\Theta\,\dot{\Theta})$ via a dedicated Ethernet connection to the IPC server. The IPC server calculates the constraint vector $\mathbf{c}_p$ and the limitation vector $\mathbf{l}_p$ for each control point using the formulas described in Chapter 7. The IPC server also derives the set of all intersections of dimension $n-2$ to 0 which follow from the hyperplanes defined by the $\mathbf{c}_p$ and $\mathbf{l}_p$. Details of the intersection of n-dimensional subspaces are presented in Appendix F. The control constraints and the affine subspaces of dimension $n-1$ to 0 are forwarded via the IPC network interface to the safety envelope. As soon as the IPC network interface receives an update of the control constraints, a new request is issued to the IPC server. This loop runs asynchronous at a frequency of approximately 200Hz. The IPC network interface waits on a semaphore from the IPC server which times out after 0.02s. If the semaphore times out the safety core assumes a network or software fault has occurred and orders the emergency shutdown of the robot.

[7]User Datagram Protocol, see for example `http://www.freesoft.org/CIE/RFC/768/index.htm`

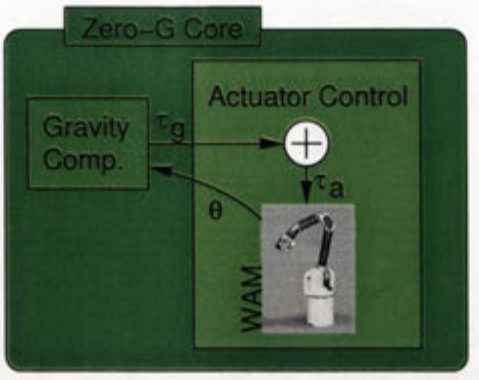


Figure 8.14: Zero-G control structure

8.5.5 Network communication

The implemented system uses parallel processing over multiple processors connected by Ethernet links. Since Ethernet does not guarantee the transmission time of a packet, the use of dedicated connections via a switched hub provide reasonable performance. The time critical connection between the robot controller and the IPC server is supervised by the safety core and the robot is shut down if real-time constraints are violated by network delays.

8.6 Passive motion experiments

Even without an external position controller a number of experiments can be conducted which demonstrate the operation of the two main modules within the safety core. Specifically, the gravity compensation module and the safety envelope. These behaviours of both modules are visible more clearly when demonstrated in isolation.

8.6.1 Zero-G mode interaction

The basic behaviour of the robot compensates for the gravitational forces acting on the robot. This passive behaviour is achieved by a feed-forward controller which calculates the torque vector τ_g required to balance the gravitational forces. Figure 8.14 shows the simple structure of the reduced safety core in this setup. The system allows a person to manipulate the robot's configuration as if it were weightless.

Figure 8.15 shows a person interacting with the robot in Zero-G mode. The robot can be moved with little effort and remains in any configuration when it is released. When pushed the robot keeps moving until the kinetic energy is consumed by the low friction of the mechanism. The robot can easily be pushed from one side of the table to the other side where it is stopped by the cushion on the table. Although the gravity compensation requires considerable torques, people perceive the robot as being completely passive. The motions of the robot which result from the manipulation are completely predictable to people. This passivity and predictability

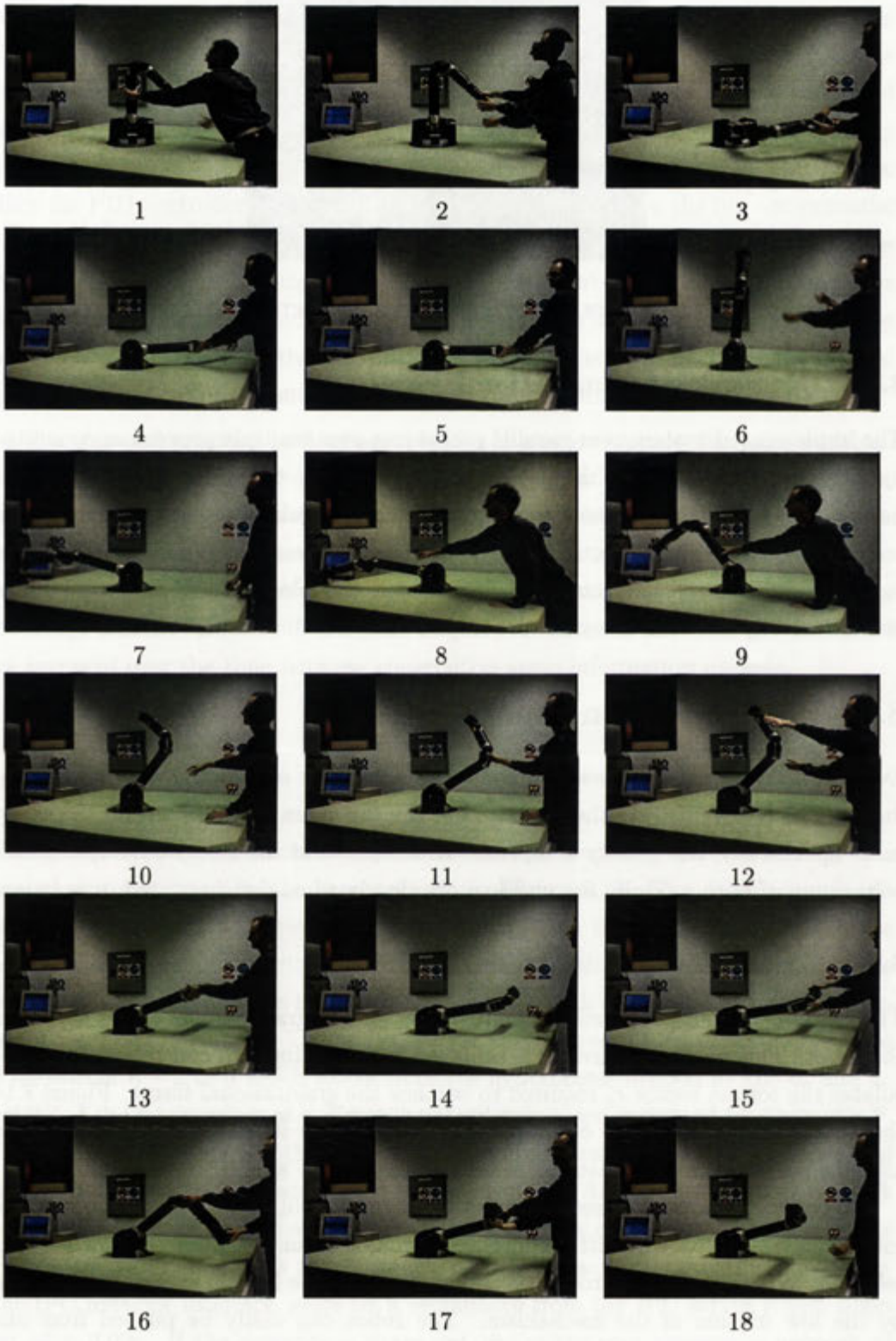


Figure 8.15: Image sequence of a Zero-G interaction experiment

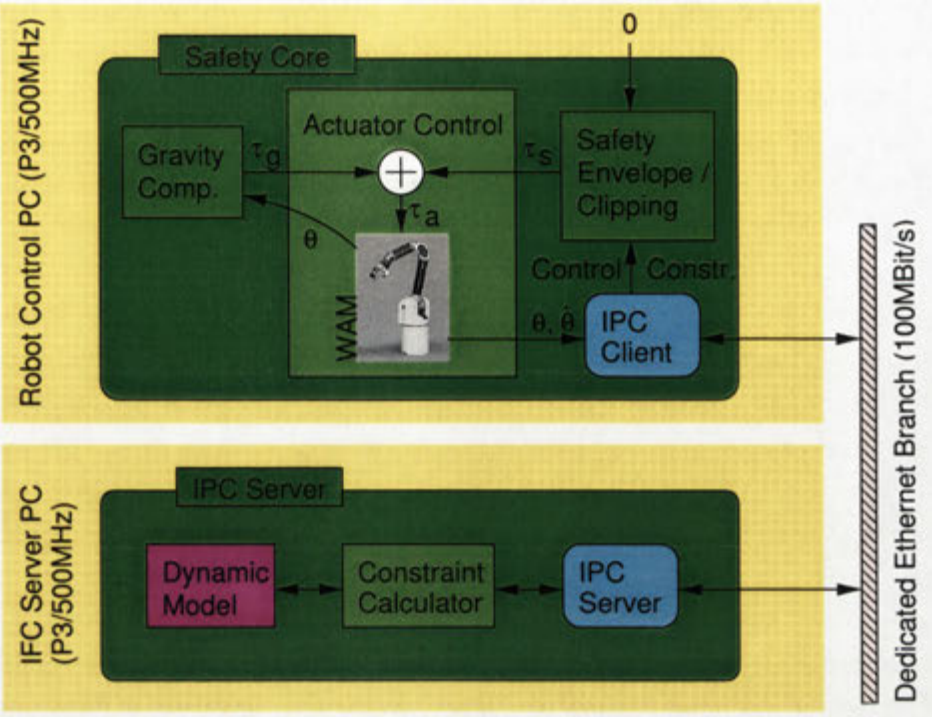


Figure 8.16: Control structure for the motion damping experiment

make the robot appear non-threatening to users. This emphasises the requirement that the users of a human-robot interaction system must feel they are in control of the system at all times and are able to predict the robot's actions to a large extent. The CD-ROM attached to this thesis contains a video sequence of this experiment. Refer to Appendix E for more details.

8.6.2 Safety envelope motion damping

The dampening effect of the safety envelope on the motions of the robot are visualised by the following experiment. The robot in Zero-G mode is moved to an upright position with a weight attached to the hand. The robot is manually held still and then released. The extra weight pulls the robots hand down until the weight rests on the table. This experiment is conducted firstly with only gravity compensation as shown in Figure 8.14 and secondly with gravity compensation *and* the safety core enabled as shown in Figure 8.16.

The second setup consists of a fully functional safety core and the IPC server to provide the required control constraints. The command torque input to the safety envelope is set to 0 which basically places the robot in Zero-G mode. However, as soon as the robot leaves the safe region of the state space, the zero-torque is no longer safe. The safety envelope generates non-zero torque vectors which dampen the motion of the robot. This dampening of motions which exceed the safety limit π_{max} is visible in the experiment with the extra weight. Initially the robot accelerates quickly but the

motion is dampened when the robot leaves the safe region. It should be noted here that the formulation of the control constraints does allow external forces to accelerate the robot in excess of the rate defined by the time constant t_c. The time constant t_c only restricts the acceleration of the robot as a result of command torques τ_e which would result in motions that give an unsafe increase of the impact potential. The time constant determines magnitude of the dampening effect of the safety envelope.

Figure 8.17 shows an image sequence of the experiment conducted in Zero-G mode *without* the safety envelope. The images are sampled at 5Hz. In the first image the robot is released and begins to fall towards the table. The arm accelerates quickly causing an overshoot when the weight hits the cushion on the table top.

Figure 8.18 shows an image sequence of the same experiment conducted *with* the safety envelope. The safety envelope is configured with $\pi_{max} = 1\text{N}$ which is a tight restriction that allows only slow motions of the arm within the safe region. The time constant for $\pi > \pi_{max}$ is set to $t_c = 0.01$ which ensures the robot quickly returns to the safe region. The rate of decrease of impact potential must be 100 times the excess of π over π_{max} in the free motion situation. With this configuration the equilibrium point where the damping torques of the safety envelope balance the extra weight is not far from the border of the safe region.

The arm is released in a similar configuration in both experiments. The manipulator accelerates in a similar way to the first experiment, but the motion of the arm is quickly dampened by the torques generated by the safety envelope. As a result the velocity of the arm is limited and the fall to the table top takes 1.5s instead of 0.8s (the first experiment). Due to the dampened motion of the arm there is almost no overshoot and the arm comes to rest close to the configuration in which the arm initially touched the table.

8.7 Motion control experiment

The PID controller described in Section 8.4 allows the system to control the position of the robot while the safety core ensures that the impact potential of the robot does not exceed the preset limit π_{max}. These characteristics of the system are demonstrated in a number of motion experiments.

The software setup for the experiments consists of the full robot control server with the safety core and the IPC server. Instead of a complex application controller a robot client is used which steps through a series of configurations in a predefined order. Whenever the robot reaches a goal configuration the next goal configuration is sent to the robot.

In the first experiment the robot is moved from its rest configuration from one side of the table to a configuration of the other side of the table and then back to the rest configuration at the start, as shown in Figure 8.19. For this motion only the joints 2 and 4 need to be moved. The same ego motion experiment is performed

Figure 8.17: Image sequence of the experiment in Zero-G mode

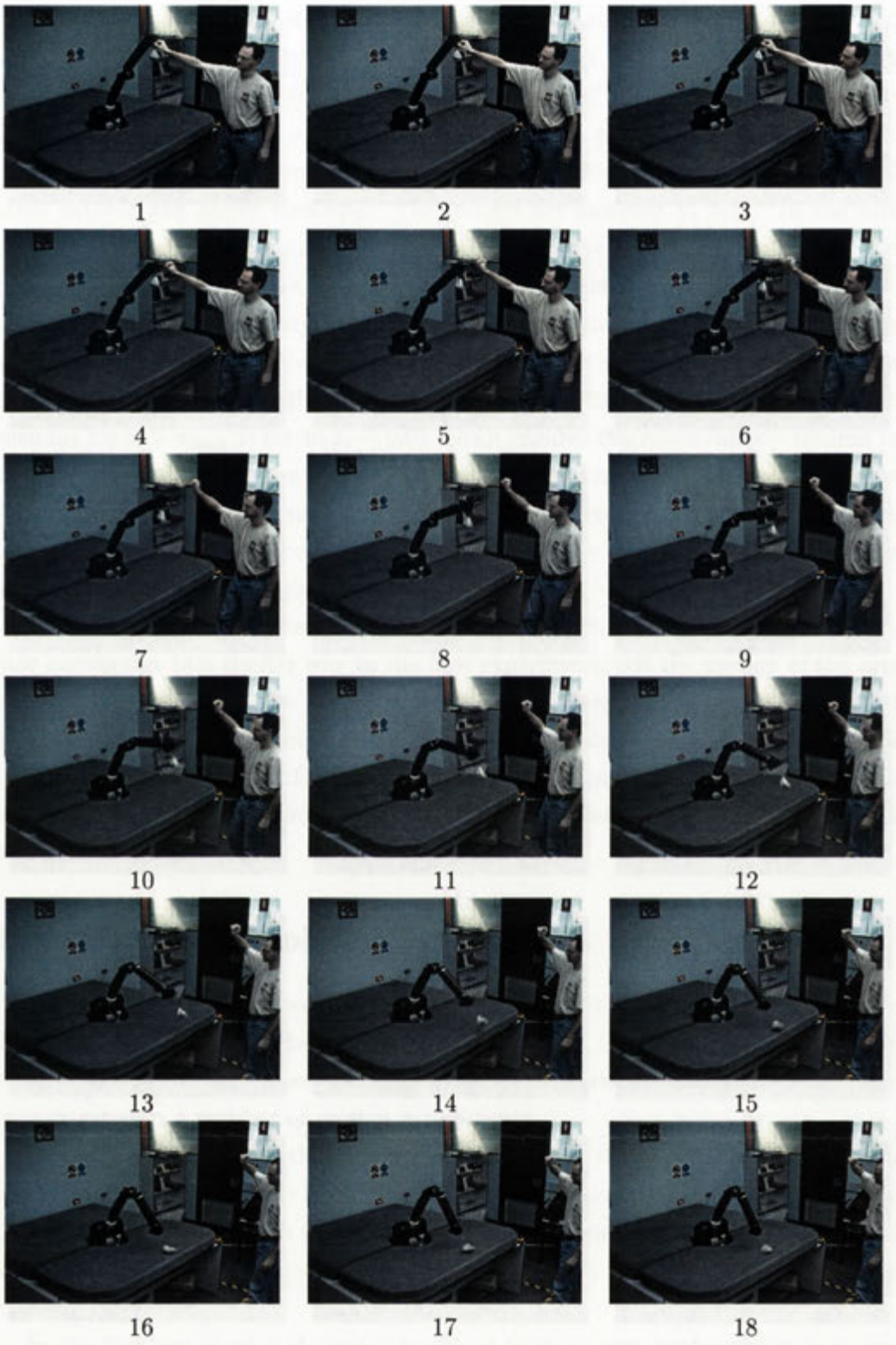


Figure 8.18: Image sequence of the experiment with safety envelope

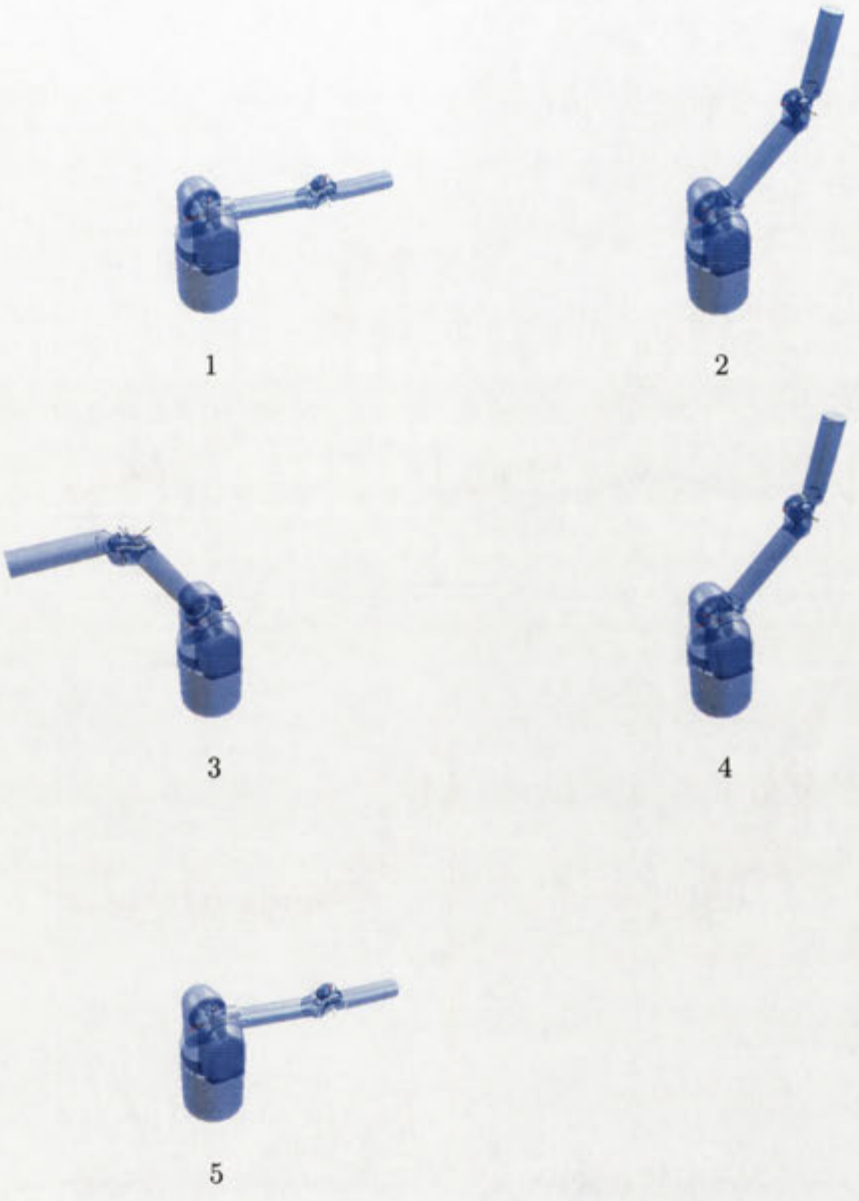


Figure 8.19: Motion sequence 1

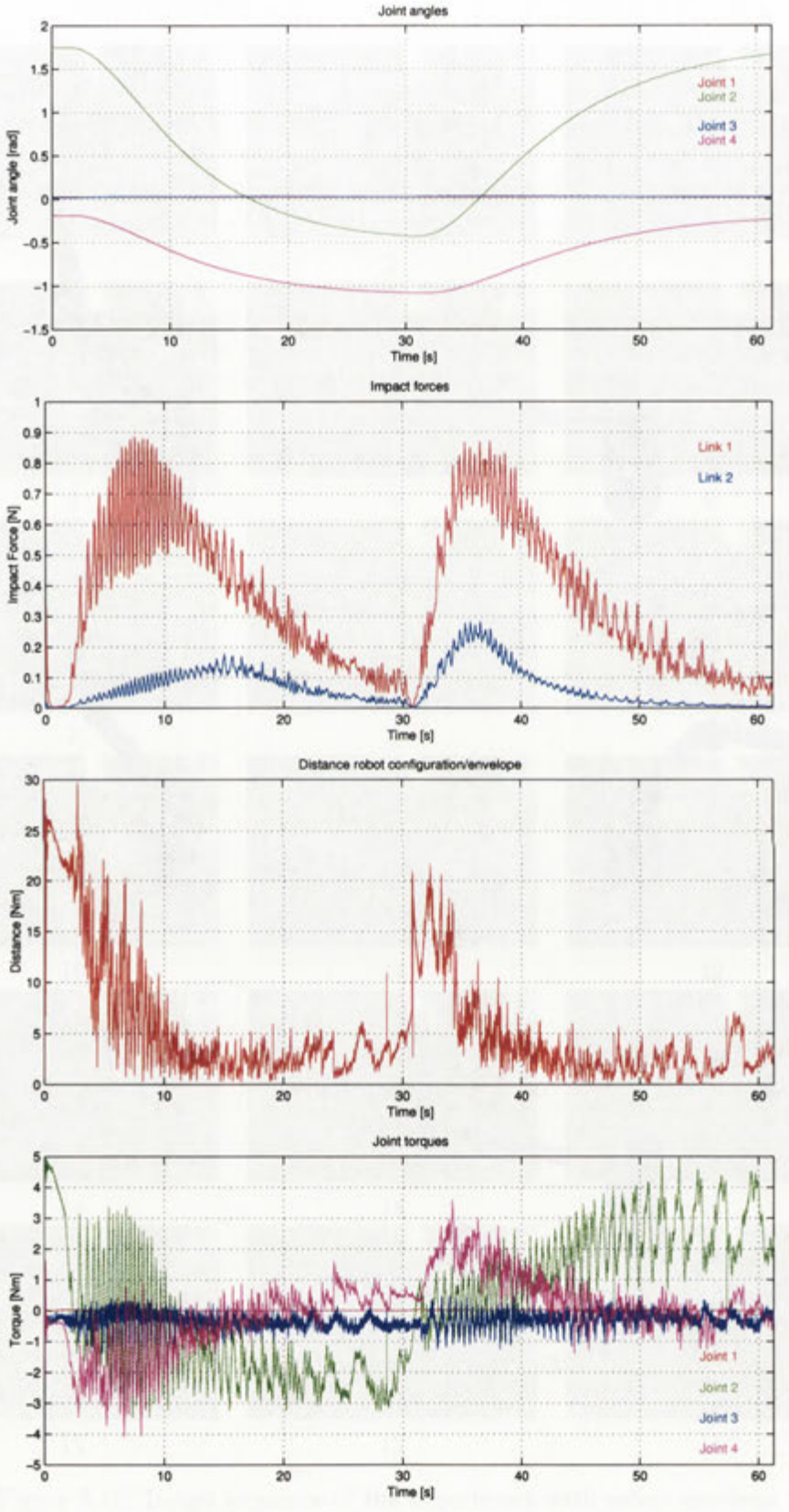


Figure 8.20: Safe robot motion using PID control and a 1N safety envelope

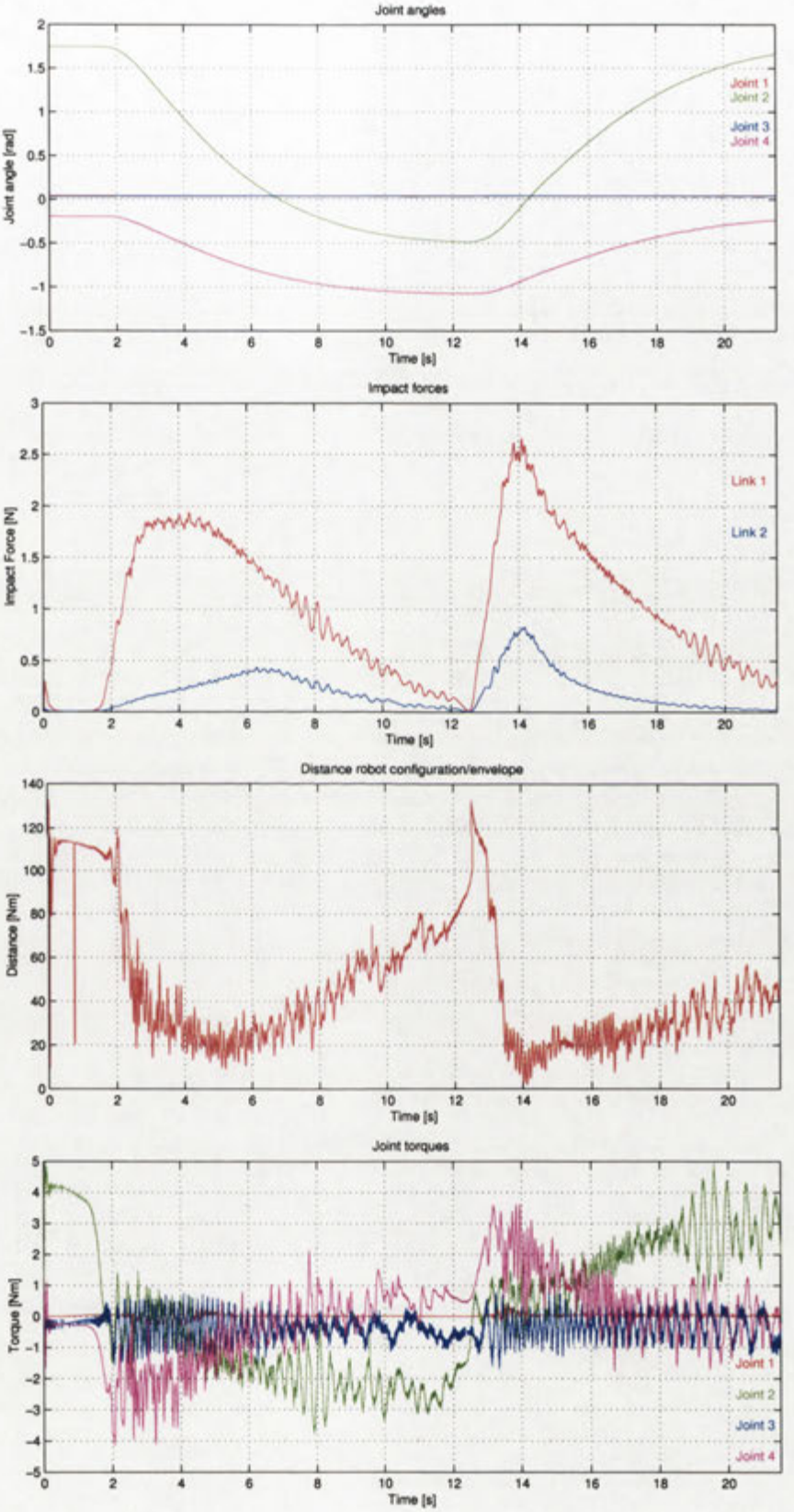


Figure 8.21: Safe robot motion using PID control and a 3N safety envelope

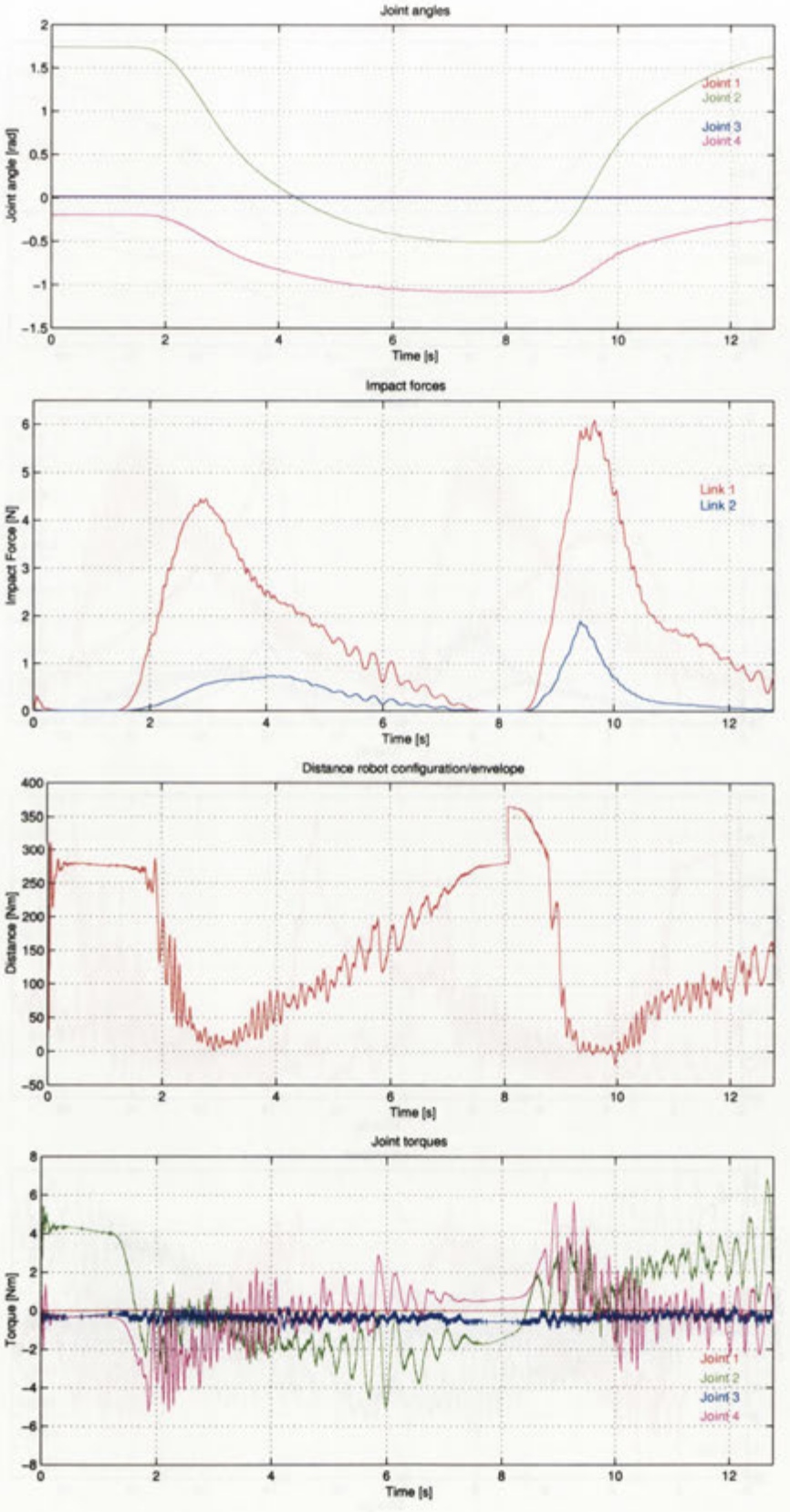


Figure 8.22: Safe robot motion using PID control and a 5N safety envelope

with three different configurations of the safety core. The settings for the maximum impact potential and the time constant are (1N, 0.01), (3N, 0.015) and (5N, 0.018). The range in the maximum impact potential allows the robot to move from quite slowly for 1N to reasonably fast for 5N. The time constants are chosen appropriately for the respective maximum impact potential as a compromise between predictability and efficiency.

Figure 8.20 shows the trace plots of all the important system parameters for the 1N experiment. The duration of the experiment was about 60s. The top plot shows the joint angles for all four joints. It shows that the PID controller generates smooth motions. The second plot shows the potential impact force of the two control points. The potential impact force of the control point is as expected much higher for a trajectory in which primarily joint 2 is in motion and the WAM is almost fully extended. The potential impact force increases quickly as the WAM accelerates both for the forward and backward motions but trails off quickly as it reaches the limit of 1N. The limitations of the simple PID controller are apparent. Although the controller keeps the joint torque close to the boundary of the safe region in torque space as can be seen in the third plot, the velocity and therefore the potential impact force slowly decrease as the robot moves towards the goal configuration. To retain a higher potential impact force and therefore achieve a better efficiency during the approach phase, a tighter coupling of the controller with the distance d of the joint torque vectors to the boundary is necessary. In the case of the PID controller a rather conservative approach is required to obtain stable control. A controller which can generate torque vectors on the boundary of the safe region would provide higher efficiency. The last plot shows the torques for the individual joints. The pattern of the joint torques for joint 2 and 4 is generally as expected, except for the oscillations. These oscillations are *not* caused by an instability of the controller but by the torque ripple of the brushless motors. This is why the frequency of the oscillations is proportional to the joint velocity of the corresponding joint. Considering the strong torque ripple which is not modelled in the safety core the motion of the robot is smooth. Also the robot successfully stayed within the limits of the safety envelope at all times.

Figures 8.21 and 8.22 show plots of the same experiment using higher impact potential limits of 3N and 5N respectively. The robot is allowed to move much faster and completes the same motion path in 21s and 13s respectively compared with 60s in the first experiment. Due to the higher speed of the robot variations in friction and stiction effects are much smaller and as a result the jitter in the potential impact force and the distance to the boundary of the safe region is much smaller. However, the ability of the PID controller to operate on the boundary of the safe region degrades. The plots of the distance of the joint torques to the boundary show the distance between the joint torque vectors and the boundary steadily increases. This is a result of the conservative tuning of the controller which places more emphasis on

stable operation rather than high efficiency.

The results of a similar set of experiments for a more complex trajectory illustrated in Figure 8.23 involving all four joints are presented in Figures 8.24, 8.25 and 8.26. Again in all cases the controller achieves smooth motions. Instances occur when the robot leaves the safe region for a short period of time as a result of the torque ripple. For example in Figure 8.25 it can be seen that at 33s the impact potential exceeded the limit by 25%. However, the safety core promptly and successfully returns the manipulator to the safe region.

The final part of the trajectory differs from the rest of the trajectory in that the control point on link 4 almost has the same potential impact force as the control point on link 3. The PID controller successfully adapts the speed of the motion so that the robot gets close to the boundary of the safe region. This results in a considerably higher angular velocity.

It is also noteworthy that in the case of the more complex trajectory the PID controller shows a better performance with respect to the distance of the generated torque vectors to the boundary of the safe region in joint torque space. In all three configurations of the safety core the controller generates command torques which are continuously close to the boundary. Even for the quicker motions the simple PID controller achieves a good performance.

8.8 Impact Experiment

The experiments in the previous section show how the robot can be controlled within the limits of the safety envelope. However, the values in the plots of the potential impact force are calculated from the joint velocities and the dynamic model of the robot. The experiments do not provide a ground truth of the actual potential impact force of the manipulator. The actual potential impact force of the robot can be verified only with experiments in which the robot collides with stationary obstacles.

In such a collision experiment a number of practical issues must be considered. The body of the robot is made of aluminium tubing, which is sufficiently strong enough to serve as a structural part of the robot, but the tubes are vulnerable to damage when colliding with a rigid object such as a load cell. Any kind of padding would falsify the measurement since the impact would not be instantaneous as assumed in the impact model.

For this reason a qualitative experiment was preferred. In this experiment the robot accelerates to the predefined maximum impact potential before it impacts with a plasticine block attached to a rigid frame. The plasticine impactor is 100mm long, 8mm thick and 35mm deep. The robot impacts with the 8mm side of the block. The impact point on the robot is on link 3 about 25cm from the shoulder joint close to the control point of this link, shown in Figure 8.27. This point has the higher potential impact force of the two control points and therefore limits the speed of the

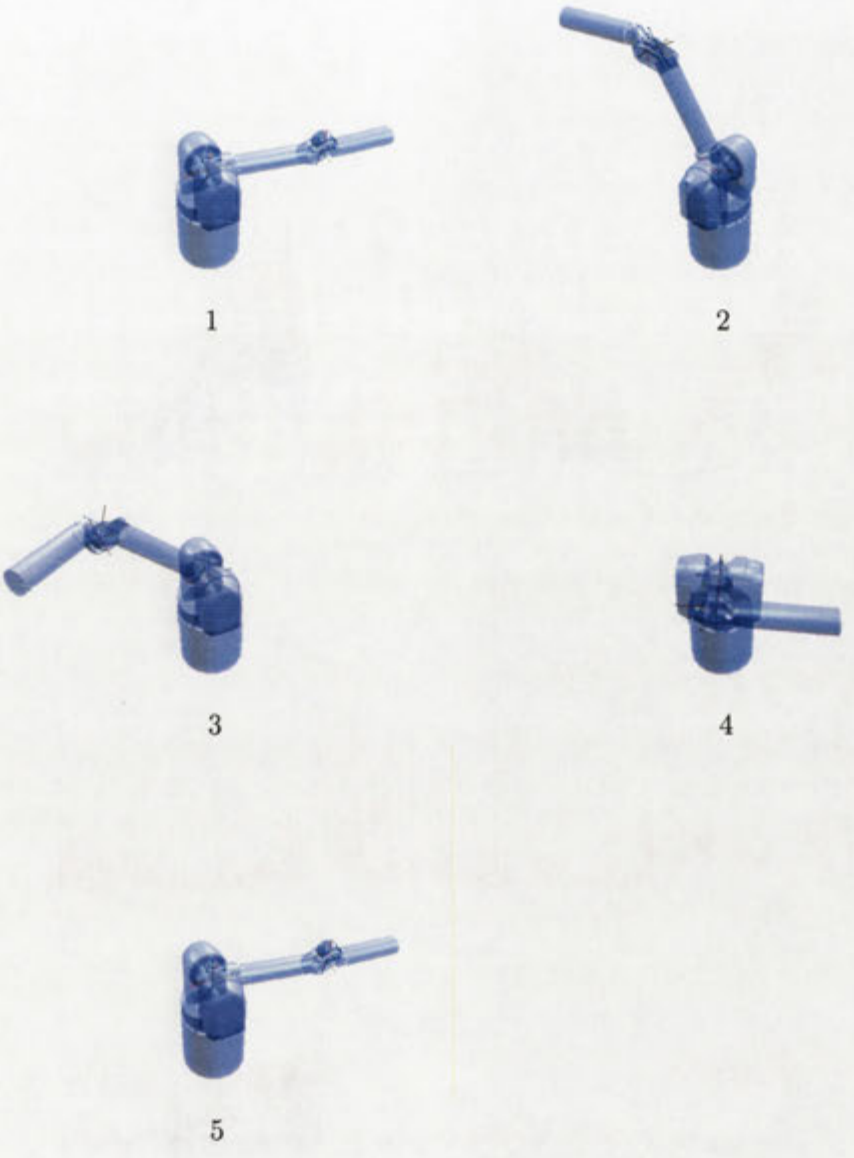


Figure 8.23: Motion sequence 2

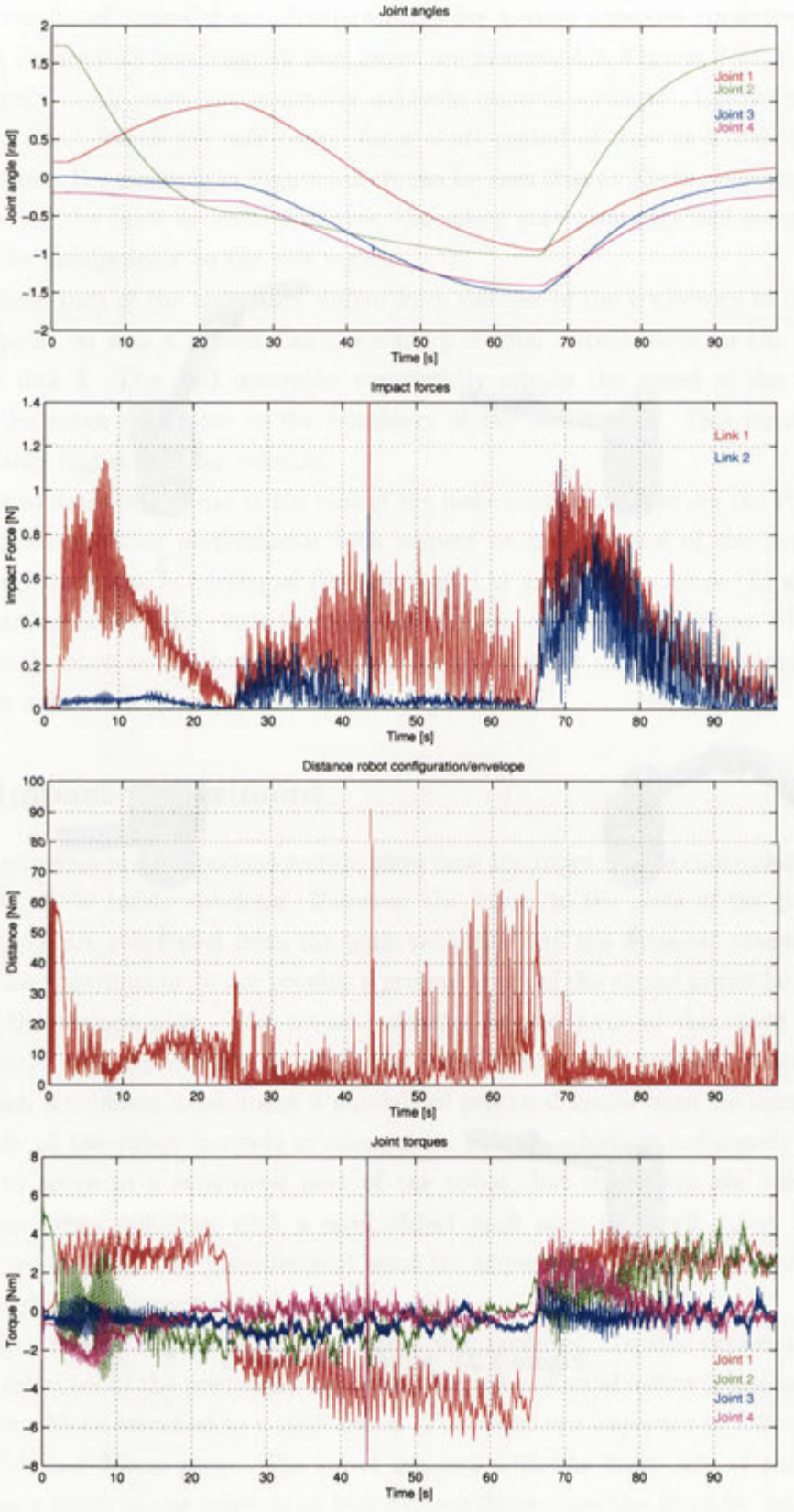


Figure 8.24: Safe robot motion using PID control and a 1N safety envelope

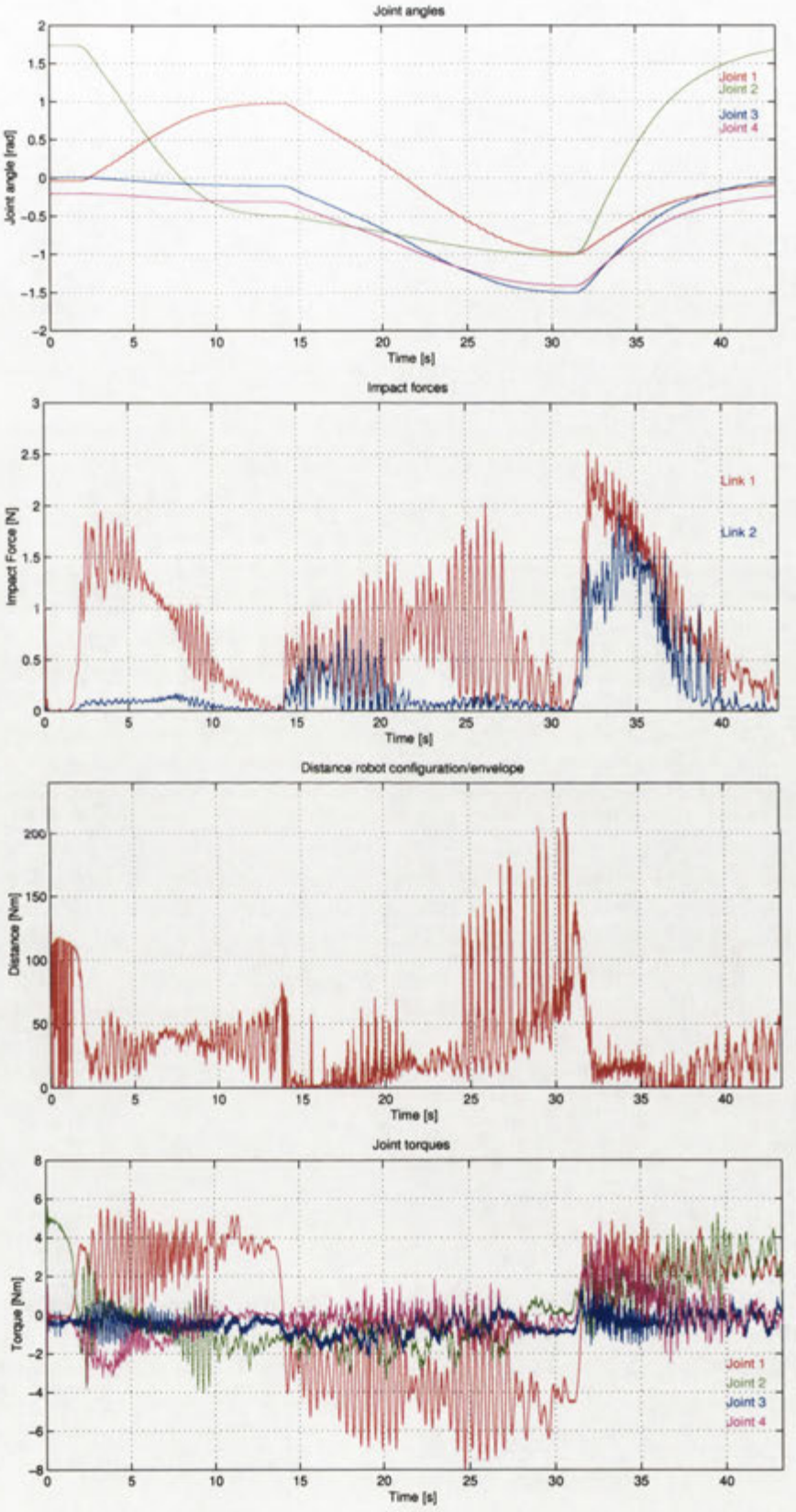


Figure 8.25: Safe robot motion using PID control and a 3N safety envelope

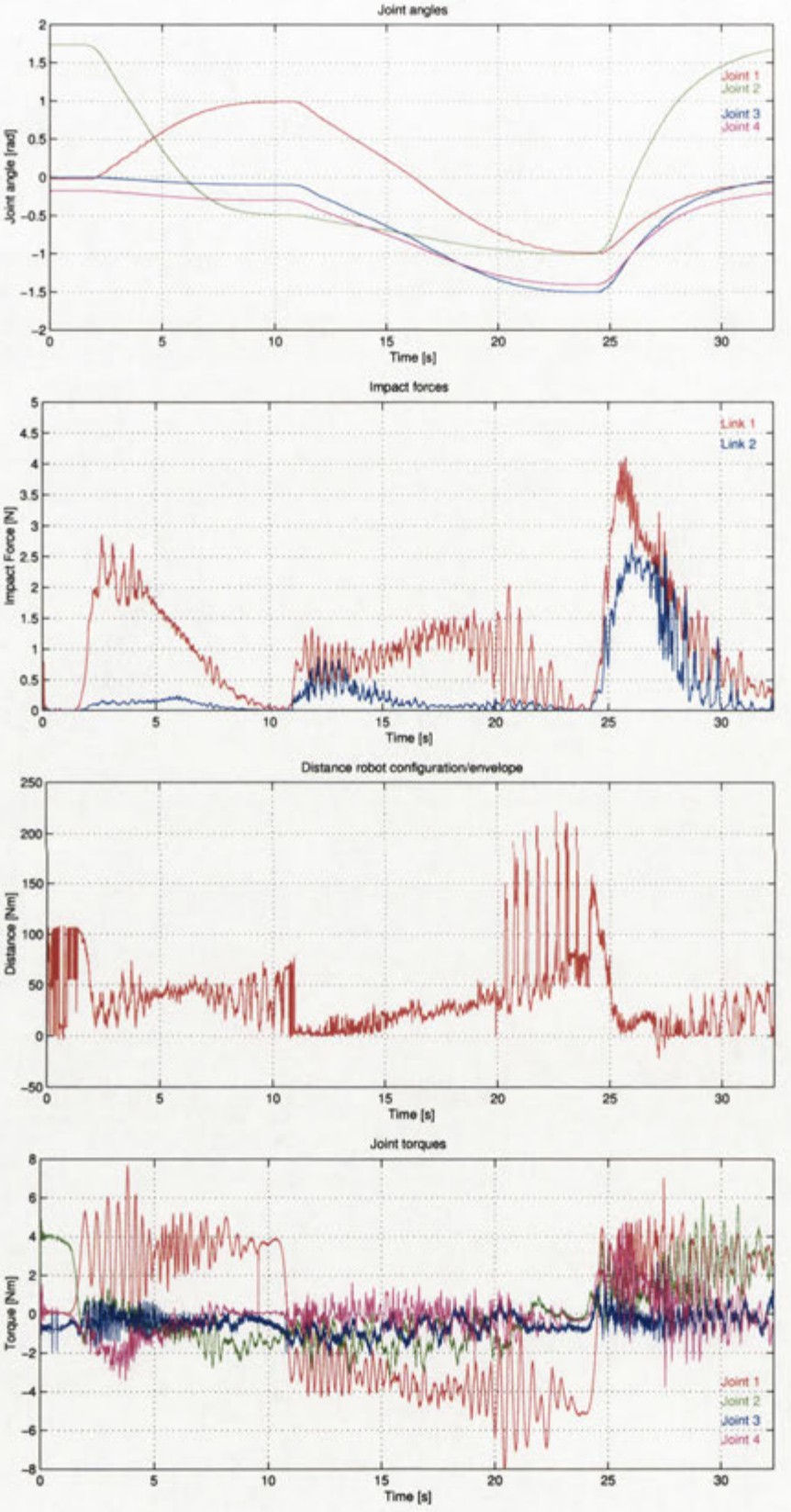
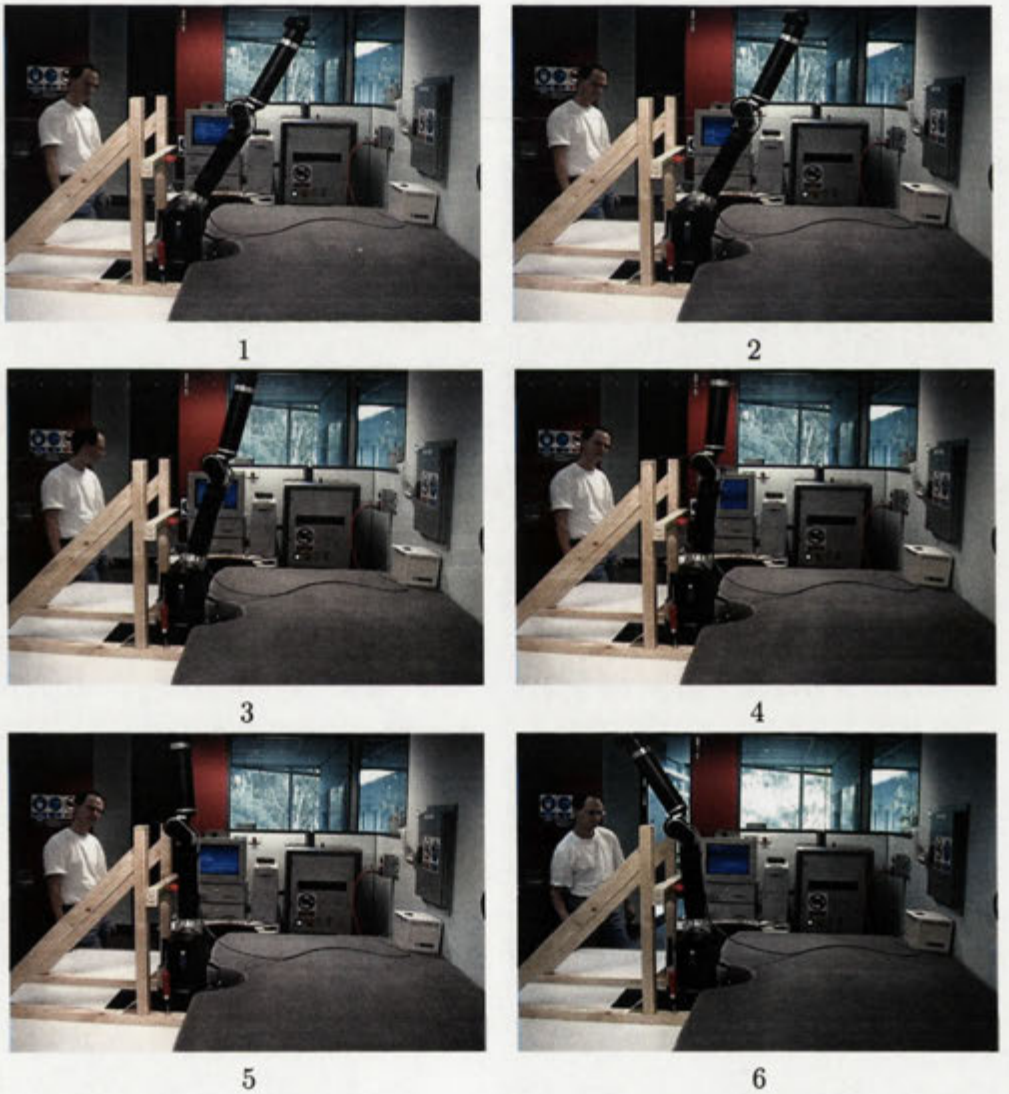


Figure 8.26: Safe robot motion using PID control and a 5N safety envelope

a) b)

Figure 8.27: Setup of the impact experiment



1 2

3 4

5 6

Figure 8.28: Image sequence of the impact experiment

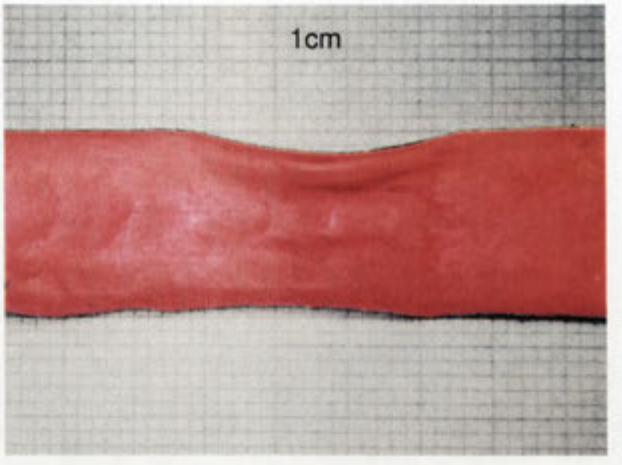
robot. Figure 8.28 shows the setup of the experiment before and after the collision. The robot rotates in a fully extended configuration (J_4 in zero position) around J_2. Joint J_3 is in zero position such that the joint axes of J_4 is perpendicular to the motion of the robot. When the robot reaches the vertical configuration (J_2 in zero position) the PID controller is disabled and the robot enters Zero-G mode. In this mode neither gravitational forces nor joint forces created by the PID controller can influence the impact. This means that during the deformation of the impactor no additional energy or forces are added by the motors. Therefore, link 4 continues to swing forward until friction stops it. Since the PID controller does not operate the robot continuously at the boundary of the safe region but accelerates initially to $\pi \approx \pi_{max}$ the point where the collision motion starts is selected such that the impact potential is between 95% and 100% of π_{max}.

The experiment was conducted with three different impact potential limits, 1N, 2N and 3N. The deformations of the impactors in the impact experiments are shown in Figure 8.29. The background paper of the plasticine impactors shows major lines every 1cm and minor lines every 2mm. The deformation in the impactor in Figure 8.29a) that resulted from an impact limit of 1N has a depth of 2.5mm. The deformations in Figure 8.29b) and c) have deformations of 4mm and 6mm caused by impacts with limits of 2N and 3N respectively.

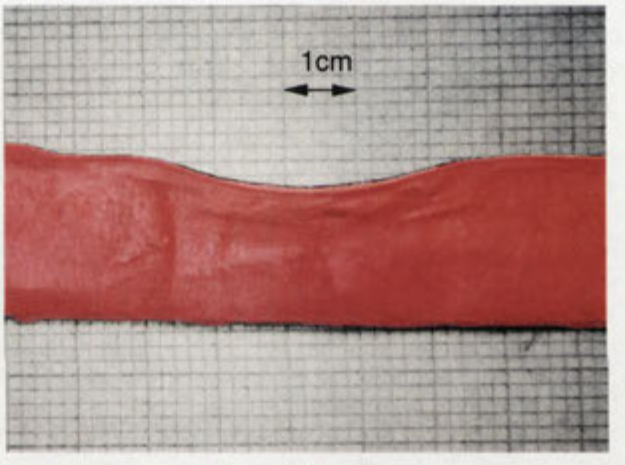
It should be noted that although the experiment provides an indication of the correct operation of the IPC scheme, the depth of the dent in the plasticine impactor does not measure the impulsive impact force of the manipulator. Since the impact is purely plastic, the depth of the dent is a function of the energy absorbed by the plasticine impactor. The deformation is the result of an impact of the manipulator in Zero-G without further joint torques which could influence the deformation and hence, the absorbed energy is purely kinetic energy of the robot. The kinetic energy is proportional to the square of the velocity while the impulsive impact force is proportional to the velocity. The increase in the deformation with increasing impact force limit therefore is only physical evidence and an indication, as opposed to the calculated potential impact force, that the IPC-scheme successfully controls the impact potential of the robot.

8.9 Summary

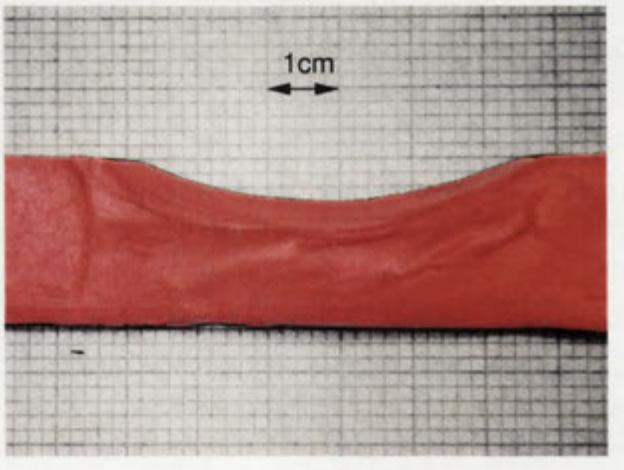
This chapter addressed a number of practical issues and presented solutions to problems that are crucial to the implementation of the IPC scheme. The choice of a suitable manipulator with regards to its kinematic and mechanical/surface design of the robot is crucial. Chapter 7 presented an example of a design of a robot base which is kinematically suited for the implementation of the IPC scheme. The WAM robot has an identical kinematic configuration and is therefore an ideal system for the IPC scheme. The discussion of poles in the potential impact force for the kinematic



a)



b)



c)

Figure 8.29: Plasticine impactors after the impact with different limits, a) 1N, b) 2N and c) 3N.

structure of the robot identified a variety of possible poles. However, most of the pole locations are inaccessible since they are embedded within the mechanism. Taking the geometry of the WAM into consideration, only two poles actually appear on the surface, one on link 2 and another on link 4 at the intersection with joint axis 1. Due to the experimental setup of the robot the base and link 2 in particular are not accessible for the user. Therefore, the pole on link 2 can be safely disregarded. The pole on link 4 however must be considered as it may appear anywhere along the cylinder of link 4. However, this pole only appears for a particular singular configuration of the robot which can easily be avoided by the motion planner using a simple heuristic, without restricting the capabilities and efficiency of the manipulator.

The WAM robot also has a number of other important characteristics. The surface of the robot has no sharp edges and mainly consists of cylindrical parts without gaps between moving parts. The light weight design of the robot allows efficient operation under the IPC scheme. It also makes physical interaction more convenient for the user. The WAM robot is equipped with a number of hardware safety features which limit the risk of injuries in the case of hardware or software faults and allow a shutdown of the robot. The joints are all driven by cables which makes the joints easily backdriveable and provides a transmission of torques with low friction in both directions. This not only allows a person to physically interact with the robot with ease, but it also allows the controller to measure external forces which act on *any* part of the manipulator based on a comparison of the dynamic model of the robot and the actual motion. The actuator effectively becomes a sensor. Although this method only allows the measurement of forces and torques along the columns of the Jacobian of the point of contact, the capability to sense external forces not only at the hand (with F/T wrist sensor) but on all links of the manipulator is highly advantageous. Overall the WAM robot is a well suited to the development of human-robot interaction systems and provides many advantages compared to typical industrial type manipulators.

The safety core requires an external position controller. As an example an adapted PID controller was proposed which used a basic PID loop to position control the robot. The controller is augmented with a method to shift the command position of the PID controller at a speed that allows operations of the robot close to the boundaries of the safe region. The general concept is to control the velocity of the robot along a trajectory in a way that the robot is moving close to the boundaries of the safe region while ensuring that the saturation effect for the generated torque vectors does not interfere with the proper operation of the controller. This general concept is also applicable to other control laws, too. The IPC scheme and the safety core/external position controller architecture are therefore particularly suited for control methods based on a learning algorithm which is able to use the distance d and the impact potential π as feedback to learn the control of the manipulator most efficiently. The adapted PID controller is a proof of concept of how position control

can be achieved using the safety core.

The software modules of the system are distributed over multiple standard personal computers. The robot server PC runs the safety core, the PID controller and the external force sensing. All three modules operate at the actuator control frequency of 1KHz. Therefore their ability to exchange data in real-time is critical. The calculation of the control constraints is computationally expensive. The update of the constraints is not as real-time critical as the verification and clipping of command torques. The calculation of the constraints was performed on a dedicated PC which produced updates of the control constraints at a frequency of 200Hz. The interface presented to an application client allowed the definition of new goal configurations of the robot, grasping commands for the gripper and provided updates of the robot's state and impact potential at a frequency of 10Hz.

This chapter also presented a number of experimental results. The passive motion experiment with an extra weight in the hand showed how the safety envelope influenced the motion of the arm and effectively dampened motions which exceeded the safety limit set by the maximum impact potential π_{max}. For robot motions which are a result of external forces the safety envelope only influences the robot behaviour if the robot leaves the safe region and the current impact potential exceeds the maximum impact limit. In this case the safety envelope generates dampening torques which depend on the time constant t_c. The selection of smaller t_c cause stronger dampening torques, and a quicker return by the robot to the safe region. Within the safe region the robot's behaviour is equivalent to the basic Zero-G behaviour. The safety envelope combines both the passiveness and the predictability of the Zero-G mode with a quantitative limitation of the impact potential of the robot.

The motion experiments demonstrated the functionality of the PID position controller. Two different motion sequences were performed by the system at three different settings of the safety core. The experiments demonstrated that the controller was successfully able to generate joint torques that were close to the control constraints. However, after the robot reached the limit of the impact potential the controller was not able to retain this close proximity to the boundaries in the state space during the remaining motion. Instead, a slow decay in the impact potential occurred. This resulted in a suboptimal efficiency in terms of speed. On the positive side the system succeeded in moving the robot along straight lines in the configuration space and in keeping the robot in the safe area of state space. These are the major prerequisites for safe human-robot interaction.

Finally a collision experiment was presented which allowed a qualitative verification of the effectiveness of the safety core. The robot collided with a stationary rigidly mounted plasticine impactor at different settings of the safety core. Although the deformation of the plasticine does not strictly speaking measures the impact potential of the robot, the increasing deformations with greater values of π_{max} indicate that the impact core works properly.

Chapter 9

System Integration

The previous chapters presented the algorithms for the vision based 3D head pose and eye gaze interface and the human-friendly robot control scheme using impact force control. This chapter shows how the two components are integrated in an example application which illustrates the feasibility of human-robot interaction systems based on natural interaction. The resulting system is an interactive pick-and-place robot which allows a user to interact with the manipulator through eye gaze and facial gestures as well as through physical contact.

Section 9.1 describes the basic system functionalities which allow the user to control the robot and manipulate objects in the environment. It also gives a brief overview of the implemented action set. Each robot action can be invoked by the user through a context-dependent facial gesture.

The physical setup of the human-robot interaction experiment is described in Section 9.2. The robot is mounted in the centre of a table top environment. The user stands in front of the table which allows convenient passing of objects between the robot and the user.

Section 9.3 briefly describes the information flow between the agents of the human-robot interaction system including the robot, the vision system and the user.

A detailed description of the finite state machine which controls the overall behaviour of the system is presented in Section 9.4. The finite state machine uses the input from the vision system and the robot to determine an appropriate high-level action for the robot.

The results of human-robot interaction experiments using gesture control and the IPC scheme are presented in Section 9.5. The functionality of the system is demonstrated by an $8\frac{1}{2}$ minute interaction experiment which shows all interaction modes of the robot system.

Finally, a summary of results is in Section 9.6.

9.1 System functions

The interactive pick-and-place application is designed to allow a human operator to place and retrieve objects in the environment using a robot. A small set of functions is sufficient to provide the the system with basic object manipulation abilities. All system functions are controlled with a few facial gestures and the users eye gaze.

9.1.1 Basic concept

The basic concept is to create a natural interaction system that allows a human operator command a robot system under impact force control to carry out simple manipulation tasks using a natural and intuitive interface that doesn't require any in depth knowledge about robots, kinematics and robot control. The visual interface based on gaze direction tracking and gesture recognition presented in the previous chapters is well suited to this task. The visual interface provides two basic functionalities required for an interactive pick-and-place system: Discrete events to issue commands and a deictic measurement mode which allows the user to reference objects and positions in the environment.

Discrete gestures such as nodding and shaking the head are used to invoke robot commands. The commands include activation and deactivation of the robot and manipulation commands such as *pick object* or *put down object*. The eye gaze is used as a deictic input device to identify the objects and positions to which commands were previously issued. In addition, gestures are also used to correct any errors that may occur. The data provided by the vision system is always prone to errors since all measurements are based on visual detection methods which cannot be 100% reliable. The errors include missed gestures, phantom gestures and falsely classified gestures and inaccurate gaze point measurements. To prevent the system from executing an erroneous command the *no* gesture, characterised by shaking the head, can be given by the user. The *no* gesture terminates the execution of the current command, depending on the robot state undoes robot actions and puts the system into the next logical rollback state.

Besides the visual perception the human-robot interface uses the contact force detection system described in Chapter 8 to allow object passing between the user and the robot. The robot can receive an object from a user by waiting in a transfer position as shown in Figure 9.1 with open fingers until it senses an external force acting on the arm. The force is detected when the user pushes an object into the hand. The object is then grasped and the success of the object grasping operation can be determined from the positions of the finger at the end of their closing. The same technique is used when the robot passes an object back to the user. The robot waits in the transfer position until it senses the user pulling at the object. It then opens the fingers and releases the objects. A simpler method that detects a change in the configuration of the robot could be used. The external forces result in a change

Figure 9.1: Transfer configuration

in the configuration of the robot which could trigger the transfer. The change in configuration depends on the control method used for the robot. If a control method which provides high stiffness is used the change may not be sufficient to provide safe triggering whereas the external force measurement is independent of the robot control.

The implementation of the interactive pick-and-place system presented in this thesis is a proof of concept only, a robot system with similar capabilities mounted on a motorised wheelchair could be used by a disabled person. In addition, the vision system could also be used to control the wheelchair similar to the system presented in [Bergasa et al. 1999] which allows a person to drive a wheel chair using facial gestures such as looking up, down, left and right.

In summary, the interactive pick-and-place robot provides the following functionalities:

- Invocation of robot action through facial gestures
- Object and position identification using the user's eye gaze
- External force measurement to allow natural object passing between the user and the robot
- Error correction and action termination through facial gestures

9.1.2 Actions of the interactive pick-and-place system

The number of basic actions required for an interactive pick-and-place robot is quite small, in fact two actions would be sufficient. To add more flexibility and to enable the user to move objects in the environment without physically handling them, the pick-and-place actions are split into two parts, the object acquisition stage and the

object placement stage. This breakup also provides an additional rollback point for the error correction.

When the system is initiated, it remains disabled until it is explicitly enabled by a user. The *yes* gesture (nodding) activates the robot and brings it into the idle configuration (shown in Figure 9.2). The robot remains in this position until it is commanded to receive an object from the user with a *look up* gesture (tilting the head back) or to grasp an object on the table with a *look down* gesture (tilting the head forward). The eye gaze point of the user after a *look down* gesture determines which object is to be grasped from the table. After grasping the object from the table or from the user's hand the robot returns to the idle position with the object grasped and remains there until another command is issued. Again the *look up* and *look down* gestures are used to either pass the object to the user or to place the object onto the table. The gestures are therefore context dependent but easy to associate for the user. A *look up* gesture always refers to an action related to approaching the user while a *look down* gesture always refers to an action related to a transfer to or from the table. If an object is to be placed onto the table, the eye gaze point after the gesture determines the position where the object is to be put down. The system is deactivated by a *no* gesture when it is in the idle state and no object is grasped.

The action commands of the robot system are summarised in the following list:

- Enable/Disable system
- Receiving objects from the hand of the user
- Grasping objects on the table (object identified by eye gaze)
- Handing grasped objects over to the user
- Putting an object down (position identified by eye gaze)

9.2 Experimental setup

The setup for the interaction experiments is shown in Figure 9.3. The robot is mounted on a desktop (~90cm high) with the operator standing in front of the table. In this configuration the robot can easily exchange objects with the operator. A Sony EVI-D30 camera, the input to the vision system, is located on the far end of the table slightly translated to the right to allow a unoccluded view of the operators face. The zoom of the camera is adjusted to provide images showing the user's face in a comparable size to the images presented in Chapter 5. Figure 9.2 showed a typical situation during an interaction experiment with the robot in the idle configuration.

The target positions shown in Figure 9.3 are visible in Figure 9.2 as five yellow dots on the table top. The dots mark the positions where objects can be placed and picked up by the system. The system does not include a scene camera that could

Figure 9.2: Experimental scene

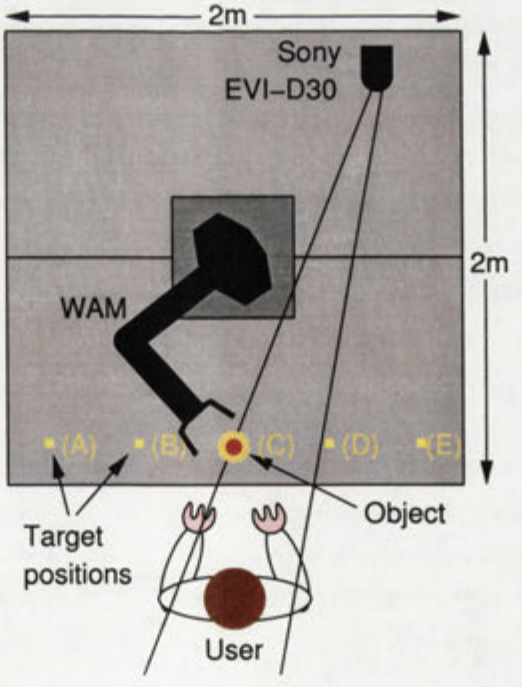


Figure 9.3: Top view of the experimental setup

locate objects in the environment and guide the grasping operation of the robot. The gaze point estimation is not sufficiently precise to allow the robot to "blindly" grasp an object. Due to the shallow viewing angle of the user with respect to the table top an angular error of 3° corresponds to about 10cm error along the projection of the gaze vector on the table plane. The arrangement of using predefined positions also simplifies the path planning considerably since the approach and grasp configurations for each target position can be manually taught in Zero-G mode. Although the kinematics of the 4 DOF configuration of the WAM are simple, the approach and grasping direction would still need to be deduced by the system. During operation the robot moves from the idle configuration through the approach configuration to the grasp configuration and back into the idle configuration to grasp objects and in the opposite order to put objects onto the table.

The hand gesture controlled robot presented by [Triesch and von der Malsburg 1998] used special non-natural gestures to provide the robot with additional information on the grasping direction for each operation. However, such methods could be perceived as cumbersome and non-intuitive by users. The main focus in the interactive pick-and-place system in this research has been given to natural interaction modes. A scene camera system with an automatic grasping planning module was beyond the scope of this thesis.

The table top setup of the experiment with the user standing freely in front of the robot suggests the use of deictic hand gestures to identify objects and positions. The use of eye gaze may appear to be slightly awkward. It is generally agreed that natural human-robot interfaces should incorporate as many interaction modes as possible, and therefore, the inclusion of deictic hand gestures would certainly improve the quality of interaction in this experimental setup. However, this experiment is only a proof of concept and is not intended to reflect all characteristics of a possible application of this technology. As mentioned in Section 9.1.1, one of the first applications of human-robot interaction systems could be in the area of helping devices for disabled persons where hand gestures are not an option.

9.3 System structure

The interactive pick-and-place system is realised by the integration of the visual gaze and gesture recognition system and the human-friendly impact force controlled WAM robot manipulator. Figure 9.4 illustrates the principal components and the information flow between the user and the components of the system. The finite state machine shown in the figure represents the system state and generates commands to the robot.

The arrows show the flow of information through the system. The outer clockwise circle closes the control loop and represents the major direction of information flow in the system. All actions of the system are initiated by the user whose gestures and

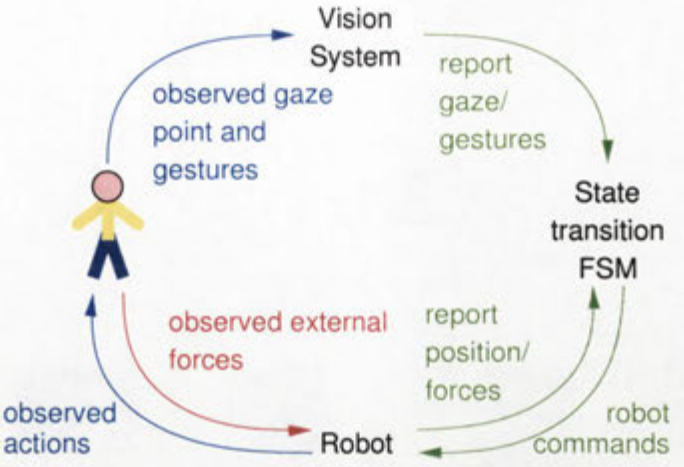


Figure 9.4: Information flow

gaze point are detected by the vision system. This information is passed to the finite state machine (FSM) which controls the state transitions. All actions of the robot are generated by transitions in the FSM. The action commands are transmitted to the robot for execution. The actions of the robot are observed by the user who determines the next command to issue according to the progress of the task and the successful execution of the actions.

In addition to this closed control loop the manipulator itself is used as a sensor to measure external forces acting on the robot. The information flow originating from forces applied by the user onto the robot forms the counterclockwise flow. The blue arrows indicate information flow based on visual observations, red arrows represent physical forces and the green arrows represent digital information transfered within the computer system.

The system was implemented using three standard PCs. The vision processing computer used a 200MHz Pentium with a Fujitsu CTRV vision system (see Chapter 3). This computer also implemented the FSM which required few resources. Two Pentium-III/500MHz computers were used to implement the control loop for the robot and the safety envelope boundaries. The three computers were networked using dedicated Ethernet links. The data exchanged between the vision computer and the robot control computer is not time critical and no latency control is required. The data includes time stamped position and external force information. This data is generated at 10Hz by the robot control computer, and position commands for the robot are issued by the FSM on the vision computer.

9.4 Finite state machine implementation

The finite state machine is central to the control of the interactive pick-and-place system. It receives all sensor data from the vision system and the robot before determining the appropriate state transitions. Figure 9.5 shows the structure of the FSM,

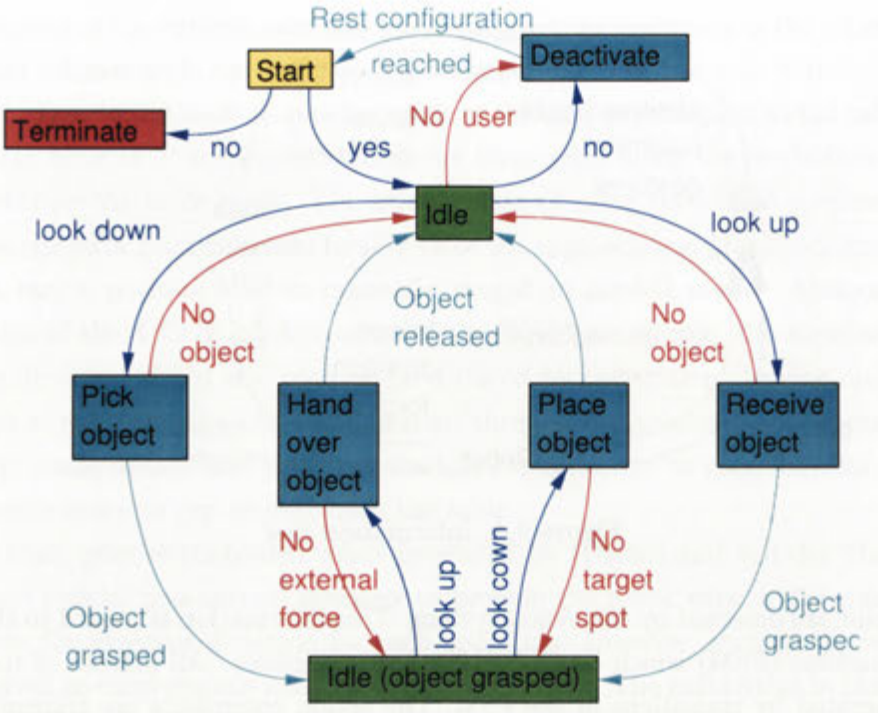


Figure 9.5: Simplified representation of the FSM

including the transitions which control the operations described in Section 9.1.2. The system enters the *Start* state when the vision system is active and the robot is deactivated. A *yes* gesture (nodding) activates the robot and brings it into the idle position. All actions are activated through gestures. Transitions initiated by gestures are marked with blue arrows, transitions caused by error conditions are marked with red arrows and transitions in turquoise are the results of actions or time-out conditions.

The green states are stable states in which the system can remain until a new command is issued or an error occurs. For example if no user can be detected by the vision system for several minutes while the system is in the idle state. The turquoise boxes denote temporary states in which the system remains only until an operation is completed or an error occurs. Without explicit user interference the system always returns to a stable green state.

Figure 9.5 shows only the gross states of the system. Each state has a number of sub-states which reflect the progress of an action performed by the state. Figure 9.6 illustrates the details of the idle state. The state has five distinct sub-states, of which the first three are temporal and are designed to bring the robot back to the rest position. If no errors have occurred and the user has not performed a *no* gesture which deactivates the robot, the system will automatically reach the stable *wait for command* sub-state. The user can then invoke different actions using the associated gestures. The diagram for the *idle (object grasped)* state is similar.

The sub-states of the *pick object* state are shown in Figure 9.7. Although this

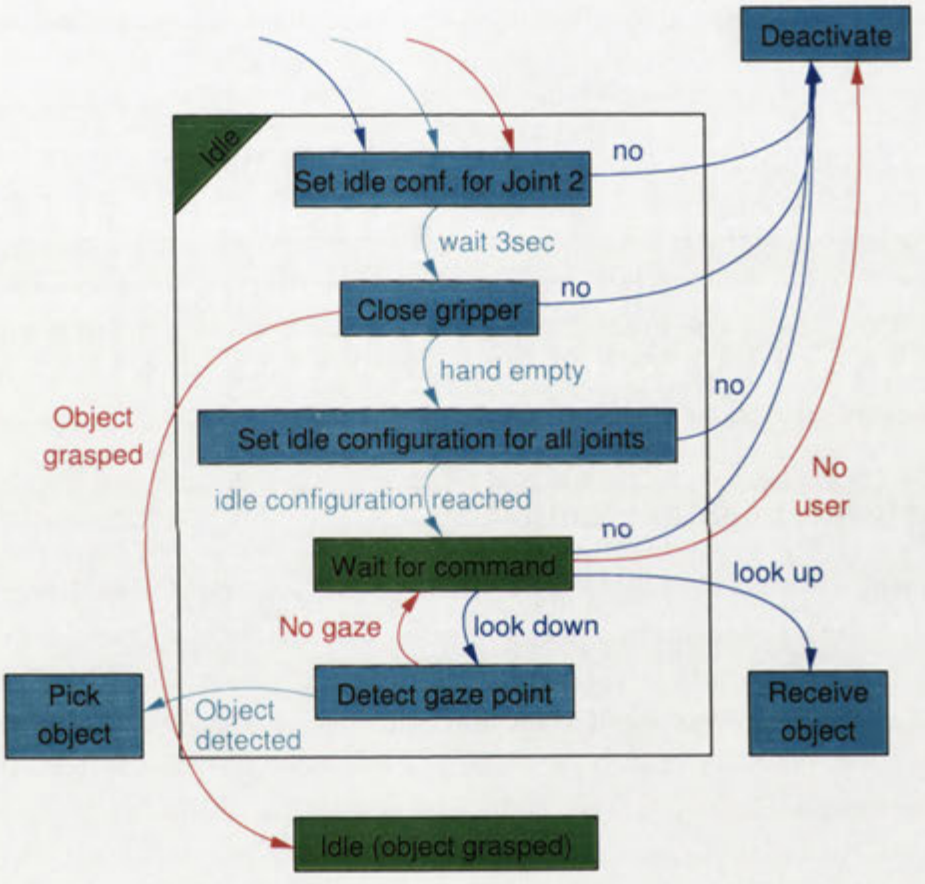


Figure 9.6: sub-states of the *idle* state

state contains more sub-states, the underlying structure is serial. The robot moves into the approach configuration, then into the grasp configuration where it grasps the object and continues into the *idle (object grasped)* state unless the user intervenes or an error occurs. If the user signals via a *no* gesture that the vision system did not determine the correct object position the system uses the next closest available object or if there are no more targets nearby it returns to the *idle* state. If the system switches to a different object a direct motion towards the new approach configuration may cause a collision with the table or objects on the table. Therefore the robot must first move away from the table into free space to allow a safe reconfiguration of the arm. Figure 9.8 shows the safe reconfiguration state. Depending on the sub-state of the pick operation the change of action to a different object requires rollback actions that ensure no collisions with the table or objects on the table.

The representation of the *put down object* state is similar. The diagrams of the other states are based on the same philosophy and do not provide any further insights into the underlying mechanisms used to achieve the desired robot behaviour and can be found in Appendix G.

In addition to the transitions shown the figures all temporal states have time-out values. If a time-out occurs the system shuts down the robot. The time-outs are designed take take effect only if operations could not be completed in an orderly way and the system must be shut down for safety reasons.

9.5 Experimental results

The interactive pick-and-place robot application has been tested extensively. This section presents the results of an experiment with a duration of 8:32 minutes which demonstrates all features of the system. A video of the entire experiment can be found on the CD-ROM included in this thesis. It shows the simultaneous view from a scene camera and from the Sony EVI-D30 camera used in the vision system. Please refer to Appendix E for details regarding the videos on the CD-ROM.

At the start of the experiment there are no objects on the table. First, the user hands an object to the robot and commands the system to place the object on the centre target position (C) onto the table. Two more objects are then placed, one at the leftmost position (1) and then at position (D). During the placement of the last object an error occurs because the vision system incorrectly determines position (E) to be the target position. The user rectifies the error using a *no* gesture and the robot then moves to position (D) before it releases the object and returns into the *idle* state. The next task is to move the object in position (1) to position (B). The user commands the robot to pick up the object in position (1) using the *look up* gesture and the system settles into the *idle (object grasped)* state. Next the user indicates with another *look down* gesture that the robot is to put down the object and gazes at position (B). The robot places the object in the correct position and

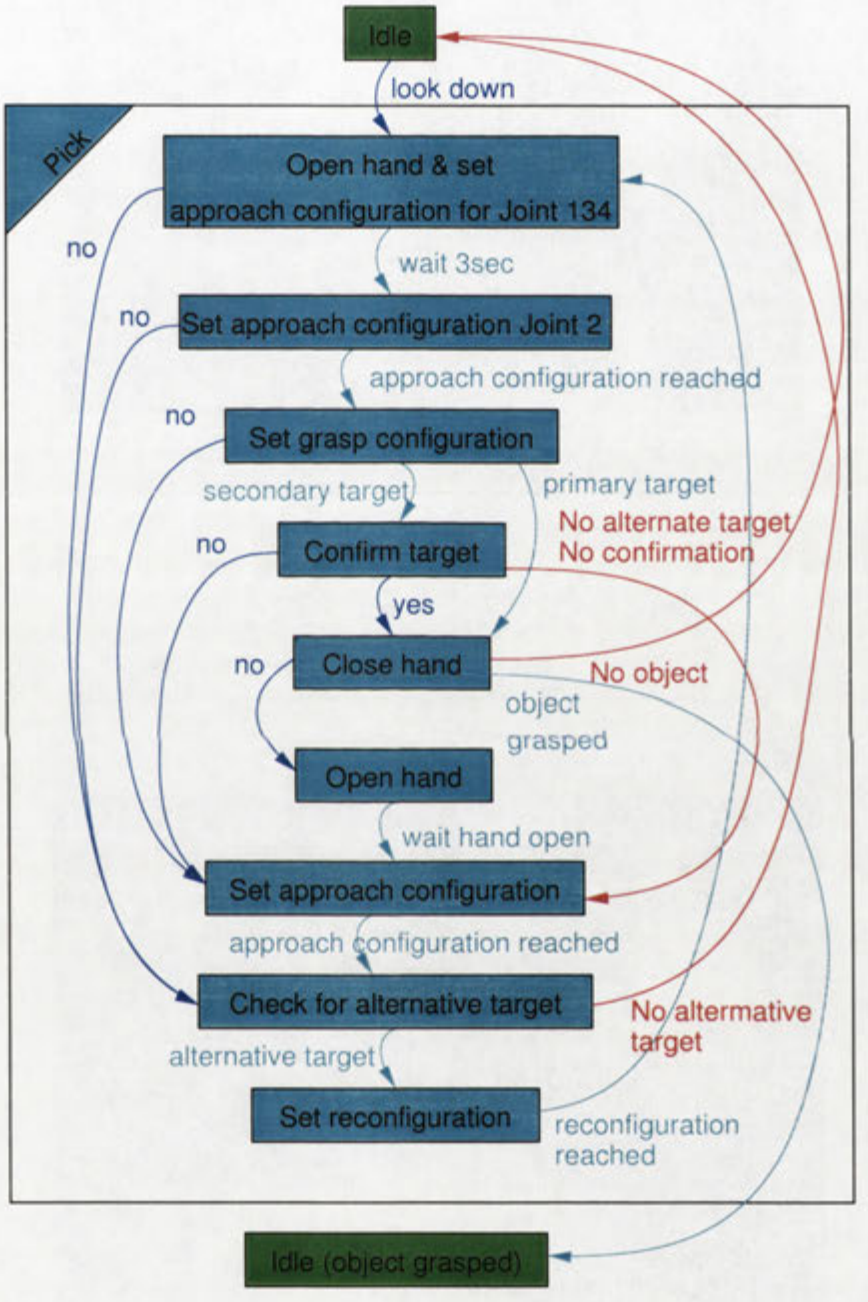


Figure 9.7: Sub-states of the *pick object* state

Figure 9.8: Safe reconfiguration pose

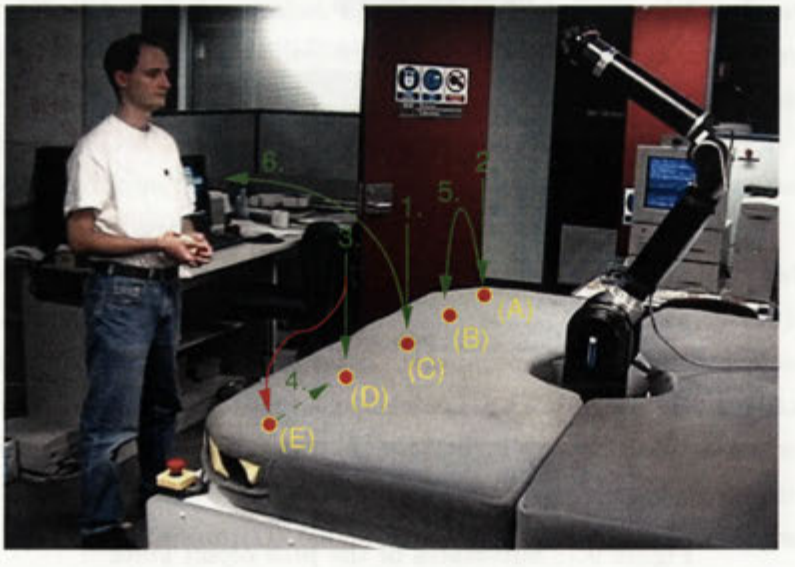


Figure 9.9: Action sequence of the experiment

returns to the *idle* state. Finally the user retrieves the object from position (C) and the robot hands the object to the user. Finally the user shuts down the system with a *no* gesture. The sequence of actions performed in the experiment is illustrated in Figure 9.9.

Figure 9.10 shows various parameter plots of the experiment. The plot a) shows the joint angles of the robot. The blue spikes coinciding with the negative red spikes indicate instances when the robot was in a grasping configuration on the left side of the table (target positions (1)-(C)), the red spike which coincides with a negative blue spike indicate grasping configurations on the right side of the table (targets (D),(E)). The instances when objects were exchanged between the user and the robot are characterised by negative green spikes that coincide with positive black spikes and zero readings on the red and blue joints. The three transfers of objects and the hand over at the end are easily identified.

The plot b) shows the angular distance between the 3D gaze vector of the user and hypothetical gaze vectors to the respective targets as illustrated in Figure 9.11. If the user fixates one of the targets, the respective angles are 0. Note that for the sake of clarity only the gaze angles with respect to the targets (A), (C) and (E) are shown. The angles to the remaining two targets lie between the plotted values. The relatively high offsets in all three plots are a result of the user observing the robot's motions. The unusual readings in the time period 330s to 380s were caused by a temporary tracking loss (please refer to the video) and subsequent incorrect head pose estimation resulting in a gaze vector directed to the far right. As soon as the vision system recovered the true head pose the gaze vector returned to its normal range.

The plot c) shows two force values, an external force value shown in blue and an impact force value in red. The external force is the result of the discrepancy of the dynamic model of the robot with the observed motions as described in Chapter 8. The magnitude (in Nm) of the error torque vector of the individual joint torques serves as a measure for the external force. The magnitude of the error torque vector is mainly constant when the robot is stationary. However, during motion the plot shows strong oscillations which increase in frequency with increasing joint velocities of the robot. The oscillations are caused by the torque ripple of the motors which is not represented in the dynamic model of the robot and therefore show up as joint torques caused by external forces. The oscillations are the reason why the user must wait when an object is exchanged with the robot. Once the robot is stationary the external force measure settles to a constant value. Only then the contact forces generated by the user can be reliably sensed. The red plot shows the maximum of the impact forces calculated at the two impact control points[1]. The impact force limit was set to 3N which was violated several times by only fractions of a second.

[1]Described in Chapter 7.

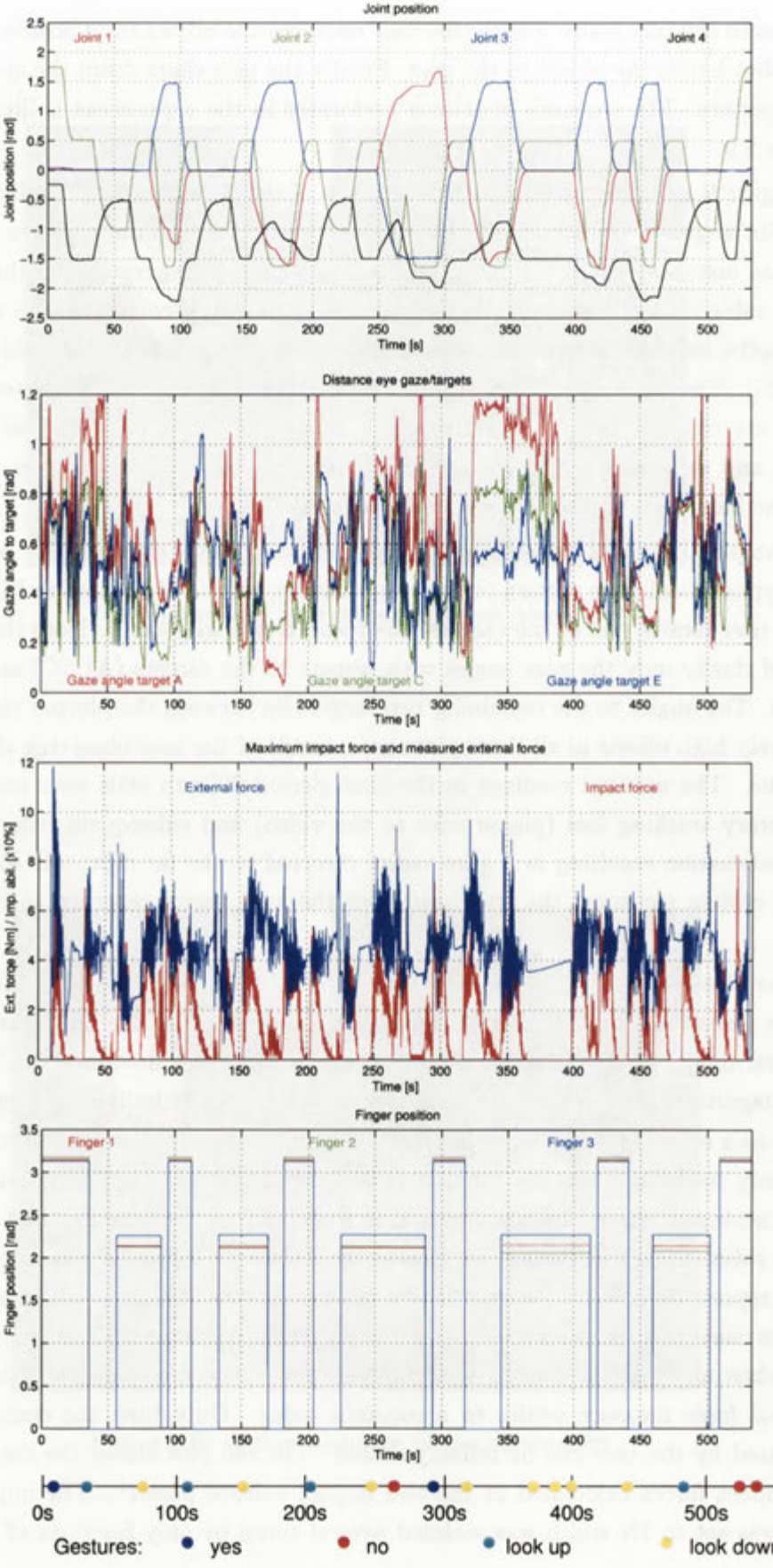


Figure 9.10: Interaction experiment (8:32 minutes)

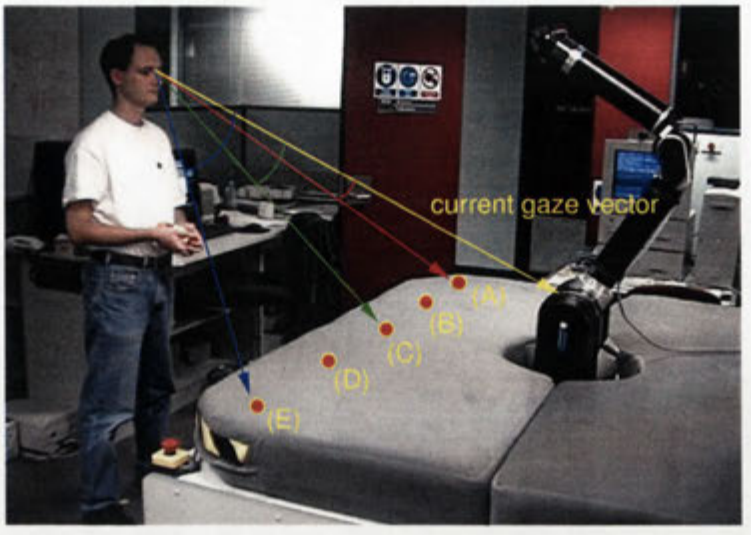


Figure 9.11: Angles between current and target gaze angles

This is also a result of the torque ripple in the motors.

The plot d) shows the position of the fingers of the robot hand. Joint angles close to 0 indicate an open hand, while angles close to π a closed hand. Angles of approximately 2.2rad indicate that an object is being grasped by the hand.

The following sections examine parts of this experiment in more detail and describe the influence the various parameters have on transitions in the FSM.

9.5.1 Place sequence

Figures 9.12 and 9.13 show snapshots of the first 110s of the interaction experiment described in the previous section viewed by a scene camera. The images are recorded with a sampling rate of 0.3Hz. The first image frames show the robot is deactivated and is waiting in its rest position for a *yes* gesture. After activating the robot reaches the idle configuration (frame 4) where the user commands the robot to receive an object at the transfer position with a *look up* gesture. The vision system recognised a *look up* gesture at 33.6s (frame 9) the robot starts to move towards the user (frame 10). At frame 14 the robot reaches the transfer position and waits for the user to place the object into it's hand.

Figure 9.14 shows the respective parameter plots for this 100s sequence. This figure is analogous to Figure 9.10. The spike in the blue external force in plot c) at 54.8s (frame 16) is clearly visible and was detected correctly by the system. The fingers of the robot hand begin to close immediately. In the motion towards the idle configuration more large spikes appear in the plot. As previously mentioned these spikes are caused by the torque ripple of the motors and could be easily confused with external contacts. Therefore objects can only be exchanged when the robot is stationary.

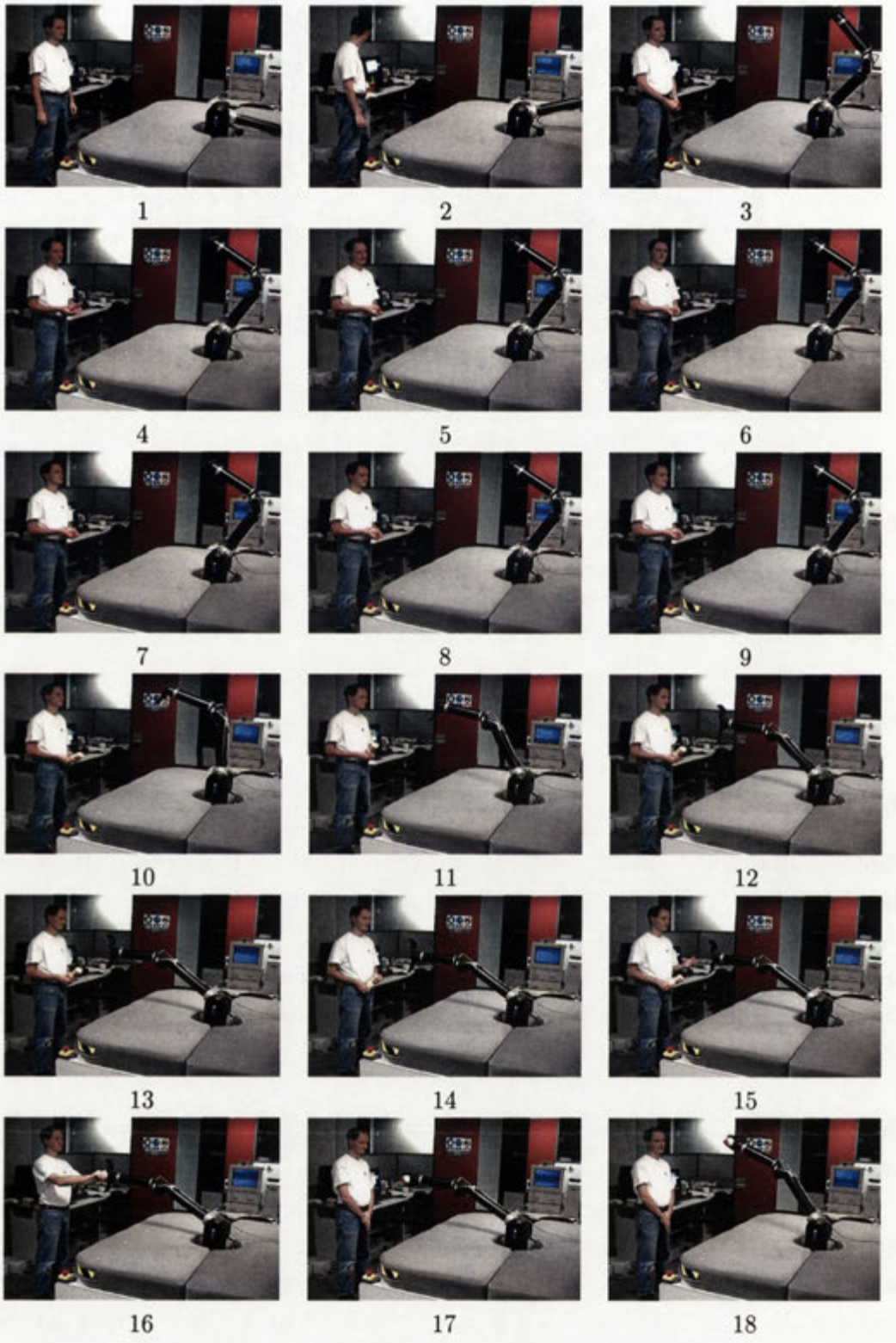


Figure 9.12: Put down object sequence (part 1)

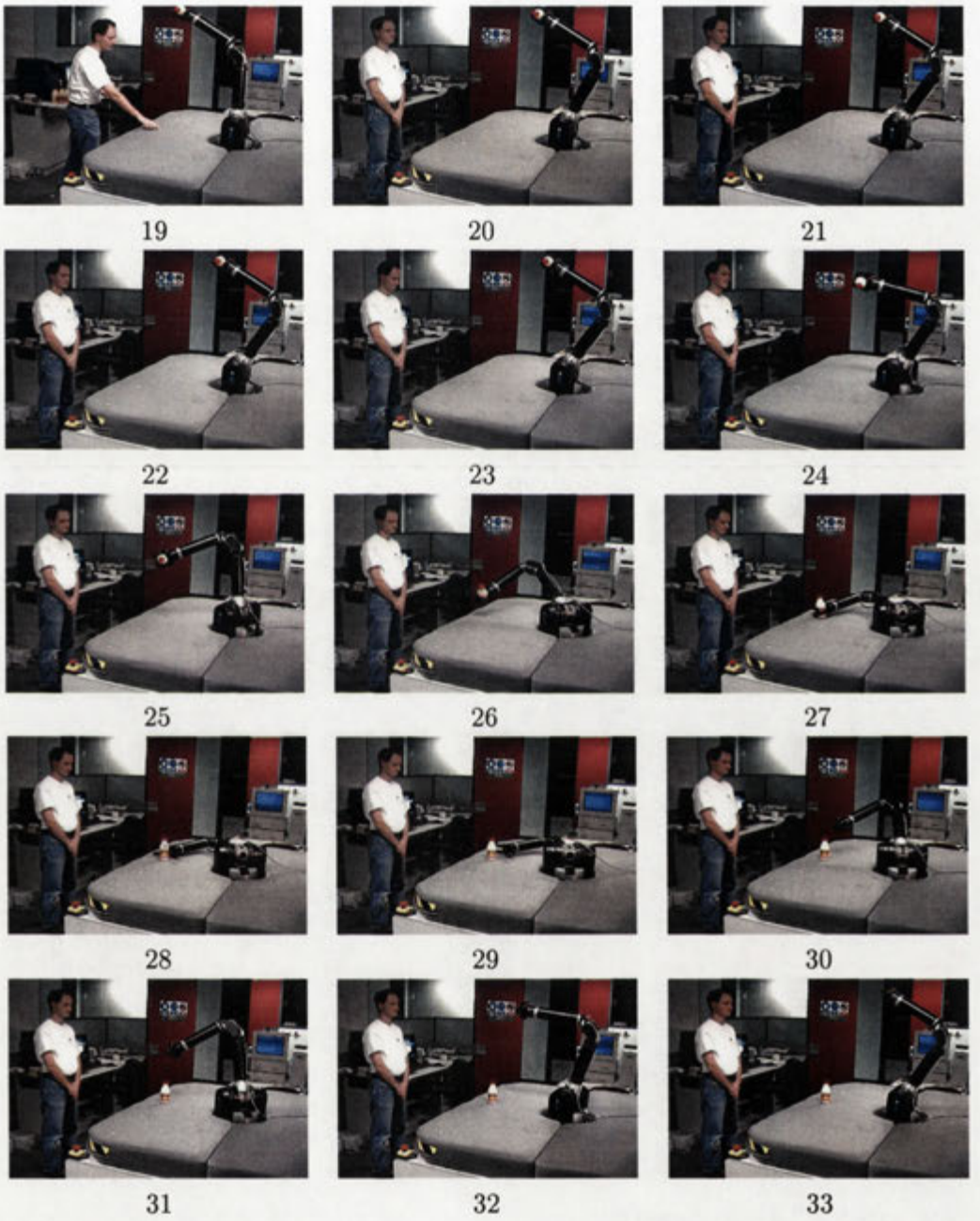
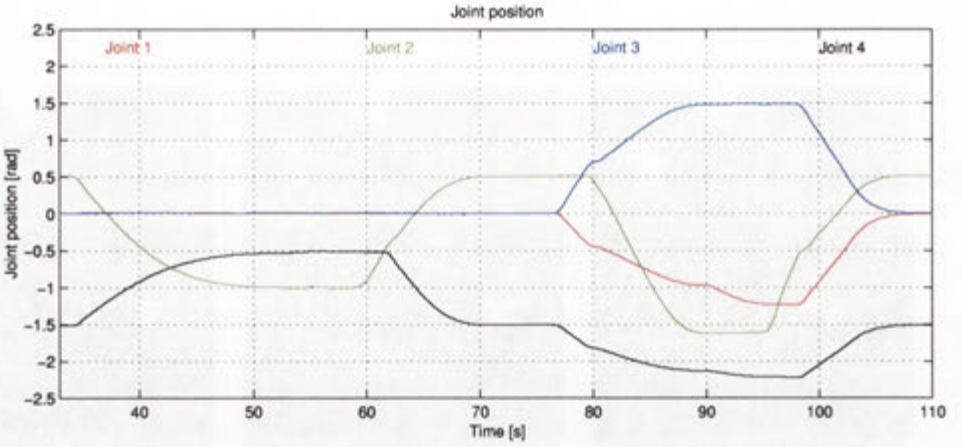
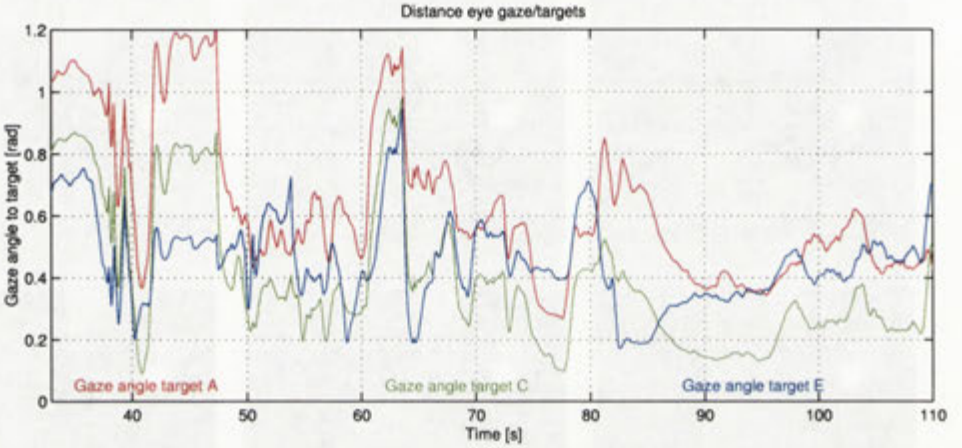


Figure 9.13: Put down object sequence (part 2)

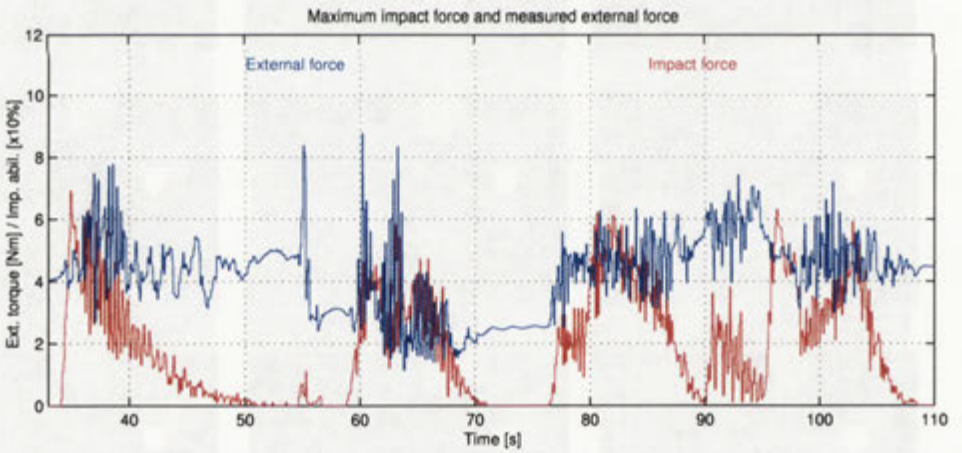
a)



b)



c)



d)

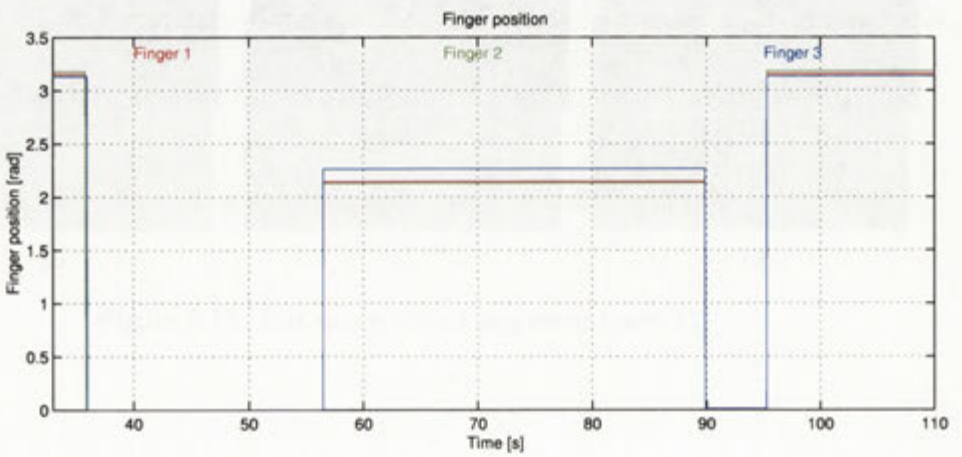


Figure 9.14: *place object* sequence

In frame 19 the user designates with eye gaze that position (C) is the location where the object is to be placed. At 73.7s (frame 22) the user performs a *look down* gesture initiating the *put down object* operation. Immediately after the gesture the gaze point clearly moves to the centre position (green plot falls to 0.1rad) which is correctly detected by the system and the robot places the object at position (C) in frame 27. The joint angle plot shows that the hand was in the approach and grasp configuration between 89s and 95s in the experiment. It can also be seen in the gaze distance graph that during this time the user's gaze point was directed at position (C).

9.5.2 Pick sequence

The time period between 435s and 514s is a typical combination of the *pick object* and *hand over object* actions. This is the last operation in the experiment. Figures 9.15 and 9.16 show an image sequence from a scene camera. The images show the robot approaching the object in position (C), grasping it in frame 8, return to the idle configuration in frame 12 and finally handing the object over to the user in frame 23.

Figure 9.17 shows the parameter plots of the subsequence. The *look down* gesture which initiated the *pick object* operation was recognised by the vision system at 437.5s in frame 1. The user's gaze at the object at position (C) is clearly visible in the angular distance plot 9.17b) at 440s in frame 2 and was properly identified by the system. The *look up* gesture was detected by the vision system at 480s (frame 15) and the robot starts to move into the transfer configuration to perform the *hand over object* operation. The robot reaches the transfer configuration at 500s (frame 21). One second later the grip of the user on the object is clearly visible as a spike in the blue external force plot 9.17c). The finger joint angle plot 9.17d) shows that the fingers immediately open and the object is released.

9.5.3 Error correction sequence

The last sequence that is discussed in detail is the error correction sequence shown in Figure 9.18. The user wants to place an object at position (D). Figure 9.19 shows the parameter plots of the sequence. First the object is handed over to the robot. The contact force again is clearly visible in the blue external force plot 9.19c) at 223s.

After the robot returned to its idle configuration the user performs a *look down* gesture at 245s which corresponds to frame 1 in Figure 9.18. The gesture was correctly detected by the system, however, the user's gaze was incorrectly measured and the system wrongly assumed that the object was to be placed at position (E). The robot begins moving towards this position in frames 2-5 when the user realises the error and performs a *no* gesture at 263s (frame 6). The robot backs up and reaches the reconfiguration state at 272s (frame 8). It then approaches the proper target po-

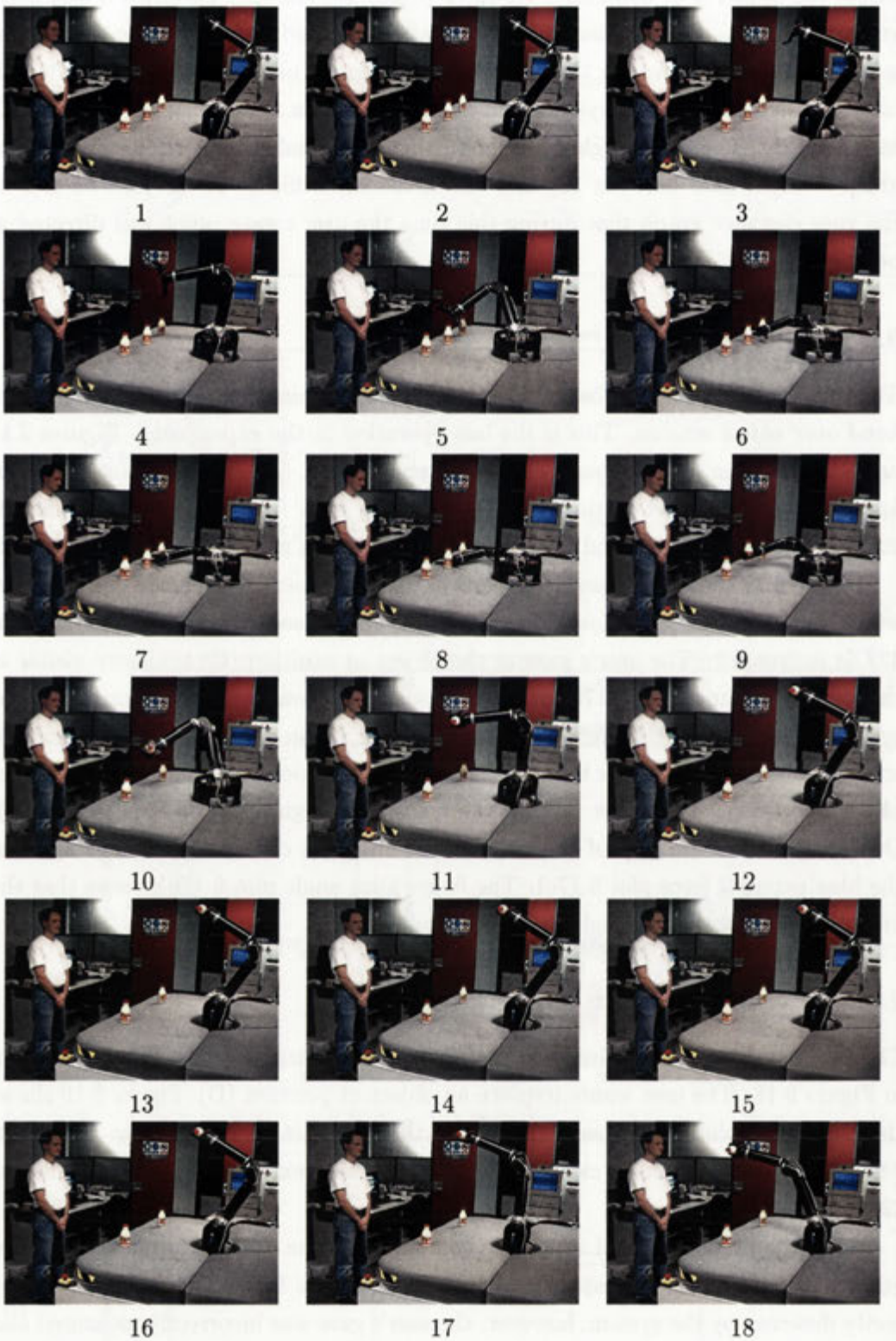


Figure 9.15: Pick object sequence (part 1)

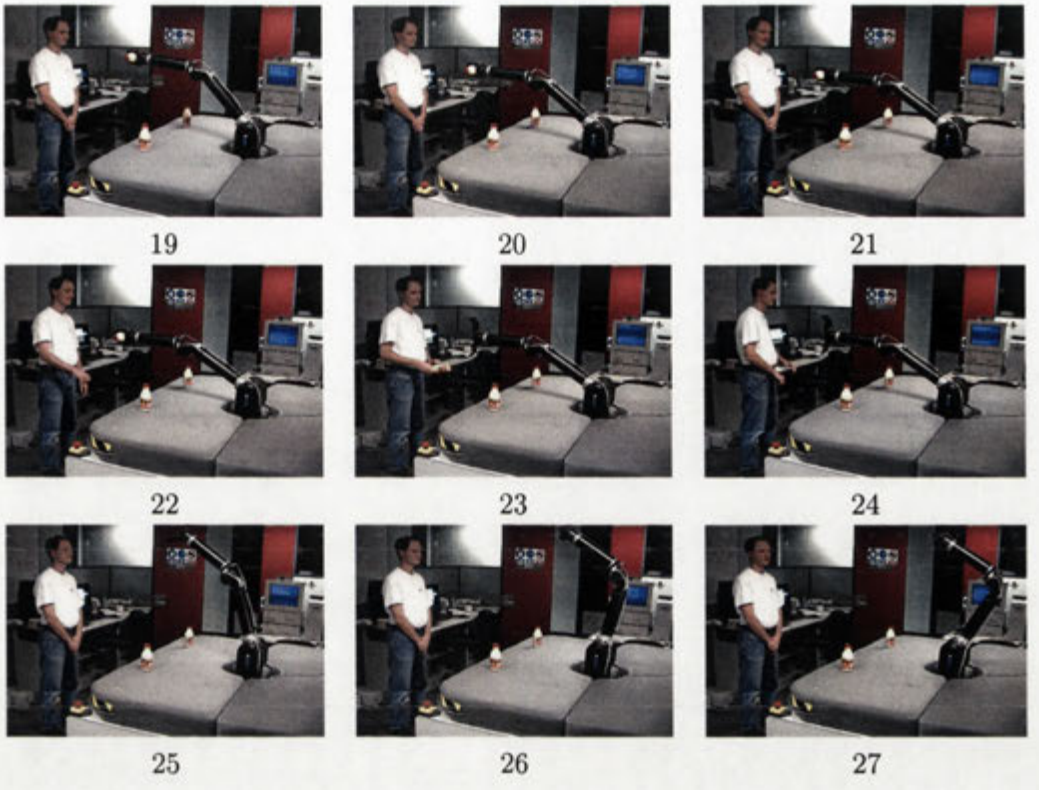


Figure 9.16: Pick object sequence (part 2)

sition in frames 9-10. Position (D) was selected as the *place object* position since it was the second closest to the measured gaze point when the command was originally given. The robot approaches position (D) but releases the object only after the user confirms the new position with a *yes* gesture at 286.0s (frame 14).

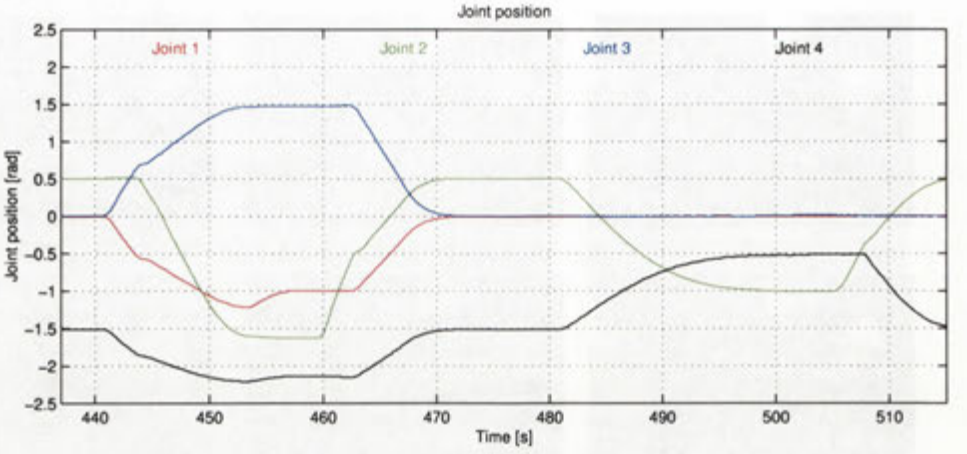
On completion of the corrected *place object* operation the robot returns into it's idle configuration in frames 16-18.

9.6 Summary

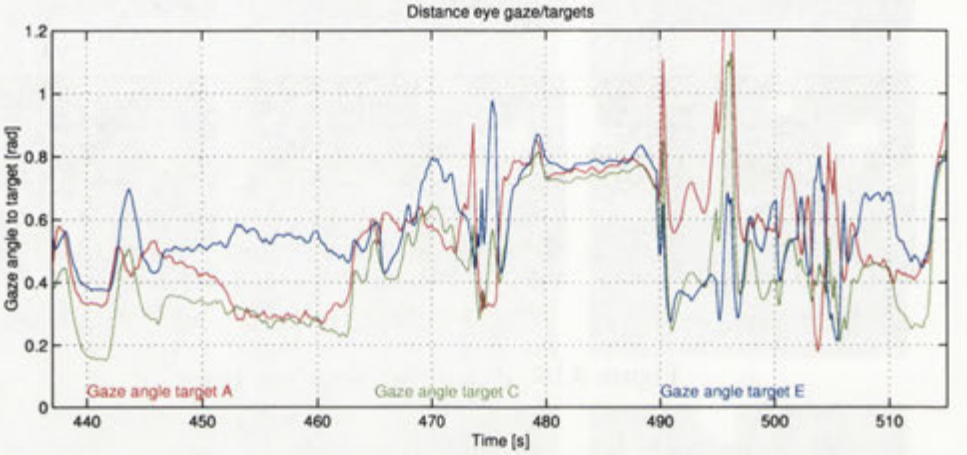
This chapter presented the results of the integration of the two primary components of the system, the visual 3D face and eye gaze tracking system and the human-friendly impact force controlled robot manipulator. An interactive pick-and-place system was selected as an example application which demonstrated that it is possible to develop robotic systems that are able to cooperate and interact safely with untrained human operators. The system allows a user to interact with the robot system through visual cues such a facial gestures and the gaze point, as well as through physical contact. A user was able to use the robot to pick and place objects on a desktop. All actions were initiated through facial gestures, while the gaze point measurements provide deictic cues for certain commands.

The system was controlled by a finite state machine (FSM) which used a vision

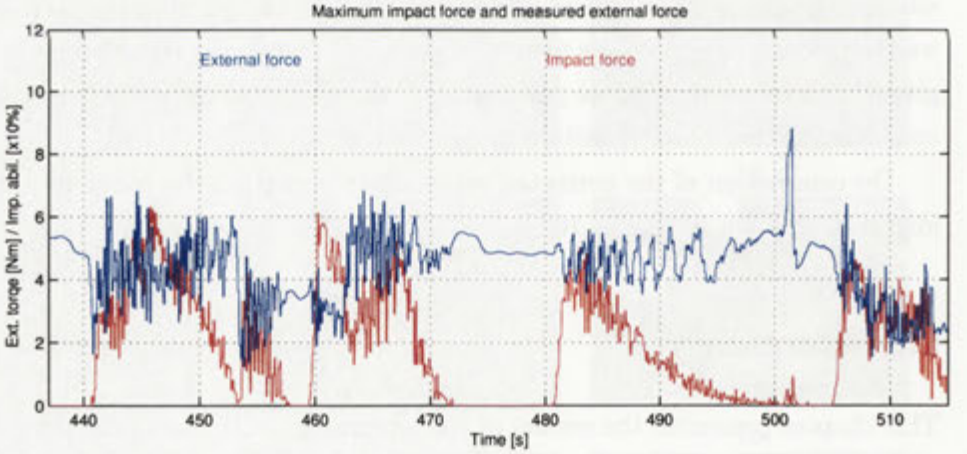
a)



b)



c)



d)

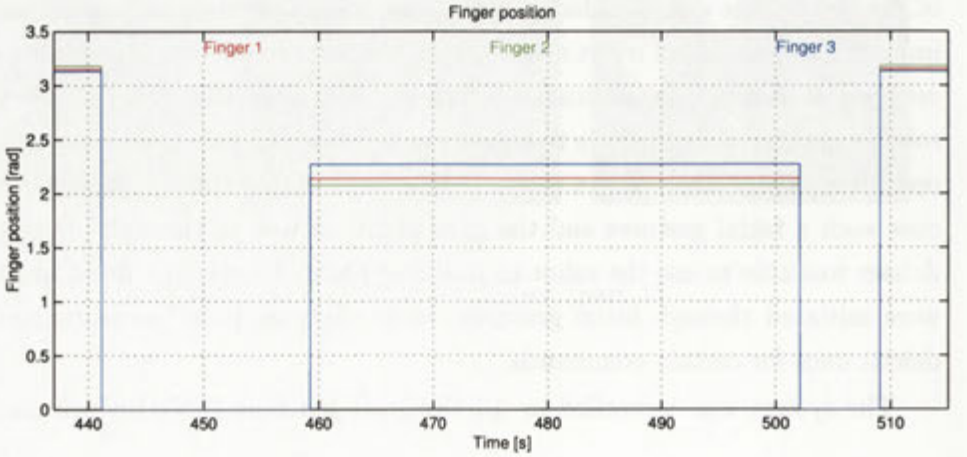


Figure 9.17: Pick object sequence

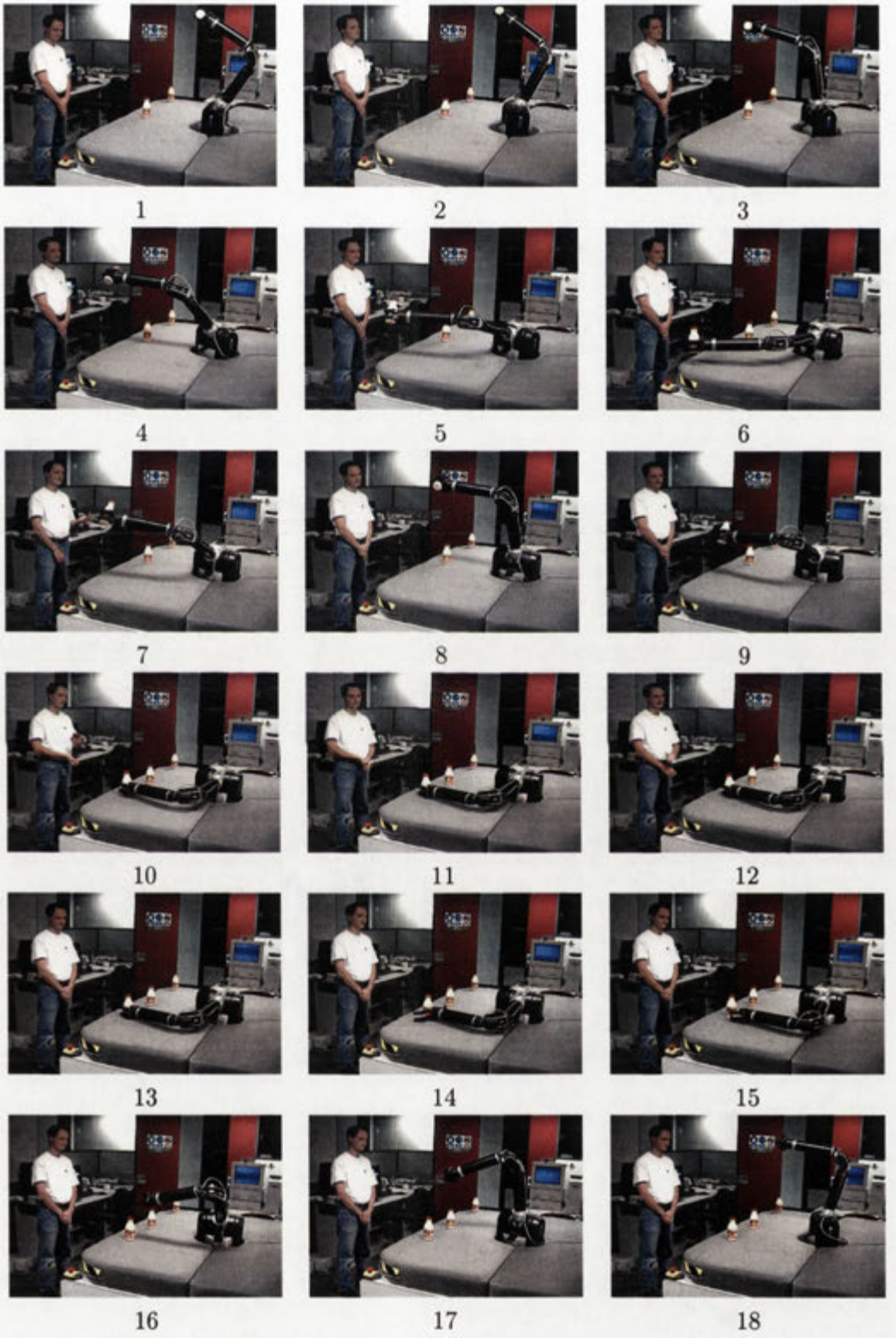


Figure 9.18: Error correction sequence

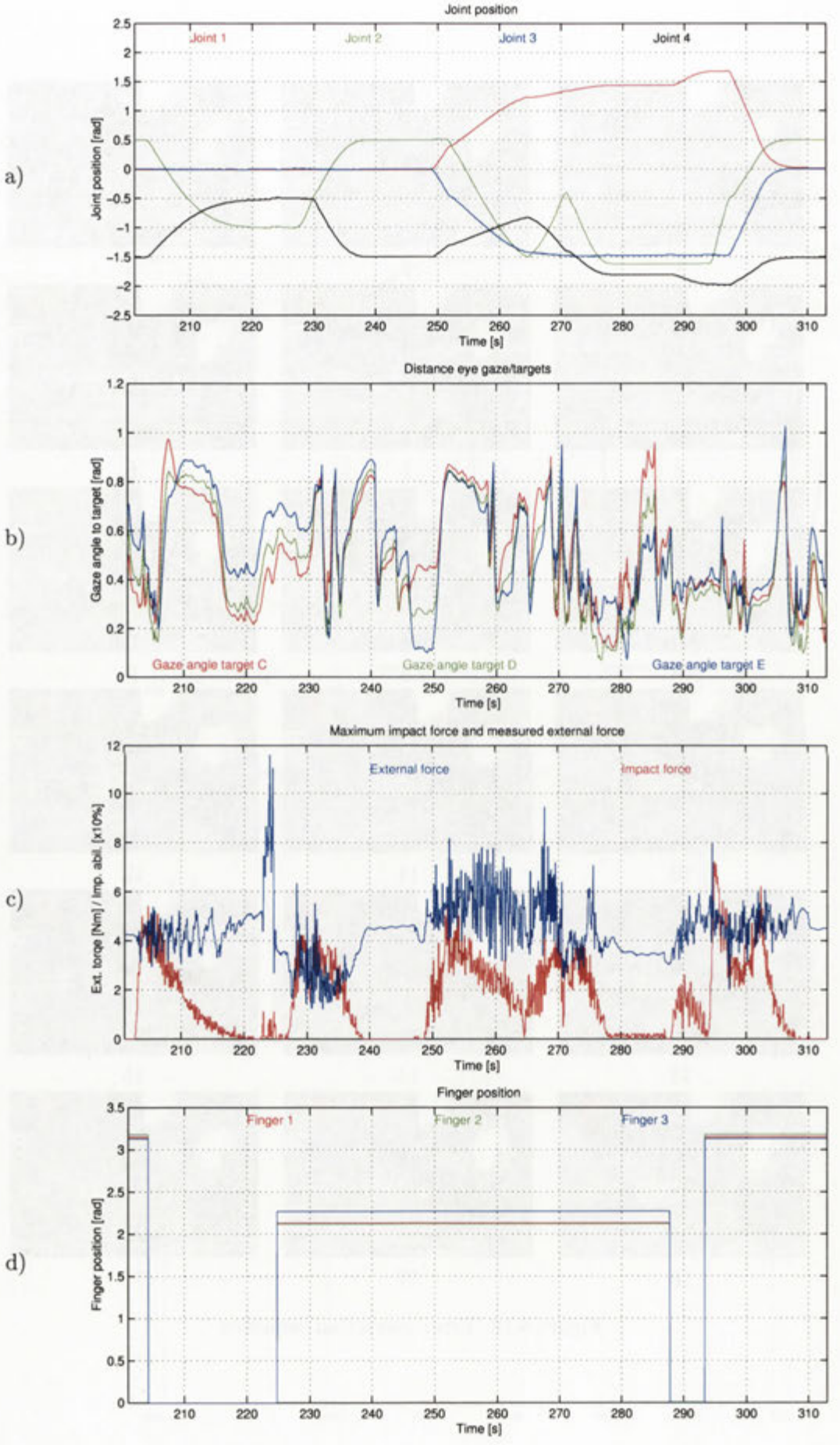


Figure 9.19: Error correction sequence

system and the robot as sensors. The FSM performed transitions according to the sensor data and the internal state of the FSM.

The interactive pick-and-place robot system was extensively tested in the laboratory. The results of a typical experiment with a duration of over 8 minutes was presented. This experiment included all the different interaction modes and robot actions that the system could perform. It showed that the system architecture was robust against tracking failures and against false measurements of the gaze point.

Overall the interactive pick-and-place robot represents a novel example of the application of visual interfaces to human-friendly robots. The implementation could have sufficient interaction modes for an application such as helping hands for the disabled. However, the system only provides a fraction of the interaction modes commonly used by humans. Obviously the inclusion of other interaction modes such as speech recognition, speech synthesis and other visual perception modes such as hand and body gestures and object recognition are required to achieve truly natural human-robot interaction. However, the prototype system presented in this research proves that it is possible to create robot systems with natural interfaces that are safe to use and can perform useful tasks under human supervision without the traditional requirement of an expert operator.

Chapter 10

Conclusions and Further Work

10.1 Conclusions

The goal of the research reported in this thesis was to contribute to the exciting new area of human-friendly robots. Emerging in the gap between teleoperated systems with high operator involvement and fully autonomous systems with low performance in unstructured environments, human-robot cooperation systems bring up two major issues: natural interaction interfaces and safety guarantees for an user. Without addressing these issues successfully no progress is possible in this research domain and, like their fully autonomous colleagues, human cooperation robots will remain inside research laboratories.

This thesis makes contributions in two research fields. The goal in the area if visual interfaces was to develop a robust and real-time capable visual 3D head pose and 3D eye gaze tracking system using only a monocular camera. The visual face and gaze tracking and gesture recognition system presented in Chapters 3-5 allowed an user to control computerised systems in a natural and intuitive way. The system uses template matching to identify a number of predefined facial features in the monocular images in order to track their 2D location in real time. A network of Kalman filters merges the 2D data of the various features according to the confidence of the correlation result. An improved version of Huttenlocher's 3-point alignment algorithm was developed to recover the 3D pose of the head. Based on this information the eye gaze vector of the user is calculated. A pattern matching algorithm recognises facial gestures from the tracking parameters of the head. The system is well suited as a human-machine interface. The gestures provide a way of invoking discrete commands while the gaze point can be used to reference objects and locations in the environment. The usefulness of the developed vision system was demonstrated in an application to control a robot manipulator. However, eye gaze tracking and gesture recognition technology is also applicable in a wide range of other human-machine interaction scenarios.

The safety of the human operator was identified as the second major issue in

human-friendly robotics. Specifically, quantifiable safety limits must be guaranteed by a human-friendly robot in order to have practical usefulness. This important issue was addressed by a novel control strategy for robot manipulators. The idea was to guarantee a limit on the impact force that any collision of the robot with static obstacles could generate. The impact force limit provides collision safety with an adjustable trade-off between impact force and the efficiency of the robot. Creating a motion bandwidth limit allows users to recognise a robot's actions and to react appropriately. This predictability of a robot's motion helps to prevent collisions with the user. The impact force limit and the limit of it's first derivative are integrated into a formulation based on the dynamic model of the robot which defines the portion of the joint torque vector space that fulfils both constraints. The resulting joint torque vector filter encapsulates the robot in a safety envelope that is independent of the control algorithm. The proposed control architecture was tested in simulations and implemented on a real robot system. Various experiments verified the feasibility and usefulness of the impact force control scheme.

The two modules, the visual interface and the impact force control strategy, were combined to develop an interactive pick-and-place system. This system allows non-experts to use the manipulator to pick and place objects in a desktop environment. Facial gestures were used to invoke the various actions of the robot while the user's eye gaze was used as a pointing device to identify objects and positions in the environment. The system also allowed physical interaction between the user and the robot. When objects were exchanged between the user and the robot, the robot sensed the forces exerted by the grip of the user onto the object and used this information to trigger the exchange operation. The two interaction modes were intuitive to the operator and allowed a natural human-robot interaction while the safety envelope guaranteed the safety of the user. The interactive pick-and-place system was a proof of concept of human-friendly robots. This work showed that human-robot cooperation systems are possible and can be implemented in a safe and intuitive way for operation by non-expert users. Possible application areas for such systems include

- Helping hands for the disabled
- Home and office robots
- Humanoid robots

10.2 Directions of future work

This thesis presented a fully operational real robot manipulator system that was controlled through a real-time vision system. The system fulfils the two major criteria

for human-friendly robots; an interface that allows natural interaction and a control strategy which provides quantifiable safety guarantees for the user.

Some of the shortcomings of the system have already been described in the previous chapters. Besides the various improvements that could be made to the algorithms and architectures developed in this thesis, the areas of visual interfaces and human-friendly robot control offer many possibilities to develop novel ideas, concepts and integrated systems.

10.2.1 Face and gaze tracking

The vision system was the first module that has been developed in this research. Some of algorithms, such as the Kalman filter network and the SAD template correlation, have been tailored to the processing capabilities of the original hardware setup (a 68030/25MHz + monochrome correlation hardware). With today's hardware a range of improvements have become feasible and some have already been implemented in other face and gaze tracking systems [Matsumoto and Zelinsky 2000] [Schwerdt and Crowley 2000].

At the lowest level more sophisticated feature detection algorithms should be used. Algorithms that include skin colour detection for gross face localisation, brightness normalised template correlation and Gabor filters will improve the tracking robustness and the ability of the system to recover after tracking failures. A systematic integration approach to multi-scale and multi-mode feature detection algorithms is required for the different visual cues [Triesch and von der Malsburg 2000] to derive an optimal unique solution from the different cues.

In Chapter 2 a variety of 3D pose estimation methods for faces have been reviewed. Feature based methods appear to have the most advantages. While the different feature based methods all provide satisfactory results, most require a specific setup and camera calibration. The system presented in this research used a monocular precalibrated (focal length and 3D pose) camera. Algorithms able to facilitate a variable number of cameras without the need for precise calibration would improve this situation. The system also requires the manual initialisation of the features, specifically the user must click on a number of features in images on a computer screen to identify their appearance. The 3D location of the features are predefined in a geometric face model which fits most individuals sufficiently but needs adaptation in some cases. This procedure is time consuming and requires expertise in the selection of appropriate features and should be automated. The acquisition of the 3D model and the feature locations should occur in parallel to the tracking based on the results of the image processing.

The vision system presented in this research project used template correlation to detect the iris for eye gaze estimation. To calculate the gaze direction more accurately, the iris/pupil should be located more precisely, for example using a Hough

transform [Matsumoto and Zelinsky 2000]. Also the 3D eye ball location and the precise diameter could be adapted online to improve the tracking results.

10.2.2 Safe robot control

The impact force control strategy was designed to guarantee safety for collisions of the manipulator with a human *at the moment* of impact. Only this situation is covered by the formulation. Although this approach does effectively provide a limitation during contact situations, the resulting limits depend on the selected impact force limit and motion bandwidth. To provide safety guarantees *during* contact an appropriate additional behaviour of the robot must first be defined and the corresponding control formulations must be derived. The external force sensing technique could then be used to switch between the two behaviours of the robot.

The control scheme developed in this research was based on the idea of a gravity compensated robot with low friction that can be controlled in a manner similar to an arm floating in zero gravity. The gravity compensation did not adapt to grasped objects, it was assumed the objects all had negligible mass. To be able to grasp heavier objects without requiring higher gain control the gravity compensation must adapt online after objects are grasped or released by the robot.

The experiments with the adapted PID controller and the safety envelope showed that it is difficult to achieve high performance by operating the manipulator close to the edge of the safety envelope. A better performance could be achieved by using the dynamic model of the robot to generate feed forward control signals which are responsible for generating the floating motion. A low gain PID controller may then be sufficient to generate the remaining control signals. Alternatively the complete control signal could be generated by a learning algorithm which learns to operate the manipulator close to the envelope. The distance to the envelope could be used as a feedback value for the learning algorithm.

To achieve superior performance on the WAM robot a good model of the torque ripple is required or torque sensors should be used on the robot's joint axes. Without precise knowledge and control of the joint torques the proposed techniques are difficult to implement.

10.2.3 Human-robot interaction

The interactive pick-and-place system represented a proof of concept implementation. The interaction through eye gaze, facial gestures and physical forces represents only a fraction of the interaction modes humans normally use. A truly natural and intuitive interface for a robot manipulator must incorporate a number of additional interaction modes. These modes include speech recognition and voice synthesiser systems and additional capabilities of the vision system. Full body tracking, body gesture recognition, in particular deictic hand gestures, and a scene camera for visual

localisation of objects in the environment would allow the robot system to perceive a more complete range of human communication modes. In this case a more sophisticated grasping planning system could be incorporated to allow the robot to manipulate a variety of different objects. With more articulated interaction modes such as speech and hand gestures a large range of commands could be provided to the robot.

10.3 Closing words

Human-robot cooperation systems pose a great challenge and opportunity for robotics to achieve one of the oldest dreams of mankind; Machines that are able to take up much of the tedious daily work and allow people to do things that are more enjoyable and worthwhile. Even though such robots still only exist in the realms of science fiction and may remain there for some time to come, in the end, it is this vision that is the driving force behind the development of human-friendly robots.

Appendix A

Asimov's Laws of Robotics

The following is an excerpt from the official Asimov-FAQ of the alt.books.isaac-asimov newsgroup. It was compiled by Edward Seiler (edseiler@clark.net) and John H. Jenkins (tseng@blueneptune.com) and contains vast resources on the life and work of Isaac Asimov.

The Three Laws of Robotics are:

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 the 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.

From the book *Handbook of Robotics, 56th Edition, 2058 A.D.*, as quoted in Asimov's *I, Robot.*

In the book *Robots and Empire* (ch. 63), the "Zeroth Law" is extrapolated, and the other Three Laws modified accordingly: 0. A robot may not injure humanity or, through inaction, allow humanity to come to harm. Unlike the Three Laws, however, the Zeroth Law is not a fundamental part of robotic engineering, is not part of all robots, and, in fact, requires a very sophisticated robot to even accept it.

Asimov claimed that the Three Laws were originated by John W. Campbell in a conversation they had on December 23, 1940. Campbell in turn maintained that he picked them out of Asimov's stories and discussions, and that his role was merely to state them explicitly.

The Three Laws did not appear in Asimov's first two robot stories, "Robbie" and "Reason", but the First Law was stated in Asimov's third robot story "Liar!", which also featured the first appearance of robopsychologist Susan Calvin. (When

"Robbie" and "Reason" were included in *I, Robot*, they were updated to mention the existence of the first law and first two laws, respectively.) Yet there was a hint of the three laws in "Robbie", in which Robbie's owner states that "He can't help being faithful, loving, and kind. He's a machine - made so." The first story to explicitly state the Three Laws was "Runaround", which appeared in the March 1942 issue of Astounding Science Fiction.

Appendix B

Commercial Vision Systems

Recently a number of companies have started to market face and gaze tracking technologies for various application areas. This appendix give a brief overview over the major companies in this market.

Seeing Machines

This startup has been founded recently to commercialise the vision technology developed at The Australian National University. The systems are unique in that they derive the eye gaze direction of the user without corneal reflections, and therefore allow free head motions. More details can be found at `http://www.seeingmachines.com`.

Applied Science Laboratory (ASL)

ASL is one of major players in the area of eye gaze detection. The company has developed a number of systems with and without head gear. More details can be found at `http://www.a-s-l.com`.

Eyematic

Eyematic is a startup company which is commercialising head pose and eye gaze tracking technology originally developed in a collaboration between the University of Bochum in Germany and the University of Souther California [Maurer and von der Malsburg 1996]. The company appears to be in the development phase. More details can be found at `http://www.eyematic.com`.

IBM

The Almaden Research Center of IBM conducts research into human-computer interfaces. The *Blue Eyes* project uses a corneal reflection based eye gaze tracker embedded in a larger system which aims to develop advanced interaction methods for desktop computers. More information on the Blue Eyes project can be found at `http://www.almaden.ibm.com/cs/blueeyes/`.

IScan

Iscan develops a number of eye gaze tracking products, most notably the ETL-400 system which uses corneal reflection. The product is aimed at the automotive sector for safety applications and human-performance/ergonomic measurements. More information on IScan can be found at `http://www.iscaninc.com`.

SMI

The Berlin-based company offers a range of optical systems. One of these products is the *Eye Mouse*, an active corneal reflection based eye gaze tracking system which allows considerable head motion. More details on the Eye Mouse and other products can be found at `http://www.smi.de/em/`.

Appendix C

Huttenlocher's Alignment algorithm

The well known formulas of the Huttenlocher's alignment algorithm can be found in [Huttenlocher and Ullman 1990] and are stated in this chapter for completeness.

C.1 The alignment algorithm in a nutshell

The raw image locations $\text{img}(\mathbf{p}_i)$ are shifted in the image plane such that the new image coordinates $\mathbf{q}_1$ of $\mathbf{p}_1$ coincide with the origin of the image plane.

$$\mathbf{q}_1 = \text{img}(\mathbf{p}_1) - \text{img}(\mathbf{p}_1) = 0 \tag{C.1}$$

$$\mathbf{q}_2 = \text{img}(\mathbf{p}_2) - \text{img}(\mathbf{p}_1) \tag{C.2}$$

$$\mathbf{q}_3 = \text{img}(\mathbf{p}_3) - \text{img}(\mathbf{p}_1) \tag{C.3}$$

This shifting operation is compensated for in the end. The top left 2×2 submatrix L of the rotation matrix R which describes the orientation of the object conforms to the following equation.

$$L = Q \cdot M^{-1} \tag{C.4}$$

where $Q \in \mathbb{R}^{2\times2}$ contains the image positions $\mathbf{q}_2$ and $\mathbf{q}_3$ in it's columns and $M \in \mathbb{R}^{2\times2}$ contains the model points in it's columns. Choosing $\mathbf{e}_1, \mathbf{e}_2 \in \mathbb{R}^2$ to be $(1,0)$ and $(0,1)$, k_1 and k_2 become

$$k_1 = -\begin{pmatrix} l_{11} \\ l_{21} \end{pmatrix} \cdot \begin{pmatrix} l_{12} \\ l_{22} \end{pmatrix} \tag{C.5}$$

$$k_2 = \left\| \begin{pmatrix} l_{11} \\ l_{21} \end{pmatrix} \right\| \cdot \left\| \begin{pmatrix} l_{12} \\ l_{22} \end{pmatrix} \right\| \tag{C.6}$$

$$c_1 = \pm\sqrt{\frac{1}{2}\left(k_2 + \sqrt{k_2^2 + 4k_1^2}\right)} \tag{C.7}$$

$$c_2 = \begin{cases} \frac{k_1}{c_1} & c_1 \neq 0 \\ \pm\sqrt{-k_2} & c_1 = 0 \end{cases} \tag{C.8}$$

$$s = \sqrt{l_{11}^2 + l_{21}^2 + c_1^2} = \sqrt{l_{12}^2 + l_{22}^2 + c_2^2} \tag{C.9}$$

$$R = \frac{1}{s}\begin{bmatrix} l_{11} & l_{12} & \frac{1}{s}(c_2 l_{21} - c_1 l_{22}) \\ l_{21} & l_{22} & \frac{1}{s}(c_1 l_{12} - c_2 l_{11}) \\ c_1 & c_2 & \frac{1}{s}(l_{11} l_{22} - l_{12} l_{21}) \end{bmatrix} \tag{C.10}$$

$$\mathbf{d} = \frac{1}{s}\begin{bmatrix} \mathrm{img}(\mathbf{p}_1).x \\ \mathrm{img}(\mathbf{p}_1).y \\ f \end{bmatrix} \tag{C.11}$$

C.2 Closed form solution

The image points are generated by applying the homogeneous transformation matrix $\mathbf{M}_1(\theta_x, \theta_y, \theta_z, 0, 0, d)$ to the general model points p_1, p_2 and p_3 and projection the resulting coordinates into the image plane using the pinhole camera model. The resulting image coordinates are ip_1, ip_2 and ip_3.

$$ip_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{C.12}$$

$$ip_2 = f\begin{bmatrix} \frac{\cos\theta_z \cos\theta_y p_2[1] + (-\sin\theta_z \cos\theta_x + \cos\theta_z \sin\theta_y \sin\theta_x) p_2[2]}{-\sin\theta_y p_2[1] + \cos\theta_y \sin\theta_x p_2[2] + d} \\ \frac{\sin\theta_z \cos\theta_y p_2[1] + (\cos\theta_z \cos\theta_x + \sin\theta_z \sin\theta_y \sin\theta_x) p_2[2]}{-\sin\theta_y p_2[1] + \cos\theta_y \sin\theta_x p_2[2] + d} \end{bmatrix} \tag{C.13}$$

$$ip_3 = f\begin{bmatrix} \frac{(\cos\theta_z \cos\theta_y p_3[1] + (-\sin\theta_z \cos\theta_x + \cos\theta_z \sin\theta_y \sin\theta_x) p_3[2])}{-\sin\theta_y p_3[1] + \cos\theta_y \sin\theta_x p_3[2] + d} \\ \frac{(\sin\theta_z \cos\theta_y p_3[1] + (\cos\theta_z \cos\theta_x + \sin\theta_z \sin\theta_y \sin\theta_x) p_3[2])}{-\sin\theta_y p_3[1] + \cos\theta_y \sin\theta_x p_3[2] + d} \end{bmatrix} \tag{C.14}$$

Huttenlocher and Ullman's algorithm calculates the rotation matrix R from coordinates ip_1, ip_2 and ip_3 which is integrated into the pose M. The symbolic expression for R can be found in equation C.15 (for $q_16 \neq 0$).

$$R = \begin{bmatrix} \frac{q_6}{\sqrt{q_{17}}} & \frac{q_{12}}{\sqrt{q_{17}}} & \pm\frac{\frac{(-q_6 q_{12} - q_{11} q_{10}) q_{11}}{\sqrt{q_{16}}} - \sqrt{q_{16}} q_{10}}{q_{17}} \\ \frac{q_{11}}{\sqrt{q_{17}}} & \frac{q_{10}}{\sqrt{q_{17}}} & \pm\frac{\sqrt{q_{16}} q_{12} - \frac{(-q_6 q_{12} - q_{11} q_{10}) q_6}{\sqrt{q_{16}}}}{q_{17}} \\ \pm\frac{\sqrt{q_{16}}}{\sqrt{q_{17}}} & \pm\frac{-q_6 q_{12} - q_{11} q_{10}}{\sqrt{q_{17}}\sqrt{q_{16}}} & \frac{q_6 q_{10} - q_{12} q_{11}}{q_{17}} \end{bmatrix} \tag{C.15}$$

where

$$q_1 = -\sin\theta_y p_3[1] + \cos\theta_y \sin\theta_x p_3[2] + d$$

$$
\begin{aligned}
q_2 &= -\sin\theta_z \cos\theta_x + \cos\theta_z \sin\theta_y \sin\theta_x \\
q_3 &= \cos\theta_z \cos\theta_y p_3[1] + q_2 p_3[2] \\
q_4 &= -\sin\theta_y p_2[1] + \cos\theta_y \sin\theta_x p_2[2] + d \\
q_5 &= \cos\theta_z \cos\theta_y p_2[1] + q_2 p_2[2] \\
q_6 &= \frac{f q_5 p_3[2]}{q_4(p_2[1]p_3[2] - p_3[1]p_2[2])} - \frac{f q_3 p_2[2]}{q_1(p_2[1]p_3[2] - p_3[1]p_2[2])} \\
q_7 &= \cos\theta_z \cos\theta_x + \sin\theta_z \sin\theta_y \sin\theta_x \\
q_8 &= \sin\theta_z \cos\theta_y p_3[1] + q_7 p_3[2] \\
q_9 &= \sin\theta_z cos\theta_y p_2[1] + q_7 p_2[2] \\
q_{10} &= \frac{f q_9 p_3[1]}{q_4(p_2[1]p_3[2] - p_3[1]p_2[2])} + \frac{f q_8 p_2[1]}{q_1(p_2[1]p_3[2] - p_3[1]p_2[2])} \\
q_{11} &= \frac{f q_9 p_3[2]}{q_4(p_2[1]p_3[2] - p_3[1]p_2[2])} - \frac{f q_8 p_2[2]}{q_1(p_2[1]p_3[2] - p_3[1]p_2[2])} \\
q_{12} &= \frac{f q_5 p_3[1]}{q_4(p_2[1]p_3[2] - p_3[1]p_2[2])} + \frac{f q_3 p_2[1]}{q_1(p_2[1]p_3[2] - p_3[1]p_2[2])} \\
q_{13} &= \sqrt{q_6^2 + q_{11}^2} \\
q_{14} &= \sqrt{q_{10}^2 + q_{12}^2} \\
q_{15} &= \sqrt{(q_{14} - q_{13})^2 + 4(-q_6 q_{12} - q_{10} q_{11})^2} \\
q_{16} &= \frac{1}{2} q_{14} - \frac{1}{2} q_{13} + \frac{1}{2} q_{15} \\
q_{17} &= q_6^2 + q_{11}^2 + q_{16}
\end{aligned}
$$

The result contains a large number of parameters. The expressions for these parameters are rather complex. This makes determining accurate bounds of the systematic error and the sensitivity difficult. Only rough approximations of the bounds have been derived [Grimson et al. 1992].

Appendix D

Gravity Compensation

The compensation of the joint torques induced by gravity is a central prerequisite for the Impact Force Control scheme.

D.1 Gravity compensation on a serial manipulator

Figure D.1 shows the kinematic configuration and dimensions of the Barrett Whole Arm Manipulator (WAM) as well as the location and orientation of the coordinate frames assigned to each link. Frame 0 is located at the intersection of the first three joint axes J_{1-3} and is oriented such that the z-axis is coincident with joint axis J_1 and and the y-axis is aligned with joint axis J_2. The following frames are assigned using the Denavit-Hartenberg convention.

The corresponding Denavit-Hartenberg transformation matrices are

$$T_1^0 = \begin{bmatrix} \cos(\theta_1) & -\sin(\theta_1) & 0 & 0 \\ \sin(\theta_1) & \cos(\theta_1) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad T_2^1 = \begin{bmatrix} \cos(\theta_2) & -\sin(\theta_2) & 0 & 0 \\ 0 & 0 & -1 & 0 \\ \sin(\theta_2) & \cos(\theta_2) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_2^3 = \begin{bmatrix} \cos(\theta_3) & -\sin(\theta_3) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -\sin(\theta_3) & -\cos(\theta_3) & 0 & L_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad T_3^4 = \begin{bmatrix} \cos(\theta_4) & -\sin(\theta_4) & 0 & L_2 \\ 0 & 0 & -1 & 0 \\ \sin(\theta_4) & \cos(\theta_4) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_4^5 = \begin{bmatrix} \cos(\theta_5) & -\sin(\theta_5) & 0 & -L_2 \\ 0 & 0 & 1 & L_3 \\ -\sin(\theta_5) & -\cos(\theta_5) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad T_5^6 = \begin{bmatrix} \cos(\theta_6) & -\sin(\theta_6) & 0 & 0 \\ 0 & 0 & -1 & 0 \\ \sin(\theta_6) & \cos(\theta_6) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_6^7 = \begin{bmatrix} \cos(\theta_7) & -\sin(\theta_7) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -\sin(\theta_7) & -\cos(\theta_7) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

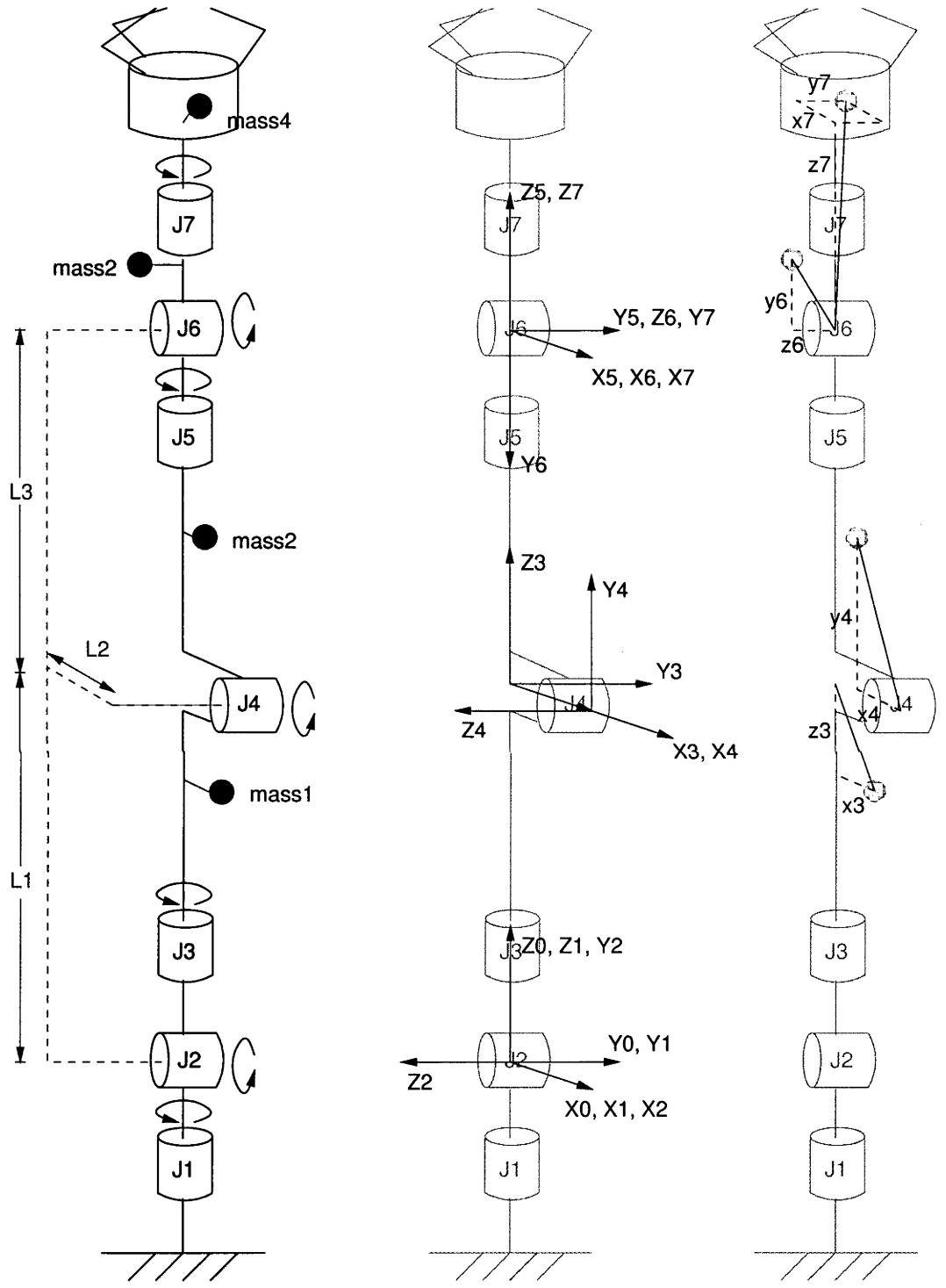


Figure D.1: Kinematic configuration and dimensions of the robot, the position of the coordinate systems and the location of the centre of mass of the links.

where the geometric constants L_{1-3} are known.

Figure D.1 also shows the location of the masses as blue dots which are considered for the gravity compensation. Since axis 1 is mounted vertically the link between axis 1 and 2 need not be considered for the gravity compensation. The next link between axis 2 and 3 is almost symmetric and the centre of mass is close to the joint axis and can also be neglected. This assumption is also valid for the link between axis 5 and 6 which therefore is incorporated in m_2. All other links are modelled as point masses m_{1-4}. The location of m_1 is expressed with respect to frame 3. Since the link is symmetric with respect to the XZ-plane of frame 3 the y-coordinate of the centre of mass is assumed to be 0. The same assumption is valid for m_2 which lies in the XY-plane of frame 4. The link between joint 6 and 7 modelled as m_3 carries the motor of joint 7 on the side but is symmetric with respect to the YZ-plane of frame 6. The hand is modelled as m_4 and no assumptions are made about its location. This also allows to model grasped objects as a part of this link.

To determine the torques required to compensate for the gravitational forces on the robot's links, the Lagrange method is used. The first step is to determine the height (z-coordinate with respect to frame 0) of the masses m_{1-4} as a function of the joint angles θ_{1-7}.

$$h_1 = \begin{pmatrix} 0 & 0 & 1 & 0 \end{pmatrix} T_0^1 T_1^2 T_2^3 \begin{pmatrix} x_3 & 0 & z_3 & 1 \end{pmatrix}^t \tag{D.1}$$

$$h_2 = \begin{pmatrix} 0 & 0 & 1 & 0 \end{pmatrix} T_0^1 T_1^2 T_2^3 T_3^4 \begin{pmatrix} x_4 & y_4 & 0 & 1 \end{pmatrix}^t \tag{D.2}$$

$$h_3 = \begin{pmatrix} 0 & 0 & 1 & 0 \end{pmatrix} T_0^1 T_1^2 T_2^3 T_3^4 T_4^5 T_5^6 \begin{pmatrix} 0 & y_6 & z_6 & 1 \end{pmatrix}^t \tag{D.3}$$

$$h_4 = \begin{pmatrix} 0 & 0 & 1 & 0 \end{pmatrix} T_0^1 T_1^2 T_2^3 T_3^4 T_4^5 T_5^6 T_6^7 \begin{pmatrix} x_7 & y_7 & z_7 & 1 \end{pmatrix}^t \tag{D.4}$$

The potential energy U of the robot can now be determined

$$U = \prod_{i=1}^{4} m_i h_i \tag{D.5}$$

According to Lagrange's method, the joint torques τ_{1-7} required to compensate for the gravitational load can be derived as the partial derivative of the potential energy U with respect to the respective joint angle τ_i.

$$\tau_i = \frac{\partial U}{\partial \theta_i} \tag{D.6}$$

The vector $\tau = (\tau_1, \ldots, \tau_7)^t$ provides the torques required to create the 'floating'-behaviour of the robot.

D.2 Parameter identification

To calculate the vector of gravity compensation torques τ the introduced constants including the 4 masses and their position parameters must be identified. This system identification process is performed using an iterative least square estimator.

Each of the torques τ_i is given by a sum of products where each product contains a number of trigonometric functions of the joint angles τ_{1-7} and a subset of the constants to be identified which always appear linearly. The first step is to separate the trigonometric part from the unknowns and express τ as the product of a matrix Γ containing the trigonometric parts and the vector u of unknowns.

$$\tau = \Gamma u = \begin{bmatrix} sc_{1,1} & \cdots & sc_{1,n} \\ \vdots & & \vdots \\ sc_{7,1} & \cdots & sc_{7,n} \end{bmatrix} \begin{bmatrix} u_1 \\ \vdots \\ u_n \end{bmatrix} \tag{D.7}$$

The $sc_{i,j}$ are products of sin's and cos's of the joint angles τ_{1-7} and the u_i are products of the constants to be identified. In general the values of the unknowns u_i do not allow the identification of the individual factors of the u_i, however, the values of the products are sufficient to calculate the desired torques. For the kinematic model shown in Figure D.1 the vector u consists of 12 elements, and thus, Γ is a 7×12 matrix.

$$u = [m_1x_3, m_1z_3, m_2, m_2x_4, m_2y_4, m_3, m_3y_6, m_3z_6, m_4, m_4x_7, m_4y_7, m_4z_7]^t \tag{D.8}$$

If a least squares algorithm is to be used, the columns of T need to be linearly independent. Otherwise the least squares solution is unstable, or in the case of precise values, the solution is a space with the dimension of the number of dependencies. The matrix derived from the model of the WAM contains 4 dependencies. This means that the respective columns $sc_{*,i}$ of Γ are linear dependant with respect to the remaining columns. To remove the dependencies for the system of equations the respective unknowns are combined.

Some of the dependencies are intuitively understandable. For example, the moments created by m_3y_6 and m_4z_7 always act together on the joints since y_6 and z_7 are aligned for all configurations of the robot. Therefore, they can not be identified individually. The linear dependencies are as follows

$$sc_{*,3} = L_2sc_{*,1} + L_1sc_{*,2} \tag{D.9}$$

$$sc_{*,6} = L_2sc_{*,1} + L_1sc_{*,2} - L_2sc_{*,4} + L_3sc_{*,5} \tag{D.10}$$

$$sc_{*,9} = L_2sc_{*,1} + L_1sc_{*,2} - L_2sc_{*,4} + L_3sc_{*,5} \tag{D.11}$$

$$sc_{*,12} = sc_{*,7} \tag{D.12}$$

There are various ways to reduce the number of unknowns to 8 using the previously described dependencies. One solution is the vector u' in which the elements 3, 6, 9 and 12 of the original vector u are integrated into the remaining elements.

$$u' = \begin{bmatrix} m_1x_3 + L_2(m_2 + m_3 + m_4) \\ m_1z_3 + L_1(m_2 + m_3 + m_4) \\ m_2x_4 - L_2(m_3 + m_4) \\ m_2y_4 + L_3(m_3 + m_4) \\ m_3y_6 + m_4z_7 \\ m_3z_6 \\ m_4x_7 \\ m_4y_7 \end{bmatrix} \tag{D.13}$$

The corresponding matrix Γ' is derived by deleting the 3rd, 6th, 9th and 12th column in Γ. The resulting matrix has dimension 7×8 and contains no further linear dependencies between rows. The the value of the 12 unknown constants in u' can not be identified individually.

D.3 4DOF configuration

By replacing the wrist part of the WAM arm with a non-articulated part (essentially a pipe with mounting features) the system can be reduced to 4DOF maintaining the same reach and appearance of the arm. This 4DOF-configuration is desirable for experiments that do not require the full dexterity of the WAM since it reduces the complexity of the system significantly. The update speed of the control cycles is increased and real-time operation can be achieved more easily. The Zero-G calculations for the 4DOF-configuration are also more simple.

The kinematic description is derived by removing the last 3 coordinate system transformation matrices and the related unknowns from the system. The second link of the arm becomes one point mass m_2 with location (x_4, y_4, z_4). The resulting vector of unknowns u'_4 is

$$u'_4 = [m_1x_3 + L_2m_2, m_1z_3 + L_1m_2, m_2x_4, m_2y_4, m_2z_4]^t \tag{D.14}$$

The components of u'_4 can be estimated using the same least squares estimation scheme that is used for the 7DOF configuration.

D.4 Least squares estimation

The vector of unknowns u' can be estimated using an iterative least squares estimator. The robot is held into various configurations using PID control loops for the joint angles. Then the average torque required to hold the robot in position is calculated

and used as the measured Zero-G torque τ. Γ can be calculated directly from the joint angles. Each measurement produces 7 equations from the 7 joints. The recursive equations of the estimator are updated for each row of Γ and the respective τ_i

$$k_{n+1} = \frac{P_n \Gamma_{1,*}}{1 + \Gamma_{1,*} P_n \Gamma_{1,*}} \tag{D.15}$$

$$\hat{u}_{n+1} = \hat{u}_n + k_{n+1}(\tau_i - \Gamma_{1,*}\hat{u}_n) \tag{D.16}$$

$$P_{n+1} = (I - k_{n+1}\Gamma_{1,*})P_n \tag{D.17}$$

where k_n is the gain vector in the n-th iteration, P_n is the covariance matrix of the estimate of the unknown vector $\hat{u}_n$ and I is the identity matrix. Before the first iteration ($n = 0$) P_0 is chosen large. The measurements from 2 different configurations of the robot are sufficient to constrain to obtain an estimate $\hat{u}$. However, in practise it is necessary to collect measurements from many points to reduce the effect of noise caused by friction and torque ripple in the mechanical system. It is therefore desirable to run the estimation process continuously during operations of the robot. Whenever the robot is stationary for a sufficiently long time to obtain a stable average for the torque vector the estimate of the parameters can be updated. This also allows the use of the Zero-G behaviour when objects are handled. If the control system has reason to assume that the parameters have changed, for example when an object is grasped or released, it is advisable to reset the covariance matrix $P = P_0$ to speed up the adaption to the changed configuration.

Figure D.2 shows the estimation process of the five unknowns required for the 4DOF configuration of the robot. The large scale variations disappear after only approximately 1000 measurements, while it takes roughly 100,000 measurements for the estimates to sufficiently converge to achieve a satisfactory result.

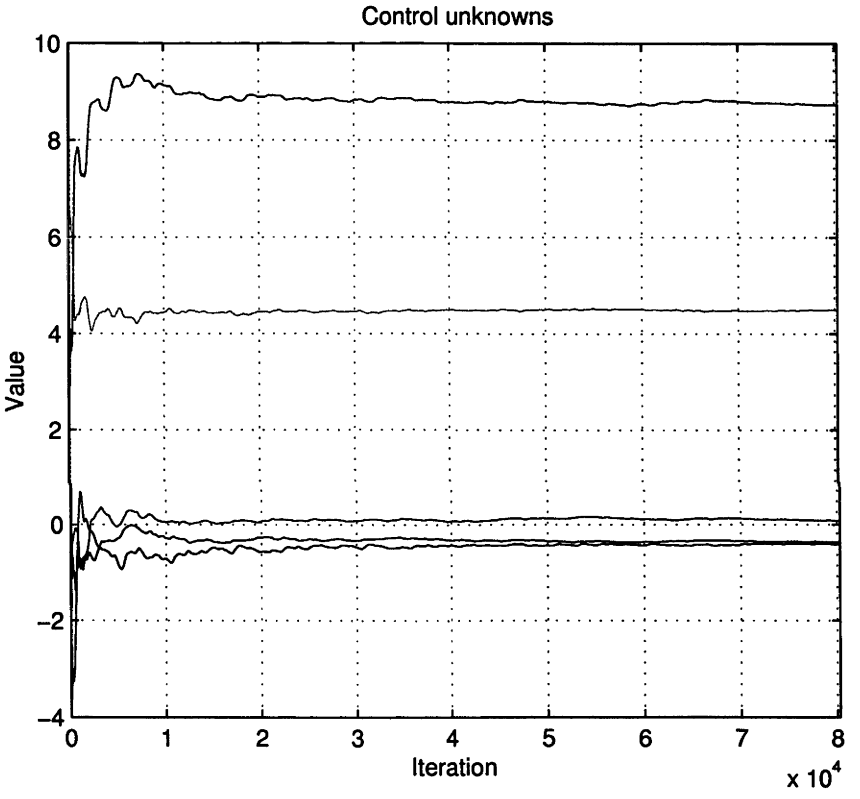


Figure D.2: Unknown estimation for the 4DOF configuration.

Appendix E

Video Index

This thesis reported on many experimental results that are best viewed as a video sequence. Located in the back cover of this thesis is a CD-ROM which contains a MPEG movie file with many of the results presented in this thesis. Similar to the thesis the videos are divided into three groups, the clips that demonstrate the visual interface, the clips regarding the safe control scheme for robot manipulators and finally the integration of the two subsystems.

E.1 Visual interface

The video clips between 0:21min and 5:06min present experiments for the visual face tracking system.

E.1.1 3D Head pose tracking

The first clip shows the visual face tracking system described in Chapter 3. The video overlay shows the facial feature positions measured by the system. Each feature is marked with a "pin" which changes its 3D orientation according the head pose. The pins of features viewed from the side are the longest while the pins of features parallel to the image plane only show the head. The person leaves the camera view several times and and returns soon afterwards. The system quickly recovers the facial feature positions since the person becomes visible again.

The second clip shows a similar experiment using a different video overlay display. Instead of the feature positions the triangles of the three feature triplets used by the system for the 3D pose estimation are displayed. The triangles are normally displayed in green and turn red when the head is rotated such that the triangle is viewed from the back side. In this case the pose estimate from the respective feature triplet is disregarded.

The clip starting at 1:53min shows the mechanism of the mannequin robot MARITA. The system consists of a 2DOF pan-and-tilt unit holding a mannequin

Start	End	Content
0:00	0:21	Video title
Visual interface experiments		
0:21	1:11	Face tracking with feature position display
1:11	1:53	Face tracking with feature triplet display
1:53	2:03	Mechanism of the mannequin robot *MARITA*
2:03	3:15	Pose estimation accuracy measurement with MARITA
3:15	3:57	Gesture recognition / eye motion detection
3:57	4:11	Experimental setup of the gaze point evaluation
4:11	5:06	Gaze point estimation
Safe robot control experiments		
5:06	6:30	Zero-G demonstration
6:30	6:54	Safety core motion damping
6:54	7:50	Collision experiment with a static obstacle
7:50	8:02	Setup the plasticine impact experiment
8:02	8:26	Closeup of a collision of the robot with the impactor
8:26	9:00	Plasticine impact experiments
9:00	9:40	Physical human-robot intervention
System integration		
9:40	10:01	Introduction to the interactive pick-and-place system
10:01	19:00	Experiments with the interactive pick-and-place system

Table E.1: Video index

head. The system allows precise positioning and therefore precise measurements of the accuracy of the visual pose estimation system with real image data.

The following clip (2:03min-3:15min) shows an excerpt from the experimental evaluation of the accuracy of the vision system described in Chapter 4. The head is moved about continuously stopping only briefly at different configurations. The scales on the borders show both the exact and measured rotations around x-, y- and z-axis. The true position of the mechanism is indicated by the red gauges, the measured position by the green gauges.

E.1.2 Gesture recognition

The video clip from 3:15min to 3:57min demonstrates the function of the facial gesture recognition system. All gestures used in the human-robot interaction system are demonstrated including additional gestures such as *maybe*. The recognised gestures are displayed in the bottom left corner of the image. The red squares in the video overlay show the measured orientation of the eyes with respect to the head. The gesture recognition system is described in Chapter 5.

E.1.3 Eye gaze point estimation

The remaining two video clips (3:57min-4:11min and 4:11-5:06min) in the visual interface section show the experimental setup and experimental results of the eye

gaze point estimation. The test subject is seated in front of the camera and a board with the six letters A-F attached to it. The person looks at the letters one after another while the vision system tracks the eye gaze point. The video overlay display shows the six letters and a cross to mark the current eye gaze point of the person. When the results of the eye gaze estimation provide sufficient evidence that the person is looking at a particular letter, the letter is highlighted. The experiment was performed twice, once with natural head motions which requires the system to incorporate both the 3D head pose and the eye ball orientation estimation and once with head motion only when the person looks straight ahead.

E.2 Safe robot control

The safe robot control section from 5:06min to 9:40min presents the various aspects of the impact force control scheme described in Chapters 7 and 8.

E.2.1 Basic behaviour: Zero-G

The basic behaviour of the robot is to compensate for its own weight. This passive behaviour allows person to move the robot with ease by pulling and pushing the links. The first clip (5:06min to 6:30) shows a person interacting with the robot in this mode. It is apparent how easily the robot can be manipulated in this mode and that this behaviour is both predictable and non-threatening to a person. Only at the end of the clip when the robot is deactivated it becomes obvious how cumbersome the robot is to move without the gravity compensation. This experiment is described in detail in Chapter 8.

E.2.2 Safety core motion damping

The clip (6:30min to 6:54) shows the influence of the safety core on the basic behaviour of the robot. Chapter 8 describes the experiment in which an extra weight is attached to the robot to induce a constant external force. This experiment was conducted once in the Zero-G mode only and once with the safety envelope active. The video clip shows the two experiments side by side and the damping effect of the safety core for motions exceeding the safe area in state space.

E.2.3 Collision experiments

The following clips (6:54min to 9:00min) demonstrate the safety envelope on ego-motion generated by an external position controller. The clips show various collision experiments.

In the first set of experiments the robots path is blocked by a foam rubber block. Various settings of the safety envelope result in different collision forces. In the lowest setting the robot is not able to push the foam rubber block over.

The second set of experiments were conducted with a plasticine impactor. The robot moved at a speed which corresponds to approximately 100% of the maximum impact force limit when it collides with the plasticine block. The resulting deformations of the plasticine impactors increase with increasing impact force limit of the safety core. This experiment is described in detail in Chapter 8.

E.2.4 Human intervention

The final clip (9:00-9:40) in the safe robot control section from 9:00min to 9:40min shows an experiment in which the robot is moving towards a goal configuration while a person interferes with this motion. The safety core impact force limit is set to 1N to ensure a slow motion by the robot. The video shows how the person can influence the path of the robot. When the person pushes the robot down the robot continues to move towards the goal configuration from the current configuration. This shows that the path in joint position space taken by the robot towards the goal configuration can be influenced by the operator or even by obstacles in the environment.

E.3 System integration: Interactive pick-and-place

The last video clips (9:40-10:01 and 10:01-19:00) demonstrates the integrated system which uses both the visual interface to issue commands for the robot and allows the user to command the robot to pick up designated objects from a table and put them down in positions using eye gaze. The video clip shows two simultaneous views, an external overview in the full video image and an additional small image in the bottom right corner showing the view through the vision system. The vision system view is enlarged temporarily when import actions take place in this view such as the recognition of facial gestures. The functional aspects of the experiment are described in detail in Chapter 9.

The video overlay display in the vision system view shows a number of gauges. The gauge in the top left shows the impact ability of the system in a range of 0 to 200%, the top right gauge shows the absolute value of the external joint torque vector[1] and the bottom right gauge shows the angular distance between the gaze line and the two closest target positions or objects. In the bottom left corner of the vision system view recognised gestures are displayed. The view is enlarged several times during the experiment to show gesture recognition results.

[1]This gauge is particularly important when an object is passed between the operator and the robot. At 12:15min the vision system view is enlarged such that the effect of the person applying forces to the manipulator can be seen.

Appendix F

Vectorspace Algebra

The formulation of the safety envelope requires the use of well known methods from the linear algebra such as the intersection of n-dimensional affine subspaces and the projection of points into n-dimensional affine subspaces. This appendix presents the underlying details formulations for the required operations.

F.1 Algebraic operations with n-dimensional subspaces

The safety constraints in Chapter 7 for the joint torques define n-dimensional halfspaces, where n is the number of joints. For each point p_i on the surface of the manipulator considered for the impact limitation scheme a set of inequalities define the constraints to be met by the n-dimensional joint torque vector τ. These inequalities define a halfspace H_i of safe torques in the n-dimensional joint torque space $\mathcal{T}$. The torques $\tau \notin \mathcal{T}$ are torques that violate the safety constraints for the point p_i, which means that the impact force of p_i with a stationary obstacle would violate Equation 7.36 (restated here):

$$\frac{d}{dt}\pi_{\mathbf{p}} \leq \frac{1}{t_c}(\pi_{max} - \pi_{\mathbf{p}}) \;\; \forall \mathbf{p} \in \mathbf{P} \tag{F.1}$$

Therefore, only those torque vectors τ that are within the safe area for all points p_i are safe to use as command torques for the robot. The intersection $H = \bigcap_i H_i$ defines the set of safe torques. If the motion control algorithm generates a command torque $\tau_c \notin H$ the closest torque vector τ_s must be determined on the surface of H to ensure safe operation of the robot.

The halfspaces H_i are defined by hyperplanes h_i separating the safe and the unsafe torque vectors. As shown in Chapter 7 τ_s can be found by calculating the intersection of the h_i and the intersection of the resulting affine subspaces and so on until the resulting affine subspaces are empty. τ_s is the orthogonal projection $p(\tau)$ of τ_c into one of those subspaces, determined by the following criteria.

$$p(\tau) \in H \text{ and } ||p\tau - \tau|| \text{ is minimal} \tag{F.2}$$

A hyperplane h is an affine subspace of $\mathcal{T}$ and can be defined by a point $o \in h$ and a set of spawning vectors $s = \{s_1, \ldots, s_k\}$ with $rank(s_1 \ldots s_k) = n - 1$. The points of h are given by $o + a_1 s_1 + \ldots + a_k s_k$ where the a_i are scalar factors.

F.1.1 Intersection of affine subspaces

The intersection of two affine subspaces $h_1 = (o_1, \{s_1, \ldots, s_k\})$ and $h_2 = (o_2, \{t_1, \ldots, t_l\})$ is an affine subspace $h_3 = (o_3, \{u_1, \ldots, u_m\})$.

For all points $v \in h_3$ there must be vectors $a = (a_1, \ldots, a_k)$ and $b = (b_1, \ldots, b_l)$ such that $v = o_1 + \sum_{i=1}^{k} a_i s_i = o_2 + \sum_{i=1}^{l} b_i t_i$. We can find o_3 and the u_i as the solution space of

$$o_1 + \sum_{i=1}^{k} a_i s_i = o_2 + \sum_{i=1}^{l} b_i t_i \tag{F.3}$$

$$\Leftrightarrow \sum_{i=1}^{k} a_i s_i - \sum_{i=1}^{l} b_i t_i = o_2 - o_1 \tag{F.4}$$

$$\Leftrightarrow (ST) \begin{pmatrix} a \\ -b \end{pmatrix} = o_2 - o_1 \tag{F.5}$$

This is a normal linear equation system of type $Ax = b$ with $A \in \mathbb{R}^{n \times k+l}$, $x \in \mathbb{R}^{k+l}$ and $b \in \mathbb{R}^n$.

The dimension of h_3 is given by $rank(U)$. U can be reduced to a basis by calculation the echelon form of U and removing the u_i for the i that are not step indices.

F.1.2 Orthogonal projection into affine subspaces

The orthogonal projection v_h of a point v in the torque space H into an affine subspace $h = (o, \{b_1, \ldots, b_k\})$ is given by $v_h = o + \sum_{i=1}^{k} u_i b_i$ where the vector of unknowns $\hat{u}$ satisfies Equation F.6.

$$\underbrace{\begin{pmatrix} < b_1|b_1 > & \cdots & < b_1|b_k > \\ \vdots & & \vdots \\ < b_k|b_1 > & \cdots & < b_k|b_k > \end{pmatrix}}_{\tilde{B}} \underbrace{\begin{pmatrix} u_1 \\ \vdots \\ u_k \end{pmatrix}}_{\hat{u}} = (v - o)^t \underbrace{\begin{pmatrix} & & \\ b_1 & \cdots & b_k \\ & & \end{pmatrix}}_{B} \tag{F.6}$$

If the basis of the affine subspace forms a ONS, $\tilde{B}$ becomes the identity matrix

I and therefore the calculation of $\hat{u}$ simplifies to

$$\hat{u} = (v - o)^t B \tag{F.7}$$

If the basis B is an ONS, the orthogonal projection v_h can then be calculated as

$$v_h = o + \sum_{i=1}^{k} < (v - o)^t | b_i > b_i \tag{F.8}$$

If h is a subspace of H and therefore $o = 0$ the calculation of v_h simplifies to

$$v_h = \sum_{i=1}^{k} < v | b_i > b_i \tag{F.9}$$

Appendix G

States of the Finite State Machine

The interactive pick and place application is controlled by a finite state machine (FSM) which was discussed in Chapter 9. Figure 9.5 showed an overview of the major FSM states. Detailed diagrams of the states *idle* and *pick object* were presented in Figures 9.6 and 9.7. The architecture of the other states is similar and are therefore presented in this appendix.

In the *idle (object grasped)* state illustrated in Figure G.1 the robot can be commanded to either hand the object over to the user or to place to object at an empty location on the table. The two actions are initiated with the *look up* and the *look down* gesture respectively. The general mechanism is similar to the *idle* state.

The *place* state is illustrated in Figure G.2 allows to command the robot to place a grasped object at a location on the table which is designated by the user's gaze. Before the object is released by the robot the user can correct false target locations with a *no* gesture. The basic mechanism of the *place* state is similar to the *pick* state.

The *receive* state allows the user to hand an object over to the robot. First the robot hand is opened and the robot is brought into the transfer configuration shown in Figure 9.1 which allows the user to conveniently put an object into the robot's open hand. When the robot senses the external force created by the user the hand is closed and the robot proceeds into the *idle (object grasped)* state.

A similar mechanism is used to hand a grasped object over to the user. The *hand over* state is illustrated in Figure G.4. The robot moves into the transfer position and waits for the user to grasp the object. As soon as the grip of the user's hand on the object is sensed by the robot, the hand releases the object and the robot proceeds into the *idle* state.

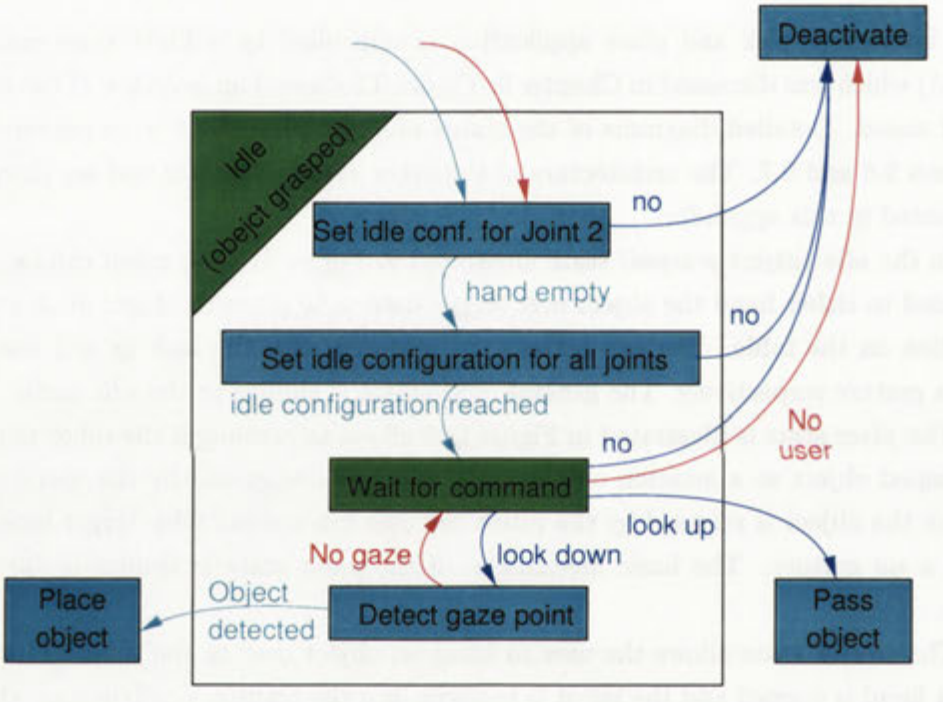


Figure G.1: Substates of the idle (object grasped) state

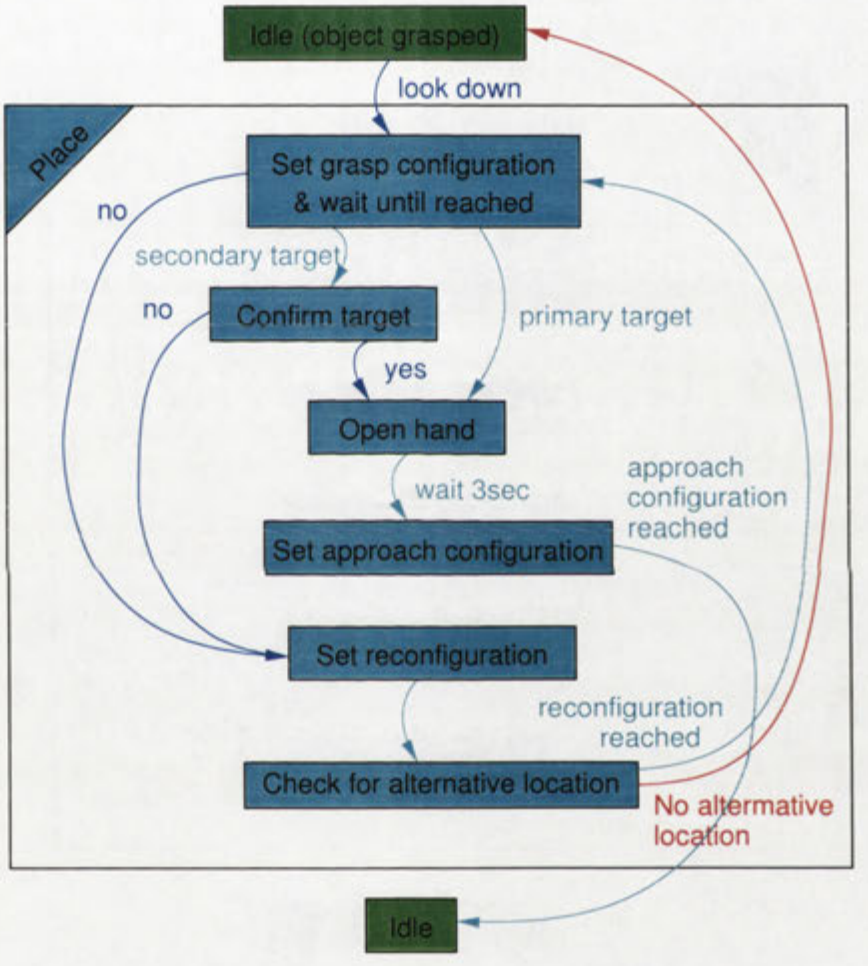


Figure G.2: Substates of the *place* state

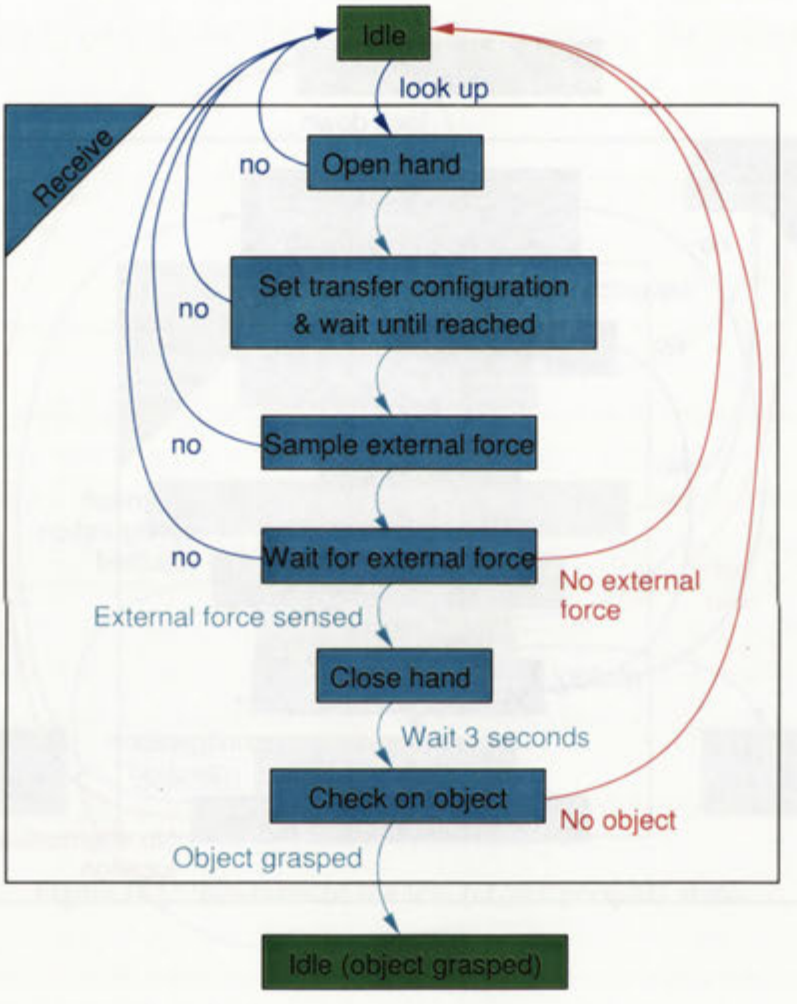


Figure G.3: Substate of the *receive* state

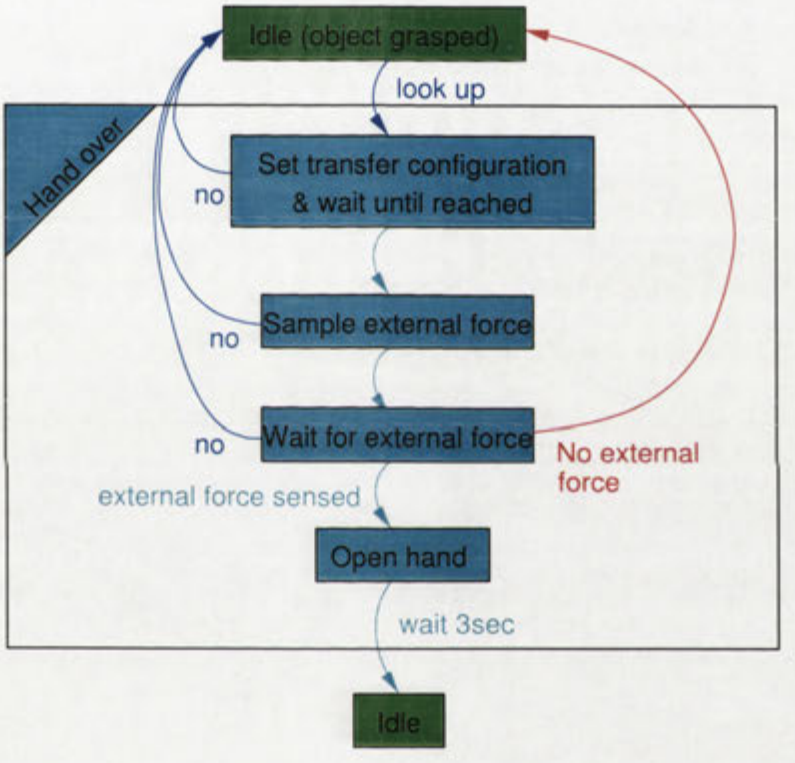


Figure G.4: Substates of the *hand over* state

Bibliography

Alter, T. D.: 92, 3d pose from three corresponding points under weak-perspective projection, *Technical Report AIM-1378*, MIT, AI Lab.

Anand, D. K. and Zmood, R. B.: 1995, *Introduction to Control Systems*, 3 edn, Butterworth-Heinemann Ltd, Oxford.

Arai, F., Kawaji, A., Fukuda, T., Matsuura, H. and Ota, H.: 1998, Safety oriented mechanism and control using er fluid in the joint, *Proc. of the Int. Conf. on Robotics and Automation*, Leuven, Belgium, pp. 2482–2487.

Azarbayejani, A., Starner, T., Horowitz, B. and Pentland, A.: 1993, Visually controlled graphics, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **15**(6), 602–605.

Azarbayejani, A., Wren, C. and Pentland, A.: 1996a, Real-time 3-d tracking of the human body, *Proc. of the Int. Conf. on Pattern Recognition, ICPR'96.*

Azarbayejani, A., Wren, C. and Pentland, A.: 1996b, Real-time 3-d tracking of the human body, *Proc. of IMAGE'COM.*

Baluja, S. and Pomerleau, D.: 1994a, Non-intrusive gaze tracking using artificial neural networks, *Advances in Neural Information Processing Systems, NIPS'94*, San Fransisco, CA.

Baluja, S. and Pomerleau, D.: 1994b, Non-intrusive gaze tracking using artificial neural networks, *Technical Report CMU-CS-94-102*, School of Computer Science, Carnegie Mellon University.

Barborak, D., Conrardy, C., Madigan, B. and Paskell, T.: 1999, Through-arc process monitoring techniques for control of automated gas metal arc welding, *Proc. of the Int. Conf. on Robotics and Automation*, Detroit, pp. 3053–3058.

Basu, S., Essa, I. and Pentland, A.: 1996, Motion regularization for model-based head tracking, *Proc. of the Int. Conf. on Pattern Recognition, ICPR'96.*

Basu, S., Oliver, N. and Pentland, A.: 1998a, 3d lip shapes from video: A combined physical-statistical model, *Speech Communication* **26**(1–2), 131–148.

Basu, S., Oliver, N. and Pentland, A.: 1998b, 3d modeling and tracking of human lip motions, *Proc. of Int. Conf. on Computer Vision ICCV'98*, Bombay, India.

Baumberg, A. and Hogg, D.: 1993, Learning flexible models from image sequences, *Technical Report 93.36*, School of Computer Studies, University of Leeds.

Baumberg, A. and Hogg, D.: 1994, An efficient method for contour tracking using active shape models, *Technical Report 94.11*, School of Computer Studies, University of Leeds.

Becker, M., E.Kefalea, Mael, E., von der Malsburg, C., M.Pagel, Triesch, J., Vorbrüggen, J., R.P.Würtz and Zadel, S.: 1999, Gripsee: A gesture-controlled robot for object perception and manipulation, *Autonomous Robots* **6**(3), 203–221.

Bergasa, L., Mazo, M., Gardel, A., Sotelo, M. and Garcia, J.: 1999, Guidance of a wheelchair for handicapped people by head movements, *Proc. of the Int. Conf. on Field and Service Robots*, pp. 150–155.

Black, M. J. and Jepson, A. D.: 1998, Recognizing temporal trajectories using the condensation algorithm, *Proc. of the IEEE Int. Conf. on Automatic Face and Gesture Recognition*, pp. 16–21.

Black, M. and Yacoob, Y.: 1995, Tracking and recognizing rigid and non-rigid facial motions using parametric models of image motion, *Proc. of the Int. Conf. on Computer Vision ICCV*, Boston, MA, pp. 374–381.

Bozic, S.: 1979, *Digital and Kalman Filtering*, London: Edward Anold.

Brand, M.: 1999, Shadow puppetry, *Proc. of the Int. Conf. on Computer Vision ICCV'99*, Kerkyra, Greece, pp. 1237–1244.

Brooks, R. A., Breazeal, C., Marjanovic, M., Scassellati, B. and Williamson, M. M.: 1998, The cog project: Building a humanoid robot, *Proc. IARP Workshop on Hum. and Human Friendly Robots*, Tsukuba, Japan, pp. I–3.

Brunelli, R. and Poggio, T.: 1993, Face recognition: Feature versus templates, *IEEE Trans. on Pattern Analysis and Machine Intelligence* **15**(10), 1042–1052.

Cai, J. and Goshtasby, A.: 1999, Detecting human faces in color images, *Image and Vision Computing* **18**(1), 63–75.

Cham, T.-J. and Rehg, J. M.: 1999, Dynamic feature ordering for efficient registration, *Proc. of the Int. Conf. on Computer Vision ICCV'99*, Kerkyra, Greece, pp. 1084–1091.

Chellappa, R., Wilson, C. L. and Sirohey, S.: 1995, Human and machine recognition of faces: A survey, *Proceedings of the IEEE* **83**(5), 705–740.

Cheng, G. and Kuniyoshi, Y.: 2000, Complex continous meaningful humanoid interaction: A multi sensory-cue based approach, *Proc. of the Int. Conf. on Robotics and Automation*, San Francisco, pp. 2235–2242.

Cheung, E. and Lumelsky, V.: 1988, Motion planning for robot arm manipulators with proximity sensing, *Proc. of Int. Conf. on Robotics and Automation, ICRA'88*, pp. 740–745.

Clark, A. F. and Kokuer, M.: 1992, Feature identification and model tracking, *Proc. of the 11th IAPR Int. Conf. on Pattern Recognition*, Vol. 3, The Hague The Netherlands, pp. 79–82.

Cohen, C. J., Conway, L. and Koditschek, D.: 1996, Dynamical system representation, generation, and recognition of basic oscillatory motion gestures, *Proc. of the Int. Conf. on Automatic Face and Gesture Recognition*, IEEE Computer Society Press, pp. 60–65.

Coianiz, T., Torresani, L. and Caprile, B.: 1995, 2d deformable models for visual speech analysis, *NATO Advanced Study Institute: Speechreading by Man Machine.*

Colombo, C. and Bimbo, A. D.: 1996, Visual speech recognition using active shape models and hidden markov models, *Proc. of ICASSP'96*, pp. 817–820.

Costall, A.: 1993, A cubist manifesto for visual science, *Image and Vision Computing* **11**(6), 334–341.

Craig, J. J.: 1986, *Introduction to Robotics*, 2 edn, Addison Wesley.

Cygnaski, D. and Orr, J.: 1985, Object recognition and orientation determination by tensor methods, *IEEE Trans. Pattern Analysis and Machine Intelligence* **7**(6).

Darrell, T., Essa, I. and Pentland, A.: 1994, Correlation and interpolation networks for real-time expression analysis/synthesis, *Proc. of Neural Information Processing Systems '94*, pp. 909–916.

Darrell, T., Essa, I. and Pentland, A.: 1995, Task-specific gesture analysis in real-time using interpolated views, *Technical Report 364*, MIT AI Lab.

Darrell, T. and Pentland, A.: 1993, Space - time gestures, *Proc. of the Conf. on Computer Vision and Pattern Recognition*, New York, pp. 335–340.

Delamarre, Q. and Faugeras, O.: 1999, 3d articulated models and multi-view tracking with silhouettes, *Proc. of the Int. Conf. on Computer Vision ICCV'99*, Kerkyra, Greece, pp. 716–721.

Dhillon, B. S. and Anude, O. C.: 1993, Robot safety and reliability: A review, *Microelectronic Reliability* **33**(3), 413–429.

Dickmanns, E. D.: 1999, An expectation-based, multi-focal, saccadic (ems) vision system for vehicle guidance, *Proc. of Int. Symposium on Robotics Research ISRR'99*, pp. 421–430.

Duta, N. and Jain, A. K.: 1998, Learning the human face concept in black and white images, *Proc. of the Int. Conf. on Pattern Recognition*, pp. 1365–1367.

Ekman, P. and Friesen, W.: 1975, *Unmasking the Face*, Prentice Hall.

Ekman, P. and Friesen, W.: 1978, *The Facial Action Coding System: A Technique for the Measurement of Facial Movement*, Consulting Psychologists Press Inc., San Francisco, CA.

Elagin, E., Steffens, J. and Neven, H.: 1998, Automatic pose estimation system for human faces based on bunch graph matching technology, *Proc. of the IEEE Int. Conf. on Automatic Face and Gesture Recognition, FG'98*, pp. 136–141.

Engelberger, J. F.: 1980, *Robotics in Services – Management and Applications of Industrial Robots*, Kogan Page.

Engelberger, J. F.: 1989, *Robotics in Service*, MIT Press, Cambridge, Massachusetts.

Etherton, J. and Sneckenberger, J. E.: 1988, A robot safety experiment varying robot speed and contrast with human decision cost, *Proceedings of the Human Factors Society 32nd Annual Meeting*, Vol. 1 of *Safety: Robotics/Industrial Safety*, p. 953.

Feng, G. C., Yuen, P. C. and Lai, J. H.: 2000, Virtual view fae image synthesis using 3d spring-based face model from a single image, *Proc. of the Int. Conf. on Face and Gesture Recognition FG'2000*, Grenoble, pp. 530–535.

Feraud, R., Bernier, O., Viallet, J. E. and Collobert, M.: 2000, A fast and accurate face detector for indexatino of face images, *Proc. of the Int. Conf. on Face and Gesture Recognition FG'2000*, Grenoble, pp. 77–82.

Gee, A. and Cipolla, R.: 1994, Non-intrusive gaze tracking for human-computer interaction, *Proc. of the Int. Conf. on Mechatronics and Machine Vision in Practice*, Toowoomba, Australia, pp. 112–117.

Gee, A. and Cipolla, R.: 1996, Fast visual tracking by temporal consensus, *Image and Vision Computing* **14**, 105–114.

Geiger, D. and Liu, T.-L.: 1996, Recognizing articulated objects with information theoretic methods, *Proc. of the Int. Conf. on Automatic Face and Gesture Recognition*, IEEE Computer Society Press, pp. 45–50.

Graf, R. and Weckesser, P.: 1998, Roomservice in a hotel, *3rd IFAC Symposium on Intellegnt Autonomous Vehicles - IAV 98*, Madrid, pp. 641–647.

Graham, J. H. and Millard, D. L.: 1991, Towards the development of inherently safe robots, *Proc. of Int. Symposium on Industrial Robots*, pp. 911–917.

Grimson, W. E., Huttenlocher, D. P. and Alter, T. D.: 1992, Recognizing 3D ojbects of 2D images: An error analysis, *Technical Memo AIM-1362*, Massachusetts Institute of Technology, Artificial Intelligence Laboratory.

Haritaoglu, I., Cutler, R., Harwood, D. and Davis, L. S.: 1999, Backpack: Detection of people carrying objects using silhouettes, *Proc. of the Int. Conf. on Computer Vision ICCV'99*, Kerkyra, Greece, pp. 102–107.

Haritaoglu, I., Harwood, D. and Davis, H. S.: 1998a, W^4: Who? when? where? what? a real time system for detection and tracking people, *Proc. of the IEEE Int. Conf. on Automatic Face and Gesture Recognition*, pp. 222–227.

Haritaoglu, I., Harwood, D. and Davis, L. S.: 1998b, Ghost: A human body part labeling system using silhouettes, *Proc. of the Int. Conf. on Pattern Recognition*, pp. 77–82.

Hayakawa, Y., Ogata, T. and Sugano, S.: 2000, A robotic co-operation system based on a self-organization approach human work model, *Proc. of the Int. Conf. on Robotics and Automation*, San Francisco, pp. 4058–463.

Hayashibara, Y., Takubo, T., Sonoda, Y., Arai, H. and Tanie, K.: 1999, Assist system for carrying a long object with a human - analysis of a human cooperation behaviour in the vertical direction, *Proc. of the Int. Conf. on Intelligent Robots and Systems IROS'99*, pp. 695–700.

Heap, T. and Hogg, D.: 1996, Towards 3d hand tracking using a deformable model, *Proc. of the Int. Conf. on Automatic Face and Gesture Recognition*, IEEE Computer Society Press, pp. 140–145.

Heinzmann, J.: 1996, *Real-time human face tracking and gesture recognition*, Master's thesis, Universität Karlsruhe, Fakultät für Informatik.

Heinzmann, J., Matsumoto, Y., Kieffer, J. and Zelinsky, A.: 1998, Smart interfaces + safe mechanisms = human friendly robots, *Proc. of IARP Int. Workshop on Humanoid and Human Friendly Robotics*, pp. II–3.

Heinzmann, J. and Zelinsky, A.: 1997, Robust real-time face tracking and gesture recognition, *Proc. of the Int. Joint Conf. on Artificial Intelligence, IJCAI'97*, Vol. 2, pp. 1525–1530.

Hills, W. D.: 1982, A high-resolution imaging touch sensor, *Int. Journal on Robotics Research* **1**(2), 33–44.

Hirai, K., Hirose, M., Haikawa, Y. and Takenaka, T.: 1998, The development of honda humanoid robot, *Proc. of the Int. Conf. on Robotics and Automation ICRA'98*, Leuven, Belgium, pp. 1321–1326.

Hirata, Y. and Kosuge, K.: 2000, Distributed robot helpers handling a single object in cooperation with a human, *Proc. of the Int. Conf. on Robotics and Automation*, San Francisco, pp. 458–490.

Hogan, N.: 1984, Impedance control: an approach to manipulation, *American Control Conference, 1984*, pp. 304–313.

Hogan, N.: 1985, Impedance control: An approach to manipulation. parts i, ii, and iii., *Trans. ASME Journal of Dynamics, Systems, Measurement, and Control* **107**(3), 1–24.

Hogg, D.: 1983, Model-based vision: A program to see a walking person, *Image and Vision Computing* **1**(1), 5–20.

Hong, P., Turk, M. and Huang, T. S.: 2000, Gesture modelling and recognition using finite state machines, *Proc. of the Int. Conf. on Automatic Face and Gesture Recognition FG'2000*, IEEE Computer Society Press, pp. 410–415.

Hongo, H., Ohya, M., Yasamoto, M., Niwa, Y. and Yamamoto, K.: 2000, Focus of attention for face and hand gesture recognition using multiple cameras, *Proc. of the Int. Conf. on Face and Gesture Recognition FG'2000*, Grenoble, pp. 156–161.

Hough, P.: 1962, Methods and means for recognising complex patterns, *Technical Report 3069654*, US patent.

Howe, R. D. and Cutkosky, M. R.: 1993, Dynamic tactile sensing: Perception of fine surface features with stress rate sensing, *Trans. on Robotics and Automation* **9**(2), 140–151.

Hsu, F.-Y. and Fu, L.-C.: 2000, Intelligent robot deburring using adaptive fuzzy hybrid position/force control, *Trans. on Robotics and Automation* **16**(4), 325–335.

Hu, C., Yu, Q., Li, Y. and Ma, S.: 2000, Extraction of parametric human model for posture recognition using genetic algorithm, *Proc. of the Int. Conf. on Automatic Face and Gesture Recognition FG'2000*, IEEE Computer Society Press, pp. 518–523.

Hu, M. K.: 1962, Visual pattern recognition by moment invariants, *IRE Trans. on Information Theory* **8**, 179–187.

Hutchinson, T. E.: 1993, Hot topics; eye gaze computer interfaces: computers that sense eye position on the display, *Computer* **26**(7), 65, 67.

Hutchinson, T. E., White Jr., K. P., Martin, W. N., Reichert, K. C. and Frey, L. A.: 1989, Human-computer interaction using eye-gaze input, *IEEE Transactions on Systems, Man, and Cybernetics* **19**(6), 1527–1534.

Huttenlocher, D. P. and Ullman, S.: 1990, Recognizing solid objects by alignment with an image, *Int. Journal of Computer Vision* **5**(2), 195–212.

Hyde, J. and Cutkosky, M.: 1994, Controlling contact transitions, *IEEE Control Systems* **14**(1), 25–30.

Iketani, A., Nagai, A., Kuno, Y. and Shirai, Y.: 1998, Detecting persons on changing background, *Proc. of the Int. Conf. on Pattern Recognition*, pp. 74–76.

Imagawa, K., Lu, S. and Igi, S.: 1998, Color-based hands tracking system for sign language recognition, *Proc. of the IEEE Int. Conf. on Automatic Face and Gesture Recognition, FG'98*, pp. 462–467.

Inoue, H., Inaba, M., Mori, T. and Tachikawa, T.: 1993, Real-time robot vision system based on correlation technology, *Proc. of Int. Symp. on Industrial Robots.*

Ioffe, S. and Forsyth, D.: 1999, Finding people by sampling, *Proc. of the Int. Conf. on Computer Vision ICCV'99*, Kerkyra, Greece, pp. 1092–1097.

Iwai, Y., Ogaki, K. and Yachida, M.: 1999, Posture estimation using structure and motion models, *Proc. of the Int. Conf. on Computer Vision ICCV'99*, Kerkyra, Greece, pp. 214–219.

Iwamoto, K. and Tanie, K.: 1997, Development of an eye movement tracking type head mounted display: Capturing and displaying real environment images with high reality, *Proc of Int. Conf. on Robotics and Automation ICRA '97*, pp. 2258–2263.

Jacquin, A. and Eleftheriadis, A.: 1995, Automatic location tracking of faces and facial features in video sequences, *Proc. of the Int. Workshop on Automatic Face- and Gesture Recognition 95*, Zürich, pp. 142–147.

Jochem, T., Pomerleau, D. and Thorpe, C.: 1993, MANIAC: A next generation neurally based autonomous road follower, *Proc. of Int. Conf. on Intelligent Autonomous Systems, IAS-3.*

Johannsen, G.: 1995, Human-machine interfaces for cooperative work, *Proceedings of the Sixth International Conference on Human-Computer Interaction*, Vol. I. Human and Future Computing of *I.11 Collaboration 2*, pp. 359–364.

Ju, S. X., Black, M. J. and Yacoob, Y.: 1996, Cardboard people: A parametrized model of articulated image motion, *Proc. of the Int. Conf. on Automatic Face and Gesture Recognition*, IEEE Computer Society Press, pp. 38–44.

Karwowski, W., Plank, T., Parsaei, M. and Rahimi, M.: 1987, Human perception of the maximum safe speed of robot motions, *Proceedings of the Human Factors Society 31st Annual Meeting*, Automation Safety, pp. 186–190.

Kassim, A., Tan, T. and Tan, K.: 1999, A comparative study of efficient generalised hough transform techniques, *Image and Vision Computing* **17**, 737–748.

Kawato, S. and Ohya, J.: 2000, Real-time detection of nodding and head-shaking by directly detecting and tracking the between-eyes, *Proc. of the Int. Conf. on Automatic Face and Gesture Recognition FG'2000*, IEEE Computer Society Press, pp. 40–45.

Khatib, O.: 1999, Mobile manipulation: The robotic assistant, *Robotics and Autonomous Systems* **26**, 175–183.

Khatib, O., Yokoi, K., Chang, K. and Casal, A.: 1997, The stanford robotic platforms, *Int. Conf. on Robotics and Automation*, Leuven, Belgium, p. Video Proceedings.

Kim, K. I. and Zhang, Y. F.: 1998, Human-robots coordination with rotational motion, *Proc. of the Int. Conf. on Robotics and Automation*, Leuven, Belgium, pp. 3480–3485.

Klingspohr, H., Block, T. and Grigat, R.-R.: 1997, A passive real-time gaze estimation system for human-machine interfaces, *Lecture Notes in Computer Science* **1296**, 718–725.

Kosecka, J., Blasi, R., Taylor, C. J. and Malik, J.: 1998, A comparative study of vision-based lateral control strategies for autonomous highway driving, *Proc. of the Int. Conf. on Robotics and Automation ICRA'98*, Leuven, Belgium, pp. 1903–1908.

Kosuge, K., Hashimoto, S. and Yoshida, H.: 1998, Human-robots collaboration system for flexible object handling, *Proc. of the Int. Conf. on Robotics and Automation*, Leuven, Belgium, pp. 1841–1846.

Kosuge, K., Sato, M. and Kazamura, N.: 2000, Mobil robot helper, *Proc. of the Int. Conf. on Robotics and Automation*, San Francisco, pp. 583–588.

Kumar, V. P. and Poggio, T.: 2000, Learning-based approach to real time tracking and analysis of faces, *Proc. of the Int. Conf. on Face and Gesture Recognition FG'2000*, Grenoble, pp. 96–101.

Kurpjuhn, T. P., Hauck, A., Nickels, K. and Hutchinson, S.: 1999, Development of a visual space-mouse, *Proc. of the Int. Conf. on Robotics and Automation*, Detroit, pp. 2527–2532.

Lee, S., Boo, K. S., Shin, D. and Lee, D. H.: 1998, Automatic lane following with a single camera, *Proc. of the Int. Conf. on Robotics and Automation ICRA'98*, Leuven, Belgium, pp. 1689–1694.

Li, P. and Horowitz, R.: 1995, Passive velocity field control of mechanical manipulators, *Proc. of the Int. Conf. on Robotics and Automation.*

li Tian, Y., Kanade, T. and Cohn, J. F.: 2000a, Dual-state parametric eye tracking, *Proc. of the Int. Conf. on Automatic Face and Gesture Recognition FG'2000*, IEEE Computer Society Press, pp. 110–115.

li Tian, Y., Kanade, T. and Cohn, J. F.: 2000b, Recognizing lower face action units for facial expression analysis, *Proc. of the Int. Conf. on Automatic Face and Gesture Recognition FG'2000*, IEEE Computer Society Press, pp. 484–490.

Lim, H.-o. and Tanie, K.: 1999, Collision-tolerant control of human-friendly robot with viscoelastic trunk, *Transactions on Mechatronics* **4**(4), 417–427.

Lim, H.-o. and Tanie, K.: 2000, Human safety mechanisms of human-friendly robots: Passive viscoelastic trunk and passively movable base, *Int. Journal on Robotics Research* **19**(4), 307–335.

Lim, H.-o., Yokoi, K., Takanishi, A. and Tanie, K.: 1999, Collision force suppression by human-friendly robots with passively movable base, *Proc. of the Int. Conf. on Intelligent Robots and Systems IROS'99*, pp. 1039–1044.

Lozano-Pérez, T.: 1985, Motion planning for simple robot manipulators, *Int. Journal on Robotics Research* **3**, 133–140.

Luh, J. Y. S. and Hu, S.: 1999, Interactions and motions in human-robot coordination, *Proc. of the Int. Conf. on Robotics and Automation*, Detroit, pp. 3171–3176.

Lumelsky, V. J. and Cheung, E.: 1993, Real-time collision avoidance in teleoperated whole-sensitive robot arm manipulators, *Trans. on Systems, Man and Cypernetics* **23**(1), 194–203.

Maggioni, C.: 1993, A novel device for using the hand as a human-computer interface, *Proc. of the HCI'93 Conf. on People and Computers VIII*, Tools and Techniques, pp. 191–202.

Maggioni, C.: 1995, Gesturecomputer: New ways of operating a computer, *Proc. of the Int. Workshop on Automatic Face- and Gesture-Recognition*, Zürich, pp. 166–171.

Maggioni, C. and Wirtz, B.: 1991, A neural net approach to 3-D pose estimation, *in* T. Kohonen, K. Mäkisara, O. Simula and J. Kangas (eds), *Artificial Neural Networks*, Vol. I, North-Holland, Amsterdam, Netherlands, pp. 75–80.

Malciu, M. and Preteux, F.: 2000, A robust model-based approach for 3d head tracking in video sequences, *Proc. of the Int. Conf. on Face and Gesture Recognition FG'2000*, Grenoble, pp. 169–174.

Marcel, S., Bernier, O., Viallet, J.-E. and Collobert, D.: 2000, Hand gesture recognition using input-output hidden markov models, *Proc. of the Int. Conf. on Face and Gesture Recognition FG'2000*, Grenoble, pp. 456–461.

Mase, K.: 1991, Recognition of facial expressions from optical flow, *IEICE Trans, Spec. Issue on Computer Vision and its Applications* **E74**(10), 3474–3483.

Matsumoto, Y., Heinzmann, J. and Zelinsky, A.: 1999, The essential components of human-friendly robots, *Proc. of the Int. Conf. on Field and Service Robots, FSR'99*, pp. 29–31.

Matsumoto, Y. and Zelinsky, A.: 2000, An algorithm for real-time stereo vision implementation of head pose and gaze direction measurement, *Proc. of the Int. Conf. on Face and Gesture Recognition FG'2000*, Grenoble, pp. 499–504.

Matthews, I., Cootes, T., Cox, S., Harvey, R. and Bangham, J.: 1998, Lipreading using shape, shading and scale, *Proc. of Int. Conf. on Auditory-Visual Speech Processing AVSP'98*, Terrigal, Australia.

Matthies, L. and Elfes, A.: 1988, Integration of sonar and stereo range data using a grid-based representation, *Proc. of the Int. Conf. on Robotics and Automation ICRA'88*, Philadelphia, pp. 25–29.

Maurer, T. and von der Malsburg, C.: 1996, Tracking and learning graphs and pose on image sequences of faces, *Proc. of the Int. Conf. on Automaic Face and Gesture Recognition '96*, pp. 176–181.

Maybeck, P.: 1979, Stochastic models, *Estimation and Control* **1**.

McKenna, S. J., Jabri, S., Duric, Z. and Wechsler, H.: 2000, Tracking interacting people, *Proc. of the Int. Conf. on Automatic Face and Gesture Recognition FG'2000*, IEEE Computer Society Press, pp. 348–353.

McLauchlan, P., Reid, I. and Murray, D.: 1994, Recursive affine structure and motion from image sequences, *Proc. of the Eur. Conf. on Computer Vision ECCV'94*, Stockholm, pp. 217–224.

Milgram, P., Zhai, S., Drascic, D. and Grodski, J.: 1993, Applications of augmented reality for human-robot communication, *Proc. IROS'93: IEEE/RSJ International Conf. on Intelligent Robots and Systems.*

Mills, J. and Lokhurst, D.: 1993, Control of robotic manipulators during general task execution: A discontinuous control approach, *Int. J. on Robotics Research* **12**(2), 146–163.

Moghaddam, B. and Pentland, A.: 1997, Probabilistic visual learning for object representation, *IEEE Trans. on Pattern Analysis and Machine Intelligence* **19**(7), 696–710.

Morimoto, C. and Flickner, M.: 2000, Real-time multiple face detection using active illuminatino, *Proc. of the Int. Conf. on Face and Gesture Recognition FG'2000*, Grenoble, pp. 8–13.

Morimoto, C., Koons, D., Amir, A. and Flickner, M.: 1998, Pupil detection and tracking using multiple light sources, *Technical report*, IBM Almaden Research Center.

Morimoto, C., Koons, D., Amir, A. and Flickner, M.: 1999, Framerate pupil detector and gaze tracker, *Proc. of the Int. Conf. on Computer Vision ICCV'99*, Kerkyra, Greece.

Morita, T., Iwata, H. and Sugano, S.: 1999, Development of human symbiotic robot: Wendy, *Proc. of the Int. Conf. on Robotics and Automation*, Detroit, pp. 3183–3188.

Morita, T., Shibuya, K. and Sugano, S.: 1998, Design and control of mobile manipulation system for human symbiotic humanoid: Hadaly-2, *Proc. of the Int. Conf. on Robotics and Automation*, Leuven, Belgium, pp. 1315–1320.

Müller, S., Eickeler, S. and Rigoll, G.: 2000, Crane gesture recognition using pseudo 3-d hidden markov models, *Proc. of the Int. Conf. on Automatic Face and Gesture Recognition FG'2000*, IEEE Computer Society Press, pp. 398–402.

Munkelt, O., Ridder, C., Hansel, D. and Hafner, W.: 1998, A model driven 3d image interpretation system applied to person detection in video images, *Proc. of the Int. Conf. on Pattern Recognition*, pp. 70–73.

Nagel, B., Wingbermühle, J., Weik, S. and Liedtke, C.-E.: 1998, Automated modelling of real human faces for 3d animation, *Proc. of the Int. Conf. on Pattern Recognition*, pp. 693–696.

Nishimura, T. and Okaa, R.: 1996, Spotting recognition of human gestures from time-varying images, *Proc. of Int. Conf. on Automatic Face and Gesture Recognition FG'96*, IEEE Computer Society Press, pp. 318–322.

Novak, J. L. and Feddema, J. T.: 1994, A capacitance-based proximity sensor for whole arm obstacle avoidance, *Proc. of the Int. Conf. on Robotics and Automation ICRA'94*, Nice, France, pp. 1307–1314.

Ohba, K., Clary, G., Tsukada, T., Kotoku, T. and Tanie, K.: 1998, Facial expression communication with fes, *Proc. of the Int. Conf. on Pattern Recognition*, pp. 1376–1378.

Ohta, H., Saji, H. and Nakatani, H.: 1998, Recognition of facial expressions using muscle-based feature models, *Proc. of the Int. Conf. on Pattern Recognition*, pp. 1379–1381.

Ohta, Y., Kanade, T. and Sakai, T.: 1980, Color information for region segmentation, *Computer Graphics and Image Processing* **13**(3), 222–241.

Pappu, R. and Beardsley, P. A.: 1998, A qualitative approach to classifying gaze direction, *Proc. of the IEEE Int. Conf. on Automatic Face and Gesture Recognition, FG'98*, pp. 160–165.

Penin, L. F., Aracil, R., Ferre, M., Pinto, E., Hernando, M. and Barrientos, A.: 1998, Telerobotic system for live power lines maintenance: Robtet, *Proc. of the Int. Conf. on Robotics and Automation*, Leuven, Belgium, pp. 2110–2115.

Perret, J.: 1998, Service robots for nuclear safety: New developments by cybernetix, *Proc. of the Int. Conf. on Robotics and Automation ICRA'98*, Leuven, Belgium, pp. 2106–2109.

Pinto-Elias, R. and Sossa-Axuela, J. H.: 1998, Automatic facial feature detection and location, *Proc. of the Int. Conf. on Pattern Recognition*, pp. 1365–1367.

Pramadihanto, D., Iwai, Y. and Yachida, M.: 1998, A flexible feature matching for automatic face and facial feature points detection, *Proc. of the Int. Conf. on Pattern Recognition*, pp. 92–95.

Raibert, M. H. and Craig, J. J.: 1981, Hybrid position/force control of manipulators, *ASME Journal on Dynamic Systems, Measurement and Control* **102**(2), 126–133.

Reinders, M. J. T., Sankur, B. and van der Lubbe, J. C. A.: 1992, Transformation of a general 3D facial model to an actual scene face, *Proc. of the 11th IAPR Int. Conf. on Pattern Recognition*, Vol. 3, The Hague The Netherlands, pp. 75–78.

Revéret, L. and Benoît, C.: 1998, A new 3d lip model for analysis and synthesis of lip motion, *Proc. of Int. Conf. on Auditory-Visual Speech Processing AVSP'98*, Terrigal, Australia, pp. 207–212.

Rigoll, G., Eickeler, S. and Müller, S.: 2000, Person tracking in real-world scenarios using statistical methods, *Proc. of the Int. Conf. on Automatic Face and Gesture Recognition FG'2000*, IEEE Computer Society Press, pp. 342–347.

Rowley, H. A., Baluja, S. and Kanade, T.: 1995, Human face detection in visual scenes, *Technical Report CMU-CS-95-158*, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA.

Rowley, H. A., Baluja, S. and Kanade, T.: 1998, Neural network-based face detection, *IEEE Trans. on Pattern Analysis and Machine Intelligence* **20**(1), 23–38.

Sakaguchi, M., Zhang, G. and Furusho, J.: 2000, Modeling and motion control of an actuator unit using er clutches, *Proc. of the Int. Conf. on Robotics and Automation*, San Francisco, pp. 1347–1353.

Sakoe, H. and Edelmann, S.: 1980, Dynamic programming optimization for spoken work recognition, *IEEE Trans. ASSP* **26**, 623–625.

Sarkar, N., Yun, X. and Ellis, R.: 1998, Live-constraint-based control for contact transition, *IEEE Trans. on Robotics and Automation* **14**(5), 743–754.

Sato, S. and Sakane, S.: 2000, A human-robot interface using an interactive hand pointer that projects a mark in the real work space, *Proc. of the Int. Conf. on Robotics and Automation*, San Francisco, pp. 589–595.

Sato, Y., Kobayashi, Y. and Koike, H.: 2000, Fast tracking of hands and fingertips in infrared images for augmented desk interface, *Proc. of the Int. Conf. on Face and Gesture Recognition FG'2000*, Grenoble, pp. 446–453.

Saulnier, A., Viaud, M.-L. and Geldreich, D.: 1995, Real-time facial analysis and synthesis chain, *Proc. of Int. Workshop on Automatic Face- and Gesture Recognition 95*, Zürich, pp. 86–91.

Schraft, R. D., Rust, H. and Gehringer, H. R.: 1999, Sensors for the object detection in environments with high heat radiation and photoresist or diffuse visibility conditions, *Proc. of the Int. Symposium on Experimental Robotics ISER'99*, Sydney, Australia, pp. 45–54.

Schweitzer, G., Tschichold, N. and Vestli, S.: 1998, Autonomy and interactivity of a mail distributing service robot, *Proc. Comp. Eng. in Systems Applications IMACS*, Nabeul-Hammamet, Tunisia.

Schwerdt, K. and Crowley, J. L.: 2000, Robust face tracking using color, *Proc. of the Int. Conf. on Face and Gesture Recognition FG'2000*, Grenoble, pp. 90–95.

Segen, J. and Kumar, S.: 1998, Fast and accurate 3d gesture recognition interface, *Proc. of the IEEE Int. Conf. on Automatic Face and Gesture Recognition*, pp. 86–91.

Sengupta, K. and Ohya, J.: 1998, Human face structure estimation from multiple images using the 2d affine space, *Proc. of the IEEE Int. Conf. on Automatic Face and Gesture Recognition, FG'98*, pp. 106–111.

Sengupta, K., Shinqin, W., Ko, C. C. and Burman, P.: 2000, Automatic face modeling from monocular image sequences using modified non parametric regression and an affine camera model, *Proc. of the Int. Conf. on Face and Gesture Recognition FG'2000*, Grenoble, pp. 524–529.

Shakunaga, T., Ogawa, K. and Oki, S.: 1998, Integration of eigentemplate and structure matching for automatic facial feature detection, *Proc. of the IEEE Int. Conf. on Automatic Face and Gesture Recognition, FG'98*, pp. 94–99.

Shimada, N., Shirai, Y. and Kuno, Y.: 1995, Hand gesture recognition using computer vision based on model-matching method, *Proceedings of the Sixth International Conference on Human-Computer Interaction*, Vol. I. Human and Future Computing of *I.1 Gestural Interface*, pp. 11–16.

Shimizu, I., Zhang, Z., Akamatsu, S. and Deguchi, K.: 1998, Head pose determination from one image using a generic model, *Proc. of the IEEE Int. Conf. on Automatic Face and Gesture Recognition, FG'98*, pp. 100–105.

Sicard, P. and Levine, M. D.: 1989, Joint recognition and tracking for robotic arc welding, *IEEE Transactions on Systems, Man, and Cybernetics* **19**(4), 714–728.

Spindler, F. and Chaumette, F.: 1997, Gaze control using human eye movement, *Proc of Int. Conf. on Robotics and Automation ICRA'97*, pp. 2258–2263.

Suita, K., Yamada, Y., Tsuchida, N., Imai, K., Ikeda, H. and Sugimoto, N.: 1995, A failrue-to-safety kyzon system with simple contact with detection and stop capabilities for safe human-autonomous robot coexistance, *Proc. of the Int. Conf. on Robotics and Automation, ICRA'95*, Nagoya, Japan, pp. 3089–3096.

Sun, Q., Huang, W. and Wu, J.: 1998, Face detection based on color and local symmetry information, *Proc. of the IEEE Int. Conf. on Automatic Face and Gesture Recognition*, pp. 130–135.

Sung, K.-K. and Poggio, T.: 1994, Example-based learning for view-based human face detection, *Technical Report 1521*, MIT AI Lab.

Sung, K.-K. and Poggio, T.: 1998, Example-based learning for view-based human face detection, *IEEE Trans. on Pattern Analysis and Machine Intelligence* **20**(1), 39–52.

Swain, M. and Ballard, D.: 1991, Color indexing, *Int. Journal of Computer Vision* **7**(1), 11–32.

Tao, H. and Huang, T. S.: 1998, Connected vibrations method for non-rigid motion tracking, *Proc. of the Int. Conf. on Pattern Recognition*, pp. 1424–1426.

Tarn, T., Wu, Y., Xi, N. and Isidori, A.: 1996, Force regulation and contact transition control, *IEEE Control Systems* **16**(1), 32–40.

Terashima, M. and Sakane, S.: 1999, A human-robot interface using an extended digital desk, *Proc. of the Int. Conf. on Robotics and Automation*, Detroit, pp. 2874–2880.

Terrilon, J.-C., David, M. and Akamatsu, S.: 1998, Detection of human faces in complex scene images by use of a skin model and of invariant fourier-mellin moments, *Proc. of the Int. Conf. on Pattern Recognition*, pp. 1350–1355.

Thomanek, F. and Dickmanns, E. D.: 1996, Autonomous road vehicle guidance in normal traffic, *Lecture Notes in Computer Science* **1035**, 499–516.

Thompson, D. W. and Mundy, J. L.: 1987, Three-dimensional model matching from an unconstrained viewpoint, *Proc. of Int. Conf. on Robotics and Automation ICRA'87*, pp. 208–220.

Tomono, A., Iida, M. and Kobayashi, Y.: 1989, A tv camera system which extracts feature points for non-contact eye movement detection, *Proc. of the SPIE Conf. on Optics, Illumination and Image Sensing for Machine Vision*, pp. 2–12.

Townsend, W. T. and Salisbury, J. K.: 1993, Mechanical design for whole-arm manipulation, *Robots and Biological Systems: Toward a New Bionics?* pp. 153–164.

Triesch, J. and von der Malsburg, C.: 1996, Robust classification of hand postures agains complex backgrounds, *Proc. of the Int. Conf. on Automatic Face and Gesture Recognition*, IEEE Computer Society Press, pp. 170–175.

Triesch, J. and von der Malsburg, C.: 1998, A gesture interface for human-robot-interaction, *FG'98, The IEEE Third International Conference on Automatic Face and Gesture Recognition, April 14–16 in Nara, Japan*, IEEE.
*ftp://ftp.neuroinformatik.ruhr-uni-bochum.de/pub/manuscripts/articles/fg98reprint.

Triesch, J. and von der Malsburg, C.: 2000, Self-organized integration of adaptive visual cues for face tracking, *Proc. of the Int. Conf. on Face and Gesture Recognition FG'2000*, Grenoble, pp. 102–1007.

Ullman, S.: 1986, An approach to object recognition: Aligning pictoral descriptions, *AI Memo 931*, MIT.

Utsumi, A., Mori, H., Ohya, J. and Yachida, M.: 1998, Multiple-human tracking using multiple cameras, *Proc. of the IEEE Int. Conf. on Automatic Face and Gesture Recognition*, pp. 498–503.

Volpe, R. and Khosla, P.: 1993, A theoretical and experimental investigation of impact control for manipulators, *Int. J. on Robotics Research* **12**(4), 351–365.

Vranish, J. M. and Chauhan, D. S.: 1990, Tri-mode collision avoidance skin for robot arms in space, *Proc. of Int. Symposium on Robotics and Manufacturing*, pp. 189–195.

Wakita, Y., Hirai, S., Hori, T., Takada, R. and Kakikura, M.: 1998, Realization of safety in a coexistent robotic system by information sharing, *Proc. of the Int. Conf. on Robotics and Automation*, Leuven, Belgium, pp. 3474–3479.

Waldherr, S., Romero, R. and Thrun, S.: 2000, A gesture based interface for human-robot interaction, *Autonomous Robots* **9**, 151–173.

Walker, I. D.: 1994, Impact configurations and measures for kinematically redundant and multiple armed robot systems, *IEEE Trans. on Robotics and Automation* **10**(5), 670–683.

Wellner, P.: 1993, Interacting with paper on the digitaldesk, *Comm. of the ACM* **36**(7), 86–96.

Wren, C., Azarbayejani, A., Darrell, T. and Pentland, A.: 1996, Pfinder: Real-time tracking of the human body, *Proc. of the IEEE Int. Conf. on Automatic Face and Gesture Recognition*, pp. 51–56.

Wren, C., Azarbayejani, A., Darrell, T. and Pentland, A.: 1997, Pfinder: Real-time tracking of the human body, *IEEE Trans. on Pattern Analysis and Machine Intelligence* **19**(7), 780–785.

Wren, C. and Pentland, A.: 1998, Dynamic models of human motion, *Proc. of the IEEE Int. Conf. on Automatic Face and Gesture Recognition*, pp. 22–27.

Wu, A., Shah, M. and da Vitoria Lobo, N.: 2000, A virtual 3d blackboard: 3d finger tracking using a single camera, *Proc. of the Int. Conf. on Face and Gesture Recognition FG'2000*, Grenoble, pp. 524–529.

Wu, H., Shioyama, T. and Kobayashi, H.: 1998, Spotting recognition of head gestures from color image series, *Proc. of the Int. Conf. on Pattern Recognition*, pp. 83–85.

Wu, Y. and Huang, T. S.: 1999, Capturing articulated human hand motion: A divide-and-conquer approach, *Proc. of the Int. Conf. on Computer Vision ICCV'99*, Kerkyra, Greece, pp. 606–611.

Wu, Y. and Toyama, K.: 2000, Wide range, person- and illumination-insensitive head orientation estimation, *Proc. of the Int. Conf. on Face and Gesture Recognition FG'2000*, Grenoble, pp. 183–188.

Xu, L. and Zheng, Y. F.: 1999, Reflexive behavior of personal robots using primitive motions, *Proc. of the Int. Conf. on Robotics and Automation*, Detroit, pp. 3189–3194.

Xu, M. and Akatsuka, T.: 1998, Detecting head pose from stereo image sequences for active face recogntion, *Proc. of the IEEE Int. Conf. on Automatic Face and Gesture Recognition, FG'98*, pp. 82–87.

Xu, Y., Hollerback, J. and Ma, D.: 1995, A nonlinear pd controller for force and contact transient control, *IEEE Control Systems* **15**(1), 15–21.

Yamada, Y.: 1997, Evaluation of human pain tolerance and its application to designing safety robot mechanism for human-robot coexistance, *Journal on Robotics and Mechatronics* **9**(1), 65–70.

Yang, J., Lu, W. and Waibel, A.: 1998, Skin color modeling and adaptation, *Technical Report CMU-CS-97-146*, CMU.

Yang, J. and Waibel, A.: 1995, Tracking human faces in real-time, *Technical Report CMU-CS-95-210*, CMU.

Yang, J. and Waibel, A.: 1996, A real-time face tracker, *Proc. of IEEE Workshop on Applications of Computer Vision WACV'96*, pp. 142–147.

Yang, M.-H. and Ahuja, N.: 1998, Extracting gestural motion trajectories, *Proc. of the IEEE Int. Conf. on Automatic Face and Gesture Recognition*, pp. 10–15.

Yokoyama, T., Yagi, Y. and Yachida, M.: 1998, Active contour model for extracting human faces, *Proc. of the Int. Conf. on Pattern Recognition*, pp. 673–676.

Zhang, Y. and Kambhamettu, C.: 2000, Robust 3d head tracking under partial occlusion, *Proc. of the Int. Conf. on Face and Gesture Recognition FG'2000*, Grenoble, pp. 499–504.

Zheng, J. Y. and Suezaki, S.: 1998, A model based approach in extracting and generating human motion, *Proc. of the Int. Conf. on Pattern Recognition*, pp. 1201–1205.

Zhu, X., Yang, J. and Waibel, A.: 2000, Segmenting hands of arbitrary color, *Proc. of the Int. Conf. on Face and Gesture Recognition FG'2000*, Grenoble, pp. 446–453.

IEEE MAGAZINES AND NEWSLETTERS

The Institute of Electrical and Electronics Engineers, Inc.· 445 Hoes Lane · Piscataway NJ 08855

Advertising Space Insertion Order

Publication: IEEE Robotics & Automation Magazine **Insertion Date(s)**

Advertiser: Size/Shape:

Color: **B/W** 2/C(specify) 4/c Premium Position

Headline:
Material: New due Repeat

Reader Service No/**Yes** Reader Response Method(Yes/No): Mail **Email** Phone Fax

Frequency Rate Card No: Rate protected: (yes/)

Space rates: Color Charge:$ Premium Position:$
Gross:$ Net

Special Instructions:
Advertiser/Agency Contact:
Bill to:
Phone
Fax:
Email:
URL:

Signature/Title___PO Attached_____ DATE

Send insertion order and/or signed PO to:
Rosalyn Snyder
Managing Editor & Advertising Sales
3603 Octavia Street
Raleigh NC 27606 USA
r.g.snyder@ieee.org
(unil April 1)
Tel: 252 995 0018
Fax 252 995 0018 (call first)

Send Materials to :
Cathline Tanis
IEEE Advertising Production Manager
445 Hoes Lane
Piscataway NJ 08855
Tel: 732 562 3949; Fax: 732 981 1855
c.tanis@ieee.org

Contract and Copy Regulations: Advertisers and advertising agencies assume liability for content (including text, representations, illustrations, sketches, maps, trademarks or other copyrighted matter) of advertisements printed , and also assume responsibility for any claims arising therein made against the Publisher. Advertising is accepted at the discretion of the Publisher. The publisher reserves the right to reject any advertising which is not in keeping with the publication's standards. The publisher reserves the right to reject or amend employment advertising phrased in such a way as to imply age or wage discrimination. The advertiser is encouraged to specify experience minimums, not maximums.
Short rates and rebates: Advertisers will be short-rated or rebated to earned rates based on the number of insertions actually used during a 12 month period. Rate Protection Policy: Orders are accepted subject to change in rates upon notice from the publisher. Contracts may be canceled at the time the rate change becomes effective without incurring a short-rate adjustment if the frequency rate upon which billing is based has been earned up to the date of cancellation. The publisher's liability for any error will not exceed the charge for the advertisement in question.

Cancellations not accepted after closing dates. Cancellations not accepted on covers and preferred positions without written notice to the publisher 30 days before closing date.

No advertisement will be placed without a signed insertion order. IEEE Magazine Representative: Rosalyn G. Snyder