Fusion and Regularisation of Image Information in Variational Correspondence Methods

Jarno Ralli

Department of Computer Architecture and Technology

University of Granada

A thesis submitted for the degree of

PhilosophiæDoctor (PhD)

2011 December

Modelos Variacionales de Correspondencia para Fusión y Regularización de la Información en Procesamiento de Imágenes

Jarno Ralli

Departamento de Arquitectura y Tecnología de Computadores

Universidad de Granada

Para optar al grado de

PhilosophiæDoctor (PhD)

2011 Diciembre

Declaración

Dr. Eduardo Ros Vidal, Catedrático de Universidad del Departamento de Arquitectura y Tecnología de Computadores de la Universidad de Granada, y Dr. Javier Alonso Díaz, profesor de Universidad del Departamento de Arquitectura y Tecnología de Computadores de la Universidad de Granada,

CERTIFICAN:

Que la memoria titulada 'Fusion and Regularisation of Information in Variational Correspondence Methods', ha sido realizada por D. Jarno Samuli Ralli bajo nuestra dirección en el Departamento de Arquitectura y Tecnología Computadores de la Universidad de Granada para optar al grado de Doctor por la Universidad de Granada.

Granada, 19 de Diciembre de 2011

Fdo. Eduardo Ros Vidal Fdo. Javier Alonso Díaz

Abstract

In this work we study, and improve, applicability of variational correspondence methods, used for calculating dense optical-flow and disparity fields, to real scenes with realistic illumination conditions. It is well known that under realistic illumination conditions and image noise, proper image representation is crucial in order to generate both correct and temporally coherent optical-flow and disparity fields. We have studied 34 different image representations and ranked these with respect to accuracy, robustness and a combination of accuracy plus robustness. In the case of well known test images with optimal (or quasi optimal) illumination conditions, effects of image representation are not that important. On the other hand, with scenes from real applications, such as robotic grasping or vision in automotive industry, influence of image representation is crucial. Also we have extended the basic models to include both spatial- and temporal-constraints. In the case of optical-flow, for example, temporal constraint reduces 'flickering' of estimations. By flickering we mean temporal changes in the displacement fields due to lack of or ambiguity of spatial features in the images. We show that by using spatial constraints in the disparity estimation, considerable improvements are possible. These constraints are due to (a) what we know of the solution before hand (e.g. roads are relatively flat surfaces, sky is far away) or (b) what we can deduce from the scene itself. Effectively, we show how these constraints can be obtained and refined in a hypothesis-forming-validation loop (HFVL).

In the example that we give of a HFVL loop, we segment an initial disparity map and form constraint(s) based on the segments and feed back these into the disparity calculation.

Not only do the disparity estimations improve, but we also segment the depth map into meaningful surfaces

Apart from introducing the principal results obtained, we also explain in detail how the models that we have used can be solved. Therefore, this work (along with the related Matlab/MEX code available at `http://http://www.jarnoralli.fi/joomla/code`) will hopefully help other scientists understand and further improve the variational methods for calculating stereo-disparity and optical-flow.

Resumen

En este trabajo se ha estudiado, y mejorado, la aplicabilidad de los métodos variacionales para el cálculo de flujo-óptico y disparidad en secuencias reales bajo condiciones realistas de iluminación. Es conocido que, bajo condiciones realistas de iluminación y ruido en la imágenes, la representación apropiada de la imagen es crucial para generar estimaciones correctas y coherentes, temporalmente, por ejemplo para el flujo-óptico y la disparidad. Hemos estudiado 34 representaciones diferentes de imágenes y estas han sido clasificadas con respecto a la precisin, la robustez y una combinación de la precisión y la robustez. En el caso de imágenes más usadas en la literatura, típicamente las condiciones de iluminación son óptimas (o casi óptimas) y, por lo tanto, el efecto de la representacin no es tan importante. Por otro lado, con escenas de las aplicaciones reales, como la de robótica o la visión en la industria automotriz, la influencia de la representación es crucial, como se muestra en este trabajo. También hemos ampliado los modelos básicos para incluir restricciones (ingl. constraints) espaciales y temporales. En el caso del flujo-óptico, por ejemplo, la restricción temporal reduce 'parpadeo' (ingl. flickering) de las estimaciones. Con el parpadeo nos referimos a los cambios temporales en los campos de flujo-óptico debido a la falta de, o a la ambigüedad de información espacial para que el modelo pueda generar estimaciones correctas. Hemos mostrado que mediante el uso de restricciones espaciales en la estimación de la disparidad se producen mejoras considerables. Estas restricciones se deben a: (a) lo que sabemos de la solución de antemano (por ejemplo las carreteras son superficies relativamente planas, el cielo está muy lejos) o (b) lo que podemos deducir de la propia escena. Efectivamente, hemos demostrado que estas restricciones pueden ser obtenidas y refinadas en bucles de formación y convalidación de hipótesis (ingl. hypothesis-forming validation loop, HFVL). En el ejemplo

que damos de un bucle de HFVL, primero segmentamos un mapa de disparidad inicial y a partir de esto, formamos la restricción (o restricciones) y retroalimentamos esta (estas) en el calculo de disparidad. Así, no solamente se mejoran las estimaciones de la disparidad, pero también a la vez segmentamos el mapa de profundidad en superficies significativas.

Aparte de introducir los resultados principales obtenidos, en este documento también explicamos, en detalle, como se pueden resolver los modelos que hemos utilizado. Por lo tanto, se espera que este trabajo (junto con el código Matlab/MEX disponible en `http://http://www.jarnoralli.fi/joomla/code`) ayude a otros científicos comprender y mejorar los métodos variacionales para el cálculo de flujo-óptico y la disparidad.

To my family

Acknowledgements

I would like to acknowledge my family, especially my wife Claudia Reyes and my parents Seppo and Marja Ralli, for their support during the process of making this PhD dissertation. Without their help, none of this would have been possible. I would like to express my gratitude for Dr. Eduardo Ros and Dr. Javier Díaz for supervising and believing in me during all these years. Sometimes, when the going got rough, their positiveness kept me going on the right track. Also, I would like to express my gratitude for Dr. Francisco Pelayo for intruducing me into the world of science and for acting as a supervisor for my MsC thesis (I really mean it Paco!!). Last, but certainly not the least, I would like to thank the following persons (not in any particular order): Dr. Florian Pokorny, from Royal Institute of Technology (Sweden) for his invaluable help with the mathematics, Dr. Karl Pauwels and B.Sc Emma Matikainen for helping with proof reading, Dr. Ville Kyrki from Lappeenranta University of Technology (Finland) for our discussions, Dr. Danica Kragic, Dr. Carl-Henrik Ek, and Dr. Mårten Björkman from Royal Institute of Technology (Sweden) for their valuable ideas, Dr. Javier Sánchez Pérez, and Dr. Luis Alvarez León from Universidad de Las Palmas de Gran Canaria (Spain) for helping me getting started with the variational methods.

I thank my collegues from CIE/CITIC, Pablo Guzmán, Mauricio Vanegas, Silvia Tolu, Niceto Luque, Jesús Garrido, Sara Granados, Richard Carillo, Matteo Tomasi, Francisco Barranco, Rodrigo Agís, Rafael Rodríguez, and Enrique Fernández for putting up with my odd finnish behaviour and humor, and for throwing all the barbacues and parties!

Finally, I thank the following projects/parties for providing funding to make this thesis: European Projects DRIVSCO (FP6-IST-FET) and TOMSY

(FP7-270436), National (Spanish) grants DINAM-VISION (DPI2007-61683), ARC-VISION (TEC2010-15396), MULTIVISION (TIC-3873) and RECVIS (TIN2008-06893-C03-02) and the FPI-program (Formación de Personal Investigador) by the Spanish Ministry of Education.

Contents

List of Figures **xi**

List of Tables **xix**

Glossary **xxi**

1 Introduction **1**

1.1 General 1

1.2 Scientific Objectives 7

1.3 Project Framework 8

1.3.1 DRIVSCO 9

1.3.2 GRASP 9

1.3.3 TOMSY 10

1.3.4 DINAM-VISION 11

1.4 Methods and Tools 11

1.5 Organization of Chapters 12

2 Introducción en Castellano **13**

2.1 General 13

2.2 Objetivos Cientificos 19

2.3 Marco de Proyectos 21

2.3.1 DRIVSCO 21

2.3.2 GRASP 22

2.3.3 TOMSY 23

2.3.4 DINAM-VISION 23

2.4 Métodos y Herramientas 24

2.5 Organización de los Capítulos 25

3 Variational Correspondence Methods 27

3.1 Introduction 27
3.2 Organisation of Sections 30
3.3 What is Meant by Calculus of Variations? 30
3.4 Motivation for Variational Correspondence 31
3.5 Optical-flow 33
3.5.1 Early Linearisation 33
3.5.2 Late Linearisation 36
3.5.2.1 Coarse-to-fine and Multi-resolution 38
3.5.2.2 Large Displacements 38
3.5.3 Summary 39
3.6 Stereo Disparity 40
3.6.1 Late Linearisation 40
3.7 Smoothness Terms and Error Functions 41
3.8 Problems with the Models 44
3.9 Robust Data Terms 48
3.9.1 Motivation 48
3.9.2 Background Material and Related Work 50
3.9.2.1 Related Work and Our Contribution 50
3.9.2.2 Sources of Error 51
3.9.2.3 Variational Stereo 51
3.9.3 Searching for Optimal Parameters with Differential Evolution . . 52
3.9.4 Image Transformations 53
3.9.4.1 RGBN (Normalized RGB) 54
3.9.4.2 Gradient 54
3.9.4.3 Gradient Magnitude 55
3.9.4.4 HS(V) 55
3.9.4.5 Spherical 56
3.9.4.6 Log-Derivative 56
3.9.4.7 Phase Component of Band-pass Filtered Image Using Quadrature Filters 56

3.9.5 Experiments 58
3.9.5.1 K-fold Cross-Validation 60
3.9.5.2 Induced Illumination Errors and Image Noise 60
3.9.5.3 Error Metric 62
3.9.6 Results 62
3.9.6.1 Ranking 62
3.9.6.2 Improvement Due to Combined Representation Spaces 66
3.9.7 Visual Qualitative Interpretation 68
3.9.8 Conclusions 71
3.10 Spatial and Temporal Constraints 72
3.10.1 Motivation 73
3.10.2 Related Work and Our Contribution 74
3.10.3 System Scheme 75
3.10.4 Extended Models 76
3.10.5 Data Terms 77
3.10.6 Regularization Terms 79
3.10.7 Spatial- and Temporal Constraints 80
3.10.8 Predicted Temporal Constraints 81
3.10.9 Experiments 82
3.10.9.1 Error Metrics 82
3.10.9.2 Quantitative Results for Spatial Constraint in Disparity Calculation 83
3.10.9.3 Quantitative Results for Temporal Constraint in Optical-flow Calculation 85
3.10.9.4 Qualitative results for spatio-temporal constraints . . . 87
3.10.10 Conclusions 90
3.11 Summary 90

4 Segmentation of Disparity Maps 91
4.1 Introduction 91
4.2 Motivation for Level-sets 91
4.3 Implicit Surfaces 92
4.3.1 Dynamic Implicit Surfaces 94

4.3.2 Mean Curvature Motion 95
4.4 Hypothesis-Forming-Validation-Loops and Segmentation 96
4.4.1 Motivation 97
4.4.2 Related Work and Our Contribution 98
4.4.3 Hypothesis-Forming-Validation-Loop 99
4.4.3.1 Variational Stereo 100
4.4.3.2 Data Terms 100
4.4.3.3 Regularization Term 101
4.4.3.4 Spatial Constraint 101
4.4.4 Segmentation 102
4.4.4.1 Two Regions 103
4.4.4.2 Standard Model 103
4.4.4.3 Surface Segmentation Model 105
4.4.4.4 Multi-Region 107
4.4.4.5 Solving the Equations 108
4.4.4.6 Segmentation Algorithm 111
4.4.5 Experiments 112
4.4.5.1 Error Metrics 112
4.4.5.2 GRASP and Middlebury Images 114
4.4.5.3 Reference Results for Middlebury 115
4.4.5.4 Segmentation for Robotic Grasping 119
4.5 Conclusions 124

5 Solving the Equations 125
5.1 Introduction 125
5.2 A Word About the Used Notation 126
5.2.1 Pixel Numbering Schemes 128
5.2.2 Pixel Neighbourhoods 128
5.3 Numerical Methods 130
5.3.1 SOR 131
5.3.2 Jacobi 131
5.3.3 Gauss-Seidel 132
5.3.4 TDMA/ALR 133

5.3.5 Multigrid 136
5.4 Equations to be Solved 140
5.5 Finite Difference Discretisation 142
5.5.1 Finite Difference Operators 143
5.5.2 Discretization of DIV Operator 144
5.6 Solving Optical-Flow 147
5.6.1 Early Linearisation 147
5.6.1.1 Coarse-To-Fine Algorithm 150
5.6.1.2 SOR-Jacobi 152
5.6.1.3 SOR-GS 152
5.6.1.4 GS-ALR 153
5.6.1.5 Results 154
5.6.2 Late Linearisation 155
5.6.2.1 Coarse-To-Fine Algorithm 160
5.6.2.2 SOR-Jacobi 163
5.6.2.3 SOR-GS 165
5.6.2.4 GS-ALR 167
5.6.2.5 Results 169
5.7 Solving Level-set Equation 170
5.7.1 SOR-GS 173
5.8 Conclusions 173

6 Conclusions 175
6.1 Summary 175
6.2 Future Work 177
6.3 Publications 177
6.4 Main Contributions 178

7 Conclusiones en Castellano 181
7.1 Sumario 181
7.2 Trabajo Futuro 183
7.3 Publicaciones 183
7.4 Contribuciones Principales 185

References **187**

A Euler-Lagrange Equations **197**

A.1 Temporal Constraint for Optical-Flow 197

A.1.1 Energy Functional 197

A.1.2 Related Euler-Lagrange Equations 198

B Solvers **199**

B.1 Optical-flow 199

B.1.1 Discrete Differential Operator 199

B.1.1.1 Early Linearisation 200

B.1.1.2 Late Linearisation 201

C Matlab/MEX Code **203**

C.1 Introduction 203

C.2 Notation Used in the Matlab Code 203

C.2.1 Early Linearisation 203

C.2.1.1 Solver 204

C.2.2 Late Linearisation 205

C.2.2.1 Solver 207

List of Figures

1.1 DRIVSCO car. (a) DRIVSCO optical system; (b) corresponding GUI (Graphical User Interface). 3

1.2 GUI showing the detected lanes in the current image and a history of prediction and steering angles. 4

1.3 System architecture. IMO, PAR and GPU denote Independently Moving Objects, Perception-Action Repository and Graphical Processing Unit, respectively. 4

1.4 Pot. (a) Random dot stereogram; (b) related disparity image. 5

1.5 Shark (a) Random dot streogram; (b) related disparity image. 6

1.6 (a) left stereo-image; (b) related disparity map; (c) segmentation. 7

1.7 Local features. (a) Lena image; (b) energy; (c) orientation; (d) phase. In the case of orientation, color is used to codify the orientation of the edge in question. 7

2.1 DRIVSCO coche. (a) sistema óptico de DRIVSCO; (b) la GUI (ingl. Graphical User Interface) correspondiente. 15

2.2 La interfaz que en este caso demuestra los carriles junto con la historia de predicción de ángulos de la dirección (volante). 16

2.3 Arquitectura del sistema. IMO, PAR y GPU significan significan, en inglés, Independently Moving Objects, Perception-Action Repository y Graphical Processing Unit. 16

2.4 Tetera. (a) Estereograma de puntos aleatorios; (b) la disparidad asociada. 17

2.5 Tiburon. (a) Estereograma de puntos aleatorios; (b) la disparidad asociada. 19

2.6 (a) Imagen izquierda; (b) el mapa de disparidad asociado; (c) resultados de segmentación. 19

2.7 Características locales. (a) Imagen de Lena; (b) la energía; (c) la orientación; (d) la fase. En el caso de la orientación, el color codifica la orientación del borde en cuestión. 20

3.1 Binocular disparity. 29

3.2 A robotics related disparity example. (a) Left stereo-image; (b) corresponding disparity map; (c) 3D-reconstruction of the object of interest. In the case of the disparity map, gray-level codifies the disparity: objects with dark tones are closer to the cameras, while objects with light tones are further away from the cameras. 29

3.3 Optical flow. (a) image at t=0; (b) corresponding optical-flow. Colour codifies direction and intensity velocity. 30

3.4 Error- and influence functions. (a) Error functions; (b) corresponding influence functions. $\epsilon = 0.3$ and $\lambda = 0.75$. 43

3.5 Optical-flow results for the Beanbags case. (d) displays the frame11 warped using the optical-flow field given in (c). 44

3.6 Right image warped by (a) calculated disparity map and (b) the ground-truth. 45

3.7 Tsukuba case: (a) left image; (b) right image; (c) symmetrical disparity, left-right; (d) symmetrical disparity, right-left; (e) ground truth; (f) occlusions (based on (c) and (d)). 46

3.8 Phase response of Cones stereo-image: (a) original image; (b) phase response corresponding to $\theta = 0°$; (c) phase response corresponding to $\theta = 22.5°$; (d) phase response corresponding to $\theta = 45°$. 58

3.9 Stereo-images from the Middlebury database used in the quantitative experiments. (a) Aloe; (b) Art; (c) Baby1; (d) Baby2; (e) Baby3; (f) Books; (g) Bowling; (h) Cloth1; (i) Cloth2; (j) Cloth3; (k) Cones; (l) Dolls; (m) Lampshade1; (n) Lampshade2; (o) Laundry; (p) Moebius; (q) Plastic; (r) Reindeer; (s) Rocks1; (t) Rocks2; (u) Teddy; (v) Tsukuba; (x) Venus; (y) Wood1; (z) Wood2. 59

3.10 Baby2. (a) Original; (b) Local multiplicative (LM) plus local additive (LA); (c) Severe luminance (LS); (d) Severe salt&pepper (SPS). 61
3.11 Results for $\nabla I(RGB)$, phase and logd. Org. means original images, while Comb. means combined error and noise (error+noise in the tables). 67
3.12 Results for $(r)\phi\theta$ (spherical), HS(V) and RGB. Org. means original images, while Comb. means combined error and noise (error+noise in the tables). 68
3.13 Cones. (a) Ground truth; (b) $\nabla I(RGB)$; (c) $\nabla I(RGB)$+phase; (d) $\nabla I(RGB)$+HS(V); (e) phase; (f) RGB. 69
3.14 DRIVSCO scene. (a) Left image; (b) $\nabla I(RGB)$; (c) $\nabla I(RGB)$+phase; (d) $\nabla I(RGB)$+HS(V); (e) phase; (f) RGB. 70
3.15 GRASP scene. (a) Left image; (b) $\nabla I(RGB)$; (c) $\nabla I(RGB)$+phase; (d) $\nabla I(RGB)$+HS(V); (e) phase; (f) RGB. 71
3.16 Use of spatial constraint in disparity calculation. Dsc (d_{sc} in the text) is the spatial constraint while D is the solution (disparity). This particular case shows how the knowledge of the geometrical setup, in this case, related to the form of the road, of the scene can be used to constrain the solution. 75
3.17 Use of spatial and temporal constraints in optical-flow calculation. Usc and Vsc (u_{sc} and v_{sc} in the text) are the spatial constraints while Utc and Vtc (u_{tc} and v_{tc} in the text) are the temporal constraints. Up and Vp (u_p and v_p in the text) is the predicted optical-flow at time $t+1$. . . 76
3.18 Influence functions $\Psi'_{CS}(s^2) = 1/(1+\frac{s^2}{\lambda^2})$ and $\Psi'_{CT}(s^2) = exp(-\frac{s^2}{\lambda^2})$ for λ of 0.2. 80
3.19 (a) Monopoly; (b) Midd1; (c) Midd2. 83
3.20 (a) Monopoly ground-truth; (b) Midd1 ground-truth; (c) Midd2 ground-truth; (d) Monopoly, mixed-diffusion without SC (e) Midd1, mixed-diffusion without SC; (f) Midd2, mixed-diffusion without SC; (g) Monopoly SC; (h) Midd1 SC; (i) Midd2 SC; (j) Monopoly, mixed-diffusion with SC; (k) Midd1, mixed-diffusion with SC; (l) Midd2, mixed-diffusion with SC. SC stands for spatial constraint. 85
3.21 (a) Monopoly; (b) Midd1; (c) Midd2. In each case the spatial constraint is 0 and mixed regularisation is used. 86

3.22 Sequences (a) Rubberwhale; (b) Grove2; (c) Grove3; (d) Hydrangea; (e) Urban3; (f) Yosemite (with clouds). 86
3.23 Frame 670. (a) Left image; (b) disparity without spatial constraint; (c) disparity with spatial constraint; (d) velocity; (e) velocity with temporal constraint; (f) velocity with temporal and spatial constraint. Only the optical-flow module is given in the figure. 88
3.24 Video-surveillance application. (a) Left image; (b) constraint; (c) disparity without spatial constraint; (d) disparity with spatial constraint. It is difficult to spot the suitcase, even for a human observer, from the left image or the disparity map without constraint. On the other hand, by using a constraint for the floor, the results for disparity improve so that detecting the object of interest becomes considerably easier. 89

4.1 (a) Graph of an implicit function $\Phi(x, y)$; (b) Graph of an implicit function $\Phi(x, y)$ with 'zero' plain (yellow) plane cutting it at $z = 0$. 92
4.2 Zero isocontour (green) of the implicit function with inside- (Ω_1) and outsize (Ω_2) regions. 93
4.3 Normal of the implicit function Φ. $\vec{N} = \nabla\Phi/|\nabla\Phi|$. 96
4.4 Data flow in the proposed method, showing only one iteration cycle of HFVL. 100
4.5 Influence functions $\Psi'_{CS\{1\}}(s^2) = 1/(1+\frac{s^2}{\lambda^2})$ and $\Psi'_{CS\{2\}}(s^2) = exp(-\frac{s^2}{\lambda^2})$ for λ of 0.2. 102
4.6 Terms p and $\bar{p}$ for $\sigma^2 = 1$, as defined in (4.19). 106
4.7 Surface segmentation for Tsukuba. (a) Disparity ground truth; (b) 1st segment; (c) 2nd segment; (d) 3rd segment; (e) 4th segment; (f) 5th segment. 107
4.8 Multi-grid V- and W-cycles. 109
4.9 Graph of the regularised Heaviside function, as defined in (4.25). 111
4.10 Initialisation and the first five iteration cycles for Tsukuba. (a) Initialisation; (b) 1st iterationM; (c) 2nd iteration; (d) 3rd iteration; (e) 4th iteration; (f) 5th iteration. 113
4.11 Middlebury images (a) Tsukuba; (b) Teddy; (c) Cones. 114

4.12 GRASP images (a) GRASP 1; (b) GRASP 2; (c) GRASP 3; (d) GRASP 4. 115
4.13 Disparity maps: (a) Tsukuba GT; (b) Teddy GT; (c) Cones GT; (d) Tsukuba calculated; (e) Teddy calculated; (f) Cones calculated. GT stands for groud-truth. 116
4.14 Segmentation results: (a) Tsukuba GT; (b) Teddy GT; (c) Cones GT; (d) Tsukuba calculated; (e) Teddy calculated; (f) Cones calculated. . . 117
4.15 (a) Disparity WO constraint; (b) Disparity with constraint; (c) Segmentation based on a; (d) Segmentation based on b. With the constraint the estimations improve for the background, especially near the table legs. WO stands for 'without'. 118
4.16 GRASP 1 (a) Disparity (none); (b) Disparity (const.); (c) Segmentation (based on a); (d) Segmentation (based on b). Constraining the solution leads to better disparity estimations for the table, especially near the objects of interest, therefore leading to better scene interpretation. . . . 119
4.17 GRASP 2 (a) Disparity (none); (b) Disparity (const.); (c) Segmentation (based on a); (d) Segmentation (based on b). Segmentation based on the constrained disparity map shows considerable improvement for the object of interest situated on the table. 120
4.18 GRASP 3 (a) Disparity (none); (b) Disparity (const.); (c) Segmentation (based on a); (d) Segmentation (based on b). 121
4.19 GRASP 4 (a) Disparity (none); (b) Disparity (const.); (c) Segmentation (based on a); (d) Segmentation (based on b). 121
4.20 Object of interest: (a) GRASP 1; (b) GRASP 2; (c) GRASP 3; (d) GRASP 4. 122
4.21 3D reconstruction of objects of interest: (a) GRASP 1; (b) GRASP 2; (c) GRASP 3; (d) GRASP 4. 123

5.1 A Cartesian grid. The blue grid defines the image pixels while the red grid is the computational grid used for solving the PDEs. We assume that $h_x = h_y$. Origin of the grid is in the left upper corner and pixel positions are defined by subindices (i, j). 127

5.2 Directions on a grid. To simplify the notation, both cardinal- and inter-cardinal directions are used. Here W, N, E, S, and C refer to west, north, east, south, and centre, respectively. 127

5.3 Column- and row-wise pixel orderings. Here, $\mathcal{I} \in \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$. 128

5.4 Pixel neighbourhoods, where central pixel, $\mathcal{I}$, is denoted with a double circle. (a) Painted circles denote neighbouring pixels $\mathcal{J}$ that belong to the neighbourhood defined by $\mathcal{J} \in N(\mathcal{I})$. (b) Painted circles denote pixels $\mathcal{J}$ that belong to the neighbourhood defined by $\mathcal{J} \in N^{-}(\mathcal{I})$, while unpainted circles denote pixels $\mathcal{J}$ that belong to the neighbourhood defined by $\mathcal{J} \in N^{+}(\mathcal{I})$. (c)-(d) Painted circles denote pixels $\mathcal{J}$ that belong to the neighbourhood defined by $\mathcal{J} \in N^{-}(\mathcal{I})$, while unpainted circles denote pixels $\mathcal{J}$ that belong to the neighbourhood defined by $\mathcal{J} \in N^{+}(\mathcal{I})$. Processing order is defined by the arrows. Due to the eliminated boundary conditions 'scheme', the pixel neighbourhood operators only point to valid neighbours, as shown in (a) and (b). 129

5.5 Column- and row-wise pixel ordering. In ALR we apply, for example, TDMA first along all the columns and then along all the rows. (a) Column wise ordering; (b) row wise ordering. In both the cases we show the processing order. 135

5.6 Pixel positions a_i and c_i of b_i, depending on the pixel ordering: (a) column wise ordering; (b) row wise ordering. Painted and unpainted circles denote the pixel neighbours, in Gauss-Seidel fashion, that are taken into account when calculating the solution: painted circle denotes a neighbour that contains a solution from the current iteration step, $l+1$, while unpainted circle denotes a neighbour that contains solution from the last iteration step l. 135

5.7 A fine (Ω_h) and a coarse (Ω_H) grid, in corresponding order. 138

5.8 Multigrid V- and W-cycles and a unidirectional Full Multigrid (FMG) cycle. The unidirectional cycle is also known as coarse-to-fine scheme. . 139

5.9 Image pyramid of Tsukuba with a scale factor of 0.5: (a) scale 1; (b) scale 2; (c) scale 3; (d) scale 4. 140

5.10 Double circle denotes the position of interest while simple circles are the neighbouring positions W,N,E,S; (b) shows the eliminated boundary conditions. 146
5.11 Early linearisation results. 155
5.12 Late linearisation results. In the last row we show (k) optical-flow and (l) saturated optical-flow. With saturated we refer to how the flow is displayed. By limiting the maximum movement, also the objects with smaller displacements can be observed. 170

B.1 201

List of Tables

3.1 Possible applications of optical-flow and disparity. 28
3.2 Pros and cons of variational correspondence methods. 32
3.3 Error functions used in the smoothness term. 43
3.4 Tested image representation combinations. 54
3.5 Ranking for combined noise+error and original images. 64
3.6 Combined ranking 65
3.7 Used notation for images and error functions. 76
3.8 Results in MAE (mean average error) and percentage of correct disparities using different regularisations without (WO) and with (W) a spatial constraint. 84
3.9 Results in AAE with and without temporal constraint. W.TEM, WO.TEM and P.ERR denote 'with temporal constraint', 'without temporal constraint' and 'prediction error' correspondingly. 87

4.1 Results in MAE (mean average error) and C (percentage of correct disparities). Const. stands for constrained. The constraint is obtained, using the segmentation algorithm, as mentioned in the text. 117

5.1 Typical parameters: coarse-to-fine algorithm for early linearisation (m and n refer to number of columns and rows). 151
5.2 Typical parameters: coarse-to-fine algorithm for early linearisation (m and n refer to number of columns and rows). 162

Glossary

(u, v)	Optical-flow (displacement) field. u is the horizontal component of movement, while v is the vertical component.
κ	Curvature of the interface, where $\kappa = \nabla \cdot \vec{N}$.
∇	Spatial gradient operator $\left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}\right)^T$.
Ω	Image domain.
Ω_h	Discretised image domain, where $\Omega_h = \Omega \cup G_h$ (see discretisation grid).
Ω_i	In the case of level sets, segments i defined by implicit functions Φ_i.
$\Phi(x, y)$	Implicit function used for defining level-sets.
$\Psi(s^2)$	Robust error function.
$\Psi'(s^2)$	Influence/penaliser function of $\Psi(s^2)$.
d	Disparity (displacement) field. Horizontal component in rectified images.
$DIV(F)$	Divergence operator, where F is a differentiable vector function.
$E(d)$	Energy of the stereo model (functional).
$E(u, v)$	Energy of the optical-flow model (functional).
G_h	Discretisation grid, where $G_h := \{(x, y) \mid x = x_i = ih_x,\ y = y_j = jh_y\,;\ i, j \in \mathbb{Z}\}$.
$I(x, y, k, t)$	Image where, t defines the time and k defines the channel (i.e. channels R, G or B in the case of RGB images).

$I_{\{L,R\},\{0,1\}}$ Stereo image, where $\{L, R\}$ refers to Left- or Right image taken at time $t = 0$ or $t = 1$.

$I_{\{L,R\}}$ Stereo image, where $\{L, R\}$ refers to Left- or Right image.

$I^w_{k,0}$ Warped image in the case of optical-flow, $I^w_{k,0} := I(x+u, y+v, t)$. All channels (k) are warped.

$I^w_{R,k}$ Warped image in the case of disparity, $I^w_{R,k} := I(x + d, y, t)$. All channels (k) are warped.

1

Introduction

1.1 General

Genesis. As per the UNECE report *Statistics of Road Traffic Accidents in Europe and North America*[1], during the decade 1998 - 2008, there were on average 150.000 killed and 5.5 million injured ANNUALLY in more than 3.8 million traffic accidents in ECE countries (Europe and North America). Therefore, it is no wonder that there is a great interest in developing both passive- and active vehicle safety systems to decrease both the number of accidents and the consequences. Broadly speaking, the term *passive safety* refers to the components of the vehicle that protect the driver and the passengers, while the term *active safety* refers to the systems that help to prevent the accident from happening in the first place. Amongst the best known passive safety systems are seatbelts, airbags, laminated windshields, side impact protection beams and so on. On the other hand, an interesting sub-class of active safety systems are so called *Advanced Driver Assistance Systems* (ADAS). Such systems are, for example, lane departure warning systems, traction control systems, infrared night vision and so on.

This thesis got started as part of a research project DRIVSCO (Learning to Emulate Perception-Action Cycles in a Driving School Scenario[2], European Commission, FP6) related to a driver assistance system capable of adapting to a particular drivers driving style: *Most technical systems, for example cars, must work reliably at key-turn.*

[1] `http://live.unece.org/trans/main/wp6/publications/stats_accidents2011.html`
[2] `http://cordis.europa.eu/fetch?CALLER=PROJ_ICT&ACTION=D&CAT=PROJ&RCN=80441`

Therefore, such systems almost always employ conventional control strategies. Biological systems, on the other hand, learn. In the beginning they are functional only at a very basic level from which they improve their skills. No-one would, however, want to use a learning car, which could in the beginning barely steer. Thus, learning techniques have not really entered turn-key applications so far.
The goal of DRIVSCO is to devise, test and implement a strategy of how to combine adaptive learning mechanisms with conventional control, starting with a fully operational human-machine interfaced control system and arriving at a strongly improved, largely autonomous system after learning, that will act in a proactive way using different predictive mechanisms. In other words, the system needs to be able to analyze the driving situation and make short term predictions of how things will evolve. To this end, it is not enough only to analyze typical information coming from the central computer of a vehicle, such as velocity, steering wheel position, gas- or brake pedal positions, or even GPS signal. It can be argued that visual perception is the single most important sensory information that we humans use when driving vehicles. This assumption is also backed by the fact that many driving simulators are based only on visual stimulus (more advanced simulators include other sensory information as acceleration and so on). Therefore, the front-end of the system, used for perception and action extraction, is based on image processing. By combining visual stimulus with the other information available from the vehicle, meaningful contextual interpretations of the situation can be derived. The visual system is based on cameras working on both the visible- and the infrared wavelengths of light, thus allowing the system to be used during the night as well.

This short introduction brings us to the subject of this thesis which is related to *artificial vision* (also known as computer vision, machine vision, or image processing). Before jumping into the actual subject of artificial vision, we will mention that there have been and are several projects similar to DRIVSCO going on at the time of writing this thesis. Here are a few just to mention some: DIPLECS (Dynamic Interactive Perception-Action Learning in Cognitive Systems[1], European Commision, FP7), COSPAL (Cognitive Systems using Perception-Action System[2], European Commission,

[1] `http://www.diplecs.eu/`
[2] `http://www.cospal.org/`

FP6) and IVSS (Intersection accidents: Analysis and Prevention [1], The Program Board for Automotive Research, Sweden).

Figure 1.1 displays both the optical system and the screen displaying information for the driver in the DRIVSCO project. Figures 1.2 and 1.3 shows both a GUI (Graphical User Interface) and system architecture related to the system detecting inconsistent driving behavior and IMOs (Independently Moving Objects) on a collision coarse [46].

(a) (b)

Figure 1.1: DRIVSCO car. (a) DRIVSCO optical system; (b) corresponding GUI (Graphical User Interface).

By the time of writing this thesis Volvo announced its new collision warning system with fully automatic breaking. Also, as will be shown later, the very same techniques researched/designed in this thesis initially in the framework of the DRIVSCO project, are general enough to be used in other fields where visual cues play an important role, such as robotics.

Apart from this thesis, several others were made (or are being written currently) related to the DRIVSCO project. Here I will mention of some them, in no particular order: Karl Pauwels (Katholieke Universiteit Leuven, Leuven, Belgium), Anders Kær-Nielsen[2] and Lars Baunegaard With Jensen[3] (University of Southern Denmark, Odense, Denmark), Irene Markelić[4] (Georg-August-University Göttingen, Göttingen, Germany) and Matteo Tomasi[5] (University of Granada, Granada, Spain).

[1] `http://www.ivss.se/`
[2] `http://www.mip.sdu.dk/people/PhD_students/akn.html`
[3] `http://www.mip.sdu.dk/people/PhD_students/lbwj.html`
[4] `http://www.markelic.de/`
[5] `http://atc.ugr.es/~mtomasi/`

Figure 1.2: GUI showing the detected lanes in the current image and a history of prediction and steering angles.

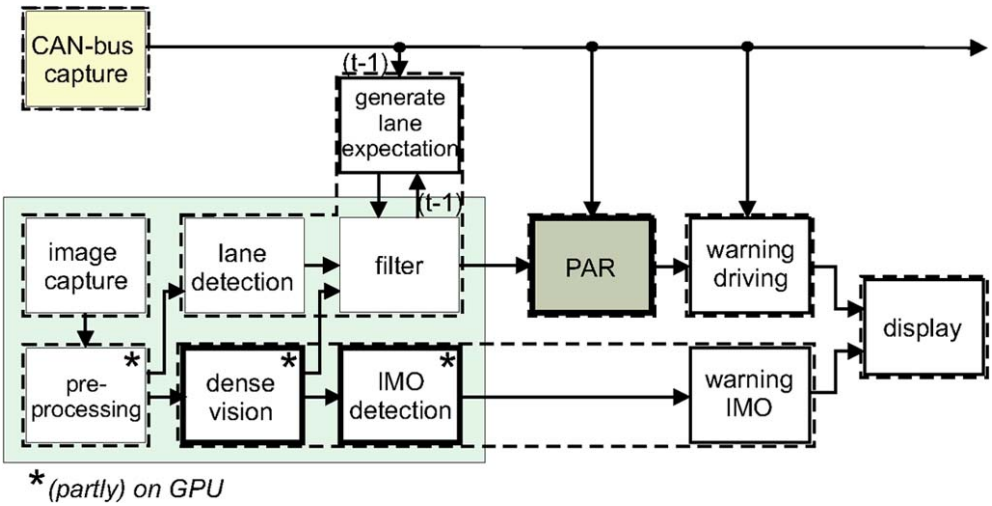
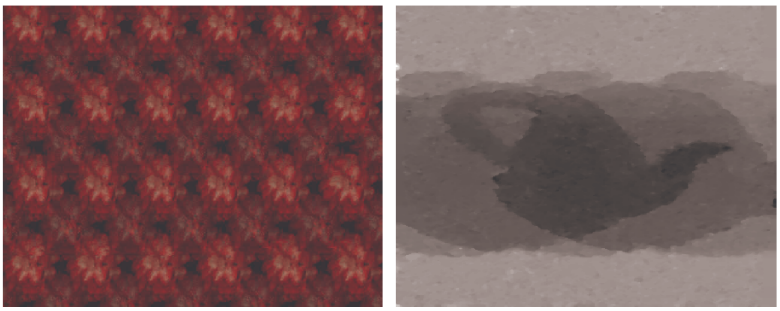


Figure 1.3: System architecture. IMO, PAR and GPU denote Independently Moving Objects, Perception-Action Repository and Graphical Processing Unit, respectively.

Artificial Vision. The field of artificial vision is typically associated with the study of *artificial intelligence* and, indeed, these fields employ many similar techniques such as pattern recognition and machine learning. Also many of the goals are similar. Since making a machine that 'understands' the scene being perceived through cameras has

shown to be much more difficult problem that was thought initially, modern approach to artificial vision adopts techniques/models from the biology: evolution has had much more time to come up with viable solutions than the mankind has even existed. This kind of a *bio-inspired* approach is also typical in the field of artificial intelligence, robotics (e.g. cerebellar based control [44][45]) and so on. In some cases bio-inspired approach has lead to the creation of concrete methods/algorithms, such as the phase-based methods [27][28] for image correspondence (for depth and movement perception), while in other cases this has lead to a system-like thinking. In the early days a complete solution to the problem of image understanding was sought within one system. Nowadays tasks are divided into separate sub-tasks or sub-systems that, running together, perform more complex tasks. These so called phase-based methods are based on the fact that Gabor filters, named after Dennis Gabor, model response of simple cells in primate visual cortex [35]. System-like thinking, on the other hand, has lead to the separation of different visual tasks into so called low-, middle-, and high level tasks, which is in line with how the process of perception and image understanding is thought to function in the case of primates [38], for example.

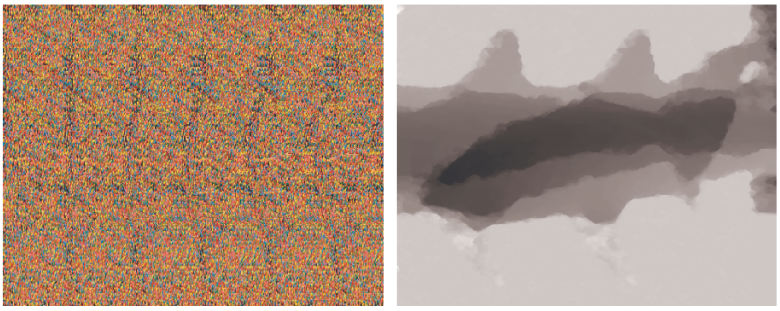


(a) (b)

Figure 1.4: Pot. (a) Random dot stereogram; (b) related disparity image.

As is indicated by Kandel in Principles of Neural Science [38], there is a consensus in the scientific community that visual perception, in the dorsal stream, is mediated by three different pathways in the visual cortex that process motion, depth and form, and colour. As it can be understood, these visual cues can be very useful when trying to come up with a system that would capable of deducing contextual information from any given image. Therefore, it is not a surprise that the kind of image cues that

we deal with in this thesis are related to motion, depth and segmentation (grouping of image pixels). Both motion and depth are considered to be low-level cues, while segmentation is thought to be a middle-level cue. Not only extracting these kind of cues is motivated by what we know of the primate vision system, but also makes sense in the DRIVSCO framework where action-perception cycles need to be extracted. Three-dimensional vision depends on both monocular depth cues and binocular disparity. Binocular disparity (stereopsis) is due to the fact that we have two eyes and thus we have two slightly different vantage points of the scene being observed. In the case of primates there is evidence that the same part of the visual system handles both motion and depth. Indeed, many methods used in the artificial vision for motion detection (optical-flow) can also be used for generating depth cues (stereo-disparity). Figures 1.4 and 1.5[1] display two different random-dot stereograms along with the related disparity maps calculated with the method described in Section 3.6. Random-dot stereograms 'hide' 3D information that can be perceived since both the eyes see the image from slightly different position. What is interesting here is that the same system that is used for obtaining depth cues from stereo-images also works with random-dot stereograms.



(a) (b)

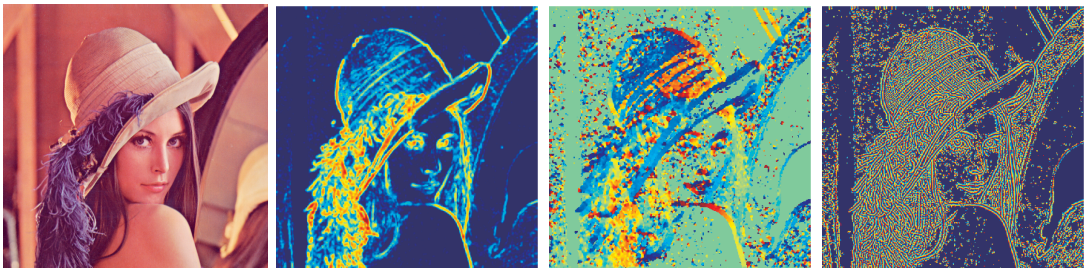
Figure 1.5: Shark (a) Random dot streogram; (b) related disparity image.

Figure 1.6 displays both the disparity map and the related segmentation for a robotic grasping scenario. Here the idea is to detect the position and type of the object(s) in the 3D-space. Based on this information the planner can decide on the best grasp type for manipulating the object of interest.

[1] `http://www.eyecanlearn.com/`

(a) (b) (c)

Figure 1.6: (a) left stereo-image; (b) related disparity map; (c) segmentation.



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

Figure 1.7: Local features. (a) Lena image; (b) energy; (c) orientation; (d) phase. In the case of orientation, color is used to codify the orientation of the edge in question.

Figure 1.7 shows an image of a Swedish model Lena Söderberg, that was published in the 1972 issue of Playboy magazine. The top part of the image, portraying the head along with the hat, has become probably the most used test image in the machine vision community. The sub-images display such local features, extracted with Gabor filters, as energy, orientation and phase. It is thought that in a primate vision the Parvocellular-interblob system (neurons in this system are sensitive, for example, to the orientation of edges) processes similar kind of cues [38].

1.2 Scientific Objectives

Scientific objectives that were put forward when starting this thesis were related to analysis of real image sequences coming from similar kind of camera assemblies (stereo-rig) as shown in Figure 1.1 in the framework of both vehicular technology and robotics.

To be more specific, the goals that were set are:

- Disambiguate low-level dense disparity approximations by using information/cues either deduced from the observed scene, for example by using a symbolic process [55], or by what is known of a possible solution beforehand.
- Study applicability of state-of-the-art algorithm(s) in the DRIVSCO and robotics frameworks. In the fall of 2006 and winter 2007 it was deduced that so called variational methods produced excellent results for both optical-flow and disparity calculation [63][13][3][5], and that also real-time implementations of these are possible running on a standard PC [18]. Therefore, variational methods were chosen for further study.
- Based on the experience related to other known correspondence forming methods, it was already known that image representation is crucial in order for any method to produce reliable cues. Therefore, it was also agreed that different representation spaces would be tested with the variational methods in order to see which ones would produce the most reliable and temporally coherent results.
- Since depth information is a very powerful cue for object detection, it was also agreed that disparity based segmentation would be researched. The main idea was to study if disparity can be used to differentiate between possible objects of interest and the background.

1.3 Project Framework

The work carried out in this thesis has been done in relation to three European research projects called DRIVSCO[1], GRASP[2], and TOMSY[3] and a national (Spanish) research project called DINAM-VISION[4]. The doctorate was initially directly funded by DRIVSCO, while participation in GRASP and TOMSY were through research exchange program: the doctorate spent 6 months at the University of Lappeenranta, Finland, in 2010 and 6 months at the Royal Institute of Technology (KTH), Sweden, in 2011. In the following there is a brief description of each of the projects.

[1] http://www.pspc.dibe.unige.it/~drivsco/
[2] http://www.csc.kth.se/grasp/
[3] http://www.cas.kth.se/tomsy/
[4] http://atc.ugr.es/dvision/index.php?option=com_content&view=frontpage

1.3.1 DRIVSCO

DRIVSCO, Learning to Emulate Perception-Action Cycles in a Driving School Scenario, EU project FP6-IST-FET, contract 016276-2.

Project description. *Most technical systems, for example cars, must work reliably at key-turn. Therefore, such systems almost always employ conventional control strategies. Biological systems, on the other hand, learn. In the beginning they are functional only at a very basic level from which they improve their skills. No-one would, however, want to use a learning car, which could in the beginning barely steer. Thus, learning techniques have not really entered turn-key applications so far.*

The goal of DRIVSCO is to devise, test and implement a strategy of how to combine adaptive learning mechanisms with conventional control, starting with a fully operational human-machine interfaced control system and arriving at a strongly improved, largely autonomous system after learning, that will act in a proactive way using different predictive mechanisms. DRIVSCO seeks to employ closed loop perception-action learning and control to cars and their drivers; combining for the first time advanced (largely hardware based) visual scene analysis techniques with supervised sequence learning mechanisms into a semi-autonomous and adaptive control system for cars and other vehicles. The central idea of this project is that the car should learn to drive autonomously from correlating scene information with the actions of the driver.

In the context of this project this system shall be tested and applied in night-vision scenarios with infra-red illumination, which is our main and commercially very relevant application domain. Here we envision a system that can learn to drive a car during daylight and apply the learned control strategies in an autonomous way to the system and augmented field of infrared night-vision.

1.3.2 GRASP

GRASP, Emergence of Cognitive Grasping Through Introspection, Emulation and Surprise, EU project IST-FP7-IP-215821.

Project description. *The aim of GRASP is the design of a cognitive system capable of performing grasping and manipulation tasks in open-ended environments, dealing with novelty, uncertainty and unforeseen situations. To meet the aim of the project, studying the problem of object manipulation and grasping will provide a theoretical and*

measurable basis for system design that is valid in both human and artificial systems. This is of utmost importance for the design of artificial cognitive systems that are to be deployed in real environments and interact with humans and other agents. Such systems need the ability to exploit the innate knowledge and self-understanding to gradually develop cognitive capabilities. To demonstrate the feasibility of our approach, we will instantiate, implement and evaluate our theories and hypotheses on robot systems with different embodiments and complexity.
GRASP goes beyond the classical perceive-act or act-perceive approach and implements a predict-act-perceive paradigm that originates from findings of human brain research and results of mental training in humans where the self-knowledge is retrieved through different emulation principles. The knowledge of grasping in humans can be used to provide the initial model of the grasping process that then has to be grounded through introspection to the specific embodiment. To achieve open-ended cognitive behaviour, we use surprise to steer the generation of grasping knowledge and modelling.

1.3.3 TOMSY

TOMSY, Topology Based Motion Synthesis for Dexterous Manipulation, EU project IST-FP7-Collaborative Project-270436.
Project description. *The aim of TOMSY is to enable a generational leap in the techniques and scalability of motion synthesis algorithms. We propose to do this by learning and exploiting appropriate topological representations and testing them on challenging domains of flexible, multi-object manipulation and close contact robot control and computer animation. Traditional motion planning algorithms have struggled to cope with both the dimensionality of the state and action space and generalisability of solutions in such domains. This proposal builds on existing geometric notions of topological metrics and uses data driven methods to discover multi-scale mappings that capture key invariances - blending between symbolic, discrete and continuous latent space representations. We will develop methods for sensing, planning and control using such representations. TOMSY, for the first time, aims to achieve this by realizing flexibility at all the three levels of sensing, representation and action generation by developing novel object-action representations for sensing based on manipulation manifolds and refining metamorphic manipulator design in a complete cycle. The methods and hardware developed will be tested on challenging real world robotic manipulation problems ranging from primarily*

'relational' block worlds, to articulated carton folding or origami and all the way to full body humanoid interactions with flexible objects.
The results of this project will go a long way towards providing some answers to the long standing question of the 'right' representation in a sensorimotor control and provide a basis for a future generation of robotic and computer vision systems capable of real-time synthesis of motion that result in fluent interaction with their environment.

1.3.4 DINAM-VISION

DINAM-VISION, Spanish national research project, DPI2007-61683.
Project description. *This project aims at developing a real-time vision system, capable of dynamically adapting the inherent characteristics (for example, the dynamic range of the spatio-temporal filters used in the low-level vision) of the used model(s) in order to improve information extraction. First stage of the system deals with low-level visual cues (e.g. local contrast changes and related magnitude, orientation and phase), while in the second stage these primitives are fused into multimodal disperse entities. The system has feed-back loops that allow feeding back information from later stages to the earlier stages, so that optimal functionality at each stage is achieved. Real-time processing is achieved by utilizing massively parallel platforms, such as FPGAs.*
The project will explore potential use of the system in different application areas, where the group has expertise, such as driver assistance systems. The system will be tested in both daylight and nigh-time scenarios, using cameras working in the visible light and infrared wavelengths. We will concentrate on IMOs (independently moving objects) and ego-motion.

1.4 Methods and Tools

The algorithms introduced in this thesis were implemented as a combination of Matlab/MEX (Mathlab EXecutable) code. The reason for choosing Matlab is that it allows for quick implementation and testing of algorithms, as well as visualisation of the results. On the other hand, Matlab is known to be slow. However, the MEX interface of Matlab allows to integrate compiled C code into Matlab code. In other words, functions programmed in C, using the MEX interface, can be called from Matlab code. In order to further improve the speed, some parts of the C code was programmed in assembler,

directly accessing FPU (Floating Point Unit) and SSE (Streaming SIMD extensions). The algorithms/methods introduced in this thesis are significantly less than half of all the different methods that were implemented during 2006-2011.

1.5 Organization of Chapters

Main part of this work is divided into three chapters as follows. In Chapter 3 we introduce the variational correspondence methods for disparity and optical-flow calculation. This chapter is divided into following sections. In sections 3.5 and 3.6 the basic correspondence models for optical-flow and stereo disparity are introduced, respectively. In Section 3.9 we study robustness of different image representations against illumination errors and image noise. Finally, in Section 3.10 the basic optical-flow and stereo disparity models are extended to include both temporal- and spatial constraints. In Chapter 4 we introduce the variational segmentation method based on level-set theorem and we will also explain how information obtained through segmentation can be used to disambiguate disparity estimations. In Chapter 5 we explain how the different models can be solved efficiently. Finally, in Chapter 6 scientific contributions of this work are discussed.

2

Introducción en Castellano

2.1 General

Génesis. Como se indica en el informe *Statistics of Road Traffic Accidents in Europe and North America*[1] de UNECE, durante la década de 1998-2008, se han muerto unos 150.000 y han resultado heridos unos 5.5 millones de personas, por AÑO, en mas de 3.8 millones de accidentes de tráfico en los países ECE (Europa y America del norte). Por lo tanto, no es una sorpresa que exista mucho interés en el desarrollo de sistemas pasivos y activos de vehículos para disminuir tanto la cantidad de accidentes como las consecuencias. Con el termino de *sistema pasivo* se refiere a componentes de los vehículos que protegen tanto al conductor como a los pasajeros, mientras que con el termino *sistema activo* se refiere a los sistemas que ayudan a evitar/prevenir los accidentes. Entre los sistemas pasivos se encuentran dispositivos tales como: cinturones de seguridad, parabrisas laminados. Por el otro lado, un subconjunto interesante de sistemas activos son los ADAS (ingl. Advanced Driver Assistance Systems). Dichos sistemas son, por ejemplo, sistemas para detección de salida de carril, control de tracción, visión nocturna (cámaras infrarrojas) etcétera.

Esta tesis se inició como parte de un proyecto europeo llamado DRIVSCO (ingl. Learning to Emulate Perception-Action Cycles in a Driving School Scenario[2], proyecto europeo, FP6) con la intención de crear un sistema ADAS capaz de adaptarse a la forma de conducción del conductor: *Mayoría de los sistemas técnicos, por ejemplo*

[1] `http://live.unece.org/trans/main/wp6/publications/stats_accidents2011.html`
[2] `http://cordis.europa.eu/fetch?CALLER=PROJ_ICT&ACTION=D&CAT=PROJ&RCN=80441`

los automóviles, deben funcionar perfectamente desde el primer uso. Por lo tanto, estos sistemas casi siempre usan las estrategias convencionales de control. Los sistemas biológicos, por otro lado, aprenden. Al principio funcionan de forma muy sub-óptima, pero el funcionamiento se va perfeccionando mientras aprenden. Nadie, sin embargo, desea emplear un vehículo 'de aprendizaje' que apenas se pueda conducir al ser usado la primera vez. Por lo tanto, las técnicas de aprendizaje no han entrado realmente en aplicaciones como esta, hasta ahora.
El objetivo de DRIVSCO es diseñar, probar e implementar una estrategia que permita combinar diferentes sistemas de aprendizaje adaptativo. La idea es partir de un sistema con una interfaz entre el usuario y la máquina y llegar a un sistema ampliamente autónomo, a través de aprendizaje, que acta de forma proactiva, utilizando diferentes mecanismos de predicción.

En otras palabras, el sistema tiene que ser capaz de analizar la situación y realizar predicciones. No es suficiente analizar solamente la información obtenida de la CPU del vehículo como, por ejemplo, la velocidad, la posición de la dirección, la posición de los pedales (acelerador, freno y/o embrague) o señal de GPS. Se puede argumentar que la percepción visual es el sentido más importante que usamos los seres humanos para conducir un vehículo. Esta suposición se ve respaldada por el hecho que varios simuladores de conducción usan principalmente información visual. Por lo tanto, la parte delantera del sistema (ingl. front-end), el cual se usa para percepción y extracción de eventos, está basado en el tratamiento de imágenes. Combinando estimulos visuales con el resto de la información disponible de la CPU del vehículo, se pueden extraer interpretaciones contextuales de la situación donde se encuentra el vehículo. El sistema visual del sistema está basado tanto en cámaras diurnas como en cámaras nocturnas (cámaras infrarrojas), el cual permite el funcionamiento del sistema incluso durante la noche.

Esta breve introducción nos lleva al tema principal de esta tesis, la cual está relacionada con *visión artificial* (también se conoce como visión maquina o visión por ordenador). Antes de seguir con el tema de visión artificial, queremos mencionar que existen (o han existido) varios proyectos similares a DRIVSCO. Aquí mencionamos algunos: DIPLECS (ingl. Dynamic Interactive Perception-Action Learning in Cognitive Systems[1], Comision Europeo, FP7), COSPAL (ingl. Cognitive Systems using

[1] `http://www.diplecs.eu/`

Perception-Action System[1], Comision Europeo, FP6) y IVSS (Intersection accidents: Analysis and Prevention [2], Suecia).

En Figura 2.1 se demuestra tanto el sistema óptico como el monitor con la interfaz del sistema, que se han utilizado en el proyecto DRIVSCO. Las figuras 2.2 y 2.3 demuestran tanto el GUI (del ingl. Graphical User Interface) como el sistema que detecta el comportamiento inconsistente del conductor tanto como los IMOs (del ingl. Independently Moving Objects) que están en rumbo de colisión con el vehículo [46].

(a) (b)

Figura 2.1: DRIVSCO coche. (a) sistema óptico de DRIVSCO; (b) la GUI (ingl. Graphical User Interface) correspondiente.

Mientras se estaba escribiendo esta tesis, el fabricante de coches Volvo dio a conocer un sistema para detectar posibles colisiones, con frenos completamente automáticos. Por otro lado, las mismas técnicas que se investigaron inicialmente en el marco del proyecto DRIVSCO, son suficientemente genéricos para ser utilizados en otros campos científicos, donde se emplea visión artificial como, por ejemplo, la robótica.

En relación a DRIVSCO se han hecho varias tesis. Adjunto hemos puesto una lista para mencionar algunas de ellas: Karl Pauwels (Katholieke Universiteit Leuven, Leuven, Belgium), Anders Kær-Nielsen[3] and Lars Baunegaard With Jensen[4] (University of Southern Denmark, Odense, Denmark), Irene Markelić[5] (Georg-August-University

[1] http://www.cospal.org/
[2] http://www.ivss.se/
[3] http://www.mip.sdu.dk/people/PhD_students/akn.html
[4] http://www.mip.sdu.dk/people/PhD_students/lbwj.html
[5] http://www.markelic.de/

Figura 2.2: La interfaz que en este caso demuestra los carriles junto con la historia de predicción de ángulos de la dirección (volante).

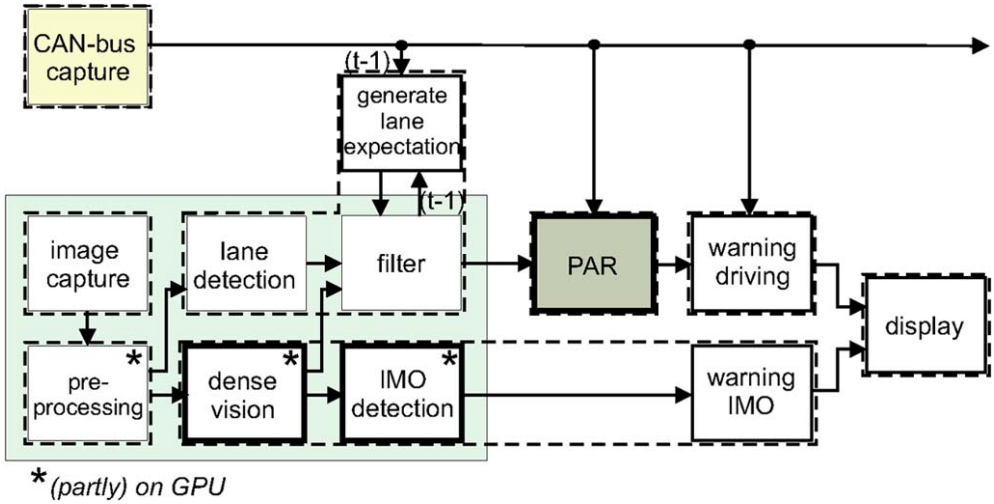


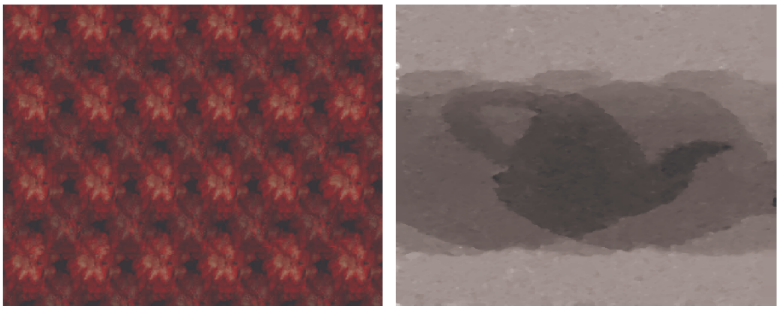
Figura 2.3: Arquitectura del sistema. IMO, PAR y GPU significan significan, en inglés, Independently Moving Objects, Perception-Action Repository y Graphical Processing Unit.

Göttingen, Göttingen, Germany) and Matteo Tomasi[1] (University of Granada, Granada, Spain).

Visión Artificial. El campo de visión artificial se asocia típicamente con el estudio

[1] `http://atc.ugr.es/~mtomasi/`

de *la inteligencia artificial* y, de hecho, estos campos emplean muchas técnicas similares como, el reconocimiento de patrones y aprendizaje automático. También, varios de los objetivos son similares. Crear una máquina que 'entiende' la escena que se percibe a través de las cámaras, ha demostrado ser un problema mucho más difícil de lo que se pensaba inicialmente. El enfoque moderno de la visión artificial adopta técnicas/modelos de la biología: la evolución ha tenido mucho más tiempo para llegar a soluciones viables antes que los seres humanos (i.e. homo sapiens) existieran. Este tipo de enfoque *bio-inspirado* también es usual en el campo de la inteligencia artificial o robótica, por ejemplo, el control basado en cerebelo [44][45]. En algunos casos el enfoque bio-inspirado ha dado lugar a la creación de métodos/algoritmos concretos, como es el caso con los métodos de correspondencia (disparidad estéreo y la percepción del movimiento) basados en la fase [27][28], mientras que, en otros casos, esto ha dado lugar a un enfoque de procesamiento dividido en sub-tareas. Los métodos basados de la fase se basan en el hecho de que los filtros de Gabor, nombrado en honor a Dennis Gabor, modelan el funcionamiento de las células simples en la corteza visual de primates [35]. Al principio se trató de resolver el problema de percepción visual utilizando un solo sistema/programa que fuera capaz de hacerlo todo. Hoy en día, el enfoque es dividir el problema en distintas sub-tareas o subsistemas que juntos realicen tareas más complejas. Este enfoque ha dado lugar a la separación de las diferentes tareas visuales en tareas de bajo, medio y de alto nivel. Existe evidencia que la percepción visual funciona de esta manera en el caso de los primates [38], por ejemplo.



(a) (b)

Figura 2.4: Tetera. (a) Estereograma de puntos aleatorios; (b) la disparidad asociada.

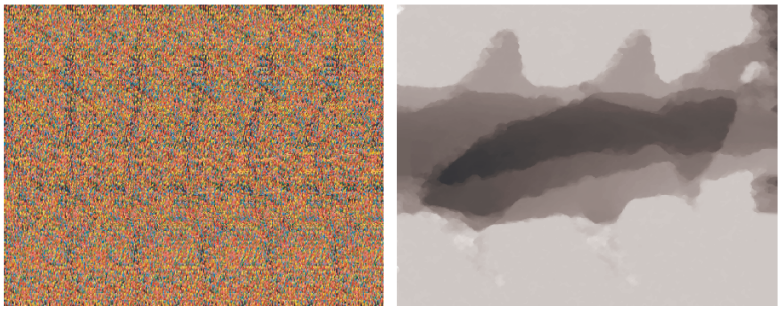
Como indica Kandel en Principios de la Neurociencia [38], existe un consenso en la

comunidad científica que la percepción visual, en la franja dorsal , está mediada por tres vías diferentes de la corteza visual que procesan el movimiento, la profundidad y la forma y el color. Como se puede entender, estas señales visuales pueden ser muy útiles cuando se trata de crear un sistema que sea capaz de deducir información contextual de cualquier imagen procesada. Por lo tanto, no es una sorpresa que el tipo de señales/rasgos que utilizamos en esta tesis estén relacionados con la detección del movimiento (flujo-óptico), la profundidad y la segmentación (agrupación de los píxeles de la imagen). Tanto el movimiento como la profundidad, se consideran señales de bajo nivel, mientras que la segmentación es una señal de nivel medio. La extracción de este tipo de señales no sólo está motivada por lo que sabemos del sistema de visión de los primates, sino que también tiene sentido en el marco de DRIVSCO donde se necesitan extraer ciclos de percepción-acción de las imágenes procesadas. La visión en tres dimensiones depende de indicios de profundidad monoculares como binoculares. La Disparidad binocular (estereoscópica) se debe al hecho de que tenemos dos ojos y por lo tanto tenemos dos puntos de vista ligeramente diferentes de la escena que observamos. En el caso de los primates, hay evidencia de que la misma parte del sistema visual se encarga tanto del procesamiento del movimiento como de la profundidad. De hecho, varios métodos utilizados en la visión artificial para detección de movimiento (flujo-óptico) también pueden ser utilizados para la generación de indicios relacionados con la profundidad (disparidad estéreo). Las figuras 2.4 y 2.5[1] muestran dos diferentes estereogramas de puntos aleatorios junto con los mapas de disparidad correspondientes, calculados con el modelo que se describe en la sección 3.6. Los estereogramas de puntos aleatorios 'ocultan' información 3D que podemos percibir ya que los ojos ven la imagen de dos posiciones ligeramente diferentes. Lo que es interesante aquí,es que el mismo sistema que se utiliza para la obtención de señales de profundidad en imágenes estéreo también funciona con estereogramas de puntos aleatorios

Figura 2.6 muestra tanto el mapa de disparidad como la segmentación, relacionados con robótica. Aquí la idea es detectar la posición y el tipo de objeto de que se trata, en el espacio 3D. Basado en esta información, el planificador puede decidir sobre el mejor tipo de agarre para manipular el objeto de interés.

Figura 2.7 muestra una imagen de una modelo sueca, Lena Söderberg, que fue publicado en la edición de 1972 de la revista Playboy. La parte superior de la ima-

[1] http://www.eyecanlearn.com/



(a) (b)

Figura 2.5: Tiburon. (a) Estereograma de puntos aleatorios; (b) la disparidad asociada.

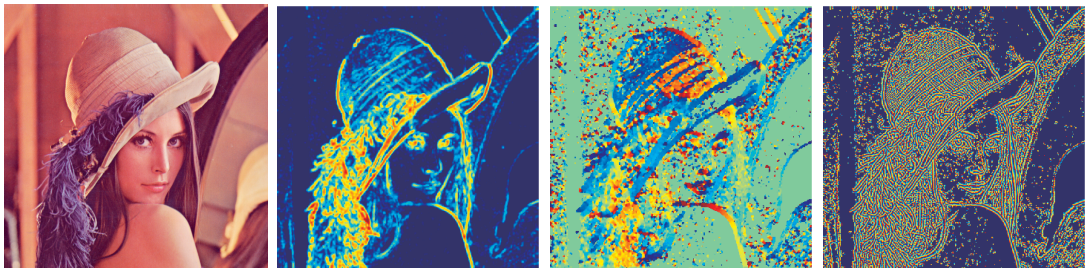
(a) (b) (c)

Figura 2.6: (a) Imagen izquierda; (b) el mapa de disparidad asociado; (c) resultados de segmentación.

gen, retratando la cabeza junto con el sombrero, se ha convertido probablemente en la imagen de prueba más utilizada en la comunidad de visión artificial. Las subfiguras muestran características tales como la energía, la orientación y la fase, obtenidas a través de filtros de Gabor. Se piensa que en el caso de los primates, el sistema parvocecular (las neuronas de este sistema son sensibles, por ejemplo, a la orientación de los bordes) procesan este tipo de señales [38]..

2.2 Objetivos Cientificos

Los objetivos científicos que se establecieron al iniciar esta tesis están relacionados con el análisis de secuencias de imágenes reales procedentes de sistemas parecidos al que sale en la figura 2.1, relacionados con tecnología vehicular y la robótica. Para ser más específicos, las metas que se establecieron son las siguientes:



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

Figura 2.7: Características locales. (a) Imagen de Lena; (b) la energía; (c) la orientación; (d) la fase. En el caso de la orientación, el color codifica la orientación del borde en cuestión.

- Desambiguar aproximaciones de disparidad mediante el uso de la información (1) deducido directamente de la escena observada (por ejemplo, utilizando un proceso simbólico [55] o (2) por lo que se sabe de antemano de la posible solución.

- Estudio de la aplicabilidad del algoritmo de estado de la técnica (s) en los marcos DRIVSCO y la robótica. En el otoño de 2006 y el invierno de 2007 se dedujo que los llamados métodos variacionales han dado excelentes resultados, tanto para el flujo óptico y el cálculo de la disparidad [63] [13], y que también, en tiempo real de las implementaciones de estos, es posible ejecutar en un PC estándar [18]. Por lo tanto, los métodos variacionales fueron elegidos para el desarrollo de nuestro sistema.

- Investigar la aplicabilidad de algoritmos de estado-del-arte tanto en el marco de DRIVSCO como en el de la robótica. Durante el otoño de 2006 y el invierno de 2007 se dedujo que los métodos variacionales daban excelentes resultados, tanto para el flujo óptico como para la disparidad [63][13][3][5], con la posibilidad de implementar esto metodos en tiempo real en un PC estandar [18]. Por lo tanto, los métodos variacionales fueron elegidos.

- Debido a la experiencia con otros métodos usados para generar correspondencias entre imágenes, ya se sabia que la representación de la imagen juega un papel crucial para que cualquier método fuese capaz de crear dichas correspondencias de alta confianza. Por lo tanto, se decidió investigar varias representaciones diferentes con el fin de ver cuales son capaces de generar resultados más fiables y

coherentes temporalmente.

- Dado que la información de profundidad es una señal muy 'determinante' para la detección de objetos, también se acordó que la segmentación basada en la disparidad sería investigado. La idea principal fue estudiar si el mapa de disparidad se puede utilizar para diferenciar entre los objetos de interés y el fondo.

2.3 Marco de Proyectos

El trabajo realizado en esta tesis se ha hecho en relación con tres proyectos europeos de investigación, denominados DRIVSCO[1], GRASP[2] , y TOMSY[3] y un proyecto nacional de España denominado DINAM-VISION[4]. El doctorando fue inicialmente financiado directamente por DRIVSCO, mientras que la participación en GRASP y TOMSY fué a través de programa de intercambio: el doctorando pasó 6 meses en la Universidad de Lappeenranta, Finlandia, en 2010 y 6 meses en el Instituto Real de Tecnología (KTH), Suecia, en 2011. A continuación, hay una breve descripción de cada uno de los proyectos.

2.3.1 DRIVSCO

DRIVSCO (ingl. Learning to Emulate Perception-Action Cycles in a Driving School Scenario), proyecto europeo FP6-IST-FET, contrato 016276-2.

Descripción del proyecto. *Mayoría de los sistemas técnicos, por ejemplo los automóviles, deben funcionar perfectamente desde el primer uso. Por lo tanto, estos sistemas casi siempre usan las estrategias convencionales de control. Los sistemas biológicos, por otro lado, aprenden. Al principio funcionan de forma muy sub-óptima, pero el funcionamiento se va perfeccionando mientras aprenden. Nadie, sin embargo, desea emplear un vehículo'de aprendizaje' que apenas se pueda conducir al ser usado la primera vez. Por lo tanto, las técnicas de aprendizaje no han entrado realmente en aplicaciones como esta, hasta ahora.*

El objetivo de DRIVSCO es diseñar, probar e implementar una estrategia que permita

[1] http://www.pspc.dibe.unige.it/~drivsco/
[2] http://www.csc.kth.se/grasp/
[3] http://www.cas.kth.se/tomsy/
[4] http://atc.ugr.es/dvision/index.php?option=com_content&view=frontpage

combinar diferentes sistemas de aprendizaje adaptativo. La idea es partir de un sistema con una interfaz entre el usuario y la máquina y llegar a un sistema ampliamente autónomo, a través de aprendizaje, que acte de forma proactiva, utilizando diferentes mecanismos de predicción.
En el contexto de este proyecto, este sistema será probado y aplicado en los escenarios de visión nocturna, que es nuestro dominio de la aplicación principal y comercialmente muy relevante. Aquí imaginamos un sistema que puede aprender a conducir un coche durante el día y aplicar las estrategias de control, aprendidos de manera autónoma, en los escenarios de conducción nocturna.

2.3.2 GRASP

GRASP (ingl. Emergence of Cognitive Grasping Through Introspection, Emulation and Surprise), projecto europeo IST-FP7-IP-215821.

Descripción del proyecto. *El objetivo de GRASP es el diseño de un sistema cognitivo capaz de realizar tareas de agarre y manipulación en entornos 'abiertos', capaz de enfrentar la incertidumbre en situaciones novedosas y no previstas. Para cumplir con el objetivo de este proyecto, el estudio del problema de la manipulación de objetos proporcionará una base teórica y 'medible' para el diseño del sistema que es válido tanto en los sistemas biológicos como artificiales. Esto es de suma importancia para el diseño de los sistemas cognitivos artificiales que van a ser usados en entornos reales y que interactúan con los seres humanos y otros agentes. Estos sistemas necesitan la habilidad de explotar el conocimiento innato y la comprensión de uno mismo para desarrollar gradualmente la capacidad cognitiva. Para demostrar la viabilidad de nuestro enfoque, vamos a crear una instancia, ejecutar y evaluar las teorías y las hipótesis sobre los sistemas de robot con diferentes formas de realización y la complejidad.*
GRASP va más allá de la clásico enfoque de percepción-actuación o actuación-percepción e implementa un paradigma de predicción-actuación-percepción, el cual está basado en los últimos hallazgos de la investigación del cerebro humano. El conocimiento de cómo agarramos objetos los seres humanos puede ser utilizado para proporcionar el modelo inicial del proceso de agarre.

2.3.3 TOMSY

TOMSY (ingl. Topology Based Motion Synthesis for Dexterous Manipulation), projecto europeo IST-FP7-Collaborative Project-270436.

Descripción del proyecto. *El objetivo de TOMSY es permitir un salto generacional en las técnicas y la escalabilidad de los algoritmos de síntesis de movimiento. Nos proponemos hacer esto por el aprendizaje y la explotación de la representación topológica adecuada y ponerlos a prueba en los dominios de manipulación de varios objetos, control de robots y la animación por ordenador. Los algoritmos tradicionales de planificación de movimiento han tenido dificultades para enfrentar la dimensionalidad del espacio de estados y de la acción y la generalización de las soluciones en tales dominios. Esta propuesta se basa en las nociones geométricas de métricos topológicos y en métodos basados en datos para descubrir mapeos (asignaciones) multi-escala que capturan invarianzas más relevantes - una mezcla entre representaciones de espacios simbólicos, discretos y continuos. Por primera vez, TOMSY aspira a lograr esto al darse cuenta de la flexibilidad en los tres niveles de percepción, representación y la generación de acciones mediante el desarrollo de nuevas representaciones de objeto-acción para percepción basado en 'manipulación variedad' (ingl. manipulation manifold) y el diseño de manipuladores metamórficos. Los métodos y hardware desarrollados se pondrán a prueba en retos del mundo real de manipulación que van desde mundos de bloque hasta doblar cartón u origami e interacciones de cuerpos humanoides con objetos flexibles.*
Los resultados de este proyecto nos proporcionarán algunas respuestas a la vieja pregunta de cual es la 'correcta' representación en un control sensorio-motora y también proporcionará una base para una futura generación de sistemas de visión robótica capaces de síntesis de movimiento en tiempo real que resultara en interactuación fluida con su entorno.

2.3.4 DINAM-VISION

DINAM-VISION, proyecto nacional de España, DPI2007-61683.

Descripción del proyecto. *El proyecto aborda el desarrollo de un sistema de visión en tiempo real con capacidad de adaptación dinámica de sus características inherentes (por ejemplo el rango dinámico de los filtros espacio-temporales en los que se basan*

los modelos de visión de bajo nivel) para mejorar la capacidad de extracción de información. El sistema de visión está formado por una primera etapa de visión de bajo nivel (en el que se extraen modalidades visuales como movimiento, profundidad a partir de dos cámaras, características de cambios locales de contraste como magnitud, orientación y fase local, etc.) y una segunda etapa de fusión de estas primitivas en entidades multi-modales dispersas. El sistema incluye lazos de realimentación (flujos de información/modelos proyectados hacia atrás) con los que el sistema puede modificar el procesamiento de imágenes en las distintas etapas que realizan la estructuración de información de acuerdo con primitivas detectadas con mayor confianza en las distintas etapas. La implementación en tiempo real de toda la plataforma utilizando procesamiento altamente paralelo en dispositivos de tipo FPGA permite su evaluación en el marco de aplicaciones reales.
El proyecto explora el potencial del sistema de visión en varios campos de aplicación en los que el grupo tiene experiencia. Nos hemos centrado en sistemas avanzados de visión para ayuda a la conducción. Evaluamos el potencial del sistema de visión de bajo y medio nivel en tiempo real con secuencias de conducción diurnas y nocturnas (tomadas con sensores específicos). Nos centramos en la detección de eventos relevantes (como objetos con movimiento independiente OMIs- y estimación de dirección de ego-movimiento heading-).

2.4 Métodos y Herramientas

Los algoritmos introducidos en esta tesis se implementaron como una combinación de código Matlab/MEX (ingl. Matlab EXecutable). La razón para elegir Matlab es que permite una rápida implementación de algoritmos, así como la visualización de los resultados. Por otro lado, Matlab es conocido por ser lento. Sin embargo, la interfaz MEX de Matlab permite integrar código C (compilado) en el código Matlab. En otras palabras, las funciones programadas en C, utilizando la interfaz de MEX, se pueden llamar desde el código de Matlab. Con el fin de mejorar aún más la velocidad, algunas partes del código de C se han programado en ensamblador, accediendo directamente a la FPU (ingl. Floating Point Unit) y SSE (Streaming SIMD Extensions). Los algoritmos/métodos introducidos en esta tesis son menos de la mitad de todos los diferentes métodos que se implementaron durante 2006-2011.

2.5 Organización de los Capítulos

Parte principal de este trabajo se divide en tres capítulos de la siguiente manera. En Capítulo 3 se introducen los métodos variacionales de correspondencia para el cálculo de disparidad y flujo óptico. Este capítulo está dividido en las siguientes secciones. En las secciones 3.5 y 3.6 se introducen los modelos básicos de flujo óptico y estéreo, respectivamente. En Sección 3.9 se estudia la robustez de diferentes representaciones de imágenes respecto a los errores en la iluminación y el ruido. Por último, en Sección 3.10 se amplían los modelos básicos de flujo óptico y disparidad para incluir las restricciones temporales y espaciales. En Capítulo 4 se introduce el método de segmentación basado en el teorema de conjunto de nivel (ingl. level sets) y también explicamos cómo la información obtenida a través de la segmentación se puede utilizar para eliminar la ambigüedad en las estimaciones de disparidad. En Capítulo 5 se explica cómo los diferentes modelos se pueden resolver de manera eficiente. Por último, en Capítulo 6 se discuten los resultados obtenidos y las contribuciones científicas de este trabajo.

3

Variational Correspondence Methods

3.1 Introduction

In this chapter we introduce the variational framework for generating pixel-wise correspondences between images and explain how these correspondences are related to the motion (optical-flow) and the depth (disparity) cues mentioned in the Section 1.1. We start by briefly explaining the concept of *calculus of variations* and its relation with the *Gauss-Seidel* equations. After this we introduce the basic models, for both optical-flow and stereo disparity, and talk about their properties. Then we study influence of the image representation upon generating correct and coherent correspondences, in disparity calculation, under illumination errors and image noise. Up to a limit, these results are valid for optical-flow as well. Finally we extend the basic models to include both spatial- and temporal-constraints. The main 'target' of our research has been improving robustness of the models, so that they could be used with real world problems. Therefore, our main contributions are (1) finding a robust image representation that allows generation of correct image correspondences under realistic illumination conditions, and (2) introducing a way of using 'a priori' information in the models.

Another term used for generating correspondences is image registration, which is a process of overlaying two or more images either taken at different times or at slightly different positions/orientations. In our case we are not so much interested in the actual overlaid images, but in the transformation needed to transform the images, so that

the given features (e.g. object borders) become super-positioned. As it was already mentioned, apparent movement of objects between two images can be due to the fact that the images were taken at different times or from different vantage points. Therefore, we identify two different cases. In the *optical-flow* case we have two (or several) images taken with the same camera at different instances of time, while in the *stereo-disparity/binocular-disparity* (or shortly disparity) case we have two (or more) cameras next to each other that take images at the same time. As it can be understood, in the first case movement (if any) is either due to the movement of the camera with relation to the scene or due to movement of the actual objects between the images taken at t and $t + 1$. In the second case parallax is due to the different vantage points between the two cameras and is related to the distance between the cameras and the objects seen in the images. *'Parallax is an apparent displacement or difference in the apparent position of an object viewed along two different lines of sight, and is measured by the angle or semi-angle of inclination between those two lines'*[1]. Figure 3.1[2] depicts binocular disparity.

Since both optical-flow and disparity problems are well known in the machine vision community, we do not spend great deal of time justifying why these are interesting problems to solve or what the related physical phenomena is. An interested reader is pointed to [31] for more information on the subject. Some examples where both optical-flow and disparity can be used, are given in Table 3.1.

Table 3.1: Possible applications of optical-flow and disparity.

-Detection of moving objects	-3D-reconstruction of the scene
-Video-surveillance	-Video-surveillance (3D)
-Tracking of objects	-Augmented reality
-Pedestrian detection vehicular technology	-Robotics (e.g. grasping)
-Between frames generation (cinema effects)	-Autonomous navigation
-Video compression	
-Image registration (e.g. medical imaging)	

Figures 3.2 and 3.3 display examples of both disparity (and possible uses) and optical-flow. In the case of the optical-flow, colour codifies the direction while intensity

[1] `http://en.wikipedia.org/wiki/Motion_parallax`
[2] `http://en.wikipedia.org/wiki/Binocular_disparity/`

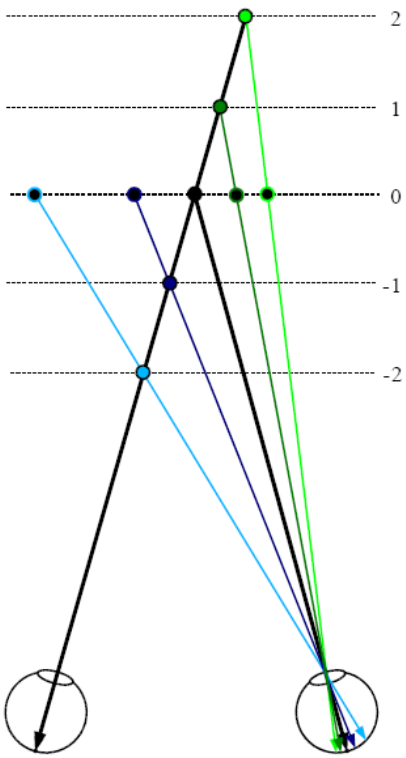
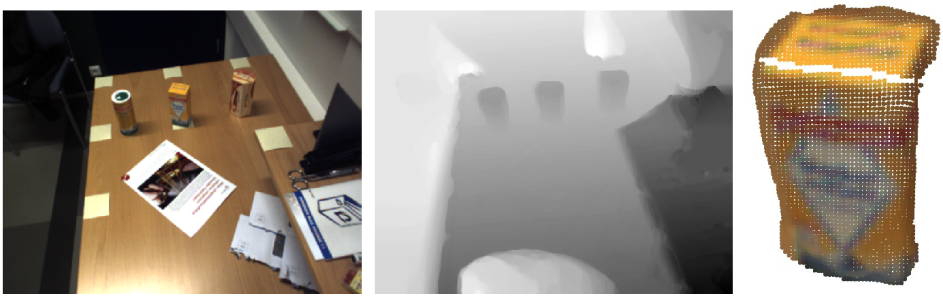


Figure 3.1: Binocular disparity.

codifies the velocity.



(a) (b) (c)

Figure 3.2: A robotics related disparity example. (a) Left stereo-image; (b) corresponding disparity map; (c) 3D-reconstruction of the object of interest. In the case of the disparity map, gray-level codifies the disparity: objects with dark tones are closer to the cameras, while objects with light tones are further away from the cameras.

(a) (b)

Figure 3.3: Optical flow. (a) image at t=0; (b) corresponding optical-flow. Colour codifies direction and intensity velocity.

3.2 Organisation of Sections

Since is the longest chapter in the thesis, we introduce the order of the sections here, hopefully making it easier for the reader to follow the rest of the chapter. In Section 3.3 we briefly cover what is calculus of variations and its relation with the Gauss-Seidel equations. In Section 3.4 we give some motivation for the use of the variational methods. In sections 3.5 and 3.6 we introduce the basic optical-flow and disparity models, in respective order. In Section 3.9 we analyse different image representations from both accuracy and robustness points of view, while in the Section 3.10 we introduce both spatial- and temporal constraints in the models.

3.3 What is Meant by Calculus of Variations?

Before proceeding any further, we will very briefly explain what is meant with *calculus of variations* and how it is related to the so called *Euler-Lagrange equations*.

Calculus of variations. *'Calculus of variations is a field of mathematics that deals with extremising functionals, as opposed to ordinary calculus which deals with functions. A functional is usually a mapping from a set of functions to the real numbers. Functionals are often formed as definite integrals involving unknown functions and their derivatives. The interest is in extremal functions that make the functional attain a*

maximum or minimum value or stationary functions those where the rate of change of the functional is precisely zero.'[1].

Now that we understand what is actually meant by calculus of variations, we will see what is its relation with the Euler-Lagrange equation, as follows.

Euler-Lagrange equation. *'In calculus of variations, the Euler-Lagrange equation, is a differential equation whose solutions are the functions for which a given functional is stationary. Because a differentiable functional is stationary at its local maxima and minima, the Euler-Lagrange equation is useful for solving optimisation problems in which, given some functional, one seeks the function minimising (or maximising) it. This is analogous to Fermat's theorem in calculus, stating that where a differentiable function attains its local extrema, its derivative is zero.'*[2] In the following we given an example of a functional and the related Euler-Lagrange equation.

$$S(f) = \int_a^b L(x, f(x), f'(x)) \tag{3.1}$$

where $f(x)$ is a function of a real variable x and L is the functional. The Euler-Lagrange equation is given by:

$$\frac{\partial}{\partial f} L - \frac{d}{dx} \frac{\partial}{\partial f'} L = 0 \tag{3.2}$$

In the optical-flow case the functional (3.3) is defined with respect to functions u and v (that are to be found), of real arguments x and y. In other words, the functional is a mapping from the functions $u(x, y)$ and $v(x, y)$ into a real number that we call energy. In the disparity case the functional (3.11) is defined with respect to a function d (that is to be found), of real arguments x and y. In this case the functional is a mapping from the function $d(x, y)$ into a real number.

3.4 Motivation for Variational Correspondence

The reason for having chosen variational methods over other methods is motivated by several issues. First of all, mathematical modelling of the problem is straight-forward and, therefore, extending the model is easy. Secondly, the same model (model with late

[1] source:`http://en.wikipedia.org/wiki/Calculus_of_variations`
[2] source:`http://en.wikipedia.org/wiki/Euler-Lagrange`

linearization), with only minor changes, can be used for both optical-flow and disparity calculation. Thirdly, the governing mathematics are well known and efficient solvers for the resulting PDEs (partial differential equations) are available, allowing even real-time implementations [19][17][18]. Fourthly, with appropriate data and smoothness terms, the results generated by the model are both accurate (for most practical applications) and robust with respect to illumination changes and image noise [59][48].

In practice almost all correspondence methods can be described in terms of minimizing (or maximizing) an energy- or a cost function. Due to a built-in smoothness term, that propagates the solution spatially (or spatio-temporally), variational methods search for a minimum of the energy function in a global fashion, and therefore are referred as *global-methods*. On the other hand, methods that do not have this kind of a spatial- or spatio-temporal smoothness term are called *local-methods*. Naturally, ad-hoc methods for increasing spatial support in the local-methods do exist. Due to the global nature of the variational methods they overcome the aperture problem. This comes at a cost of increased computational effort. Therefore, it can be said that local methods are more suitable for real-time implementations, especially in low-performance platforms.

Table 3.2: Pros and cons of variational correspondence methods.

Pros	Cons
-Easily understandable modeling	-Computational complexity
-Extendibility	-Density (100% dense fields)
-Same model for both optical-flow and disparity	
-Well known mathematics	
-Efficient solvers exist	
-Sub-pixel accuracy	
-Density (100% dense fields)	

In Table 3.2 we have listed some of pros and cons of variational correspondence methods. Without entering philosophical discussions, density of these methods can be considered a double edged sword. These methods generate smooth, 100% dense displacement fields which, of course, can be considered a positive feature. At the same time, in the case of insufficient spatial features, these methods might propagate incorrect values. However, in general, optical-flow calculation methods based on calculus of

variations are amongst the most accurate ones[1] and also give reasonable results for disparity.

3.5 Optical-flow

In this section models for calculating the optical-flow, denoted by (u, v), are introduced. First a 'simple' model with early linearisation is covered, after which a slightly more complex model with warping and late linearisation is discussed. One of the main benefits of the late linearisation model with warping is that it is suitable for estimating large displacements. As it was already mentioned, optical-flow means apparent movement of objects in the camera plane which can be either due to ego-motion or due to the movement of the perceived 3D objects. The model with late linearisation is of special interest, since it can be used for both optical-flow and disparity calculation with a very small change. Before going any further we introduce the generalised energy functional describing the model. The optical-flow field that we are searching for is the one that minimises the 'generic' energy functional in (3.3).

$$E(u, v) = \int_{\Omega} \Big(\underbrace{D(I_0, I_1)}_{\text{data}} + \alpha \underbrace{S(\nabla I_1, \nabla u, \nabla v)}_{\text{smoothness}} \Big) \mathbf{dx} \tag{3.3}$$

where $D(I_0, I_1)$ is the data term, while $S(\nabla I_1, \nabla u, \nabla v)$ is the regularisation term, $I_{\{0,1\}}$ denotes images taken at time $t = 0$ and $t = 1$, α is the 'weight' of the smoothness term (i.e. how smooth the solution should be), $\nabla = [\frac{\partial}{\partial x}, \frac{\partial}{\partial y}]$ is the spatial gradient operator and Ω is the image domain. As it was already mentioned, this is the 'generic' energy functional. Depending on how we define the data- and the smoothness-terms, we obtain different instances of the functional and, therefore, different models. In the following sections both the data- and smoothness terms will be introduced and discussed in more details.

3.5.1 Early Linearisation

In order to calculate the optical-flow we need a constancy assumption that allows us to formulate the problem mathematically. We assume that intensity level of an observed image point does not change in time or, in other words, as the pixel 'moves' from one

[1] `http://vision.middlebury.edu/flow/`

position to another it maintains its intensity level. Therefore, any observed change in intensity level is due to movement (e.g. shift in position) $(x-u, y-v)$ in the image plane as time changes from t to $t+1$, and thus we obtain the following constraint:

$$I(x,y,t) = I(x-u, y-v, t+1) \tag{3.4}$$

As it can be observed from Equation (3.4), this is a nonlinear equation in u and v. In the case of early linearisation, we directly approximate the equation using first order Taylor series expansion, and thus obtain:

$$\frac{\partial I}{\partial t} - \frac{\partial I}{\partial x}u - \frac{\partial I}{\partial y}v = 0 \tag{3.5}$$

This linearised version of the (3.4) is so called *Optical Flow Constraint* (OFC)[33]. So far we have formulated the constancy assumption only for intensity levels, but the same holds for any vector valued, e.g. RGB image:

$$\begin{aligned} R(x,y,t) &= R(x-u, y-v, t+1) \\ G(x,y,t) &= G(x-u, y-v, t+1) \\ B(x,y,t) &= B(x-u, y-v, t+1) \end{aligned} \tag{3.6}$$

The linearisation is done exactly in the same way as in the intensity image case. By examining the OFC (3.5) more closely, we notice two very important things: (1) we have only one equation but two unknowns, u and v; and (2) where the spatial derivatives are zero (e.g. in smooth zones), the OFC fails to give information of the optical-flow field. Therefore, the problem is said to be ill-posed. As proposed by Horn and Schunck [33], these problems are overcome by introducing an additional constraint called the *smoothness term*. The displacement field (u, v) that is sought, is the one that minimises the energy functional given in (3.7). This is the Horn&Schunck model that is based on early linearisation.

$$E(u,v) = \int_\Omega \sum_{k=1}^{K} \left(\underbrace{\left(\frac{\partial I_k}{\partial t} - \frac{\partial I_k}{\partial x}u - \frac{\partial I_k}{\partial y}v\right)^2}_{\text{data}} + \alpha \underbrace{\left(|\nabla u|^2 + |\nabla v|^2\right)}_{\text{smoothness}} \right) \mathbf{dx} \tag{3.7}$$

where the sub-index k refers to the channels (e.g. R, G or B) of a vector valued image I_k, and α is the weight of the smoothness term and the spatial gradient operator is given by $\nabla := \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}\right)^T$. As can be observed from (3.7), the model has two terms: (1) a data term, here OFC; and (2) a smoothness term which in this case is a Tikhonov regulariser [73]. Because of the smoothness term, the expected solution is 'smooth', since changes in the displacement field are penalised. The smoothness term also has a fill-in effect: where the OFC does not provide information, solution is propagated from the neighbourhood. A necessary condition for the minimum of the energy functional (3.7) is that its corresponding Euler-Lagrange equations, given in (3.8), are zero.

$$\begin{aligned} \sum_{k=1}^{K} \left(\frac{\partial I_k}{\partial t} - \frac{\partial I_k}{\partial x} u - \frac{\partial I_k}{\partial y} v \right) \frac{\partial I_k}{\partial x} + K\alpha DIV(\nabla u) =& 0 \\ \sum_{k=1}^{K} \left(\frac{\partial I_k}{\partial t} - \frac{\partial I_k}{\partial x} u - \frac{\partial I_k}{\partial y} v \right) \frac{\partial I_k}{\partial y} + K\alpha DIV(\nabla v) =& 0 \end{aligned} \tag{3.8}$$

with reflecting boundary conditions $\partial_n u = 0$ and $\partial_n v = 0$, where n denotes the normal to the image boundary $\partial\Omega_h$. We need boundary conditions, since the images captured by the cameras have finite domain. We can see from the Equation (3.8) that the smoothness term has a *divergence* operator that is defined as follows: for a differentiable vector function $F = Ui + Vj$ the divergence is defined as $DIV\Big(F\Big) = \frac{\partial U}{\partial x} + \frac{\partial V}{\partial y}$. Here, physical interpretation of the divergence is, in a sense, that of *diffusion* [53]. Interestingly, we can see that the DIV operator emerges, not only in the optical-flow and disparity case, but also in the segmentation model based on level-set:s. This basic model has several weaknesses that are discussed next. First, the Tikhonov regulariser [73] smoothens across object borders. What this means is, that different objects with different optical-flow fields will propagate information from one displacement field to other. Another way of saying this is, that the model does not admit borders. Second, the model does not take into account outliers in the data. These outliers are due to, for example, changes/errors in the illumination levels or noise present in the images. Influence of the outliers grows quadratically and the model is fitted to these. Third, the model only works with small displacement fields. This is due to the fact that we approximate derivatives using finite filters: if we increase size of the filter we can approximate greater displacements but loose accuracy and vise versa. Fourth, the OFC

does not use all the information available in the images, but only the first derivatives. In the next section the model with late linearisation, that addresses the mentioned weaknesses, is introduced. One more problem arises from the data term itself: a change in illumination between t and $t+1$ would be registered as movement, as is suggested by the OFC. This will be studied more in detail in Section 3.9.

3.5.2 Late Linearisation

The model that we introduce here is based on late linearisation of the constancy term(s), with non-quadratic data- and smoothness terms. By late linearisation we mean that linearisation of the constancy term(s) is postponed until discretisation of rest of the terms [50][3][15]. This model addresses several of the weaknesses that the early linearisation model suffers from: (1) it works with large displacements; (2) the model takes into account outliers in the data term; and (3) the displacement fields are piece-wise smooth. Energy functional for the model is given in (3.9).

$$E(u,v) = \int_\Omega \sum_{k=1}^{K} \left(\underbrace{\Psi_D\left(\left(I_{k,1} - I^w_{k,0}\right)^2\right)}_{\text{data}} + \alpha \underbrace{\Psi_R\left(|\nabla u|^2 + |\nabla v|^2\right)}_{\text{smoothness}} \right) \mathbf{dx} \tag{3.9}$$

where the sub-index k refers to the channels (e.g. R, G or B) of a vector valued image I_k, α is the weight of the smoothness term, $I_{k,t} = I(x,y,k,t)$ and $I^w_{k,t} = I(x+u, y+v, k, t)$ refers to a 'warped' image. Typical warping transformations are bilinear- or bicubic interpolation. In other words, we are looking for a parametrised transformation (that we call warping), defined by (u,v), that transforms the image taken at $t=0$ into the image taken at $t=1$. The purpose of the warping operation is to reduce motion (displacement) in a coarse-to-fine processing approach (for a more detailed explanation, see Section 3.5.2.2). The spatial gradient operator is given by $\nabla := \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}\right)^T$. $\Psi_D(s^2)$ and $\Psi_R(s^2)$ are robust error functions with the idea of admitting outliers in the model. These kind of error functions can also be incorporated in the early linearisation model. In the smoothness term case, the error function makes the solution piece-wise smooth, or in other words, the model does not smoothen across object boundaries. In the data term case outliers are, for example, occlusions and image structures not seen in the other image or noise and illumination errors. The specific error function in the date term case, that we have used, is $\Psi_D(s^2) = \sqrt{s^2 + \epsilon^2}$ (see Figure 3.4) [15][13]

which is applied individually to each channel [87][59] (ϵ is used for stabilisation [1]). The smoothness term that we have used in Equation (3.9) is based on the flow. In Section 3.7 we will give detailed information about different smoothness terms and error functions. In order to have a better insight how these error functions work in practise, Euler-Lagrange equations corresponding to (3.9) are given in (3.10).

$$
\begin{aligned}
&(E_k)_D = (I_{k,1} - I^w_{k,0})^2 \\
&E_R = |\nabla u|^2 + |\nabla v|^2 \\
&\sum_{k=1}^{K} \Psi'_D\Big((E_k)_D\Big)\Big(I_{k,1} - I^w_{k,0}\Big)\frac{\partial I^w_{k,0}}{\partial x} + K\alpha DIV\Big(\Psi'_R(E_R)\nabla u\Big) = 0 \\
&\sum_{k=1}^{K} \Psi'_D\Big((E_k)_D\Big)\Big(I_{k,1} - I^w_{k,0}\Big)\frac{\partial I^w_{k,0}}{\partial y} + K\alpha DIV\Big(\Psi'_R(E_R)\nabla v\Big) = 0
\end{aligned}
\tag{3.10}
$$

where $\Psi'_D(s^2)$ and $\Psi'_R(s^2)$ are the influence functions of their corresponding error functions. Again we use reflecting boundary conditions $\partial_n u = 0$ and $\partial_n v = 0$, where n denotes the normal to the image boundary $\partial\Omega_h$. Since the smoothness terms and the error functions are the same, or behave similarly, in both the optical-flow and the disparity cases, we talk more about these in Section 3.7. By introducing the models and their components in this fashion we avoid having to repeat some of the information. So far we have only mentioned the 'pros' of this model versus to the early linearisation model. However, there are 'cons' as well. Here we only briefly mention some of the problems related to this model. These will be discussed in more detail in Section 5.6.2, where we also explain how this model can be solved. Firstly, due to the robust error terms, the Euler-Lagrange equations are non-linear. Therefore, we need additional linearisation steps. Secondly, Because of the non-linear data term, the energy functional can be non-convex. This means, that several local minima might exist [3].

Since this model is of particular interest in this work, a few words of how it came to be are in order. The original model is that of Horn&Schunck [33] that we already saw in Section (3.5.1). An important step towards the model seen here was made by Nagel&Enkelmann [50]. Their model already used late linearisation of the constancy term. However, while the optical flow constraint is centred in $I_{k,1}$, the smoothness term (anisotropic) is based on $I_{k,0}$. This 'inconsistency' was corrected in the model by Alvarez et al. [3]: in their model both the optical flow constraint and the (anisotropic) smoothness terms are based on $I_{k,1}$. They also used a linear scale-space framework, i.e.

coarse-to-fine processing, in order to avoid convergence to a physically irrelevant local minima. Finally, Brox et al. [15][13] combined the ideas of TV-regulariser [36], late linearisation and robust error function for the data term, resulting in the model that we already saw in (3.9). This is the basic model that we use and extend in this work.

3.5.2.1 Coarse-to-fine and Multi-resolution

In order to explain how and why this model can cope with large displacements, we have to briefly introduce some concepts first. These concepts are *multi-resolution*, *multigrid* and *coarse-to-fine processing*. With multi-resolution we refer to an image pyramid of downscaled images. Downscaling is closely related to the scale-space theory: we expect that any structure on a coarse scale has a 'cause' on a finer scale. In other words, we expect that no new structures are generated when downscaling the images. Therefore, before sub-sampling the image, the image is smoothed, typically by applying a linear filter (e.g. Gaussian). We could build the image pyramid simply based on the scale-space theory and represent the image as a one parameter family of smoothed images and omitting the sub-sampling. However, from computational efficiency point of view (see Section 5.3.5), the multi-resolution approach is more attractive: less data to process translates into better efficiency.

Multigrid is a method, or a group of methods, used in numerical analysis for efficiently solving differential equations using a hierarchy of discretisations [74]. Multigrid is closely related to the scale-space theory and the downscaling of images. An iterative solver can be considered a 'smoothener' [74]: before transferring the problem to a coarse grid, the smoothener is applied. A smooth error term can be approximated well on a coarse grid (see Section 5.3.5).

Coarse-to-fine approach means that the image processing task is done from coarse to fine direction (i.e. starting at the coarsest scale), independently of how the image pyramid was built.

3.5.2.2 Large Displacements

By embedding it in a coarse-to-fine scheme, this model can cope with large displacements. Both linear scale-space and multi-resolution (based on image downscaling) approaches have been tested successfully. For example, in [3] Alvarez et al. used a coarse-to-fine scheme based on a linear scale-space, while in [15][13] Brox et al. used

the multi-resolution approach. In this work, we use the latter approach in order to improve the computational efficiency (see Section 5). Typically, when we start searching for the solution, we start with an approximation $u \equiv v \equiv 0$ for the flow. This is a reasonable 'guess' and produces good results for small displacements. However, for larger displacements this might not work so well. The main reasons that this model can cope with large displacements, when embedded in the coarse-to-fine scheme, are as follows. Firstly, the scheme avoids getting stuck on physically irrelevant local minima by searching solution to a simplified version of the problem on a coarse scale and gradually working through to the finest scale, while propagating the solution. This kind of processing bears similarity to the Graduated Non-Convexity (GNC) [12] method. Secondly, due to warping the displacements are kept small when changing from coarser to finer scale (see Section 5.6.2.1). This means that we can correctly approximate the derivatives using filters with finite size: if the derivatives are not properly approximated, the method will not converge to a correct solution (unless by an accident). It is easy to see that in the multi-resolution case the displacements, due to sub-sampling, are kept small on the coarse scales. However, in the linear scale-space case the situation is somewhat similar: image information is 'mixed' due to the smoothening process and, therefore, image derivatives can correctly be approximated for the structures present at the given scale.

3.5.3 Summary

To summarise, in this section optical-flow approximation models based on both early- and late-linearisation of the constancy terms were introduced. The early linearisation model that was introduced is basically that of Horn&Schunck. Similar kind of robust error functions that are used in the late linearisation model, could also be used in the early linearisation model. Also, warping with a coarse-to-fine approach could be incorporated, allowing the model to cope with larger displacements. However, the data term in early linearisation case, based on the OFC, only takes into account first order derivatives, while in the late linearisation case all the information available is taken into account.

At this stage we have a model that admits outliers in both the data- and the smoothness terms and copes with large displacements. In the following we show how

this same model can be used for disparity calculation. In Chapter 5 we show how the governing PDEs can be solved efficiently.

3.6 Stereo Disparity

In this section we introduce the model used for approximating disparities, denoted by d, for rectified stereo-images. Here the term rectification means two things: (a) correcting the lens aberration and (b) transforming the left- and right images so that the corresponding vertical lines between the images are aligned, meaning that the search for correspondences is limited to a line. Second part of the transformation can be included directly in the model by parameterising the displacement based on the fundamental matrix F and the disparity d as in [5]. However, in our model we expect that the images are fully rectified. The disparity field that we are searching for is the one that minimises the 'generic' energy functional given in (3.11).

$$E(d) = \int_{\Omega} \Big(D(I_{L,k}, I_{R,k}, d) + \alpha S(\nabla I_L, \nabla d) \Big) \mathbf{dx} \tag{3.11}$$

where $D(I_{L,k}, I_{R,k}, d)$ is the data term, while $S(\nabla I_{L,k}, \nabla d)$ is the regularisation term, $I_{\{L,R\},k}$ refers to a k:th channel of left or right image (defined by sub-index L or R) and $\nabla = [\frac{\partial}{\partial x}, \frac{\partial}{\partial y}]$ is the spatial gradient operator and Ω is the image domain. By channel here we mean channel of a vector valued image, such as RGB. Without k written explicitly, all channels are referred. $\alpha > 0$ is the weight of the smoothness term.

3.6.1 Late Linearisation

As it was already mentioned earlier, the model is basically the same as the late linearisation model for optical-flow and was first described by Slesareva et al. [67]. The energy functional of the model is given in (3.12).

$$E(d) = \int_{\Omega} \sum_{k=1}^{K} \left(\underbrace{\Psi_D \left(\left(I_{L,k} - I^w_{R,k} \right)^2 \right)}_{\text{data}} + \alpha \underbrace{\Psi_R \left(|\nabla d|^2 \right)}_{\text{smoothness}} \right) \mathbf{dx} \tag{3.12}$$

where $I_{L,k} = I_L(x, y, k)$ and $I^w_{R,k} = I_R(x + d, y, k)$ refers to a 'warped' image. Sub-indices L and R refer to left and right images, correspondingly. Similarly as in the optical-flow case, we are looking for a parametrised transformation, defined by d, that transforms the right image into the left image. As in the optical-flow case, $\Psi_D(s^2)$ and $\Psi_R(s^2)$ are robust error functions with the idea of admitting outliers in the model: the solution is piece-wise smooth and the model admits outliers in the date. As in the optical-flow case, the specific error function that we have used for the data term is $\Psi_D(s^2) = \sqrt{s^2 + \epsilon^2}$ (see Figure 3.4). The smoothness term in Equation (3.12) is based on the flow (disparity), but could be based on the image itself. For a more detailed explanation of the smoothess terms and error functions, see Section 3.7. Corresponding Euler-Lagrange equation is given in (3.13)

$$\begin{aligned}
&(E_k)_D = (I_{L,k} - I^w_{R,k})^2 \\
&E_R = |\nabla d|^2 \\
&\sum_{k=1}^{K} \Psi'_D\Big((E_k)_D\Big)\Big(I_{L,k} - I^w_{R,k}\Big)\frac{\partial I^w_{R,k}}{\partial x} + K\alpha DIV\Big(\Psi'_R(E_R)\nabla d\Big) = 0
\end{aligned} \tag{3.13}$$

where $\Psi'_D(s^2)$ and $\Psi'_R(s^2)$ are the influence functions of their corresponding error functions. These are discussed in Section 3.7.

To summarise, we now have a model for approximating correspondences in rectified stereo-images [31]. The model admits outliers in the data and the resulting disparity map is piece-wise smooth. Due to the versatility of the variational calculus, modelling of the problem is 'transparent'. Therefore, using this mathematical machinery has allowed us to model both the problems (optical-flow and disparity) in a concise way.

3.7 Smoothness Terms and Error Functions

The smoothness terms that we have used in this work are *isotropic* or, in other words, diffusion is scalar valued. Another choice would be *anisotropic* diffusion, where directional information is taken into account by means of a diffusion tensor. Whereas isotropic diffusion stops the diffusion when an object boundary is encountered (i.e. where the flows differ significantly), anisotropic diffusion stops the diffusion normal to the boundary, but encourages diffusion tangent to the boundary, therefore, in general,

generating smoother results. We chose to use isotropic diffusion in order to reduce the computational complexity of the model. Typically methods with anisotropic diffusivity term are computationally more expensive. Apart from being isotropic or anisotropic, the smoothness term can be based on the image (image-driven) or the flow (flow-driven), or a combination of both (e.g. Constraint Adaptive Regulariser of Zimmer et al. [87]).

In the case of image-driven regulariser, diffusion is based on the image itself, i.e. diffusivity weights are calculated based on the image. The idea here being that 'big' gradient values in the image correspond to object borders. This is similar to the diffusion term used by Perona and Malik [53]. While this certainly makes sense, however, not all image borders correspond with object borders, and the results can be, naturally depending on the used image sequence, spurious. On the other hand, in the case of flow-driven regulariser, diffusion is based on the flow field itself. Here the idea is to reduce diffusion when there is a 'big' change (i.e. gradient) in the flow field, thus making the solution piece-wise smooth. However, since the calculated flow fields are only approximations of the true fields, the edges of the object borders might not be as sharp as in the case of image-driven regularisers. Equation (3.14) displays the smoothness terms used in this thesis for disparity, while Equation (3.15) shows the same for the optical-flow.

$$S(\nabla I_L, \nabla d)\Big) = \begin{cases} g(|\nabla I_L|^2)(|\nabla d|^2) & \text{, image-driven} \\ \Psi_R(|\nabla d|^2) & \text{, flow-driven} \end{cases} \tag{3.14}$$

$$S(\nabla I_{L,1}, \nabla u, \nabla v) = \begin{cases} g(|\nabla I_{L,1}|^2)(|\nabla u|^2 + |\nabla v|^2) & \text{, image-driven} \\ \Psi_R(|\nabla u|^2 + |\nabla v|^2) & \text{, flow-driven} \\ (|\nabla u|^2 + |\nabla v|^2) & \text{, flow-driven (Tikhonov)} \end{cases} \tag{3.15}$$

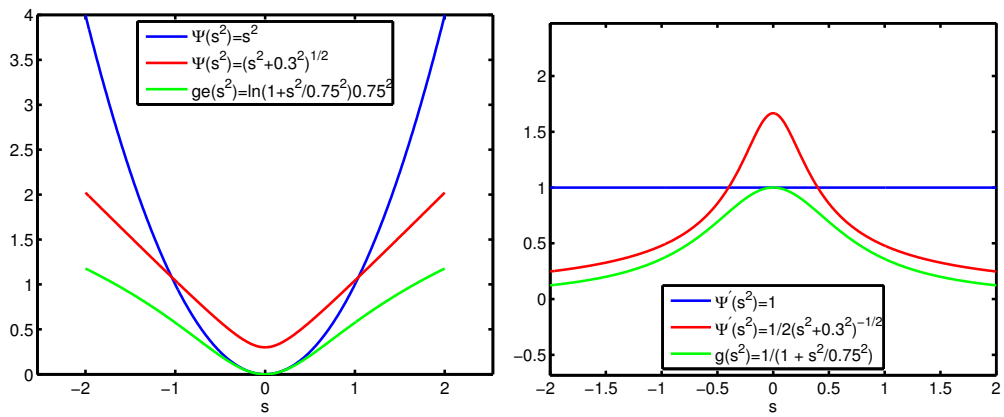
where $\Psi_R(s^2)$ and $g(s^2)$ are functions which purpose is to reduce diffusivity at the presence of borders. The specific error functions (and corresponding influence functions) that we have used in the smoothness term are given in Table 3.3.

In Table 3.3: the first one is known as Tikhonov regulariser; the second one is known as total variance regulariser (TV) [36]; while the third is the same used by Perona and Malik [53]. Graphs of the above mentioned error- and influence functions are depicted in Figure 3.4.

Figure 3.4 depicts both error- and influence functions for quadratic, TV and logarithmic error functions. As it can be observed, influence functions of the corresponding

Table 3.3: Error functions used in the smoothness term.

ERROR AND CORRESPONDING INFLUENCE FUNCTIONS	
$\Psi_R(s^2) = s^2$	$\Psi'_R(s^2) = 1$
$\Psi_R(s^2) = \sqrt{s^2 + \epsilon^2}$	$\Psi'_R(s^2) = 1/\sqrt{s^2 + \epsilon^2}$
$ge(s^2) = ln(1 + s^2/\lambda^2)\lambda^2$	$g(s^2) = 1/(1 + s^2/\lambda^2)$



(a) Error functions. (b) Influence/penalty functions.

Figure 3.4: Error- and influence functions. (a) Error functions; (b) corresponding influence functions. $\epsilon = 0.3$ and $\lambda = 0.75$.

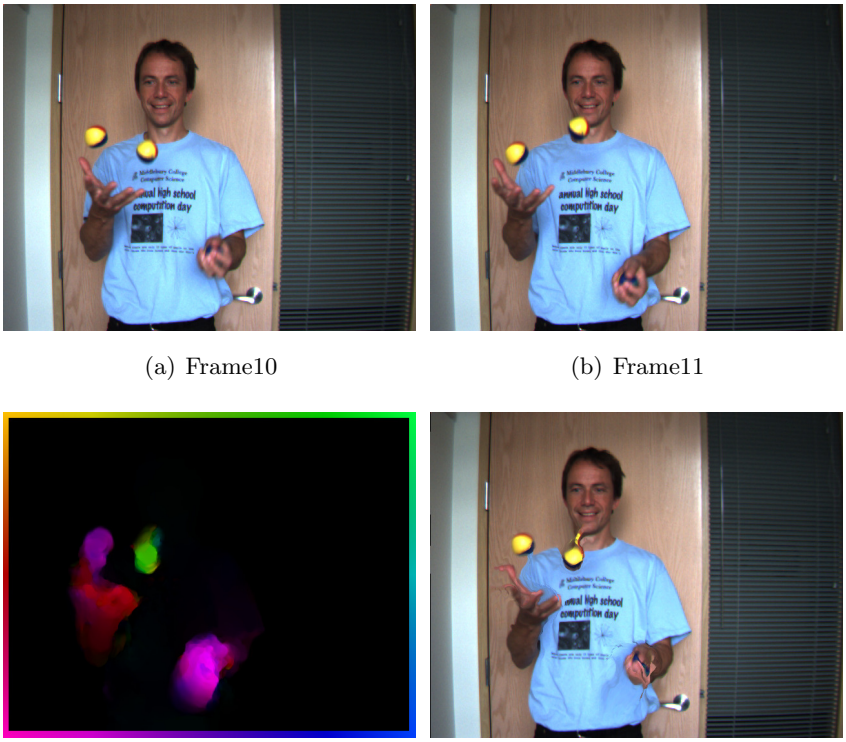
TV and logarithmic error functions are asymptotically decreasing: as the error increases, the influence of the term in question decreases. This has the effect of making the solution piecewise smooth: diffusion is decreased if $|\nabla u|^2 + |\nabla v|^2$, $|\nabla d|^2$ or $|\nabla I_L|^2$ has a 'big' value.

Data Term Error Function. We have used the TV error function in the data term as well: if the error in the data term is very big, it might be due to an occlusion (i.e. image structure is only seen one of the images), for example, and less weight would be given to these cases. Otherwise the model would be driven by occlusions.

Robust Error Functions and Early Linearisation. Although in this work we use a Horn&Schunck type of energy functional in the early linearisation case, robust error functions can be used in this model as well. If the displacements are expected to be reasonably small, then this model (with robust error function) should give reasonably good results.

3.8 Problems with the Models

Here we briefly discuss some of the problems associated with the late linearisation optical-flow and disparity models. While the coarse-to-fine strategy, together with warping, addresses the problem of large displacements and non-convexity, as indicated in Section 3.5.2.2, it only solves the problem of large displacements partially, as will be explained here. The basic idea is that displacements are smaller at coarser scales and by warping the images correspondingly, we can keep the displacements relatively small at finer scales and, therefore, approximate the image derivatives properly. Figure 3.5 shows both the original frames 10 and 11, and the frame 11 transformed (warped) by the optical-flow for the Beanbags[1] case.



(a) Frame10 (b) Frame11

(c) Optical-flow. (d) Frame11 warped.

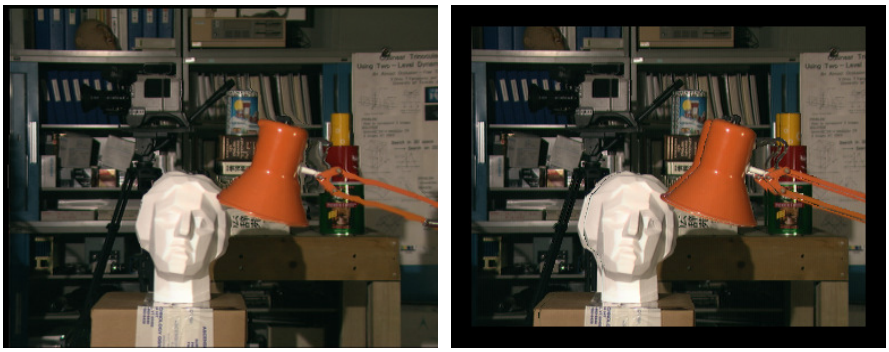
Figure 3.5: Optical-flow results for the Beanbags case. (d) displays the frame11 warped using the optical-flow field given in (c).

Now, the problem is that small image structures are not 'visible' at the coarse

[1] `http://vision.middlebury.edu/flow/data/`

scales. If these small image structures only have small displacements, then this will not be a problem. However, it becomes a problem when the small structures contain big displacements and these do not get good estimations from the coarse scales due to the fact that they are not visible. This problem has been addressed in the variational framework by Steinbruecker et al. in [68]. They address the problem by omitting warping and by conducting a 'brute force' type search in order to find the correlated pixels. By using the parallel nature of GPU (Graphical Processing Unit) architectures, this kind of a search can be parallelised very effectively. While we do not doubt the problem related to the small image structures and the coarse-to-fine algorithms, we still get acceptable results for the Beanbags[1] case, see Figure 3.5, as opposed to the results presented in [68].

Another problem is related to the occlusions and the warping. The displacement fields that we obtain with the variational methods are 100% dense. Although the data term with the robust error function $\Psi_D(s^2)$ gives less influence to the outliers (such as occlusions), we still have the occluded areas in the approximations. In Figure 3.6 we show the right Tsukuba stereo-image warped by (a) a calculated disparity map and (b) the ground-truth. Surprisingly, the results are better using the calculated disparity map. This is simply due to the warping process, which in our case is bilinear interpolation. In fact, the disparity map tells us what is the displacement of each of the pixels (i.e. where it can be found in the other image), but it is not a parameter of a bilinear interpolation operation.

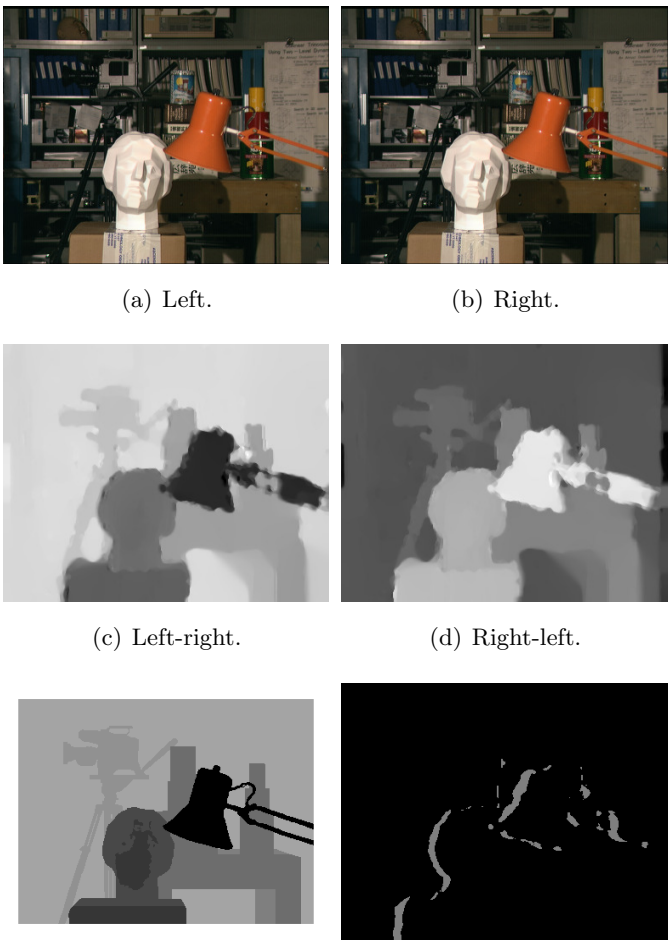


(a) Right warped. (b) Right warped gt.

Figure 3.6: Right image warped by (a) calculated disparity map and (b) the ground-truth.

[1] `http://vision.middlebury.edu/flow/data/`

One way of detecting occlusions, keeping the warping operation as it is, is by using symmetrical estimation of optical-flow or disparity fields. This has been addressed by Alvarez et el. in [4]. In Figure 3.7 we give an example related to symmetrical disparity calculation.



(a) Left. (b) Right.

(c) Left-right. (d) Right-left.

(e) Ground-truth. (f) Occlusions.

Figure 3.7: Tsukuba case: (a) left image; (b) right image; (c) symmetrical disparity, left-right; (d) symmetrical disparity, right-left; (e) ground truth; (f) occlusions (based on (c) and (d)).

Typically displacement fields are calculated only in one direction, for example displacement field between I_0 and I_1, at the same time emitting the displacement field between I_1 and I_0. As it can be understood, the displacement fields should be 'symmetrical'. In variational methods the symmetry constraint can be added, and emphasised,

as an additional term in the model. Based on these symmetrical displacement fields, we can deduce possible occlusions. In disparity case, this is very similar with left-to-right consistency check. An interested reader is pointed to [4] for more information.

3.9 Robust Data Terms

In this section we describe an analysis carried out in order to find a data representation space that would be robust under realistic illumination conditions. This is a critical issue when dealing with real world problems.

3.9.1 Motivation

The optical flow constraint [33], based on the Lambertian reflection model, states that a change in brightness of a pixel is proportional to a change in its position, i.e. the grey level of a pixel is assumed to stay constant temporally. This same constancy concept can also be used in disparity calculation by taking into account the epipolar geometry of the imaging devices (e.g. a stereo-rig). The grey level constancy, that does not hold for surfaces with non-Lambertian behavior, can be extended for vector valued images with different image representations. In this work we use a method based on calculus of variations for approximating the disparity map. Variational correspondence models typically have two terms, the first one being a data term (e.g. based on the grey level constancy), while the second one is a regularisation term used to make the solution smooth. In order to make the data term more robust with respect to non-Lambertian behaviour, different image representations can be used. Some of the problems in establishing correspondences arise from the imaging devices (e.g. camera/lens parameters being slightly different, noise due to imaging devices) and some, from the actual scene being observed (e.g. lighting conditions, geometrical deformations due to camera setup). It is clear that the underlying image representation is crucial in order for any correspondence method to generate correct, temporally coherent estimates in 'real' image sequences.

Here we propose we study how combinations of different image representations behave with respect to both illumination errors and noise, ranking the results accordingly. We believe that such information is useful to that part of the visual community that concentrates on applications, such as obstacle detection in vehicle related scenarios [34], segmentation [9] and so on. Although other authors address similar issues (e.g. that of Mileva et al. [48]), we find these to be somewhat limited in scope due to a reduced 'test bench', e.g. small amount of test images and/or image representations. Also, in most of the cases it is not satisfactorily explained how parameters related to

the model(s) have been chosen. Therefore, the main contribution of this section is an analysis of the different image representations supported by a more detailed and systematical evaluation methodology. For example, we show how optimum (or near optimum) parameters for the algorithm, related to each representation space, can be found. This is a small but important contribution in the case of real, non controlled, scenarios. The standard image representation is the RGB-space, the others being (obtained via image transformations): gradient, gradient magnitude, log-derivative, HSV, $r\phi\theta$, and phase component of an image filtered using a bank of Gabor filters.

This work is a comparative study of the chosen image representations, and it is beyond the scope of this study to explain why certain representations perform better than others in certain situations. Under realistic illumination conditions, with surfaces both complying and not complying with the Lambertian reflection model, theoretical studies can become overly complex, as we show next. It is typically thought that chromaticity spaces are illumination invariant, but under realistic lightning conditions, this is not necessarily so [47]. One of the physical models that explains light reflected by an object is the Dichromatic Reflection Model [64] (DRM), which in its basic form assumes that there is a single source of light [64], which is unrealistic in the case of real images (unless the lightning conditions can be controlled). A good example of this is given by Maxwell et al. in their Bi-Illuminant Dichromatic Reflection paper [47]: in a typical outdoor case, the main illuminants are sunlight and skylight, where objects fully lit are dominated by the sunlight while objects in shadow are dominated by skylight. Thus, as the illumination intensity decreases, the hue of the colour observed becomes bluish. For the above mentioned reasons, chromatic spaces (e.g HSV, $r\phi\theta$) are not totally illumination invariant under realistic lightning conditions. Therefore, in general, we do not speak of illumination invariance in this work but of illumination robustness or robust image representation with respect to illumination changes and noise. By illumination error we refer to varying illumination conditions between the left- and right stereo cameras.

Next, Section 3.9.2 presents the relevant related work, and some sources of error. In Section 3.9.3 we explain how the parameters related to each representation can be found. Section 3.9.4 introduces the image transformations, while Section 3.9.5 describes the conducted experiments in details. Finally, conclusions are discussed in Section 3.9.8.

3.9.2 Background Material and Related Work

3.9.2.1 Related Work and Our Contribution

The idea of robust 'key-point' identification is an important aspect of many vision related problems and has lead to such concepts as SIFT [43] (scale invariant feature transform) or SURF [8] (speeded up robust features). This work can be seen related to identifying robust features as well, however, in the framework of variational stereo. Several works comparing different data- and/or smoothness terms for optical-flow exist, for example, those of Bruhn [17] and Brox [13]. A similar work to the one presented here, carried out in a smaller scale for the optical-flow, has been done by Mileva et al. [48]. On the other hand, in [82] Wöhler et al. describe a method for 3D reconstruction of surfaces with non-Lambertien properties. Those were just to name a few. However, many comparative studies do not typically explain in detail how the parameters for each different competing algorithm and/or representation were obtained. Also, sometimes it is not mentioned, if the learn and test sets for obtaining the parameters were the same. This poses problems related to biasing and over-training. If the parameters have been obtained manually, they are prone to bias from the user: expected results might get confirmed. On the other hand, if the learn and test sets were the same and/or they were too small, it is possible that over-training has taken place and, therefore, the results are not generalisable. We argue that in order to properly rank a set of representation spaces and/or different algorithms, with respect to any performance measure, optimum parameters related to each case need to be searched consistently, with minimum human interference, avoiding over-fitting.

Our contribution. Where our work differs from the rest is that (a) we use an advanced optimisation scheme to automatically optimise the parameters related to each image representation space, (b) image sets for optimisation (learning) and testing are different in order to avoid over fitting, (c) we study the robustness of each representation space with respect to several image noise and illumination error models, and (d) we combine the results for both noise and illumination errors. Thus, the methodology can be considered to be novel.

3.9.2.2 Sources of Error

Since the approach of this study is more experimental than theoretical, we only quickly cover some of the sources of error that the correspondence methods suffer from. Although optical-flow and stereo are similar in nature, they differ in a very important aspect: in stereo, the apparent movement is due to a change of position of the observer (e.g. left and right cameras), whereas in the optical-flow case, both the observer and the objects in the scene can move with respect to each other. Thus, stereo and optical-flow do not suffer from exactly the same shortcomings. For example, in the case of stereo, shadows cast upon objects due to illumination conditions can provide information when searching corresponding pixels between images. In the case of optical-flow, a stationary shadow cast upon a moving object makes it more difficult to find the corresponding pixels. Also, as it was already mentioned in Section 3.9.1, the imaging devices also cause problems in the form of noise, motion blur, and so on. Thus, an image representation space should be robust with respect to (a) small geometrical deformations (geometrical robustness), (b) changes in the illumination level (illumination robustness), both global and local, and (c) noise present in the images (e.g. due to the acquisition device). Our analysis was carried out for stereo but is directly applicable to optical-flow as well.

3.9.2.3 Variational Stereo

The energy functional to be minimised is given in (3.16). The difference between this model and the one seen previously is that here the data term is composed of two different 'components', namely $D_1()$ and $D_2()$, each weighted by $b1$ and $b2$.

$$E(d) = \int_\Omega \Big(b_1 D_1(I_L, I_R, d) + b_2 D_2(I_L, I_R, d) \Big) \mathbf{dx} + \alpha \int_\Omega S(\nabla d) \mathbf{dx} \tag{3.16}$$

where $D_1(I_L, I_R, d)$ and $D_2(I_L, I_R, d)$ are the data terms, $S(\nabla d)$ is the regularisation term, and $I_{\{L,R\}}$ refers to the left and right images (all the channels). $b_1 \geq 0$, $b_2 \geq 0$ and $\alpha \geq 0$ are the parameters of the model, defining weight of each of the terms. Both the data and the regularisation terms are defined in Equation (3.17).

$$D_{\{1,2\}}(I_1, I_2, d) = \begin{cases} \sum_{k=1}^{K} \Psi\Big(\big(I_{L,k} - I^w_{R,k}\big)^2\Big) & \text{, type 1} \\ \sum_{k=1}^{K} \Psi\Big(|\nabla I_{L,k} - \nabla I^w_{R,k}|^2\Big) & \text{, type 2} \end{cases} \tag{3.17}$$
$$S(\nabla d) = \Psi(|\nabla d|^2)$$

where $I_{L,k} = I\big(x, y\big)_{L,k}$ is the k:th channel (e.g. R, G or B channel of a RGB image) of the left image, $I^w_{R,k} = I\big(x + d(x, y), y\big)_{R,k}$ is the k:th channel of the right image warped as per disparity $d = d(x, y)$, and $\Psi(s^2) = \sqrt{s^2 + \epsilon^2}$ is the non-quadratic error function. Usage of type 1 and type 2 is explained in the section 3.9.4. Now with both the energy functional and the related data terms described, a physical interpretation of the model can be derived: we are looking for a transformation described by d that transforms the right image into the left image with the d being piecewise smooth. By transforming the right image into the left image we mean that the image features described by the data term(s) align.

3.9.3 Searching for Optimal Parameters with Differential Evolution

Since the main idea of this work is to rank the chosen image representation spaces with respect to robustness, we have to find an optimum (or near optimum) set of parameters [b_1, b_2, α] for each different case, avoiding over-fitting. As was already mentioned, using a human operator would be prone to bias. Therefore, we have decided to use a gradient free, stochastic, population based function minimiser called Differential Evolution[1] (DE) [70][71]. The rationale for using DE is that it has empirically been shown to find the optimum (or near optimum), it is computationally efficient, and the cost function evaluation can be parallelised efficiently. The principal idea behind DE is to represent the parameters to be optimised as vectors where each vector is a population member whose fitness is described by the function value. Two members (parents) are stochastically combined into a new one (offspring) which then competes against the rest on the coming cycles. By recurring to the survival of the fittest theorem, the 'fittest' members contribute more to the coming populations and thus, their characteristics overcome those of the weak members, therefore minimising (or maximising) the function value [70][71]. In our case, this implies that the function value is the error between

[1] www.icsi.berkeley.edu/∼storn/code.html

the calculated disparity map and ground truth (see Equation (3.28)) and the member represents the algorithm's parameters. DE itself is computationally cost efficient, the problem being that several function evaluations (one per population member) per cycle are needed. However, the optimisation can be parallelised by keeping the members on the master computer and by calculating the function values on several slave computers simultaneously, which was the adopted strategy. This method of parallelizing DE is certainly not new and has been reported earlier by, for example, Plagianakos et al. [54], Tasoulis et al. [72] and Epitropakis et al. [23].

In order to compare the results obtained using different combinations of the image representations, we adopt a strategy typically used in pattern recognition: the input set (a set of stereo-images) is divided into a learning, a validation, and a test set. The learning set is used for obtaining the optimal parameters while the validation set is used to prevent over fitting: during the optimisation, when the error for the validation set starts to increase, we stop the optimisation process, therefore keeping the solution 'general'. This methodology is completely general and it can be applied to any other image registration algorithm with only some small modifications.

3.9.4 Image Transformations

In this section, we describe the image transformations that we have decided to evaluate. We have chosen the most common image representations as well as other transformations that have been proposed in the literature, due to their robustness and possibility of real-time implementation. The different tested data term combinations that have been studied are given in Table 3.4. The first column (1st term) gives the image representation for $D_1(I_L, I_R, d)$, while the second column (2nd term) gives the image representation for $D_2(I_L, I_R, d)$ in Equation (3.17). Apart from the $|\nabla I(RGB)|^2$ (gradient magnitude) case, in the rest of the image representations type 1 data term is used (see Equation (3.17)). As it was already mentioned previously, Mileva et al. tested some of the same representations earlier in [48].

Table 3.4: Tested image representation combinations.

1st term	2nd term							
	none	RGB	RGBN	$\|\nabla I(RGB)\|^2$	HS(V)	$(r)\phi\theta$	phase	logd
none								
RGB	X		X	X	X	X	X	X
RGBN	X			X	X	X	X	X
$\nabla I(RGB)$	X	X	X	X	X	X	X	X
HS(V)	X			X		X	X	X
$(r)\phi\theta$	X			X			X	X
phase	X			X				
logd	X			X				

In the following, we briefly describe the different input representations under study.

3.9.4.1 RGBN (Normalized RGB)

In the RGBN case, the standard RGB representation is simply normalised by using a factor N. In our tests, both images are normalised by using their own factor which is $N_i = max(R_i, G_i, B_i)$, i being the image in question (e.g., left or right image). The transformation is given by Equation (3.18).

$$\begin{bmatrix} R & G & B \end{bmatrix}^T \mapsto \begin{bmatrix} \frac{R}{N} & \frac{G}{N} & \frac{B}{N} \end{bmatrix}^T \tag{3.18}$$

RGBN is robust with respect to global multiplicative illumination changes.

3.9.4.2 Gradient

The gradient ($\nabla I(RGB)$) constancy term matches components of the gradient between the images. The transformation is given by Equation (3.19).

$$\begin{bmatrix} R & G & B \end{bmatrix}^T \mapsto \begin{bmatrix} R_x & G_x & B_x & R_y & G_y & B_y \end{bmatrix}^T \tag{3.19}$$

where sub-index states with respect to which variable the term in question has been derived. Gradient constancy term is robust with respect to both global and local additive illumination changes.

3.9.4.3 Gradient Magnitude

The gradient magnitude ($|\nabla I(RGB)|^2$) constancy term matches Euclidean norm of the gradient as suggested in Equation (3.17) (type 2). The transformation is given by Equation (3.20).

$$\begin{bmatrix} R & G & B \end{bmatrix}^T \mapsto \begin{bmatrix} R_x & G_x & B_x & R_y & G_y & B_y \end{bmatrix}^T \tag{3.20}$$

where sub-index states with respect to which variable the term in question has been derived. In general, this term is illumination robust with respect to both local and global additive illumination changes. As it can be observed, the actual image transformation is the same as in the $\nabla I(RGB)$ case, however, embedding in the data term is different as can be seen from Equation (3.17) (in this case type 2).

3.9.4.4 HS(V)

HSV(Hue Saturation Value) is a cylindrical representation of the colour-space where the angle around the central axis of the cylinder defines 'hue', the distance from the central axis defines 'saturation' and, the position along the central axis defines 'value' as defined in Equation (3.21).

$$\begin{aligned}
&\begin{bmatrix} R & G & B \end{bmatrix}^T \mapsto \begin{bmatrix} H & S & V \end{bmatrix}^T \\
&H = \begin{cases} 0 & \text{, if } max = min \\ (60^\circ \times \frac{G-B}{max-min})\ mod\ 360^\circ & \text{, if } max = R \\ 60^\circ \times \frac{B-R}{max-min} + 120^\circ & \text{, if } max = G \\ 60^\circ \times \frac{R-G}{max-min} + 240^\circ & \text{, if } max = B \end{cases} \\
&S = \begin{cases} 0 & \text{, if } max = 0 \\ \frac{max-min}{max} & \text{, otherwise} \end{cases} \\
&V = max
\end{aligned} \tag{3.21}$$

where $min = min(R, G, B)$ and $max = max(R, G, B)$. As can be understood from Equation (3.21), the H and S components are illumination robust while the V component is not and, therefore, we exclude the V component from the representation. In the rest of the text, HS(V) refers to image representation with only the H and S components.

3.9.4.5 Spherical

While HSV describes colours in a cylindrical space, $r\phi\theta$ does so in a spherical one. r indicates the magnitude of the colour vector while ϕ is the zenith and θ is the azimuth, as described by Equation (3.22).

$$\begin{bmatrix} R & G & B \end{bmatrix}^T \mapsto \begin{bmatrix} r & \theta & \phi \end{bmatrix}^T$$

$$\begin{aligned} r &= \sqrt{R^2 + G^2 + B^2} \\ \theta &= arctan(\tfrac{G}{R}) \\ \phi &= arcsin(\tfrac{\sqrt{R^2+G^2}}{\sqrt{R^2+G^2+B^2}}) \end{aligned} \tag{3.22}$$

As can be observed from (3.22), both the ϕ and θ are illumination robust while magnitude vector r is not and, therefore, we exclude r from the representation. In the rest of the text, $(r)\phi\theta$ and *spherical* refer to an image representation based on the ϕ and θ.

3.9.4.6 Log-Derivative

The transformation is given by Equation (3.23).

$$\begin{bmatrix} R & G & B \end{bmatrix}^T \mapsto \begin{bmatrix} (ln\ R)_x & (ln\ G)_x & (ln\ B)_x & (ln\ R)_y & (ln\ G)_y & (ln\ B)_y \end{bmatrix}^T \tag{3.23}$$

where sub-index states with respect to which variable the term in question has been derived. The log-derivative image representation is robust with respect to both additive and multiplicative local illumination changes.

3.9.4.7 Phase Component of Band-pass Filtered Image Using Quadrature Filters

We define the phase component of a band-pass filtered image as a result of convolving the input image with a set of filters [61][28][27] as proceeds. The complex-valued Gabor filters are defined as in Equation (3.24)

$$h(x; f_0, \theta) = h_c(x; f_0, \theta) + ih_s(x; f_0, \theta) \tag{3.24}$$

where $x = (x, y)$ is the image position, f_0 denotes the peak frequency, θ the orientation of the filter in reference to the horizontal axis, and $h_c()$ and $h_s()$ denote the even (real)

and odd (imaginary) parts. The filter responses (band-pass signals) are generated by convolving an input image with a filter as defined in Equation (3.25)

$$Q(x;\theta) = I * h(x; f_0, \theta) = C(x;\theta) + iS(x;\theta) \tag{3.25}$$

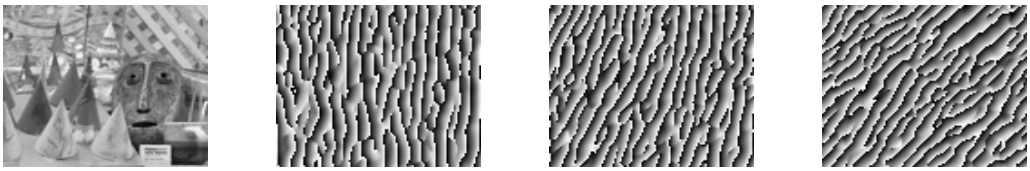
where I denotes an input image, $*$ denotes convolution, and $C(x;\theta)$ and $S(x;\theta)$ are the even and odd responses corresponding to a filter with an orientation θ. From the even and odd responses, two different representation spaces can be built, namely phase and energy, as indicated by Equation (3.26).

$$\begin{array}{l} E(x;\theta) = \sqrt{C(x,\theta)^2 + S(x;\theta)^2} \\ \omega(x;\theta) = atan(\frac{S(x;\theta}{C(x;\theta)}) \end{array} \tag{3.26}$$

where $E(fx;\theta)$ is the energy response and $\omega(x;\theta)$ is the phase response of a filter corresponding to an orientation θ. As can be observed from (3.25), the input image I can contain several components (eg. RGB, HSV) where each component would be convolved independently to extract energy and phase. However, in order to maintain the computation time reasonable, the input images are first converted into grey-level images, after which the filter responses are calculated. Therefore, the transformation can be defined as indicated by (3.27).

$$\begin{bmatrix} R & G & B \end{bmatrix}^T \mapsto \begin{bmatrix} \omega(x;\theta) \end{bmatrix} \quad \theta = 1, 2, 3, \ldots, 8 \tag{3.27}$$

The reason for choosing the phase representation is threefold: (a) the phase component is robust with respect to illumination changes; (b) cells with a similar behaviour have been found in the visual cortex of primates [35] which might well mean that evolution has found this kind of representation to be meaningful (even if we might not be able to exploit it completely yet); and (c) the stability of the phase component with respect to small geometrical deformations (as shown by Fleet and Jepson [28][27]).



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

Figure 3.8: Phase response of Cones stereo-image: (a) original image; (b) phase response corresponding to $\theta = 0°$; (c) phase response corresponding to $\theta = 22.5°$; (d) phase response corresponding to $\theta = 45°$.

3.9.5 Experiments

The purpose of the experiments was to study, both quantitatively and qualitatively, how each of the chosen image representations performs using both the original images and images with induced illumination errors and/or noise. This kind of an analysis not only allows us to study how each of the representations behaves using the original images (naturally containing some noise due to the imaging devices), but also gives an insight of how robust each of the representations actually is: those representations that produce similar results with or without induced errors can be regarded to be robust. Due to the availability of stereo-images (with different illumination/exposure times) at the Middlebury[1] database, with ground-truth, these were used for the quantitative experiments. For qualitative analysis and functional validation, images from the DRIVSCO[2], and the GRASP[3] projects were used.

[1] http://vision.middlebury.edu/stereo/data/
[2] http://www.pspc.dibe.unige.it/∼drivsco/
[3] http://www.csc.kth.se/grasp/

Figure 3.9: Stereo-images from the Middlebury database used in the quantitative experiments. (a) Aloe; (b) Art; (c) Baby1; (d) Baby2; (e) Baby3; (f) Books; (g) Bowling; (h) Cloth1; (i) Cloth2; (j) Cloth3; (k) Cones; (l) Dolls; (m) Lampshade1; (n) Lampshade2; (o) Laundry; (p) Moebius; (q) Plastic; (r) Reindeer; (s) Rocks1; (t) Rocks2; (u) Teddy; (v) Tsukuba; (x) Venus; (y) Wood1; (z) Wood2.

Even if no vigorous image analysis was used when choosing the images, both the learn- and test sets were carefully chosen by taking the following into consideration:

(a) none of the sets contains known cases where the variational method is known to fail completely; (b) both very textured (e.g. Aloe and Cloth1) and less textured cases are included (e.g Plastic and Wood1). The reason for not including cases where the algorithm fails is that in these cases the effect of the used image representation would be negligible and thus, would not convey useful information for our study. We have researched the above mentioned cases using spatial (and temporal) constraints and obtained good results even in these more challenging cases [58].

3.9.5.1 K-fold Cross-Validation

Because of the limited size of the data set for the quantitative experiment, a set of 25 different stereo-images, we have used a technique called *k-fold cross-correlation* [22][40] to statistically test how well the obtained results are generalisable. In our case, due to the size of the data set, we use a 5-fold cross-correlation: the data set is broken in five sets, each containing five images. Then, we run the DE and analyse the results five times using three of the sets for learning, one for validation and one for testing. In each run the sets for learning, validation and testing will be different. Results are based on all of the five runs. Below is a list of sets for one of the runs.

Learn: Lampshade2, Cloth1, Rocks2, Baby3, Reindeer, Baby2, Cones, Plastic, Tsukuba, Art, Wood1, Rocks1, Dolls, Cloth3, Cloth2

Test: Aloe, Baby1, Books Lampshade1, Wood2

Validation: Bowling2, Laundry, Moebius, Venus, Teddy

3.9.5.2 Induced Illumination Errors and Image Noise

Following is a list of the types of illumination errors that we have used:

- global additive (GA)
- global multiplicative (GM)
- global multiplicative plus additive (GMA)
- local additive (LA)
- local multiplicative (LM)

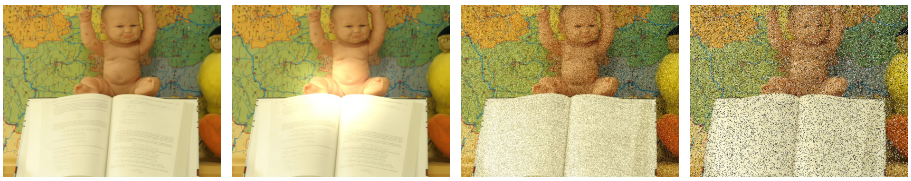
- local multiplicative plus additive (LMA)

Local illumination error was simulated using a 2D Gaussian probability density function. For image noise, the following types were used:

- luminance mild (LM), luminance severe (LS)
- chrominance mild (CM), luminance severe (CS)
- salt&pepper mild (SPM), salt&pepper severe (SPS)

Luminance and chrominance noise were simulated using a Gaussian distribution with the following parameters: (a) mild with mean 0 and deviation of 10 and (b) severe with mean 0 and deviation of 30. Salt&pepper type of noise was simulated with the following probabilities: (a) mild $P(s) = 0.05$ and $P(p) = 0.05$ and (b) severe $P(s) = 0.1$ and $P(p) = 0.1$, where $P(s)$ and $P(p)$ denote the probability of salt and pepper noise, correspondingly, being present at each image position [29]. We chose to use the above mentioned types to simulate illumination errors, since the global error is though to simulate a difference between the left- and right-cameras (i.e due to different aperture), while the local illumination error is though to simulate a flare type of error. On the other hand, luminance, chrominance and salt&pepper type of noise are though to simulate the kind of noise that we might encounter in the case of digital imaging devices.

Fig. 3.10 displays some of the illumination errors and image noise for the Baby2 image.



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

Figure 3.10: Baby2. (a) Original; (b) Local multiplicative (LM) plus local additive (LA); (c) Severe luminance (LS); (d) Severe salt&pepper (SPS).

It can be argued that the possible noise present in the images could be reduced or eliminated by using a de-noising pre-processing step, and thus, the study should

be centred more towards illumination type of errors. However, if any of the tested image representations proves to be sufficiently robust with respect to both illumination errors and image noise, this would mean that less pre-processing steps would be needed. This certainly would be beneficial in real applications, possibly suffering from restricted computational power.

3.9.5.3 Error Metric

The error metric that we decided to use is the mean average squared error (MASE) as given in (3.28).

$$MASE := \frac{1}{N} \sum_{i=1}^{N} \left((d)_i - (d_{gt})_i \right)^2 \tag{3.28}$$

where N is the number of pixels, d is the calculated disparity map, and d_{gt} is the ground truth.

3.9.6 Results

In this section we present the results, both quantity and visual quality wise. First, the results are given by ranking how well each representation has done, both accuracy and robustness wise. Then we study how combining different representations has affected the accuracy and the robustness of these combined representations. After this we present the results some for real applications visual quality wise, since ground-truth is not available for these.

3.9.6.1 Ranking

Here we rank each of the representation spaces in order to gain a better insight on the robustness and accuracy of each representation. By robustness and accuracy we mean the following: (a) a representation is considered robust when results based on this are affected only little by noise and/or image errors; (b) a representation is considered accurate when results based on this gives good results using the original (i.e. noiseless) images. While this may not be the standard terminology, we find that using these terms makes it easier to explain the results. In Table 3.5, each of the representations is ranked with respect to (a) the original images and (b) the combined illumination errors and noise types, while Table 3.6 combines the aforementioned results into a single ranking.

Mean value in the tables is the mean of MASE based on all the five different runs (see Section 3.9.5.1).

Table 3.5: Ranking for combined noise+error and original images.

Rank	Error+noise	mean	Original images	mean
1.	$\nabla I(RGB)$+phase	72.3	$\nabla I(RGB)$+HS(V)	35.1
2.	$\nabla I(RGB)$+none	83.6	HS(V)+logd	37.5
3.	phase+$\lvert\nabla I(RGB)\rvert$	84.1	$\nabla I(RGB)$+rgb	39.0
4.	$\nabla I(RGB)$+logd	87.4	$\nabla I(RGB)$+rgbn	39.4
5.	phase+none	92.3	HS(V)+phase	40.6
6.	$(r)\phi\theta$+phase	92.4	$(r)\phi\theta$+phase	42.2
7.	$\nabla I(RGB)$+$\lvert\nabla I(RGB)\rvert$	92.6	$\nabla I(RGB)$+none	44.9
8.	rgbn+logd	92.8	$\nabla I(RGB)$+logd	45.6
9.	logd+none	97.5	$\nabla I(RGB)$+phase	46.0
10.	rgb+phase	102.7	rgb+logd	46.0
11.	logd+$\lvert\nabla I(RGB)\rvert$	105.5	rgbn+logd	46.8
12.	rgbn+$\lvert\nabla I(RGB)\rvert$	111.5	rgbn+phase	47.1
13.	HS(V)+$\lvert\nabla I(RGB)\rvert$	112.0	phase+$\lvert\nabla I(RGB)\rvert$	47.7
14.	rgb+$\lvert\nabla I(RGB)\rvert$	112.9	rgb+phase	47.9
15.	$\nabla I(RGB)$+rgbn	114.8	phase+none	48.3
16.	rgbn+phase	120.0	logd+none	50.7
17.	HS(V)+phase	120.2	HS(V)+$(r)\phi\theta$	53.2
18.	rgb+logd	125.7	$(r)\phi\theta$+none	53.6
19.	$\nabla I(RGB)$+$(r)\phi\theta$	134.1	logd+$\lvert\nabla I(RGB)\rvert$	55.2
20.	$(r)\phi\theta$+$\lvert\nabla I(RGB)\rvert$	139.8	HS(V)+$\lvert\nabla I(RGB)\rvert$	55.9
21.	$\nabla I(RGB)$+rgb	175.0	$(r)\phi\theta$+$\lvert\nabla I(RGB)\rvert$	56.9
22.	$\nabla I(RGB)$+HS(V)	180.4	rgb+$\lvert\nabla I(RGB)\rvert$	57.7
23.	HS(V)+logd	278.4	$\nabla I(RGB)$+$\lvert\nabla I(RGB)\rvert$	59.8
24.	HS(V)+none	293.8	rgbn+$\lvert\nabla I(RGB)\rvert$	62.1
25.	rgb+HS(V)	360.8	rgbn+HS(V)	74.8
26.	rgbn+HS(V)	373.8	$\nabla I(RGB)$+$(r)\phi\theta$	99.3
27.	rgbn+none	374.4	$(r)\phi\theta$+logd	103.7
28.	rgb+none	380.7	rgb+$(r)\phi\theta$	119.4
29.	rgb+rgbn	394.3	HS(V)+none	134.3
30.	$(r)\phi\theta$+logd	394.8	rgbn+$(r)\phi\theta$	166.3
31.	HS(V)+$(r)\phi\theta$	563.8	rgb+HS(V)	178.8
32.	$(r)\phi\theta$+none	712.2	rgb+none	224.8
33.	rgbn+$(r)\phi\theta$	716.7	rgb+rgbn	239.1
34.	rgb+$(r)\phi\theta$	727.4	rgbn+none	260.3

Table 3.6: Combined ranking

Rank	Representation space	Combined ranking
1.	$\nabla I(RGB)$+none	9
2.	$\nabla I(RGB)$+phase	10
3.	$\nabla I(RGB)$+logd	12
4.	$(r)\phi\theta$+phase	12
5.	phase+$\lvert\nabla I(RGB)\rvert$	16
6.	$\nabla I(RGB)$+rgbn	19
7.	rgbn+logd	19
8.	phase+none	20
9.	HS(V)+phase	22
10.	$\nabla I(RGB)$+HS(V)	23
11.	$\nabla I(RGB)$+rgb	24
12.	rgb+phase	24
13.	HS(V)+logd	25
14.	logd+none	25
15.	rgb+logd	28
16.	rgbn+phase	28
17.	$\nabla I(RGB)$+$\lvert\nabla I(RGB)\rvert$	30
18.	logd+$\lvert\nabla I(RGB)\rvert$	30
19.	HS(V)+$\lvert\nabla I(RGB)\rvert$	33
20.	rgb+$\lvert\nabla I(RGB)\rvert$	36
21.	rgbn+$\lvert\nabla I(RGB)\rvert$	36
22.	$(r)\phi\theta$+$\lvert\nabla I(RGB)\rvert$	41
23.	$\nabla I(RGB)$+$(r)\phi\theta$	45
24.	HS(V)+$(r)\phi\theta$	48
25.	$(r)\phi\theta$+none	50
26.	rgbn+HS(V)	51
27.	HS(V)+none	53
28.	rgb+HS(V)	56
29.	$(r)\phi\theta$+logd	57
30.	rgb+none	60
31.	rgbn+none	61
32.	rgb+rgbn	62
33.	rgb+$(r)\phi\theta$	62
34.	rgbn+$(r)\phi\theta$	63

As can be observed from Table 3.5, the most robust representation was $\nabla I(RGB)$+phase, while the second was $\nabla I(RGB)$ without any combinations. Since both $\nabla I(RGB)$ and phase represent different physical quantities (gradient and phase of the image signal as the names suggest), and both of these have been shown to be robust, it is not surprising that a combination of these was the most robust representation. In general, representations based on both $\nabla I(RGB)$ and phase were amongst the most robust representations, while $\nabla I(RGB)$+HS(V) was the most accurate representation with the original images (i.e. without induced errors or noise). In general, representations based on $\nabla I(RGB)$ have produced good results with the original images. On the other hand, as can be observed from Table 3.6, the best combined ranking was produced by $\nabla I(RGB)$ alone. Also it can be noted that the first three are all based on $\nabla I(RGB)$. However, $\nabla I(RGB)$+phase is slightly more robust than $\nabla I(RGB)$ alone, but not as accurate. This is clear from the figures presented in Section 3.9.7.

3.9.6.2 Improvement Due to Combined Representation Spaces

In the following we show how each of the basic representations (1st term in Table 3.4) has benefited, or deteriorated, by being combined with different representations (2nd term in Table 3.4). In other words, we show, for example, how the error for $\nabla I(RGB)$ changes when combined with $|\nabla I(RGB)|^2$, therefore, allowing us to deduce if $\nabla I(RGB)$ benefits from the combination. In the figures 'grad', 'gradmag', and 'spherical' respectively denote $\nabla I(RGB)$, $|\nabla(I(RGB)|^2$, and $(r)\phi\theta$. Results are given with respect to error, thus a positive change in the error naturally means greater error and vice versa. Figure 3.11 displays the results for $\nabla I(RGB)$, phase, and logd while Figure 3.12 gives the same for $(r)\phi\theta$ (spherical), HS(V) and RGB. We have left out results for RGBN on purpose since this was the worst performer and the results, in general, were similar to those of RGB. In the figures comb. means combined error and noise (error+noise).

As it can be observed, combining $\nabla I(RGB)$ with any of the representations, apart from $(r)\phi\theta$, has improved both accuracy and robustness. Combining $(r)\phi\theta$ with $\nabla I(RGB)$ improves robustness but at the same time has worsens accuracy. The situation with phase is similar: combining phase with other representations, apart from $\nabla I(RGB)$, has improved both accuracy and robustness; when combined with $\nabla I(RGB)$ accu-

racy worsens slightly while robustness improves. Note that less robust representations benefit more from the combined representations than those that are already robust.

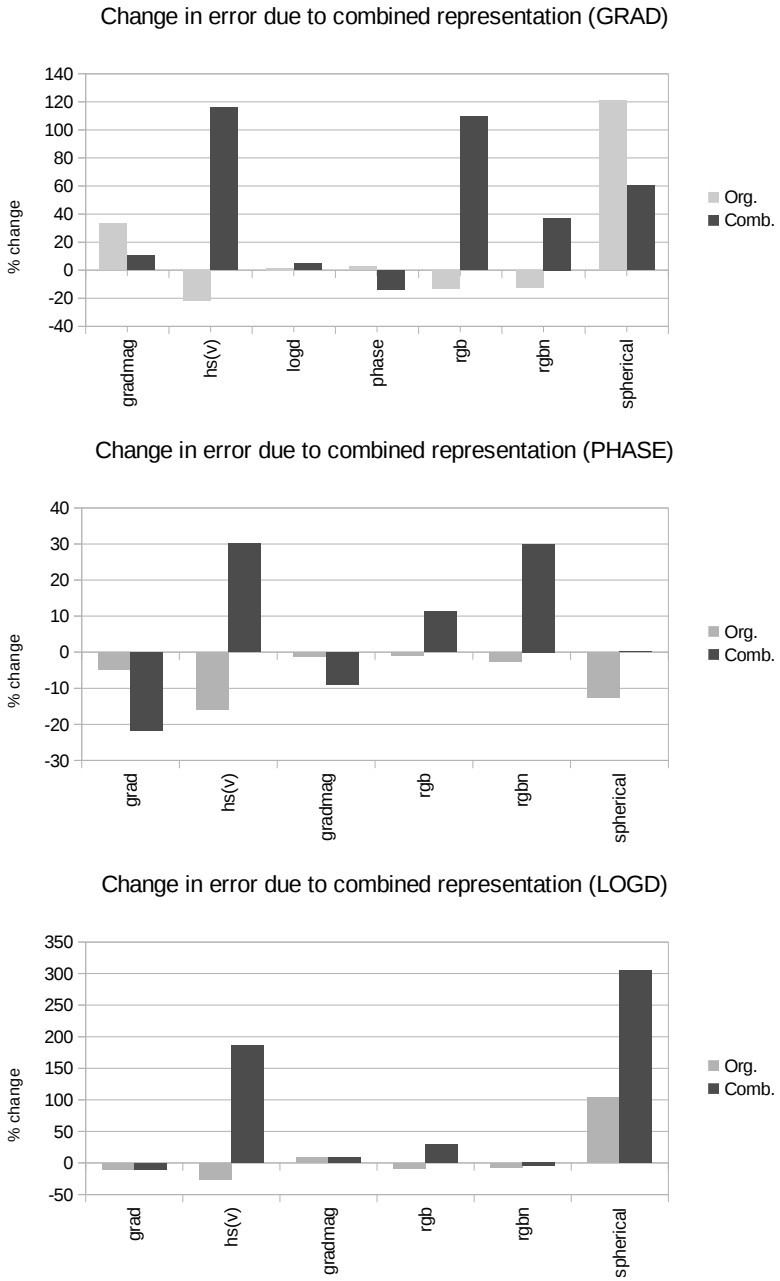


Figure 3.11: Results for $\nabla I(RGB)$, phase and logd. Org. means original images, while Comb. means combined error and noise (error+noise in the tables).

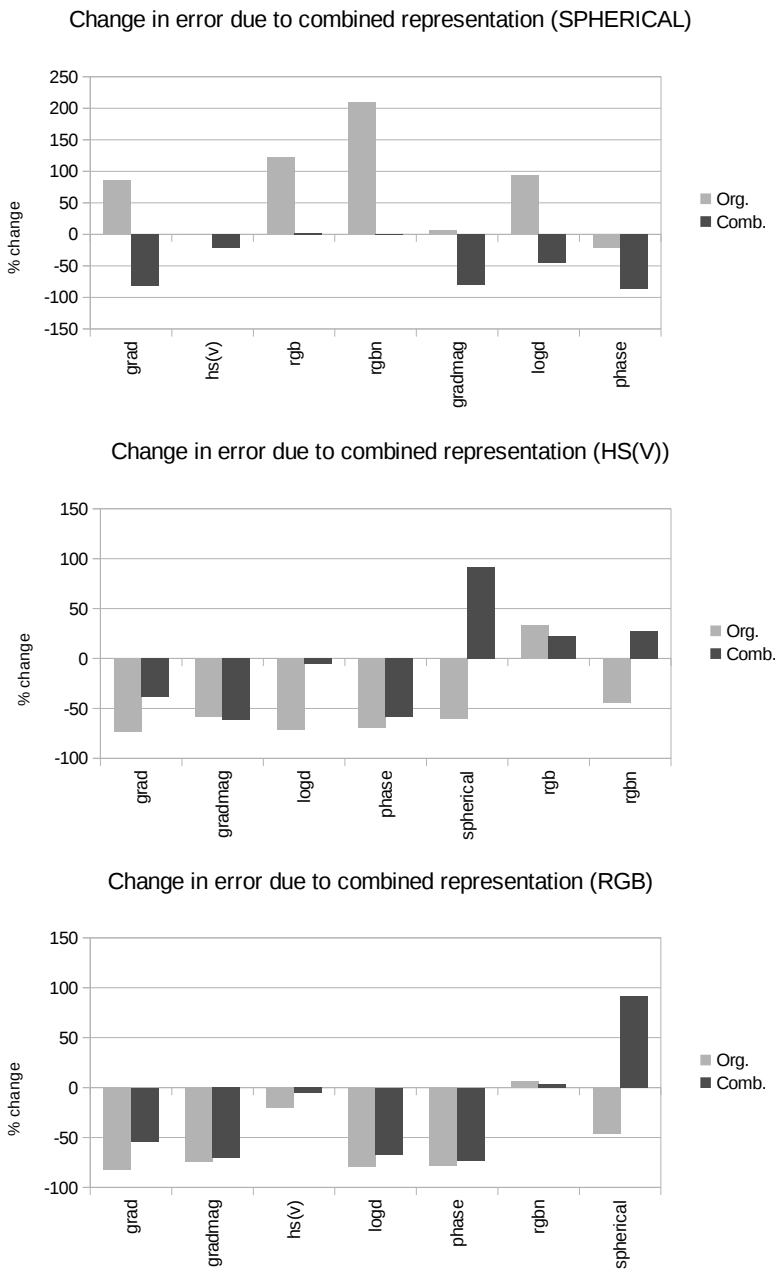
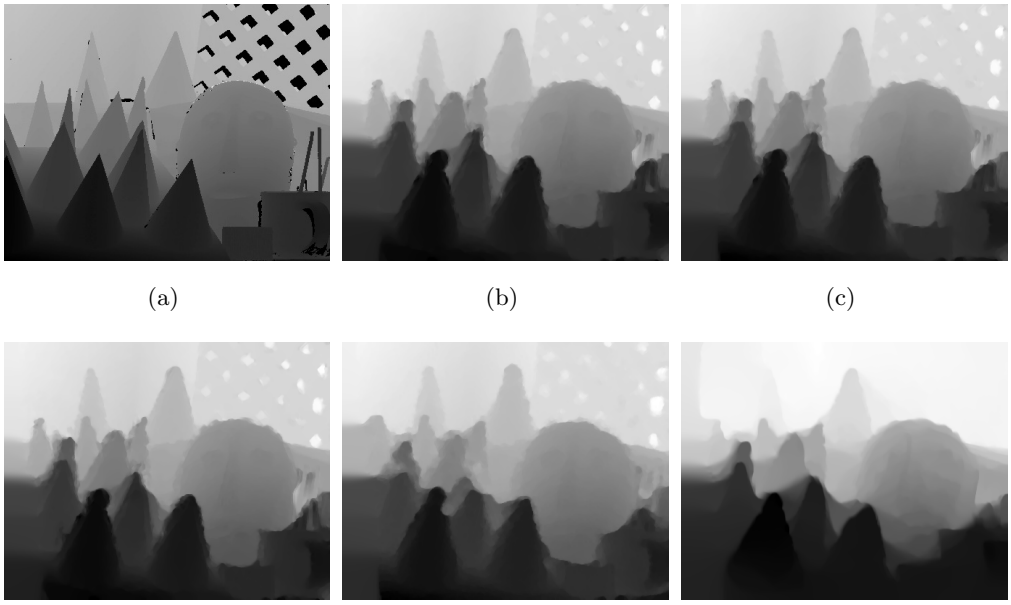


Figure 3.12: Results for $(r)\phi\theta$ (spherical), HS(V) and RGB. Org. means original images, while Comb. means combined error and noise (error+noise in the tables).

3.9.7 Visual Qualitative Interpretation

Figs. 3.13, 3.14 and 3.15 display results visually for the Cones, DRIVSCO, and GRASP cases, using the following image representations: $\nabla I(RGB)$, $\nabla I(RGB)$+phase, $\nabla I(RGB)$+HS(V),

phase, and RGB. A video of the results for DRIVSCO is available at [1]. These representations were chosen since (a) $\nabla I(RGB)$ was the overall 'winner' for the combined results (see Table 3.6); (b) $\nabla I(RGB)$+phase was the most robust; (c) $\nabla I(RGB)$+HS(V) was the most accurate; (d) phase is both robust and accurate and (e) RGB is the 'standard' representation from typical cameras. The parameters used were the same in all the cases presented here and are those from the 1st run (out of five) for the 5-fold cross-validation. The reasoning here is, confirmed by the results, that any robust representation should be able to generate reasonable results for any of the parameters found in the cross-validation scheme.



(a) (b) (c)

(d) (e) (f)

Figure 3.13: Cones. (a) Ground truth; (b) $\nabla I(RGB)$; (c) $\nabla I(RGB)$+phase; (d) $\nabla I(RGB)$+HS(V); (e) phase; (f) RGB.

As can be observed from Fig. 3.13, the results are somewhat similar for all the representations. However, as it can be observed, RGB has visually produced slightly worse results.

[1] `http://atc.ugr.es/$\sim$jarnor/index.php/publications/more-information`

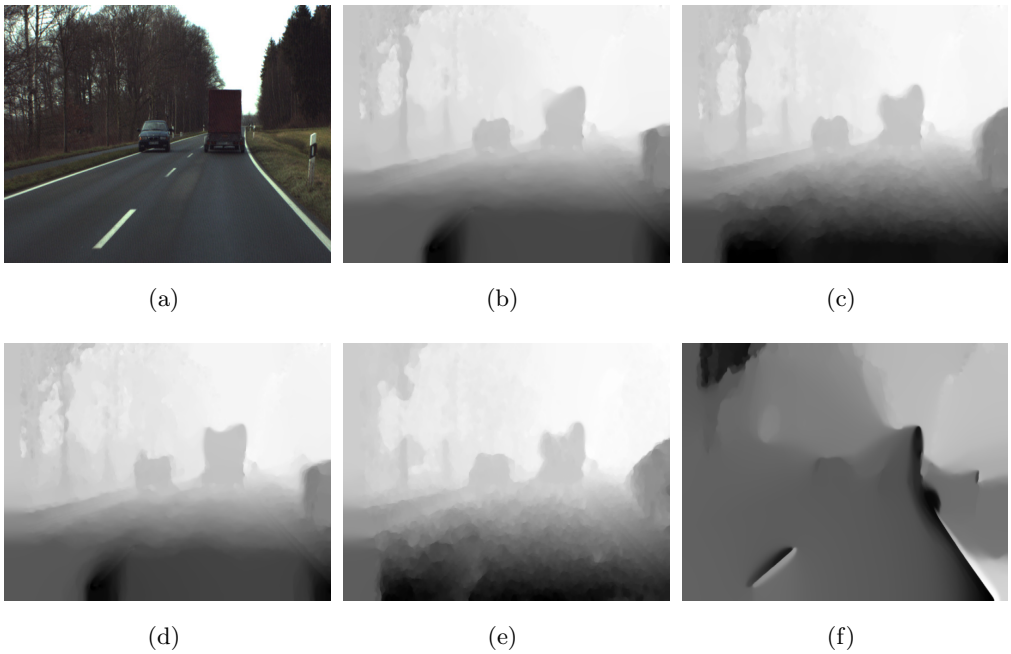
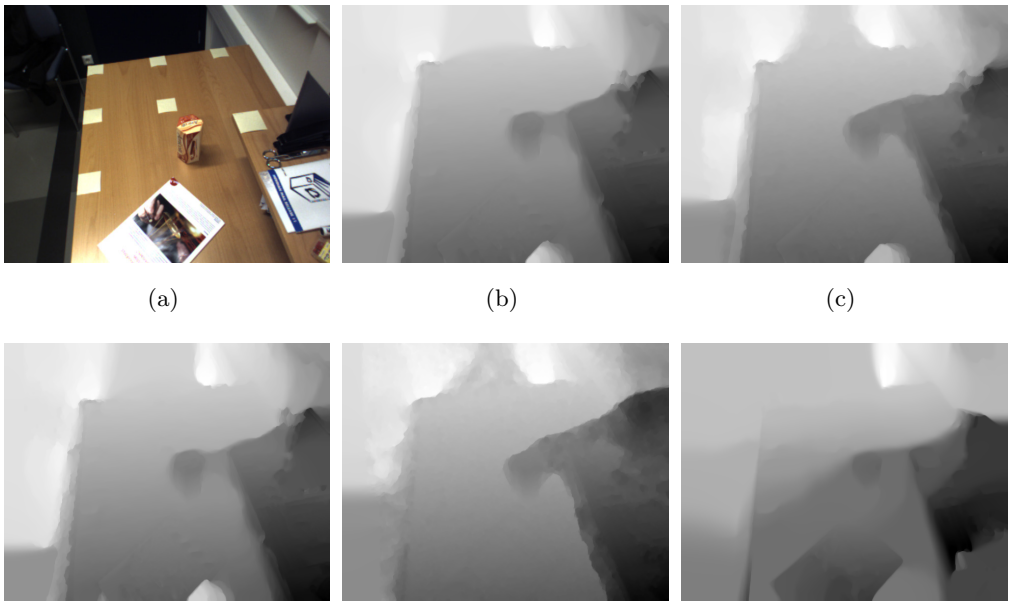


Figure 3.14: DRIVSCO scene. (a) Left image; (b) $\nabla I(RGB)$; (c) $\nabla I(RGB)$+phase; (d) $\nabla I(RGB)$+HS(V); (e) phase; (f) RGB.

Fig. 3.14 shows results for the DRIVSCO sequence. Here $\nabla I(RGB)$+phase has produced the most concise results: results for the road are far better than with any of the other representations. On the other hand, $\nabla I(RGB)$+logd has produced the best results for the trailer: obtaining correct approximations for the trailer is challenging since it tends to 'fuse' with the trees. RGB has produced very low quality results and, for example, scene interpretation based on these would be very challenging if not impossible.



(a) (b) (c)

(d) (e) (f)

Figure 3.15: GRASP scene. (a) Left image; (b) $\nabla I(RGB)$; (c) $\nabla I(RGB)$+phase; (d) $\nabla I(RGB)$+HS(V); (e) phase; (f) RGB.

Fig. 3.15 shows results for a robotic grasping scene. Both $\nabla I(RGB)$ and $\nabla I(RGB)$+HS(V) have produced good results: the object of interest laying on the table is recognisable in the disparity map. $\nabla I(RGB)$+phase or phase alone has increased 'leakage' of disparity values between the object of interest and the shelf. On the other hand, phase representation has produced the best results for the table, especially for the lowest part. has caused 'leakage' of values between the object of interest on the table and the shelf. Again, RGB has produced low quality results.

Altogether, visual qualitative interpretation of the results using real image sequences is in line with the quantitative analysis. Both $\nabla I(RGB)$ and $\nabla I(RGB)$+phase produce good results even with real image sequences. However, the former produces slightly more accurate results while the latter representation is more robust.

3.9.8 Conclusions

We have shown that the quality of a disparity map, generated by a variational method, under illumination changes and image noise, depends significantly on the used image

representation type. By combining different representations, we have generated and tested 34 different cases and found several complementary spaces that are affected only slightly even under severe image distortions. Accuracy differences of 7-fold (without noise) and 10-fold (with noise and illumination errors) were found between the best and worst representation maps, which highlights the relevance of an appropriate input representation for low level estimations such as stereo. This accuracy enhancing and robustness to noise can be of critical importance in specific application scenarios with real uncontrolled scenes and not just well behaving test images (e.g. automatic navigation, advanced robotics, CGI). Amongst the tested combinations, $\nabla I(RGB)$ representation stood out as one of the most accurate and least affected by illuminations errors and/or noise. By combining $\nabla I(RGB)$ with phase, the joined representation space was the most robust one amongst the tested. This finding was also confirmed by the qualitative experiments. Thus, we can say that the aforementioned representations complement each other. These results were also confirmed in a qualitative evaluation of natural scenes in uncontrolled scenarios.

There are some studies similar to ours, carried out in a smaller scale. However, the other studies typically provide little information related to how the optimum (or near optimum) parameters of the algorithm are achieved, related to each representation space: in this work, we have used a well known, derivative free, stochastic algorithm called Differential Evolution (DE) for the reasons given in the text. We argue that manually obtained parameters are subjected to a bias from the human operator and therefore can be expected to to confirm expected results. Three different sets of images were used for obtaining the parameters and testing each of the representations, in order to avoid over-fitting. The proposed methodology for estimating model parameters can be extended to many other computer vision algorithms. Therefore, our contribution should lead to more robust computer vision systems capable of working with real applications.

3.10 Spatial and Temporal Constraints

In this section we propose a new method for optical-flow and stereo estimation based on the inclusion of both spatial and temporal constraints in a variational framework. These constraints bound the solution based on a priori information, or in other words,

based on what is known of a possible solution or how it is expected to change temporally. This knowledge can be something that (a) is known since the geometrical properties of the scene are known or (b) is deduced by a higher-level algorithm capable of inferring this information. This improves robustness and quality of the results, as is shown in this work.

We continue by giving motivation why we think that constraining the solution is important in Section 3.10.1 and discuss related work in Section 3.10.2. After that the extended model including the spatial- and temporal constraints are introduced in Section 3.10.4. Both quantitative and qualitative results of the conducted experiments are given in Section 3.10.9. Finally, the obtained conclusions are discussed in Section 3.10.10.

3.10.1 Motivation

Even though results of the latest correspondence forming methods are impressive (see e.g. Middlebury Computer Vision Pages[1] for stereo and optical-flow), in some cases, the lack of a meaningful structure in the data deteriorates performance, which at worst can render the method useless for the task at hand (e.g. object detection/recognition and/or manipulation). However, in real image sequences something can typically be assumed of the solution: the sky is, relatively speaking, far away from the cameras (disparity zero or near zero); a road is a relatively flat surface; in the case of automatic video surveillance something is known regarding the background; floors, ceilings, walls, and other man made structures tend to be relatively flat surfaces; if the capture rate of the camera(s) is high enough, then movements in the real world translate into small movements between frames in the camera plane and so on. In the above mentioned cases, it would be beneficial if this knowledge of the geometrical and/or temporal setup could be plugged in the variational framework in order to constrain the solution based on what is known. Also, as evidence is accumulated temporally to support a certain solution, the lack of evidence in the immediate future (next frame or so) should not change the solution, unless the data (evidence) points otherwise. This kind of accumulation and propagation of evidence is called temporal coherency and is an important aspect of real applications. On the other hand, outputs produced by higher level vision (interpretation of cues contrasted with world models) are seldom used as a

[1] http://vision.middlebury.edu/

feedback to refine low level estimations. This is paradoxical since it is at this high-level where reasoning based on the extracted low-level cues takes place. For example, in the case of object recognition, the low-level cues are compared with previously generated object models. Therefore, once the object in question has been identified, the high-level model of the object could be used for improving the noisy and sometimes ambiguous low-level cues. This leads to the idea of a signal-symbol loop [57][41][37] where high-, middle- and low-level vision systems could interchange and fuse data: e.g. a hypothesis formed by the higher level method would either be accepted by the low-level algorithm, if it fits the data, or rejected. Such a hypothesis forming and testing cycle would then eventually lead to improved estimates and coherency generated on all the levels.

3.10.2 Related Work and Our Contribution

The idea of using temporal information in optical-flow calculation is certainly not new. Amongst the first works on the use of using both spatial and temporal information as energy terms are those by Black and Anandan [11][10]. In their work, they propose causality of the solution in the form of the temporal term. Roughly speaking, energy based methods incorporating temporal information can be divided into *causal- and batch processing*. In the case of causal processing, a solution calculated at t is propagated forward in time, for example to $t + 1$, and then is used to improve temporal coherence of the solution starting from $t + 1$. On the other hand, in the case of batch processing, the complete sequence of interest is processed at once: in this case, a 3D-regularisation (smoothness) term is needed. Batch type processing methods are less suitable for real-time implementations than the causal type due to their increased demand of processing power. More recent methods incorporating temporal terms are those of Werlberger et al. [81], Weickert and Schnörr [78], and by Salgado and Sánchez [62]. The first, by Werlberger, can be regarded to be causal whereas the latter belong to the batch type. As is mentioned in both [62] and [81], incorporating temporal information raises an additional challenge of modeling the movement between several frames. If the movement is modeled by being symmetrical both forward and backward in time between several frames, this imposes restrictions on the movement: it is expected to be of a constant velocity (acceleration zero). As can be expected, and is shown in [81], models accepting only constant velocities do fine as long as this assumption is not broken and actually perform worse when it does not hold. In our causal model, movement

is modelled by having both the velocity and acceleration components, as in [11], and therefore, the model does not suffer from this shortcoming.

Our contribution. Our work differs from the above mentioned ones in that we use geometrical knowledge of the scene for constraining the disparity solution and incorporate both the spatial- and temporal constraints simultaneously for the optical-flow. We also give an example of a hypothesis forming-validation loop: a hypothesis of a plane is made based on the data after which, iteratively, the hypothesis is verified against the data and used to constrain the solution.

3.10.3 System Scheme

First of all, we describe the system schematically, making it therefore easier to follow the rest of the text. By d we denote disparity and by (u, v) optical-flow (apparent movement of pixels in the camera plane), while subscripts sc and tc denote spatial- and temporal constraints, respectively. Fig. 3.16 shows the use of spatial constraint, d_{sc}, in disparity calculation; while Fig. 3.17 shows the same for optical-flow calculation with the generation of predicted temporal constraints u_p and v_p.

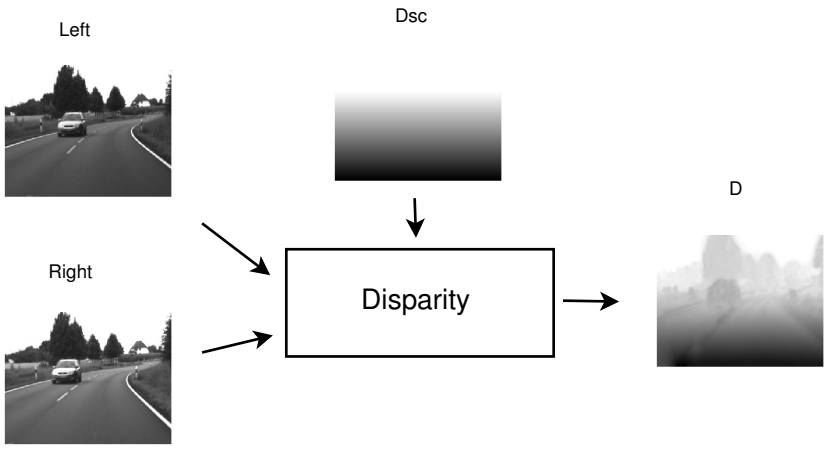


Figure 3.16: Use of spatial constraint in disparity calculation. Dsc (d_{sc} in the text) is the spatial constraint while D is the solution (disparity). This particular case shows how the knowledge of the geometrical setup, in this case, related to the form of the road, of the scene can be used to constrain the solution.

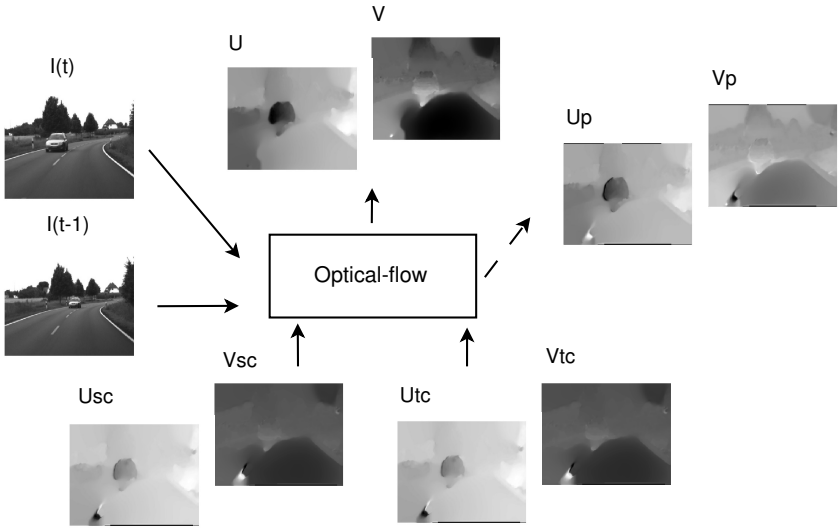


Figure 3.17: Use of spatial and temporal constraints in optical-flow calculation. Usc and Vsc (u_{sc} and v_{sc} in the text) are the spatial constraints while Utc and Vtc (u_{tc} and v_{tc} in the text) are the temporal constraints. Up and Vp (u_p and v_p in the text) is the predicted optical-flow at time $t+1$.

3.10.4 Extended Models

We start by introducing the used notation and then, continue to the energy functionals with the added spatial and temporal constraint terms. Finally, each of the terms is described in more detail.

Table 3.7: Used notation for images and error functions.

DATA TERMS	
$I_{\{L,R\},k,t} = I_{\{L,R\}}(x, y, k, t)$	
$I^w_{\{L,R\},k,t} = I_{\{L,R\}}(x+u, y+v, k, t)$	optical-flow
$I^w_{\{L,R\},k,t} = I_{\{L,R\}}(x+d, y, k, t)$	disparity
ERROR AND CORRESPONDING INFLUENCE FUNCTIONS	
$\Psi_D(s^2) = \sqrt{s^2+\epsilon^2}$	$\Psi'_D(s^2) = 1/\sqrt{s^2+\epsilon^2}$
$\Psi_R(s^2) = \sqrt{s^2+\epsilon^2}$	$\Psi'_R(s^2) = 1/\sqrt{s^2+\epsilon^2}$
$\Psi_{CS}(s^2) = ln(1+s^2/\lambda^2)\lambda^2$	$\Psi'_{CS}(s^2) = 1/(1+s^2/\lambda^2)$
$\Psi_{CT}(s^2) = exp(-s^2/\lambda^2)(-\lambda^2)$	$\Psi'_{CT}(s^2) = exp(-s^2/\lambda^2)$

Table 3.7 has been included for convenience, since three different types of error

functions are used in the model. The use of each error function will be justified later on in the text. In the data terms, $I_{\{L,R\},k,t}$ refers to a k:th channel of left or right image, defined by sub-index L or R, at time t. By channel here we mean channel of a vector valued image, such as RGB. Without k written explicitly, all channels are referred. $I^{w}_{\{L,R\},k,t}$ refers to a warped version of the image [3][15]. In the error function case, sub-indices refer to functionality or, in other words, how the error function in question is used in the model. Thus, D, R, CS, and CT refer to data, regularisation, constraint-spatial, and constraint-temporal, respectively. The functionals for stereo and optical-flow are given in (3.29) and (3.30), respectively:

$$\begin{aligned} E(d) = &\int_{\Omega} \Big(D(I_{L,1}, I_{R,1}, d) + \alpha S(\nabla I_{L,1}, \nabla d) \Big) \mathbf{dx} \\ &+ \gamma_s \int_{\Omega} \Big(C_s(d_{sc}, d) \Big) \mathbf{dx} \end{aligned} \tag{3.29}$$

$$\begin{aligned} E(u,v) = &\int_{\Omega} \Big(D(I_{L,1}, I_{L,0}, u, v) + \alpha S(\nabla I_{L,1}, \nabla u, \nabla v) \Big) \mathbf{dx} \\ &+ \gamma_s \int_{\Omega} \Big(C_s(u_{sc}, v_{sc}, u, v) \Big) \mathbf{dx} \\ &+ \gamma_t \int_{\Omega} \Big(C_t(u_{tc}, v_{tc}, u, v) \Big) \mathbf{dx} \end{aligned} \tag{3.30}$$

where the data terms for stereo and optical-flow are $D(I_{L,1}, I_{R,1}, d)$ and $D(I_{L,1}, I_{L,0}, u, v)$, while $S(\nabla I_{L,1}, \nabla d)$ and $S(\nabla I_{L,1}, \nabla u, \nabla v)$ are the regularisation terms, respectively, and $\alpha > 0$ is the weight of the smoothness term. The spatial constraint for stereo is $C_s(d_{sc}, d)$, where d_{sc} is the constraining value. $C_s(u_{sc}, v_{sc}, u, v)$ is the spatial constraint for optical-flow where u_{sc} and v_{sc} are the constraints arising from geometry.
$C_t(u_{tc}, v_{tc}, u, v)$ is the temporal constraint for optical-flow, where u_{tc} and v_{tc} are the predicted values. $\gamma_s > 0$ and $\gamma_t > 0$ are the spatial- and temporal constraints' weights, defining the influence of the new terms.

In Appendix A we give the Euler-Lagrange equations related to optical-flow with a temporal constraint. Derivation of the spatial constraint is similar.

3.10.5 Data Terms

For completeness' sake, the data- and smoothness terms are introduced here again. We have chosen to use a combination of a gradient and a gradient magnitude, since these are

capable of producing reliable results under both (a) illumination errors and (b) image noise [59]. As it was mentioned in Section 3.9.5.1, we used a 5-fold cross-correlation when searching for the most robust representation. In the tests a combination of gradient and gradient magnitude came first once and third and fourth in two other experiments. However, it failed twice, thus decreasing its overall ranking. In any case, it produces good results with the real scenes used here. As can be observed from (3.31) and (3.32), late linearisation of the data terms is used [3][50].

$$\begin{aligned}
D(I_{L,1}, I_{R,1}, d) =& b_1 \sum_{k=1}^{K} \Psi_D\Big((\frac{\partial I_{L,1,k}}{\partial x} - \frac{\partial I^w_{R,1,k}}{\partial x})^2 \Big) + b_1 \sum_{k=1}^{K} \Psi_D\Big((\frac{\partial I_{L,1,k}}{\partial y} - \frac{\partial I^w_{R,1,k}}{\partial y})^2 \Big) \\
&+ b_2 \sum_{k=1}^{K} \Psi_D\Big(|\nabla I_{L,1,k} - \nabla I^w_{R,1,k}|^2 \Big) \\
=& b_1 \sum_{k=1}^{K} \Psi_D\Big((\frac{\partial I_{L,1,k}}{\partial x} - \frac{\partial I^w_{R,1,k}}{\partial x})^2 \Big) + b_1 \sum_{k=1}^{K} \Psi_D\Big((\frac{\partial I_{L,1,k}}{\partial y} - \frac{\partial I^w_{R,1,k}}{\partial y})^2 \Big) \\
&+ b_2 \sum_{k=1}^{K} \Psi_D\Big((\frac{\partial I_{L,1,k}}{\partial x} - \frac{\partial I^w_{R,1,k}}{\partial x})^2 + (\frac{\partial I_{L,1,k}}{\partial y} - \frac{\partial I^w_{R,1,k}}{\partial y})^2 \Big)
\end{aligned} \tag{3.31}$$

$$\begin{aligned}
D(I_{L,1}, I_{L,0}, u, v) =& b_1 \sum_{k=1}^{K} \Psi_D\Big((\frac{\partial I_{L,1,k}}{\partial x} - \frac{\partial I^w_{L,0,k}}{\partial x})^2 \Big) + b_1 \sum_{k=1}^{K} \Psi_D\Big((\frac{\partial I_{L,1,k}}{\partial y} - \frac{\partial I^w_{L,0,k}}{\partial y})^2 \Big) \\
&+ b_2 \sum_{k=1}^{K} \Psi_D\Big(|\nabla I_{L,1,k} - \nabla I^w_{L,0,k}|^2 \Big) \\
=& b_1 \sum_{k=1}^{K} \Psi_D\Big((\frac{\partial I_{L,1,k}}{\partial x} - \frac{\partial I^w_{L,0,k}}{\partial x})^2 \Big) + b_1 \sum_{k=1}^{K} \Psi_D\Big((\frac{\partial I_{L,1,k}}{\partial y} - \frac{\partial I^w_{L,0,k}}{\partial y})^2 \Big) \\
&+ b_2 \sum_{k=1}^{K} \Psi_D\Big((\frac{\partial I_{L,1,k}}{\partial x} - \frac{\partial I^w_{L,0,k}}{\partial x})^2 + (\frac{\partial I_{L,1,k}}{\partial y} - \frac{\partial I^w_{L,0,k}}{\partial y})^2 \Big)
\end{aligned} \tag{3.32}$$

where the spatial gradient operator is given by $\nabla := (\partial_x, \partial_y)^T$ and $b_1 > 0$ and $b_2 > 0$ are the weights of the data terms. The benefit of non-linearised constancy terms (or late linearisation) is that the model copes better with large displacements, especially when used together with multi-resolution strategy where the solution is propagated from coarse to finer levels together with warping. Another benefit is that the full range of

information available in the images is used. Instead of using quadratic error function, we use $\Psi_D(s^2) = \sqrt{s^2 + \epsilon^2}$ [15][13], which is applied individually to each channel in each term [87]. ϵ is used for stabilisation [1] where s^2 is near zero. This kind of a robust error function gives less importance to outliers in the data, such as occlusions and image noise.

3.10.6 Regularization Terms

For regularisation, we have used both image- and flow-driven isotropic-regularisations and a combination (mixed regularisation) of the aforementioned. In the case of the mixed regularisation, we propose swapping between image- and flow-driven regularisations. For example, on every fourth iteration, we use image-driven regularisation instead of flow-driven one. Regularisation based solely on image information can prevent smoothening within objects since not all image borders necessarily adjust with object borders. On the other hand, where image borders indeed do coincide with object borders, such information can prevent over smoothening across objects. Mixed regularisation approach is both computationally attractive, since image-driven regularisation is linear and thus, the diffusion weights need to be calculated only once per image, and it also combines information of both the image- and the flow-field. There are more complex and elaborated ways of combining both the image- and flow information in order to achieve excellent results [87], and we expect that using such regularisation terms would further improve the obtained results. Regularisation terms for stereo- and optical-flow are given in (3.33) and (3.34), respectively.

$$S(\nabla I_{L,1}, \nabla d)\Big) = \begin{cases} g(|\nabla I_{L,1}|^2)(|\nabla d|^2) & \text{if image driven,} \\ \Psi_R(|\nabla d|^2) & \text{if flow driven} \end{cases} \tag{3.33}$$

$$S(\nabla I_{L,1}, \nabla u, \nabla v) = \begin{cases} g(|\nabla I_{L,1}|^2)(|\nabla u|^2 + |\nabla v|^2) & \text{if image driven,} \\ \Psi_R(|\nabla u|^2 + |\nabla v|^2) & \text{if flow driven} \end{cases} \tag{3.34}$$

where the error functions are $\Psi_R(s^2) = \sqrt{s^2 + \epsilon^2}$ as in [13] and $g(s^2) = 1/(1 + s^2/\lambda^2)$ [2][53]. The purpose of the error functions is to prevent the regularisation term from smoothening across object boundaries and thus, to make the solution piece-wise smooth. In the $g(s^2)$ case, λ is a parameter indicating which 'strength' of the edges of the image are regarded important and thus, it controls diffusion strength. Again, ϵ is used for stabilisation where s^2 is near zero.

3.10.7 Spatial- and Temporal Constraints

The last things that need to be defined are the spatial- and temporal constraint terms, for both stereo and optical-flow, which are given in (3.35), (3.36), and (3.37).

$$C_s(d_{sc}, d) = \Psi_{CS}\big((d_{sc} - d)^2\big) \tag{3.35}$$

$$C_s(u_{sc}, v_{sc}, u, v) = \Psi_{CS}\big((u_{sc} - u)^2\big) + \Psi_{CS}\big((v_{sc} - v)^2\big) \tag{3.36}$$

$$C_t(u_{tc}, v_{tc}, u, v) = \Psi_{CT}\big((u_{tc} - u)^2\big) + \Psi_{CT}\big((v_{tc} - v)^2\big) \tag{3.37}$$

where $\Psi_{CS}(s^2) = ln(1+\frac{s^2}{\lambda^2})\lambda^2$ and $\Psi_{CT}(s^2) = exp(-\frac{s^2}{\lambda^2})(-\lambda^2)$ are robust non-quadratic error functions. u_{tc} and v_{tc} are the temporal constraints, whereas d_{sc}, u_{sc}, and v_{sc} are the spatial constraints. The constraints function as priors, therefore, in a sense guiding the solution towards the constraint. λ is a parameter, that depends on the image scale, used for determining the shape of the influence function: where the constraint does not fit the data, its influence upon the solution is rejected. Influence functions of the corresponding error functions are displayed graphically in Fig. 3.18.

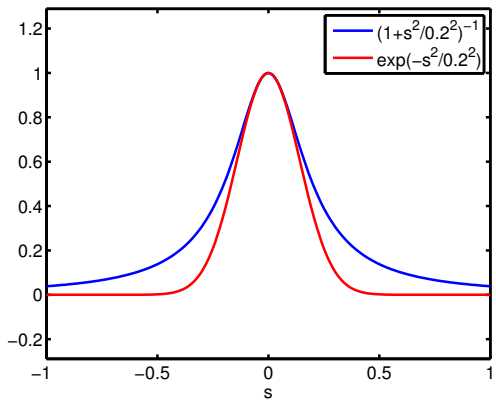


Figure 3.18: Influence functions $\Psi'_{CS}(s^2) = 1/(1 + \frac{s^2}{\lambda^2})$ and $\Psi'_{CT}(s^2) = exp(-\frac{s^2}{\lambda^2})$ for λ of 0.2.

As can be observed from Fig. 3.18, the influence function based on the exponential error function approaches zero faster than the influence function based on the logarithmic error function. For temporal constraints, we have chosen to use the exponential error function, since we expect that the temporal displacements are small and, therefore, steeper influence functions are preferred. Thus, if there is no proper temporal

continuum (e.g. a new object enters in the image) the temporal constraint should be rejected and the solution should be based solely on the data and smoothness terms. On the other hand, the expected dynamic range of the disparities is higher and, therefore, in this case, it is beneficial that the influence function has a longer non-zero tail. The shape of the function is controlled by λ. This is clearly beneficial, since the displacement depends on the scale (multi-resolution processing) and thus, the shape of the function can be adapted as per scale.

Now that the new terms have been introduced, the question from where do we obtain the actual constraints remains. Partially we answered this question in Section 3.10.1. Spatial constraints reflect our knowledge related to the spatial properties of the scene setup. In [57] Ralli et al. show how this kind of information related to the scene setup can be deduced from the initial disparity map. In the case of optical-flow, the spatial constraint can be embedded, for example, in the fundamental matrix F [25][30] and it has been studied in [77][76]. On the other hand, the temporal constraint is used for applying temporal coherency upon the solution and thus, reflects knowledge related to how the observed scene is expected to change.

3.10.8 Predicted Temporal Constraints

For the temporal constraints, we need approximations of $u(x, y, t)$ and $v(x, y, t)$ at $t+1$. At first sight this seems simple enough, but actually, the problem is two folded. We already know the apparent movement in the camera plane at time t, since this is exactly what we have calculated and is expressed by (u, v). Therefore, we know how each of the pixels move between times t and $t + 1$, but what we do not know is what is the actual value of u and v at $t + 1$. In other words, we have to know how the optical-flow changes temporally. This can be expressed, for example, using Taylor series as given in (3.38).

$$\begin{aligned} u_p &= u(x, y, t + 1) \\ &= u(x, y, t) + \frac{\partial u(x, y, t)}{\partial t} + \sum_{n=2}^{\infty} \frac{u^{(n)}(x, y, t)}{n!} \\ v_p &= v(x, y, t + 1) \\ &= v(x, y, t) + \frac{\partial v(x, y, t)}{\partial t} + \sum_{n=2}^{\infty} \frac{v^{(n)}(x, y, t)}{n!} \end{aligned} \tag{3.38}$$

where superscript (n) stands for the n:th derivative of the function, and u_p and v_p are the predicted values. We have used an approximation up to the first order (discarding higher order terms), where the first order is approximated from the current and last approximation: for example $\frac{\partial u(x,y,t)}{\partial t} \approx u(x,y,t) - u(x,y,t-1)$. Physical interpretation of these terms ($\partial u(x,y,t)/\partial t$ and $\partial v(x,y,t)/\partial t$) is the acceleration of the optical-flow. In order to account for the current movement, as explained earlier, each of the predicted terms needs to be warped as expressed in Equation (3.39).

$$\begin{aligned} u_{tc} &= u_p(x-u, y-v) \\ v_{tc} &= v_p(x-u, y-v) \end{aligned} \tag{3.39}$$

where u_{tc} and v_{tc} are the actual temporal constraints.

3.10.9 Experiments

We have evaluated effects of the spatial and temporal constraints, both quantitatively and qualitatively, using known test images from the Middlebury[1] database [32][6] and images from the DRIVSCO[2] and GRASP[3] projects. Quantitative results justify the model, but in fact, we are more interested in knowing how the model behaves with real images.

3.10.9.1 Error Metrics

Formulae of the used error metrics are given in Equations (3.40), (3.41), and (3.42).

$$MAE := \frac{1}{n}\sum_{i=1}^{n} abs\Big((d)_i - (d_{gt})_i\Big) \tag{3.40}$$

$$C := \frac{1}{n}\#\Big\{i|abs\Big((d)_i - (d_{gt})_i\Big) \leq 1\Big\} \tag{3.41}$$

$$AAE := \frac{1}{n}\sum_{i=1}^{n} \frac{(u_{gt})_i u_i + (v_{gt})_i v_i + 1}{\sqrt{\Big((u_{gt})_i^2 + (v_{gt})_i^2 + 1\Big)\Big((u)_i^2 + (v)_i^2 + 1\Big)}} \tag{3.42}$$

[1] http://vision.middlebury.edu/
[2] http://www.pspc.dibe.unige.it/∼drivsco/
[3] http://www.csc.kth.se/grasp/

where n is the number of pixels, d_{gt} u_{gt}, and v_{gt} are the ground truths of the disparity and optical-flow maps. MAE is the mean average error, C is the percentage of disparities with absolute error of 1 or smaller, and AAE is the average angular error [7].

3.10.9.2 Quantitative Results for Spatial Constraint in Disparity Calculation

The aim of this experiment is to demonstrate how spatial knowledge of the scenery, especially in difficult cases where disparity calculation normally tends to fail, can be used to enhance the results. The reason for not testing with the standard set of Middlebury images (Tsukuba, Venus, Teddy, and Cones) is that we consider those fairly 'simple', with more or less planar surfaces containing texture, and therefore, these do not reflect the challenges related to real applications. Fig. 3.19 shows the used stereo-images. Parameters are the same through all the experiments.

(a) (b) (c)

Figure 3.19: (a) Monopoly; (b) Midd1; (c) Midd2.

It can be seen that in the case of the chosen test images (Fig. 3.19), the backgrounds do not contain clearly visible structure that could be used for establishing correspondences. Therefore, it is expected that any method generating correspondences based on image structures (such as edges, corners, or differentiable pixels) will fail for most of the background. The idea of using spatial constraint, therefore, is to use a priori knowledge of the solution in order to improve the results in such cases. Fig. 3.20 shows the results with and without the use of a spatial constraint. The spatial constraint used in this case is taken from the ground-truth and is the background. Results corresponding to Fig. 3.20 are given numerically in Table 3.8.

Table 3.8: Results in MAE (mean average error) and percentage of correct disparities using different regularisations without (WO) and with (W) a spatial constraint.

	MAE					
	WO spatial constraint			W spatial constraint		
	flow	image	mixed	flow	image	mixed
Monopoly	8.3	7.5	8.0	4.9	2.4	2.1
Midd1	6.2	5.1	5.6	1.6	1.8	1.6
Midd2	6.3	5.5	5.9	1.9	2.3	1.9
	Percentage of correct disparities					
	WO spatial constraint			W spatial constraint		
	flow	image	mixed	flow	image	mixed
Monopoly	66.3%	64.2%	66.3%	61.1%	76.9%	83.4%
Midd1	47.0%	46.9%	47.2%	81.9%	79.3%	82.3%
Midd2	48.4%	46.5%	48.3%	73.1%	72.5%	79.3%

In Fig. 3.20, the first row displays the ground-truths, the second row the results with flow-driven regularisation, the third row shows the used spatial constraints for generating the results seen on the fourth row (with mixed-regularisation and spatial constraint). As can be observed, the results improve significantly, both visually and numerically, using the spatial constraint. It can also be seen from Table 3.8 that by mixing both image- and flow-field information (mixed regularisation) the results have further improved.

It can be argued that the given example (and therefore, the results) are artificial since the spatial constraints were obtained from the known ground-truths. However, from the point of view of object detection, scene interpretation, or segmentation based of disparity maps, it is not necessary for the used spatial constraint to be correct (or even near correct). From the object detection and segmentation (based on disparity) point of view, the main problem, in this case, is that the objects and the background tend to 'fuse' together. Even if we did not know the correct value for the backgrounds, we could still make a guess, in order to improve object separability from the background. The guess that we have made here is that the background is far away (zero disparity) from the cameras. As can be observed from Fig. 3.21, such as a guess has also improved object separability notably: we can still separate the background clearly from the

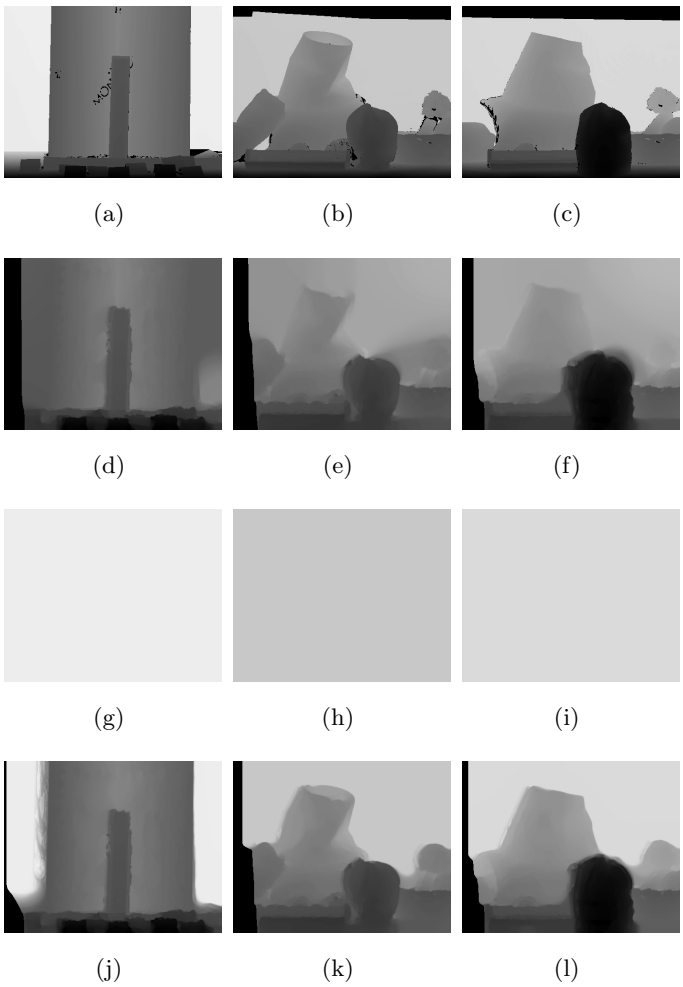
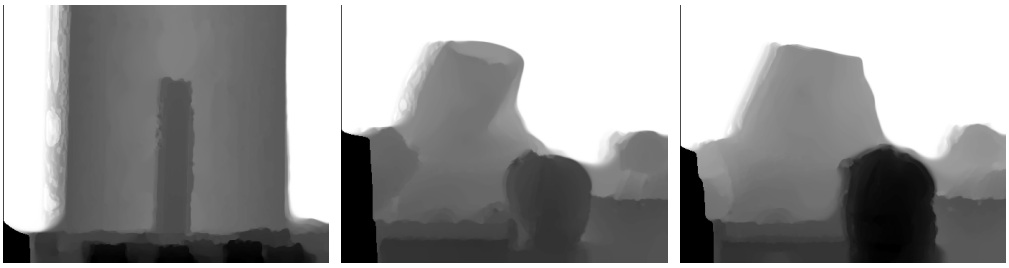


Figure 3.20: (a) Monopoly ground-truth; (b) Midd1 ground-truth; (c) Midd2 ground-truth; (d) Monopoly, mixed-diffusion without SC (e) Midd1, mixed-diffusion without SC; (f) Midd2, mixed-diffusion without SC; (g) Monopoly SC; (h) Midd1 SC; (i) Midd2 SC; (j) Monopoly, mixed-diffusion with SC; (k) Midd1, mixed-diffusion with SC; (l) Midd2, mixed-diffusion with SC. SC stands for spatial constraint.

foreground even if the used spatial constraint is not correct.

3.10.9.3 Quantitative Results for Temporal Constraint in Optical-flow Calculation

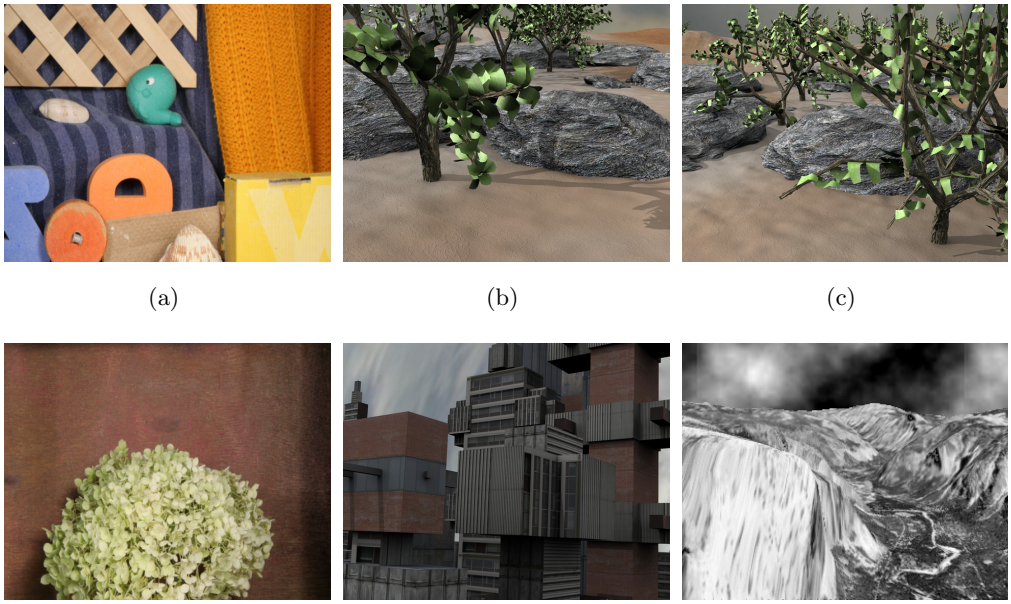
Here, we study the effects of the temporal constraint upon optical-flow calculation. Fig. 3.22 shows the used image sequences while Table 3.9 displays the results. The



(a) (b) (c)

Figure 3.21: (a) Monopoly; (b) Midd1; (c) Midd2. In each case the spatial constraint is 0 and mixed regularisation is used.

parameters are the same through all the experiments.



(a) (b) (c)

(d) (e) (f)

Figure 3.22: Sequences (a) Rubberwhale; (b) Grove2; (c) Grove3; (d) Hydrangea; (e) Urban3; (f) Yosemite (with clouds).

In the case of (a) Rubberwhale (b) Grove2 (c) Grove3 (d) Hydrangea and (e) Urban3 frames 8 to 10 are used for calculating the optical-flow whereas in the (g) Yosemite frames 5 to 8 are used. P.ERR (prediction error) is the angular error of the temporal constraint (approximated from previous solutions).

As can be observed from the Table 3.9 using temporal constraint has reduced the

Table 3.9: Results in AAE with and without temporal constraint. W.TEM, WO.TEM and P.ERR denote 'with temporal constraint', 'without temporal constraint' and 'prediction error' correspondingly.

	AAE		
	W.TEM	WO.TEM	P.ERR
Rubberwhale	4.9°	4.9°	8.4°
Grove 2	2.8°	3.0°	4.6°
Grove 3	6.6°	7.1°	9.9°
Hydrangea	2.6°	2.7°	9.6°
Urban 3	5.2°	5.4°	7.0°
Yosemite	3.2°	4.0°	5.0°

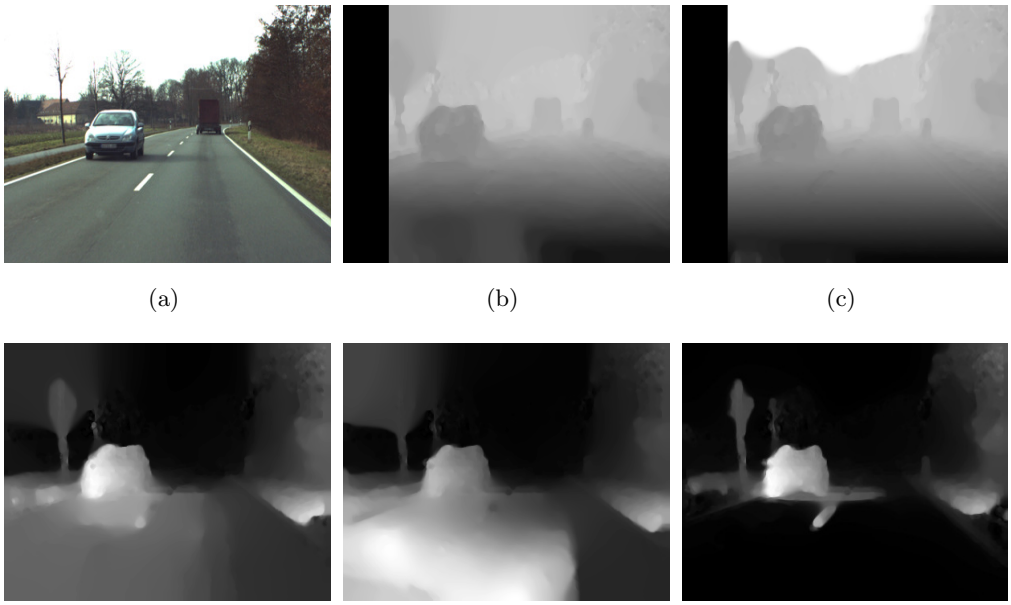
error in all the cases, apart from the Rubberwhale sequence, where the error has remained the same. Yosemite sequence has benefited the most. This is not surprising, since in this case the movement is somewhat smooth which results in a lower prediction error. Also it can be noted that even in the case where the predicted error is high, like in the Hydrangea sequence, the robust non-quadratic error function, Equation (3.37), quickly suppresses those predicted values that do not fit the data, and therefore, even these cases can slightly benefit from the temporal coherency. In real applications, however, typically high frame rates are favoured, resulting in a smaller movement between frames and thus, smaller prediction errors can be expected.

3.10.9.4 Qualitative results for spatio-temporal constraints

Here we give results for both disparity and optical-flow using a real stereo-image sequence from the DRIVSCO project and for a stereo-image pair from a video-surveillance application. Complete videos of the results for DRIVSCO are available at [1]. In the disparity case results are given for (a) no spatial constraint and (b) spatial constraint. For DRIVSCO the spatial constraint is based on the road and the sky, while in the video-surveillance case the constraint is based only on the floor. In the optical-flow case results are given for (a) no spatial- or temporal constraint (b) temporal constraint and (c) both spatial and temporal constraints. In the optical-flow case a spatial constraint of zero was used for both u_{sc} and v_{sc}. The rationale for testing with a spatial constraint

[1] http://atc.ugr.es/~jarnor/index.php/publications/more-information

of zero was to study how such a constraint affects 'flickering' of the estimations in a real sequence where no sufficient spatio-temporal structure is always present. In the following we present images from the complete processed sequence. Complete video of the results is available at [1]



(a) (b) (c)

(d) (e) (f)

Figure 3.23: Frame 670. (a) Left image; (b) disparity without spatial constraint; (c) disparity with spatial constraint; (d) velocity; (e) velocity with temporal constraint; (f) velocity with temporal and spatial constraint. Only the optical-flow module is given in the figure.

Several observations can be made from Fig. 3.23, although the observations are clearer from the complete processed sequence. In the disparity case, without the spatial constraint, the road has a 'wavy' form that changes constantly and the background (sky) fuses with the foreground (forest), since the sky has no spatial information usable for establishing correspondences. This can make scene interpretation based on disparity or 3D-reconstructed points very difficult or even impossible [58]. In the optical-flow case, using a temporal constraint reduces 'flickering' of the estimations and also makes these more concise for the lower part of the road. By flickering, we mean erroneous temporal changes in the solution. The problem with the road is that it contains very

[1] `http://atc.ugr.es/$\sim$jarnor/index.php/publications/more-information`

few spatial features. However, the middle lane marker can be used for calculating good approximations and propagating this information forward in time is beneficial, especially when the lane marker is not present. Using a spatial constraint of zero constrains the solution, as expected, to zero where no sufficient features are present (even though movement would be present). In this case, the results are somewhat similar to those from sparse methods: the values on the road assigned to zero movement can be understood as 'non-valid' estimations (any other value apart from zero can be used as well). In this way, the proposed method automatically provides a way to detect unreliable solutions. Depending on the task at hand, this can be beneficial: high quality movement estimations are available for those areas with enough spatio-temporal features and these can easily be identified from the non-zero movement estimations.

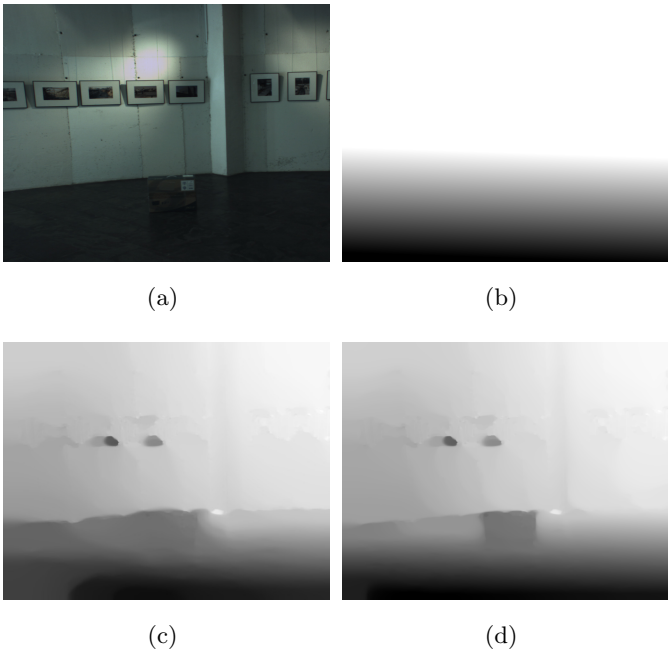


Figure 3.24: Video-surveillance application. (a) Left image; (b) constraint; (c) disparity without spatial constraint; (d) disparity with spatial constraint. It is difficult to spot the suitcase, even for a human observer, from the left image or the disparity map without constraint. On the other hand, by using a constraint for the floor, the results for disparity improve so that detecting the object of interest becomes considerably easier.

Fig. 3.24 displays results for a video-surveillance application. In this case, the object of interest has been intentionally camouflaged and is difficult to spot, even for a human

observer, by taking a quick look at the image. Since the floor does not contain enough spatial features, disparity calculation without a spatial constraint understandably fails here and, therefore, the foreground fuses with the background. The spatial constraint used here is a plane for the floor that covers the whole image space. In Fig. 3.24 (b) the constraint is scaled so that the grey level values correspond to those with the resulting disparity map. True disparity range in this case is approximately $[-40..-10]$, while the constraint contains values between $[-42..33]$ (in Fig. 3.24 (b) values greater than -10 become saturated and are therefore white). This clearly shows that where the constraint does not agree with the data, it is discarded.

3.10.10 Conclusions

Here we have proposed a new method for optical-flow and stereo estimation based on the inclusion of both spatial and temporal constraints in a variational framework. It was shown that by using such constraints significant improvements can be obtained. We have illustrated this with several real-world examples and with standard benchmark sequences. For example in the case of disparity calculation, we showed that considerable improvements are achievable using spatial information. This is true especially for cases containing surfaces with very little spatial information. On the other hand, if enough spatial information is available then the improvements will be less significant. The obtained results indicate that with the evaluated constraints (spatial and temporal terms) the results improve to a point where tasks such as object recognition and grasping, fore- and background segmentation based upon the disparity and/or optical-flow become easier for higher level vision stages.

3.11 Summary

In this chapter we have introduced early- and late linearisation models for calculating optical-flow fields and a late linearisation model for calculating stereo-disparity. We have shown that proper image representation is crucial when trying to establish correct image correspondences in real scenes under realistic illumination conditions. We have shown how a priori information, spatial or temporal, can be used to constraint the solution. Also, in the disparity case, we have shown how such spatial constraints can be generated automatically.

4

Segmentation of Disparity Maps

4.1 Introduction

Typically segmentation methods aim at grouping image pixels into *meaningful objects*, i.e. the segments describe objects more or less in the same sense as we humans perceive these. Naturally, notation of an object can change accordingly to the problem at a hand. Typical features, amongst others, used for grouping the pixels into segments are colour, texture- and shape descriptors. We would like to point out that segmentation per se is not the final objective in our case, but to be able to generate automatically meaningful constraints from the segments and use these in disparity and/or optical-flow calculation as introduced in Section 3.10. Therefore, we aim at segmenting the disparity maps into *meaningful surfaces* that can be used as constraints. To summarise, in this chapter we introduce segmentation based on level-sets theory and show how disparity maps can be segmented into meaningful surfaces.

4.2 Motivation for Level-sets

In this part, which is adapted from the book *Level Set Methods and Dynamic Implicit Surfaces* [51], we give motivation for the level-set based methods. This way a non-expert reader should find it easier to read and understand the part that actually deals with the segmentation model.

4.3 Implicit Surfaces

In two spatial dimensions the *interface* $\partial\Gamma$ is defined as an isocontour[1] of the implicit function $\Phi(x, y)$ and is mathematically given by $\partial\Gamma = \{(x, y)|\Phi(x, y) = 0\}$. There is nothing special about the zero isocontour and any other isocontour could have been chosen instead without affecting the properties of the implicit function. The interface separates *inside* and *outside* regions that are defined as in (4.1).

$$\begin{cases} \Gamma \equiv \partial\Gamma = \{(x, y) \,|\, \Phi(x, y) = 0\} \\ inside(\Gamma) \equiv \Omega_1 = \{(x, y) \,|\, \Phi(x, y) \geq 0\} \\ outside(\Gamma) \equiv \Omega_2 = \{(x, y) \,|\, \Phi(x, y) < 0\} \end{cases} \tag{4.1}$$

In Fig. 4.1 both an implicit function $\Phi(x, y)$ and a zero plane cutting the function are shown. That part of the function that is above the zero level defines the inside region while the part that is below the zero level defines the outside region.

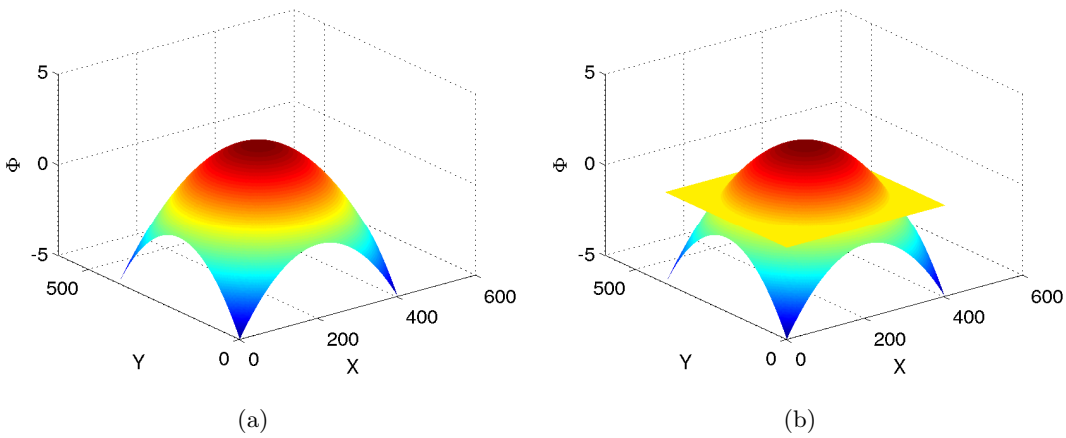


Figure 4.1: (a) Graph of an implicit function $\Phi(x, y)$; (b) Graph of an implicit function $\Phi(x, y)$ with 'zero' plain (yellow) plane cutting it at $z = 0$.

Fig. 4.2 displays the corresponding zero isocontour (green line) and the inside- and outside regions.

As it can be understood from the definitions given in (4.13), the interface is given implicitly, i.e. it is an isocontour of the function $\Phi(x, y)$, and has one dimension less than the implicit function. If the implicit function has $\Re^n$ dimensions, then the interface is $\Re^{n-1}$ dimensional. Although at first it might seem wasteful to encode the interface

[1] isocontour is a curve along which the function has a constant value

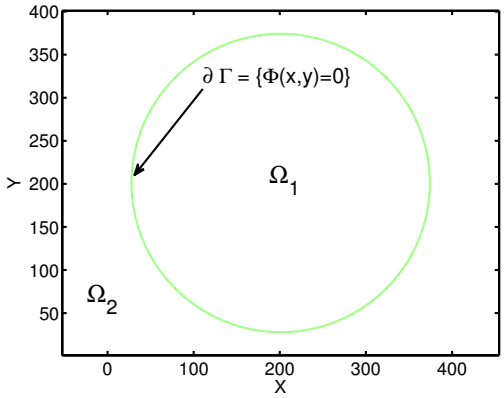


Figure 4.2: Zero isocontour (green) of the implicit function with inside- (Ω_1) and outsize (Ω_2) regions.

as a higher dimensional object, the implicit formulation has several advantages over the explicit formulation, which will be discussed next. In the explicit formulation the points defining the interface need to be stored 'explicitly'. In general, the curve is parametrised by a vector function $\vec{x}(s) = (x(s), y(s))$, where s is defined in $[s_s, s_f]$. Since we are only interested in closed curves, this implies $\vec{x}(s_s) = \vec{x}(s_f)$. In the explicit formulation, determining if a given point belongs to the inside- or the outside region is not that straight forward, since only information of the curve itself is stored. In practise, this can be achieved, for example, by drawing a line between the point of interest and some point that is known to lie in the outside region and count how many times the line intersects with the curve (interface): odd number means that it is inside the curve, i.e. it belongs to the inside region, while an even number means that it is outside the curve and thus belongs to the outside region. In the implicit formulation only sign of the $\Phi(x, y)$ needs to be checked. Topological changes are handled implicitly in the level-set formulation: if a segment breaks into two or two (or more) segments fuse together, the encoding of the interface(s) remain the same. In the explicit formulation fusing or breaking of the segments must be handled by re-parametrisation of the curve(s). While in two dimensional case this is relatively easy to do, in three or more dimensions this becomes increasingly difficult to handle and one has to deal with issues such as connectivity and so on. One more issue that we would like to bring up is discretisation. Direct, and the most common way, of approximating an explicit representation is to discretise the s, or in other words, to represent it by a set $s_s < ... < s_{i-1} < s_i < s_{i+1} < ... < s_f$. The

intervals $[s_{i-1}, s_i]$ are not necessarily of equal size: representing 'kinks' or places with high curvature requires more sampling points. Since we are interested in interfaces that change over time, i.e. *front tracking* [75], the curvature is not constant but changes and, thus, this means that the discretisation might change over time. Also, if the topology of the interface changes, this might affect the discretisation as well. If the discretisation is not addressed properly, so that both smoothness and regularity of the interface are maintained, numerical results can deteriorate to a level where they are of no use at all [85][51]. In the case of the implicit representation, the function Φ, in $\Re^2$, is discretised by sampling at discrete points (x_i, y_i) where $i = 1..N$. This set of data points is also called a grid. By far the most used grid is a *Cartesian grid* $\{(x_i, y_j)|1 \leq i \leq m, 1 \leq j \leq n\}$. If the sub-intervals $[x_i, x_{i+1}]$ and $[y_i, y_{i+1}]$ are equal in size, then it is a *homogeneous* Cartesian grid. In the case of dynamic implicit interfaces we do not track explicitly the interface, but evolve the implicit function temporally. Therefore, from numerical point of view, it is enough if the function Φ is smooth and regular. One way of assuring this is setting Φ to a *signed distance function*[51].

4.3.1 Dynamic Implicit Surfaces

As it was already mentioned, interfaces that evolve with respect to time are of particular interest. Suppose that the speed $\vec{V}(\vec{x})$ is known for all the points on the interface $\vec{x}$ such that $\Phi(\vec{x}) = 0$ (any other isocontour apart from 0 could have been chosen). Given the velocity field $V = (u, v)$ we wish to know how the interface evolves temporally. In *Lagrangian formulation* the system can be described as given in (4.2).

$$\frac{d\vec{x}}{dt} = \vec{V}(\vec{x}) \tag{4.2}$$

which is an ordinary differential equation (ODE). However, we want to avoid the problems mentioned in Section 4.3 related to the explicit interfaces and use different formulation. By using *Eulerian formulation* we can use the implicit function Φ for both encoding the interface and evolving it with a *convection equation* (4.3).

$$\begin{aligned} &\frac{\partial \Phi}{\partial t} + \vec{V} \cdot \nabla \Phi = 0 \\ &\frac{\partial \Phi}{\partial t} + u\frac{\partial \Phi}{\partial x} + v\frac{\partial \Phi}{\partial y} = 0 \end{aligned} \tag{4.3}$$

which is a partial differential equation (PDE) and it describes how the interface $\Phi(\vec{x}) = 0$ evolves temporally in a general velocity field $\vec{V}$. Equation (4.3) is also known as the *Level-set Equation* and it was first introduced by Osher and Sethian in their landmark paper [52].

4.3.2 Mean Curvature Motion

The equation (4.3) describes how the implicit interface evolves in a general, possibly external, velocity field $\vec{V}$. Here we describe a different kind of a velocity that is induced by the curvature of the interface itself and, as will be shown later, this is of particular interest from segmentation point of view. The motion induced by the curvature is known as *Mean Curvature Motion* (or MCM for short) which can be formulated entirely in terms of the implicit function Φ, as is shown next. Suppose that the velocity field is described by components tangential and normal to the interface and, therefore, we have $\vec{V} = V_n\vec{N} + V_t\vec{T}$ and therefore the convection equation (4.3) can be written as given in (4.4).

$$\frac{\partial \Phi}{\partial t} + (V_n\vec{N} + V_t\vec{T}) \cdot \nabla\Phi = 0 \tag{4.4}$$

Now, if we suppose that the tangential component of the movement is zero (mean curvature of the interface is defined as the divergence of the normal), i.e. $V_t = 0$, and write $\vec{N}$ in terms of the implicit function, the movement can be described as in (4.5).

$$\begin{aligned} &\frac{\partial \Phi}{\partial t} + V_n\vec{N} \cdot \nabla\Phi = 0 \\ &\frac{\partial \Phi}{\partial t} + V_n\frac{\nabla\Phi}{|\nabla\Phi|} \cdot \nabla\Phi = 0 \\ &\frac{\partial \Phi}{\partial t} + V_n|\nabla\Phi| = 0 \end{aligned} \tag{4.5}$$

where V_n is the *normal velocity*, i.e. velocity in the normal direction. Fig. 4.3 depicts normal of the implicit function.

Thus, mean curvature motion is characterised by $V_n = -b\kappa$, where κ is the curvature and is defined as the divergence of the normal as in 4.6 [51].

$$\kappa = \nabla \cdot \ \vec{N} = \nabla \cdot \left(\frac{\nabla\Phi}{|\nabla\Phi|}\right) = DIV\left(\frac{\nabla\Phi}{|\nabla\Phi|}\right) \tag{4.6}$$

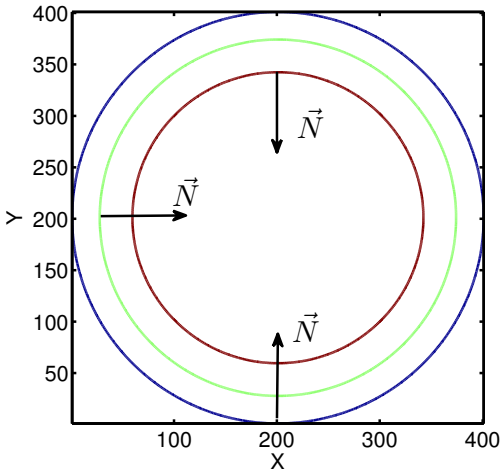


Figure 4.3: Normal of the implicit function Φ. $\vec{N} = \nabla\Phi/|\nabla\Phi|$.

Now, with the above definitions, the mean curvature motion (MCM) is given by (4.7).

$$\frac{\partial\Phi}{\partial t} = b|\nabla\Phi|DIV\left(\frac{\nabla\Phi}{|\nabla\Phi|}\right) \tag{4.7}$$

where b is the strength of the curvature term.

4.4 Hypothesis-Forming-Validation-Loops and Segmentation

In this section we show how disparity estimations can be enhanced using spatial constraints based on segmentation, therefore leading to improved scene interpretation. By using a Hypothesis-Forming-Validation-Loop (HFVL) our method effectively fuses low- and middle-level vision cues, thus increasing coherency and quality of the estimations. We describe a segmentation scheme based on physical model abstractions (polynomials as generalised surfaces of interest) that can be efficiently used as middle-level module to produce feedback cues towards enhancing low level disparity calculation methods. Improvements are considerable, especially in difficult cases without sufficient spatial features (e.g weakly textured scenes), where dense disparity methods typically tend to fail, possibly leading to an incorrect scene interpretation.

4.4.1 Motivation

The lack of discernible features in stereo can have a profound negative impact on the quality of the resulting dense disparity maps [83], making any scene interpretation based on these very difficult. In this work we propose a method for improving both coherency and quality of the dense disparity maps at areas with insufficient spatial stereo features. As it was already mentioned in the introduction of this chapter, our main goal is to make the existing methods more robust and thus more usable in real applications. Such 'difficult' cases can be, amongst other things, areas without spatial features (e.g. tables, walls and so on).

Improving quality of stereo disparity maps is not only of academic interest, especially in the case of real scenes, but is motivated by real applications such as robotic vision. For example, in robotics the problem is often circumvented by placing textured dummy objects, such as a tablecloth, in the scene. The lack of visual structure is especially prominent in indoor environments where little natural texture exists. In everyday environments modifying the environment is often undesirable, and therefore other solutions need to be devised in order for the dense stereo to be useful. In order to cope with the lack of features, information propagation is typically used in stereo algorithms. This can be achieved, for example, by using a regularisation (or smoothness) term or local segmentation. Such information propagation performs well locally but may fail over longer distances or when the original disparity approximations are incorrect to start with. In these cases, feedback from a higher level of the visual processing would be useful, but often such priors are not directly available. Higher-level feedback is also strongly motivated by the human visual system, where the majority of connections to most areas are in fact feedback processing pathways [38]. Here we propose to automatically form a hypothesis of a possible scene interpretation (spatial knowledge) based on segmentation of an initial disparity map, and then to use this information as a constraint (or a priori information of the scene). If the hypothesis and the data are in agreement, both the coherency and quality of the constrained disparity map is increased, thus reinforcing correct scene interpretation. On the other hand, if the hypothesis and the data disagree the hypothesis is rejected. Since segmentation is generally considered to be a middle-level visual task, whereas disparity approximation is a low-level task, the method proposed in this work effectively combines both

low- and middle-level cues for enhancing the scene interpretation. It is important to note that the primary aim of this paper is not to propose another disparity estimation or segmentation algorithm, but instead to exploit the idea of a *Hypothesis-Forming-Validation-Loop* (HFVL), introduced in [60], in this context and show that it can be used to significantly improve the coherence of scene interpretation. We would like to point out that the hypothesis-forming-validation loop is not confined to any particular segmentation method, or even to middle-level information, but cues from higher levels of abstraction can be used as well. In more detail, the contributions of this paper are as follows: The idea of hypothesis-forming-validation loop introduced in [60] is extended by showing how hypotheses can be generated based on segmentation. In addition, to the best of our knowledge, the surface segmentation model introduced in sec. 4.4.4.3 is novel. The segmentation scheme makes efficient use of physical model abstractions as polynomials of different degrees.

Next, section 4.4.2 presents the relevant related work. Then, the proposed general approach as well as the disparity estimation and segmentation methods used in this work are presented in sec. 4.4.3 and sec. 4.4.4, correspondingly. In sec. 4.4.5, related to the experiments, we first validate the segmentation method using well known stereo-images and then give an example of robotic grasping based on dense disparity. Finally conclusions are discussed in Section. 4.5.

4.4.2 Related Work and Our Contribution

As it was already mentioned in Section 3.10.2, using constraints in variational correspondence methods in itself is nothing new. For example in [11][10] Black and Anandan introduced temporal continuity as an energy term. In their case, the temporal term can be considered causal in the sense that the temporal information is progated forward in time. A more recent work with similar kind of causality is that of Werlberger et al. [81]. On the other hand, in [79] Weickert and Schnörr propose a spatio-temporal smoothness constraint where temporal information is propagated over the complete sequence. A more recent work, with similar kind of processing, is that of Salgado and Sánchez [62]. However, we would like to point out that although the previously mentioned works indeed use constraints, most of them stay on the low-level, whereas in this work be propose combining middle- and low-level cues.

Previously, several methods have been introduced which use color segments for improving disparity calculation in areas with insufficient spatial stereo features. Yang et al. [83] applied a window-based stereo algorithm to a stereo-image pair and then fitted planes to the segments. Other methods assume that planes in disparity space coincide with color-segments, improving pixel assignment to their respective disparity plane, e.g. [39]. Dellen and Wörgötter [21] computed sparse disparities from stereo-segment silhouettes, segmented correspondences, and created dense disparities from these by interpolating disparity inside segments.

Our contribution. In Section 3.10 [60] we introduced the idea of the hypothesis-forming-validation-loop (HFVL) by using a spatial constraint as an energy term in variational disparity calculation. Here, the hypothesis is formed based on segmenting the initial disparity map, therefore arriving at a higher abstraction level of the scene via scene interpretation. This idea bears similarity to that of Brox et al. [16], where they combine segmentation and optical-flow calculation, but on the other hand is different in that (a) we directly search for meaningful surfaces (b) the way how constraints are imposed upon the solution [60] and (c) to the best of our knowledge the segmentation model devised in this work is novel.

4.4.3 Hypothesis-Forming-Validation-Loop

The idea of HFVL was put forward by Ralli et al. in [60]: in this work we proposed constraining disparity based on what is known of the solution. Thus information from different levels of abstractions can be used to 'drive' the solution towards a more coherent interpretation. Before moving on, we describe the data flow of the proposed method, in order to make it easier to follow the rest of the paper. Since the method is not confined to any particular choice of disparity and/or segmentation method, it can be considered to be a framework. The data flow shown in Fig. 4.4 is as follows: first an initial disparity map is calculated. It is then segmented in order to generate the hypothesis used as a constraint in the final disparity calculation stage. Although here we only do one iteration cycle of HFVL, several iterations could be carried out. If the initial approximations are good enough, the such an iterative scheme should converge to a more correct scene interpretation. As a last step, the constrained disparity map is segmented. In the next sections the particular instances of the used disparity and segmentation methods are described.

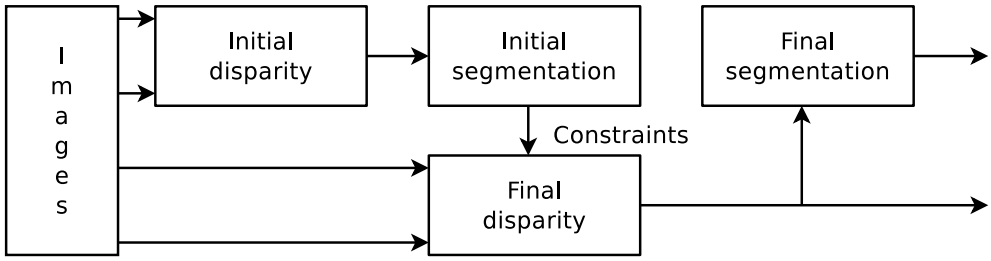


Figure 4.4: Data flow in the proposed method, showing only one iteration cycle of HFVL.

4.4.3.1 Variational Stereo

The variational stereo model used here was already introduced in Section 3.10.4, but, in order for this section to be self contained, the model is briefly explained here. The reason for using a variational method is threefold: a) extending the model is straightforward; b) the same mathematical formalism can be used for both optical-flow and stereo; and c) the governing differential equation(s) can be solved efficiently. An interested reader is pointed to [60][15][17][3]. The energy functional for stereo can be written

$$\begin{aligned} E(d) = &\int_\Omega \Big(D(I_L, I_R, d) + \alpha S(\nabla I_L, \nabla d) \Big) \mathbf{dx} \\ &+ \gamma_s \int_\Omega \Big(C_s(d_{sc}, d) \Big) \mathbf{dx} \end{aligned} \tag{4.8}$$

where d is the disparity of a rectified stereo-image pair, and the images are referred by $I_{i,k}$, where $i \in \{L, R\}$ indicates either left- or right image of a stereo pair, and k defines the channel of a vector valued image (without k written explicitly all channels are referred). The data term is $D(I_L, I_R, d)$, $S(\nabla I_L, \nabla d)$ is the regularisation term, and $C_s(d_{sc}, d)$ is the spatial constraint. $\alpha > 0$ is the weight of the smoothness term, and $\gamma_s > 0$ is the weight of the spatial constraint.

4.4.3.2 Data Terms

The used image representation has a profound effect on the quality of the resulting disparity map, especially under realistic illumination conditions, as we showed in Section 3.9 [59]. The representation chosen here is a combination of gradient and gradient magnitude due to its capability of producing reliable results under both realistic illu-

mination conditions and image noise. The data constancy term is given in (4.9).

$$\begin{aligned} D(I_L, I_R, d) =& b_1 \sum_{k=1}^{K} \Psi_D\Big((\frac{\partial I_{L,k}}{\partial x} - \frac{\partial I^w_{R,k}}{\partial x})^2 \Big) \\ &+ b_1 \sum_{k=1}^{K} \Psi_D\Big((\frac{\partial I_{L,k}}{\partial y} - \frac{\partial I^w_{R,k}}{\partial y})^2 \Big) \\ &+ b_2 \sum_{k=1}^{K} \Psi_D\Big(|\nabla I_{L,k} - \nabla I^w_{R,k}|^2 \Big) \end{aligned} \tag{4.9}$$

where $I_{L,k} \equiv I\big(x, y\big)_{L,k}$ and $I^w_{R,k} \equiv I\big(x + d(x, y), y\big)_{R,k}$ are the left and right (warped as per disparity $d(x, y)$[15][3]) stereo-images, correspondingly, with k indicating the channel in question. $I_{\{K,R\}}$ refers to all the channels of either left or right image. The spatial gradient operator is given by $\nabla \equiv (\partial_x, \partial_y)^T$, and $b_1 > 0$ and $b_2 > 0$ are weights of the terms.

4.4.3.3 Regularization Term

As it was already mentioned in Section 3.10.6, the smoothness term can be image- or flow-driven or a combination of the aforementioned (mixed regularisation). The regularisation term that we have used here, however, is flow-driven, as given in Equation (4.10). The reason for not having used mixed regularisation here is, that we were more interested in the hypothesis-formation-validation-loop and its applicability to real problems, than the actual numerical results. Also, visually flow-driven regularisation produced acceptable results. Nevertheless, we have no reasons to believe that mixed regularisation would not improve the results (quantitatively at least).

$$\begin{aligned} S_I(\nabla I_L, \nabla d) &= g(|\nabla I_L|^2)(|\nabla d|^2) \\ S_F(\nabla I_L, \nabla d) &= \Psi_R(|\nabla d|^2) \end{aligned} \tag{4.10}$$

where $\Psi_R(s^2) = \sqrt{s^2 + \epsilon^2}$. The purpose of $\Psi_R(s^2)$ is to prevent the regularisation term from smoothing across object boundaries, and thus make the solution smooth piece-wise.

4.4.3.4 Spatial Constraint

The spatial constraint term is given in (4.11).

$$
\begin{aligned}
C_s(d_{sc}, d) &= \Psi_{CS\{1,2\}}\big((d_{sc} - d)^2\big) \\
\Psi_{CS\{1\}}(s^2) &= ln(1 + \frac{s^2}{\lambda^2})\lambda^2 \\
\Psi_{CS\{2\}}(s^2) &= exp(-\frac{s^2}{\lambda^2})(-\lambda^2)
\end{aligned}
\tag{4.11}
$$

where d_{SC} is the spatial constraint, $\Psi_{CS\{1\}}$ and $\Psi_{CS\{2\}}$ are robust error functions, and λ is a parameter, that depends on the image scale, used for determining shape of the influence function: where the constraint does not fit the data its influence upon the solution is reduced. Fig. 4.5 displays shapes of the corresponding influence functions for $\lambda = 0.2$.

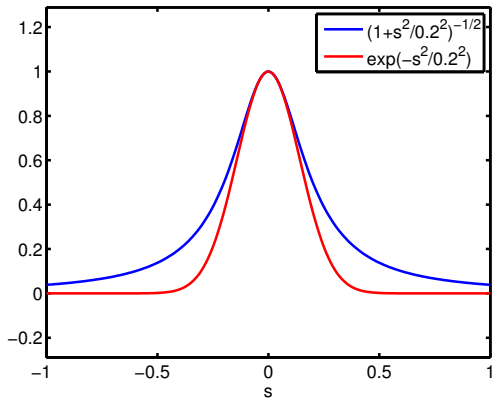


Figure 4.5: Influence functions $\Psi'_{CS\{1\}}(s^2) = 1/(1 + \frac{s^2}{\lambda^2})$ and $\Psi'_{CS\{2\}}(s^2) = exp(-\frac{s^2}{\lambda^2})$ for λ of 0.2.

In Section 3.10.7 we used $\Psi_{CS\{1\}}(s^2)$ as error function in the spatial constraint for disparity. Here, on the other hand, we have used $\Psi_{CS\{2\}}(s^2)$. The reason is that in Section 3.10.7 we expected more errors, especially in the DRIVSCO sequence, and, therefore, we needed an error function with 'broader' influence area. Here, on the other hand, we want a more precise error function, in order to not to fuse foreground objects with the background. As it can be observed from Figure 4.5, the exponential influence function approaches zero faster than the one based on the logarithmic error function.

4.4.4 Segmentation

The energy based segmentation model derived in this work is based on the *Active Regions* [20] by Chan and Vese and *Region Competition* [86] by Zhu and Yuille, while

the multi-region version is inspired by the model of Brox and Weickert [14][13]. Even though here we segment disparity maps, the same model can be used for segmenting optical-flow [14][13], or incorporate other cues apart from disparity, like disparity and colour.

4.4.4.1 Two Regions

We first describe the standard model, that is a combination of [20] and [86] (see also [13]), which is used for segmenting the image of interest into two non-lapping segments, and then move on to describing our version of the model. Since our task is to segment a disparity map into *meaningful* segments the model needs to be modified slightly: even though disparity maps can be presented as grey-valued images, they reflect information related to 3D structure of the scene. By meaningful segments in this context we mean points belonging to the same surface.

4.4.4.2 Standard Model

The evolving curve $\Gamma(x(s), y(s)) \equiv \partial\Omega$ is defined in the domain $\Omega \in \mathbb{R}^2$ as the closed boundary separating the regions Ω_1 and Ω_2, with $\Omega = \Omega_1 \cup \Omega_2$ and $\Omega_1 \cap \Omega_2 = \emptyset$. Thus we can define $inside(\Gamma) \equiv \Omega_1$ to be the inside region separated by the closed boundary Γ with $outsize(\Gamma) \equiv \Omega_2$ being the outside region. We expect that the disparity values inside the regions are homogeneous, in some sense, and are generated by the probability distribution functions $p(d|\alpha_i)$, where i indicates the region in question, with α_i being the parameters [86]. α_i depends on the used probability distribution and will be defined later on.

$$\begin{aligned} E(\Gamma, \alpha_1, \alpha_2) =& \mu \int_\Gamma ds \\ &- \int_{\Omega_1} log\, p\left(d|\alpha_1\right) \mathbf{dx} \\ &- \int_{\Omega_2} log\, p\left(d|\alpha_2\right) \mathbf{dx} \\ p\left(d|\alpha_i\right) =& p\left(\{d(x,y) : (x,y) \in \Omega_i\}|\alpha_i\right) \end{aligned} \tag{4.12}$$

where the first term is the the length of the boundary curve, and the second and the third terms are the cost of 'coding' the disparity values according to $p(d|\alpha_i)$ for each region. Instead of using explicit curve representation we use implicit representation

where the interface (curve) is defined as the isocontour (or level-set) of the function Φ as defined in (4.13).

$$\begin{cases} \Gamma \equiv \partial\Gamma = \{(x,y) \,|\, \Phi(x,y) = 0\} \\ inside(\Gamma) \equiv \Omega_1 = \{(x,y) \,|\, \Phi(x,y) \geq 0\} \\ outside(\Gamma) \equiv \Omega_2 = \{(x,y) \,|\, \Phi(x,y) < 0\} \end{cases} \tag{4.13}$$

Therefore using the level-set representation the energy functional (4.12) can be defined as given in (4.14).

$$\begin{aligned} E(\Phi, \alpha_1, \alpha_2) = & \mu \int_\Omega \delta(\Phi) |\nabla\Phi| \mathbf{dx} \\ & - \int_\Omega H(\Phi) \, log \, p\,(d|\alpha_1) \; \mathbf{dx} \\ & - \int_\Omega H(1-\Phi) \, log \, p\,(d|\alpha_2) \; \mathbf{dx} \end{aligned} \tag{4.14}$$

where $H(\Phi)$ is the Heaviside function and δ is the one dimensional Dirac measure as defined in (4.15).

$$\begin{aligned} H(\Phi) &= \begin{cases} 1, & \text{if } \Phi \geq 0 \\ 0, & \text{if } \Phi < 0 \end{cases} \\ \delta(\Phi) &= \frac{d}{d\Phi} H(\Phi) \end{aligned} \tag{4.15}$$

Now the only thing that needs to be defined is the probability density function $p\,(d|\alpha_i)$. Without loss of generality, in this work a Gaussian distribution has been used, since we expect the measured disparity values to be corrupted by a large number of random processes. Therefore if $\alpha_i = (\mu_i, \sigma_i^2)$, then the probability distribution function is defined as in (4.16).

$$p\,(\{d(x,y) : (x,y) \in \Omega_i\}|\alpha_i) = \frac{1}{\sqrt{2\pi\sigma_i}^2} exp\left(\frac{-\big(d(x,y) - \mu_i\big)^2}{2\sigma_i^2}\right) \tag{4.16}$$

The problem with (4.16) is that when it is used for modelling stereo-disparity $d(x,y)$ (which is inherently related to the 3D setup of the scene) it only successfully describes surfaces with a constant disparity value. In order to account for more complex surfaces the model has to be modified to account for the physical model behind the disparity.

4.4.4.3 Surface Segmentation Model

As it was already stated before, a better physical model is needed to account for the disparity values. By reformulating (4.16) we obtain (4.17).

$$p_r\left(\{d(x,y):(x,y)\in\Omega_i\}|\alpha_i\right)=\frac{1}{\sqrt{2\pi\sigma_i^2}}exp\left(\frac{-r_i^2}{2\sigma_i^2}\right) \tag{4.17}$$

where $r = d - \hat{d}$, with d being the observed disparity values, and $\hat{d}$ being the disparity values generated by the physical model. Without loss of generality, the models used in this work are parametric multivariate polynomials of first (linear) and second (quadratic) degree, such that $\hat{d} = Am_i$ where m_i are the parameters (of the physical model) for the region i and A is a 'position' matrix, that depends on the degree of the multivariate polynomial. Therefore, the complete parameter vector would be $\alpha_i = (m_i, \sigma_i^2)$. The parametric physical model explains the observed disparity values based on the position (x, y) of the observation and, therefore, in linear case $A = [X\,Y\,1]$, while in the quadratic case $A = [X^2\,Y^2\,XY\,X\,Y\,1]$ with X and Y being the coordinate vectors. The surface model in (4.16) would be a polynomial of 0:th degree (constant) with $A = [1]$, $m_i = \mu_i$ and $r_i = d - \mu_i$ and thus would only be suitable for describing surfaces with constant disparity values, as was argued earlier. Both of the segments, $i = \{1, 2\}$, are modelled by their respective parametric surfaces Am_1 and Am_2.

Now depending on the disparity map being segmented (into two), it may well be that the two multivariate polynomials of degree one or two do not describe well the two segments, since the 3D scene might consist of more than two distinct surfaces. Instead, we segment the domain Ω into those points that belong to the surface in question, and those that do not: Ω_1 defines the segment described by the surface, while Ω_2 is the segment that does not fit the surface, with $\Omega = \Omega_1 \cup \Omega_2$. If we go back to (4.14), we can see that there are two terms, namely $p\,(d|\alpha_1)$ and $p\,(d|\alpha_2)$, that 'compete' for $d(x, y)$. Thus we need to define $p\,(d|\alpha_2)$ so that it somehow represents Ω_2. One possibility is to define $p\,(d|\alpha_2)$ in terms of $p\,(d|\alpha_1)$. Thus the energy functional for the surface

segmentation becomes as in (4.18).

$$\begin{aligned} E(\Phi, \alpha_1) =& \mu \int \delta(\Phi)|\nabla\Phi|\mathbf{dx} \\ &- \int H(\Phi)\, log\, p_r\left(d|\alpha_1\right)\, \mathbf{dx} \\ &- \int H(1-\Phi)\, log\, \overline{p_r}\left(d|\alpha_1\right)\, \mathbf{dx} \end{aligned} \tag{4.18}$$

where the physical model for the surface is either of first or second order, as described earlier. Now if $p_r\left(d(x,y)|\alpha_1\right)$ is as in (4.17), we want to define $\overline{p_r}\left(d(x,y)|\alpha_1\right)$ so that there is balanced 'competition' between the regions Ω_1 and Ω_2. Since the univariate Gaussian distribution is described by $N(r^2, \sigma^2) = \left(1/\sqrt{2\pi\sigma^2}\right) exp\left(-r^2/2\sigma^2\right)$, where the first term (between parentheses) can be seen as a coefficient fixing shape (and thus height) of the curve, therefore in order to have balanced competition between the regions, we can approximate $\overline{p_r}\left(d|\alpha_1\right)$, for example, with (4.19).

$$\begin{aligned} p_r\left(d|\alpha_1\right) &= \frac{1}{\sqrt{2\pi\sigma_1^2}} exp\left(\frac{-r_1^2}{2\sigma_1^2}\right) \\ \overline{p_r}\left(d|\alpha_1\right) &= \frac{1}{\sqrt{2\pi\sigma_1^2}} - p_r\left(d|\alpha_1\right) \end{aligned} \tag{4.19}$$

Figure 4.6 gives an example for terms p and $\overline{p}$ ($\sigma^2 = 1$).

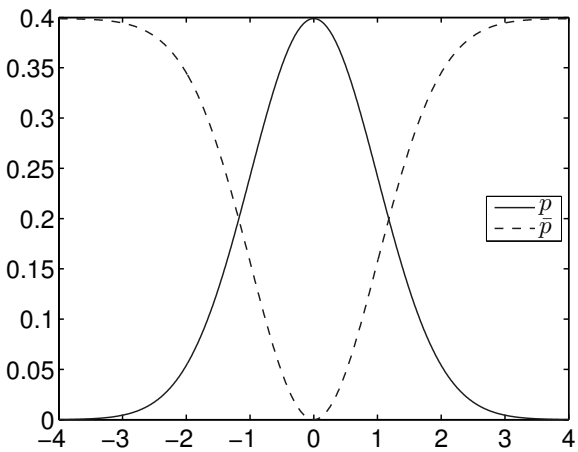


Figure 4.6: Terms p and $\bar{p}$ for $\sigma^2 = 1$, as defined in (4.19).

Figure 4.7 shows the results of the surface segmentation for the Tsukuba case. This case is, of coarse, the simplest possible case since all the objects are fronto-parallel.

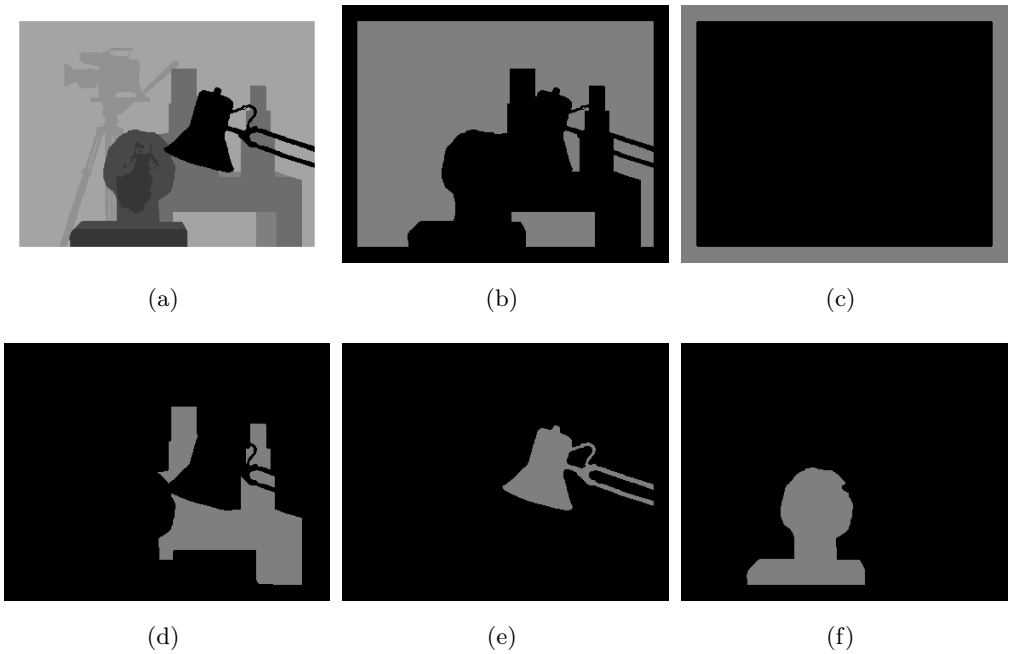


Figure 4.7: Surface segmentation for Tsukuba. (a) Disparity ground truth; (b) 1st segment; (c) 2nd segment; (d) 3rd segment; (e) 4th segment; (f) 5th segment.

4.4.4.4 Multi-Region

Since our surface segmentation model segments the disparity map into a meaningful segment Ω_1 (describing a surface) in order to find the rest of the surfaces one simply needs to segment, successively, the remaining Ω_2 (Ω_1 is omitted from the successive segmentations). Therefore we end up having $\Omega = \cup_{i=1}^{N} \Omega_i$ where N is the number of segments represented by $N-1$ level-set functions. Even though each of the $N-1$ segments (obtained by successive application of the two-region surface segmentation model) describe a single meaningful surface, it is possible, due to incomplete competition, that the segments contain points corresponding to other surfaces and therefore it is necessary to establish competition between all the regions meaning that coupling between the different level-set functions is needed. As already mentioned, we segment the image into $N-1$ meaningful segments where the N:th segment contains those positions that do not belong to any meaningful surface and therefore do not have a model. Our reasoning is that due to the 3D setup of the scene not necessarily all of the points can be explained by a surface: for example far away points may belong to a surface

but due to the distance (from the camera) the whole surface is seen as only few points in the camera. Therefore $\Omega = \cup_{i=1}^{N}\Omega_i$ where $\Omega_N = \Omega\neg\left(\cup_{i=1}^{N-1}\Omega_i\right)$ and $\Omega_i \cap \Omega_j = \emptyset$ if $i \neq j$. The above mentioned means, simply, that segments $\left(\cup_{i=1}^{N-1}\Omega_i\right)$ contain the segments that we have found by applying the segmentation process, while the segment Ω_N contains the pixels that have not been assigned to any particular segment.

The energy functional of our multi-region surface segmentation model is as given in (4.20).

$$\begin{aligned} E(\Phi_i, \alpha_i, \alpha_j) =& \mu \int \delta(\Phi_i)|\nabla\Phi_i|\mathbf{dx} - \int H(\Phi_i)\, log\, p_r\left(d|\alpha_i\right)\, \mathbf{dx} \\ &- \int H(1-\Phi_i) \max_{\substack{H(\Phi_j)>0 \\ i\neq j}} \Big(log\, \overline{p}_r\left(d|\alpha_i\right), log\, p_r\left(d|\alpha_j\right)\Big)\mathbf{dx} \end{aligned} \tag{4.20}$$

where the last term defines the competitive coupling between the different regions. In the two region case the competing term was easy to deduce: it is the other segment. In the multi-region case this is not so straightforward anymore. We have to deduce, in the case of a segment Ω_i, which of the segments Ω_j, with $i\neg j$, is competing the most for the same positions, in order to have 'balanced' competition between the segments. By defining the energy functional as given in (4.20), after we have deduced the most competing segments for all of the existing segments, we have solve the corresponding level-set equations all at once.

4.4.4.5 Solving the Equations

Now that we have defined the energy functionals for both the two- and the multi-region segmentation models, we move on to describing how these can be solved efficiently. In Section 4.4.4.6 we give a more detailed explanation of the actual algorithm and also discuss about the initialisation. In order to avoid getting stuck on local minima, and for increasing convergence speed, the energy functionals are solved in multi-grid fashion [13][14][74][19]. We search for an initial solution on the finest scale and propagate the solution sequentially to coarser scales and return back to to the finest scale. Since each coarser scale is a simplified version of the finer scale problem, resolving the equation in this fashion effectively avoids getting stuck on local minima and increases the convergence speed. In the two region case, where we search for the initial segments, V-cycle is used whereas in the multi-region case we use W-cycles. The rationale for this is that

since in the multi-region case the competition is balanced, this is where most of the work minimising the functionals should be done (avoiding the local minima simultaneously). Both V- and W-cycles are depicted in Fig. 4.8. At each scale the functionals (4.18)

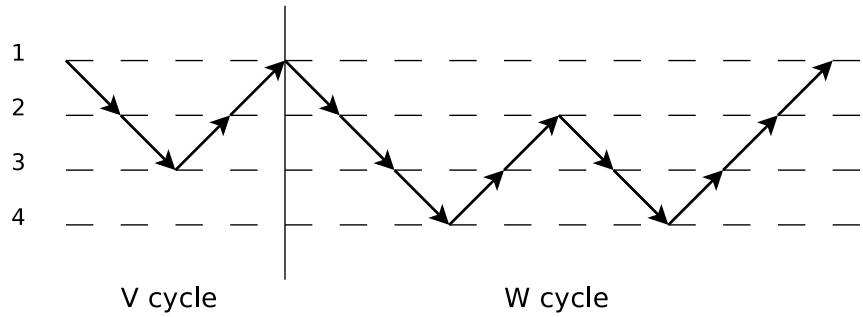


Figure 4.8: Multi-grid V- and W-cycles.

and (4.20) depend on both the level-set function Φ_i and the parameters α_i. Therefore, we use the same kind of algorithm, as proposed in [86], consisting of two steps: in the first step Φ_i is kept constant while the functional is minimised for α_i and in the second step α_i is kept constant while the functional is minimised for Φ_i. A single iteration step consists of both the steps. The steps are as follows:

1. *Minimisation of* α

   Minimising the energy functionals keeping the Φ_i constant (using the last known value of Φ), as per Bayes rule, corresponds to maximising the conditional probability as given (for discrete case) in (4.21)

$$\alpha_i = arg \max_{\alpha_i} \prod_{(x,y)\in\Omega_i} p\left(\alpha_i | \{d(x,y) : (x,y) \in \Omega_i\}\right) \tag{4.21}$$

   where $\alpha_i = (m_i, \sigma_i^2)$. Since our 'generalised' probability density functions depend on the physical model Am_i, we need to estimate the model's parameters m_i. This is done using random sample consensus (RANSAC) due to it's ability to find structures consisting of substantially less than half of the data points [69][26]. One of the crucial parameters of the RANSAC method is the minimum number of data required to fit the model. We call this simply RAN_n. If the Ω_1 is initialised so that it contains positions from the whole disparity domain, it is expected that in the first iteration cycles only a small fraction of the data will fit the model found by the RANSAC. However as the Ω_1 converges towards the surface the

number of data that fits the model is expected to increase and thus should reflect RAN_n. Therefore we start with a RAN_n covering 5% of the data and arrive at 60-70%. The variance of the Gaussian distribution is estimated as given in (4.22).

$$\sigma_i^2 = max(\frac{\int r_i^2 H(\Phi_i)}{\int H(\Phi_i)}, \sigma_{min}) \tag{4.22}$$

where $r = d - \hat{d}$, with d being the observed disparity values, and $\hat{d}$ being the disparity values generated by the physical model. The purpose of σ_{min} is to limit the variance. Equation (4.22) is derived from the fact that, in our case, variance is calculated by $\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(d_i - \hat{d}_i)^2$. We simply use the Heaviside function to define the segment of interest in question. If the variance is not limited, due to the greedy nature of the algorithm it tends to over segment in some cases which is effectively avoided by limiting the minimum allowed variance. Values between 0.25 and 1.0 were typically used for σ_{min}.

2. *Minimisation of* Φ

   Minimising the energy functionals with respect to Φ_i, keeping the α_i constant, is performed using gradient descent where descent direction is parametrised by artificial time $t \geq 0$ as in [20]. This involves solving the associated Euler-Lagrange equations of the form $\Phi_i(t, x, y)$. The Euler-Lagrange equation for (4.18), keeping the α_1 fixed, is given in (4.23).

$$\partial_t \Phi = H'_\epsilon(\Phi)\left(log\, p_r\left(d|\alpha_1\right) - log\, \overline{p_r}\left(d|\alpha_1\right) + DIV\left(\frac{\nabla\Phi}{|\nabla\Phi|}\right)\right) \tag{4.23}$$

   The Euler-Lagrange equation for (4.20), keeping the $\alpha_{\{i,j\}}$ fixed, is given in (4.24).

$$\begin{aligned}\partial_t \Phi_i = H'_\epsilon(\Phi_i)\Bigg(& log\, p_r\left(d|\alpha_i\right) \\ & - \max_{\substack{H(\Phi_j)>0 \\ i\neq j}} \left(log\, \overline{p_r}\left(d|\alpha_i\right), log\, p_r\left(d|\alpha_j\right)\right) \\ & + DIV\left(\frac{\nabla\Phi_i}{|\nabla\Phi_i|}\right)\Bigg)\end{aligned} \tag{4.24}$$

   where $H_\epsilon(\Phi)$ is a regularised version of the Heaviside function [20], where ϵ defines the amount of spatial support, as given in (4.25) and shown in Fig. 4.9.

$$H_\epsilon(\Phi) = \frac{1}{2}\left(1 + \frac{2}{\pi} arctan\left(\frac{\Phi}{\epsilon}\right)\right) \tag{4.25}$$

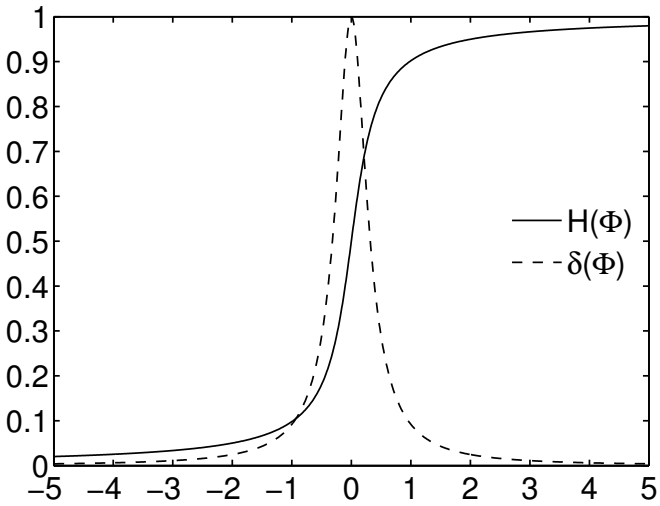


Figure 4.9: Graph of the regularised Heaviside function, as defined in (4.25).

Both (4.23) and (4.24) are solved in semi-implicit fashion using an additive operator splitting scheme (AOS)[80] similar to [42].

4.4.4.6 Segmentation Algorithm

Now that the different parts of the algorithm are clear the complete segmentation algorithm is given in Alg. 1. The $surfaceSegment()$ function returns two segments: points that belong to the surface found in the data (first region) and points that do not belong to the surface (second region). An important part of the algorithm that

Algorithm 1 Segmentation algorithm where d is the disparity map to be segmented, Ω is the domain of segmentation, reg is the region to be segmented, $\Omega(n)_1$ are the level-set functions describing the segments and n is the number of regions.

$reg = \Omega,\ n = 1$
while $size(reg) > thr$ **do**
  $[\Omega(n)_1\ \Omega_2] = surfaceSegment(d,\ reg)$
  $reg = \Omega_2$
  $n = n + 1$
end while
$\Omega_1 = regionCompetition(d,\ \Omega_1)$

has not been addressed so far is the initialisation of the surface segmentation model that produces the initial segments. By initialisation we mean the status of the level-set function $\Phi(0, x, y)$ (at time $t = 0$). This presents a problem since there must be enough points for the algorithm to be able to find all the surfaces of interest (including fairly small surfaces) but on the other hand if there are too many initial points the RANSAC method might find non-meaningful surfaces. We found out that by using a somewhat tight initial grid (every fifth position in both x- and y-direction was set to 1 the rest being -1) and by gradually changing the RAN_n as explained earlier and solving the equations in multi-grid fashion produces desirable results. However, we must add, that better initialisation is expected to produce even better segmentation results. On interesting possibility would we using segmentation results based on colour, for example, to initialise our method. Figure 4.10 shows the initialisation and the first five iterations, in the Tsukuba case, for the first segment.

4.4.5 Experiments

The purpose of the experiments is twofold. Firstly, we validate the surface segmentation method using well known test images. Secondly, we show how the proposed hypothesis-forming-validation loop enhances scene interpretation in a real, uncontrolled, scenario. More results, including videos, are available at `http://atc.ugr.es/~jarnor/index.php/investigation/results`.

4.4.5.1 Error Metrics

Formulae of the used error metrics for disparity are given in equations (4.26) and (4.27).

$$MAE := \frac{1}{N} \sum_{i=1}^{N} abs\Big((d)_i - (d_{gt})_i\Big) \tag{4.26}$$

$$C := \frac{\#\left\{ i \mid abs\Big((d)_i - (d_{gt})_i\Big) \leq 1 \right\} 100}{N} \tag{4.27}$$

where N is the number of pixels and d_{gt} is the ground truth. MAE stands for mean average error while C is percentage of correct disparities (± 1 disparity level).

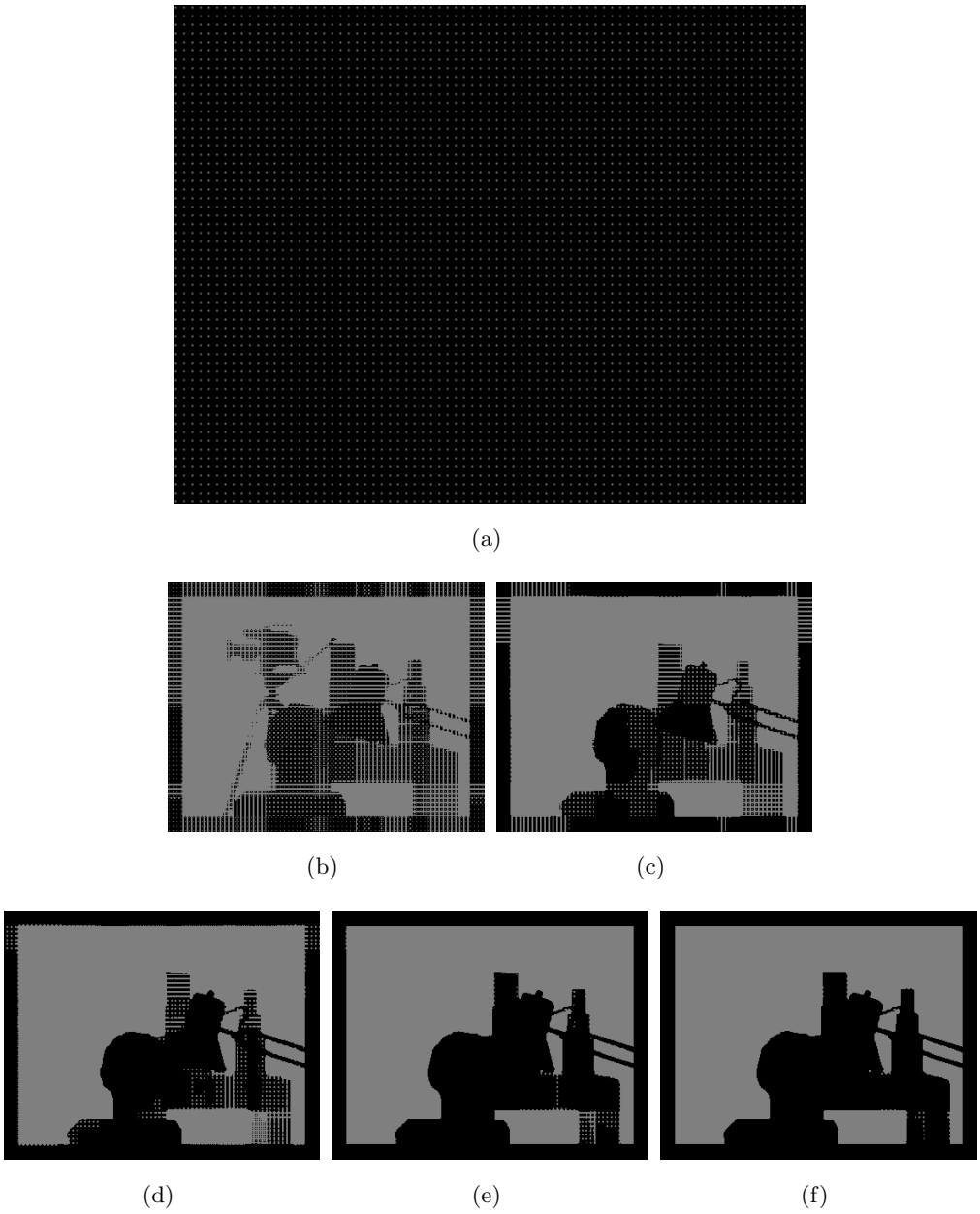


Figure 4.10: Initialisation and the first five iteration cycles for Tsukuba. (a) Initialisation; (b) 1st iterationM; (c) 2nd iteration; (d) 3rd iteration; (e) 4th iteration; (f) 5th iteration.

4.4.5.2 GRASP and Middlebury Images

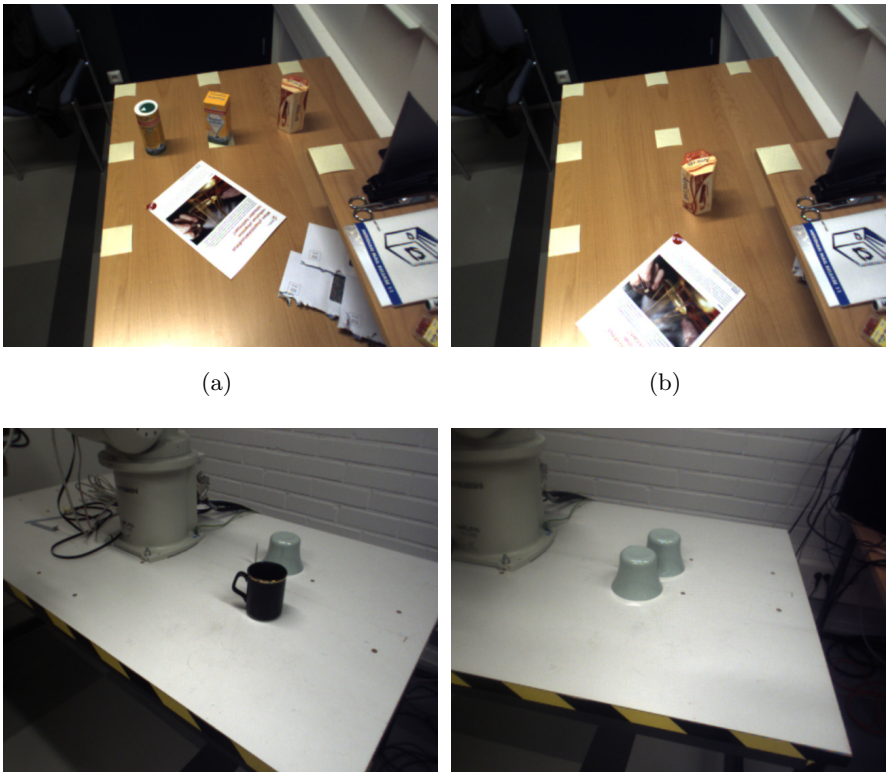
Both well known test images from Middlebury[1] [32][6] database and application specific images from the GRASP[2] project for robotic grasping were used. Fig. 4.11 shows three

(a) (b) (c)

Figure 4.11: Middlebury images (a) Tsukuba; (b) Teddy; (c) Cones.

of the four Middlebury images used for ranking disparity calculation methods. As it can be observed, Tsukuba contains only fronto-parallel objects, making it relatively easy to segment, whereas Teddy and Cones are more demanding due to number of objects and scene setup. However, at this point, we would like to point out that these images contain sufficient spatial features for most disparity calculation methods to calculate disparity maps good enough to be used for scene interpretation. Fig. 4.12 shows the application specific images related to robotic grasping. GRASP 1 and 2 cases can be considered easy, due to the number of objects and texture present in the table, whereas GRASP 3 and 4 cases are considerably more difficult: the table contains only very few useful spatial features for stereo and in the GRASP 4 case the objects on the table are both textureless and of the same colour. GRASP 1 and 2 can be considered typical cases where either additional objects are placed on the surface (table in this case) or the surface itself contains sufficient spatial features for extracting the disparity correctly. However, many man made objects, with painted or otherwise finished surfaces, tend to lack texture and/or spatial features, as can be seen in GRASP 3 and 4. This is not only true for tables, but also walls and paved roads (just to mention a few) [57] can present problems for the dense disparity methods.

[1] http://vision.middlebury.edu/
[2] http://www.csc.kth.se/grasp/



(a) (b)

(c) (d)

Figure 4.12: GRASP images (a) GRASP 1; (b) GRASP 2; (c) GRASP 3; (d) GRASP 4.

4.4.5.3 Reference Results for Middlebury

We start by validating the segmentation algorithm and demonstrate qualitatively, that although the disparity maps generated by the used disparity calculation method are far from being perfect, the segmentation algorithm is capable of producing satisfactory results for scene interpretation. Therefore, for validation purposes, segmentation results are given for both GT (ground-truth) and calculated disparity maps. Figs. 4.13 and 4.14 depict the disparities and the segmentation results. Segmentation parameters are the same throughout the tests, and the physical model of the segments is a multivariate polynomial of a second degree.

Fig. 4.13 shows the disparities for the test images. Upper row displays ground-truths (a to c) while lower row contains the calculated disparities (d to f) without any constraints.

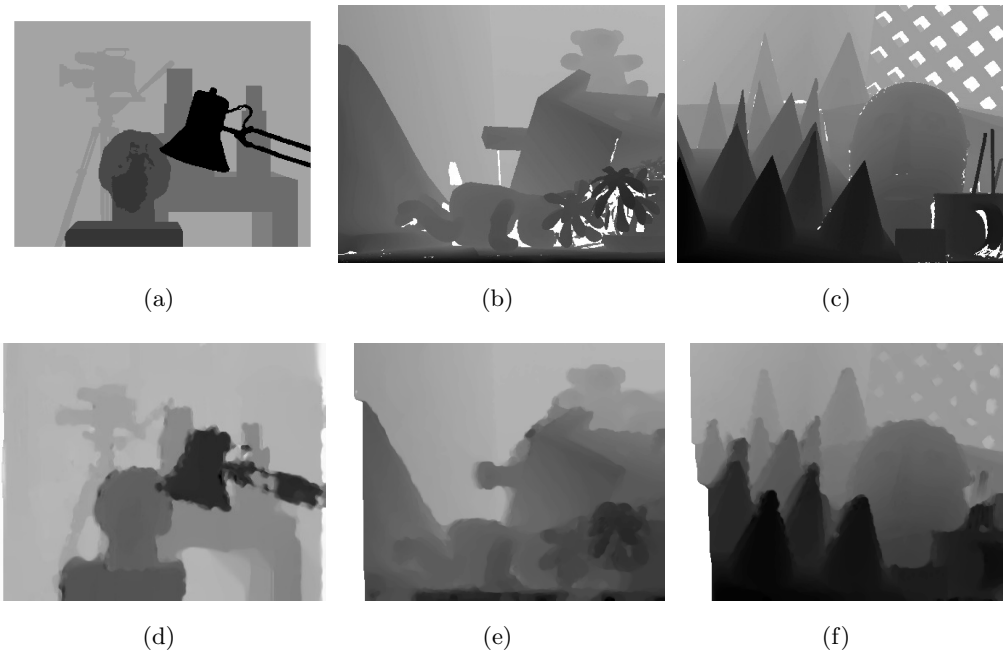


Figure 4.13: Disparity maps: (a) Tsukuba GT; (b) Teddy GT; (c) Cones GT; (d) Tsukuba calculated; (e) Teddy calculated; (f) Cones calculated. GT stands for groud-truth.

Fig. 4.14 shows the segmentation results (upper row based on GTs while lower row is based on the calculated disparities). Interestingly enough the segmentation algorithm is capable of finding the table in the Cones case: a task which is seemingly difficult even for a human observer when based on the disparity. In general, the results based on calculated disparity are on par with the ones based on GT. In the following we apply a constraint for the background in the Tsukuba case and segment the resulting disparity map. The constraint is obtained by searching for a single 'significant' plane in the disparity map, using the segmentation algorithm, which happens to be the background. This clearly improves approximations for the background, especially near the table legs, therefore leading to better scene interpretation. Without the constraint a part of the background, near the table legs, is erroneously segmented as an independent segment, even though it in reality belongs to the background, as can be observed in Fig. 4.15. For Teddy and Cones cases we have ignored the first 35 columns from the error calculation due to lack of stereo information.

The improvement of 3.6% approximately can also be seen in Table 4.1 by comparing

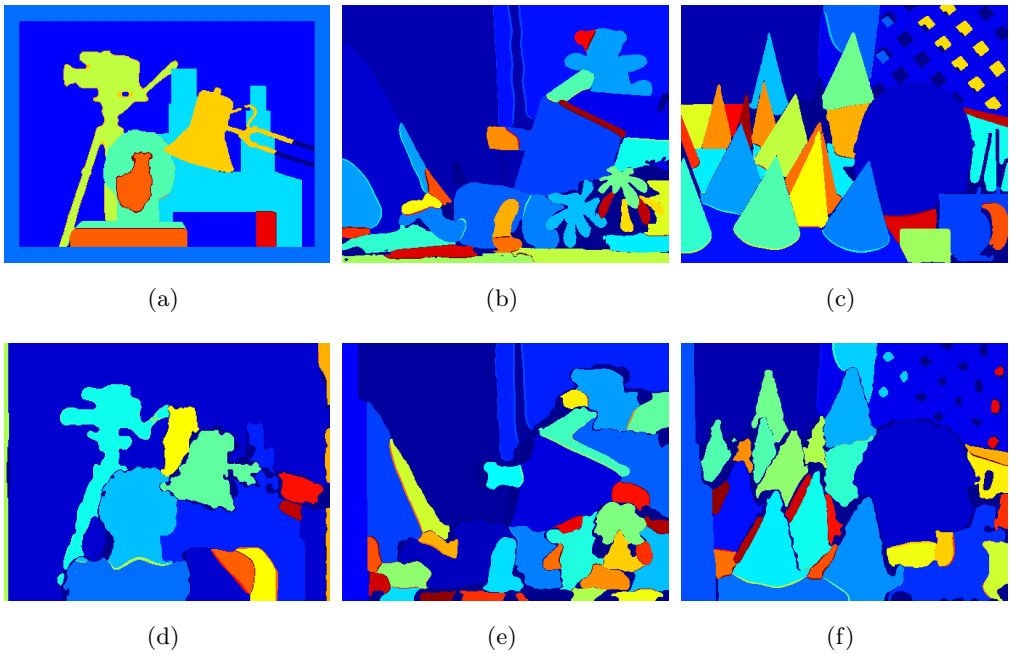
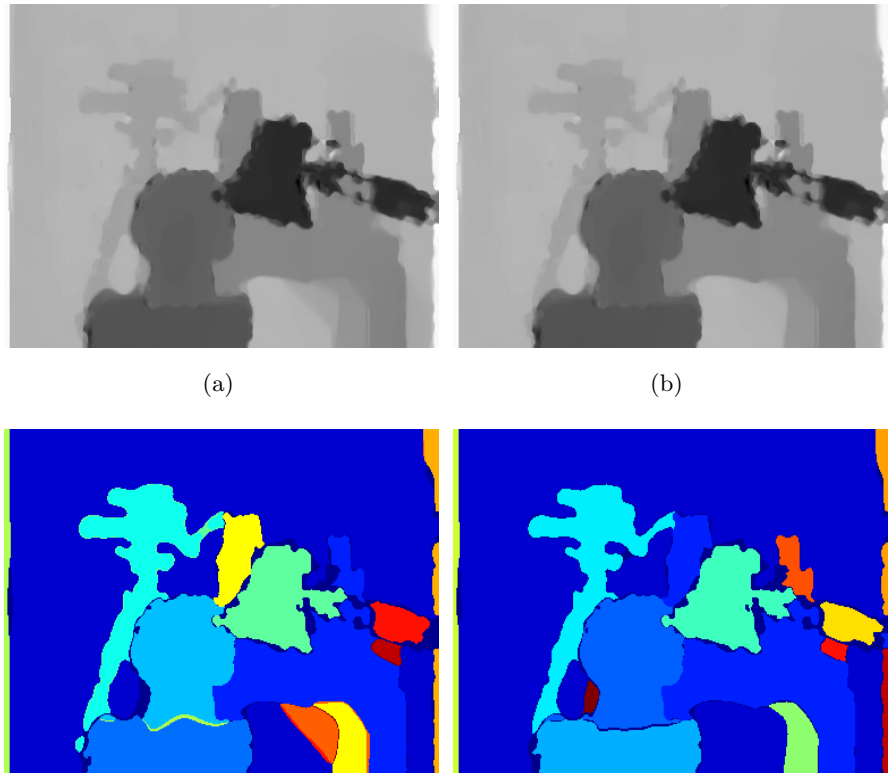


Figure 4.14: Segmentation results: (a) Tsukuba GT; (b) Teddy GT; (c) Cones GT; (d) Tsukuba calculated; (e) Teddy calculated; (f) Cones calculated.

Table 4.1: Results in MAE (mean average error) and C (percentage of correct disparities). Const. stands for constrained. The constraint is obtained, using the segmentation algorithm, as mentioned in the text.

Image	MAE	C (%)
Tsukuba	0.55	90.5
Tsukuba (const.)	0.53	90.9
Teddy	1.06	82.5
Cones	0.99	85.4

'Tsukuba' and 'Tsukuba (const.)'. This seemingly modest improvement in MAE (or percentage of correct disparities) leads to a clearly better scene interpretation, as can be observed in Fig. 4.15. Disparity approximations near the posterior table leg have improved, leading to better segmentation results in that area.



(a) (b)

(c) (d)

Figure 4.15: (a) Disparity WO constraint; (b) Disparity with constraint; (c) Segmentation based on a; (d) Segmentation based on b. With the constraint the estimations improve for the background, especially near the table legs. WO stands for 'without'.

4.4.5.4 Segmentation for Robotic Grasping

Here we demonstrate qualitatively how the proposed method is used for (a) generating useful spatial constraints, (b) generating high quality disparity maps and (c) segmenting the disparity map. We show that spatial constraining effectively improves coherency of the disparity estimations, leading to improved segmentation results and, therefore, to better scene interpretation. Spatial constraint is obtained automatically by segmenting the initial disparity map into a single planar object: since the table is the biggest planar object available in the images, this is found as expected. The physical model of the constraint is of first degree, while the final segments are of second degree. Imposing a planarity constraint (arising from the table) in the whole disparity map domain has effectively corrected the false estimations. Where the spatial constraint does not fit the data it gets rejected, therefore allowing correct disparity estimation for rest of the objects. In the figures const. is the constrained disparity solution.

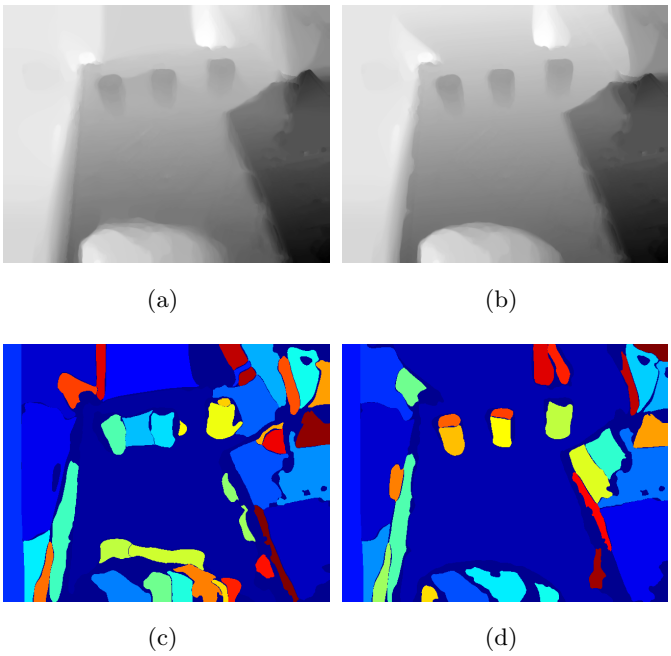


Figure 4.16: GRASP 1 (a) Disparity (none); (b) Disparity (const.); (c) Segmentation (based on a); (d) Segmentation (based on b). Constraining the solution leads to better disparity estimations for the table, especially near the objects of interest, therefore leading to better scene interpretation.

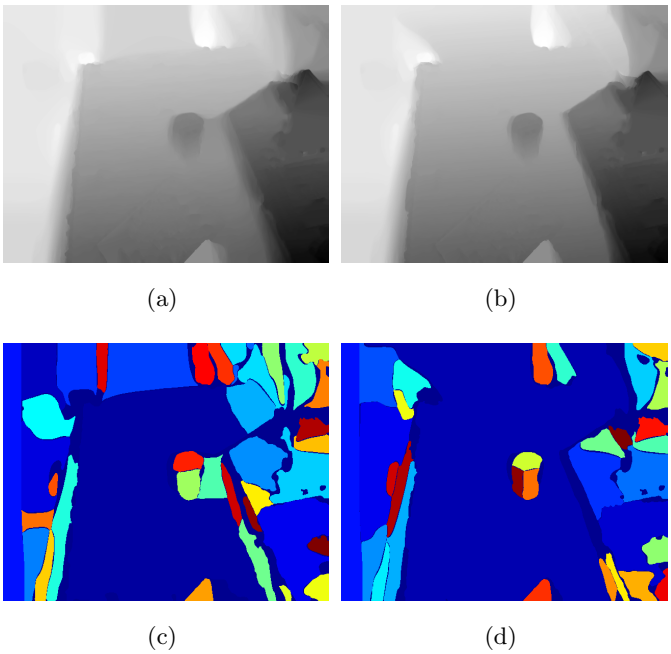


Figure 4.17: GRASP 2 (a) Disparity (none); (b) Disparity (const.); (c) Segmentation (based on a); (d) Segmentation (based on b). Segmentation based on the constrained disparity map shows considerable improvement for the object of interest situated on the table.

From figs. 4.16-4.19 it can be observed that the spatial constraint considerably improves quality of the resulting disparity maps and therefore scene interpretation based on the segmentation. With the spatial constraint, the disparity values change smoothly, as a function of the distance, for the table.

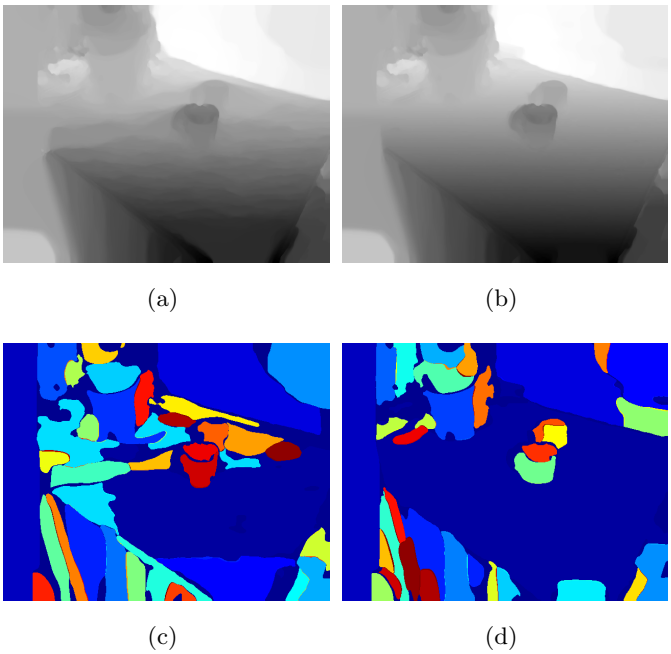


Figure 4.18: GRASP 3 (a) Disparity (none); (b) Disparity (const.); (c) Segmentation (based on a); (d) Segmentation (based on b).

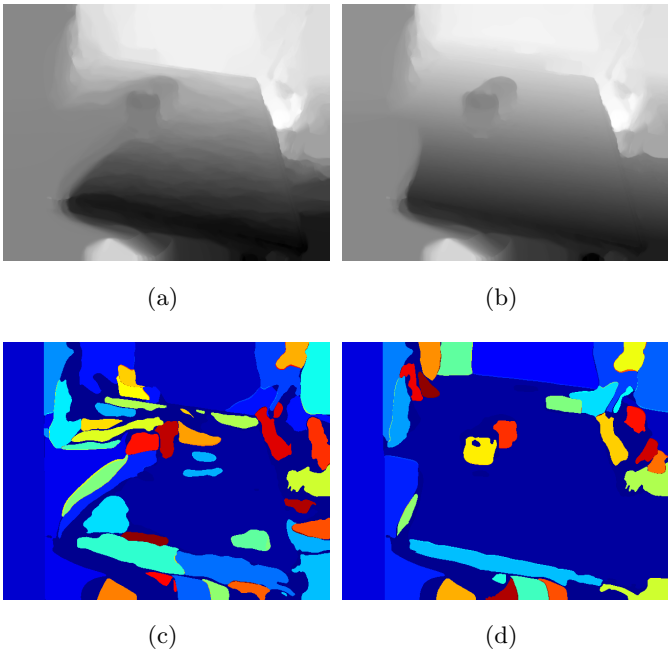
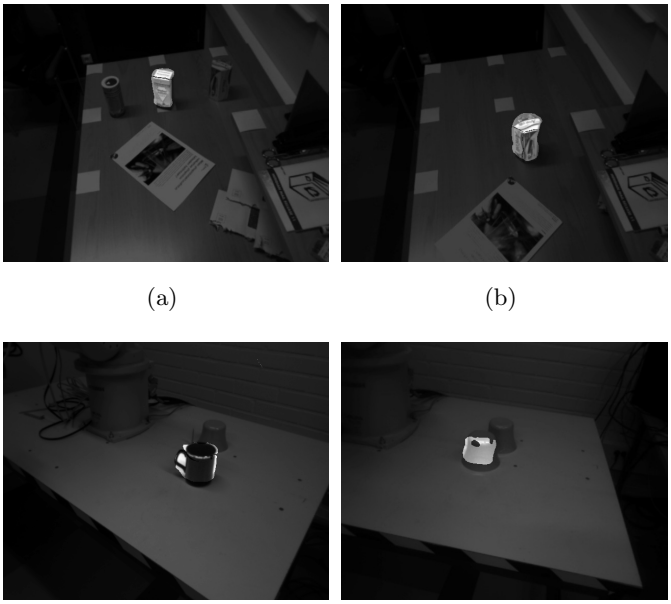


Figure 4.19: GRASP 4 (a) Disparity (none); (b) Disparity (const.); (c) Segmentation (based on a); (d) Segmentation (based on b).

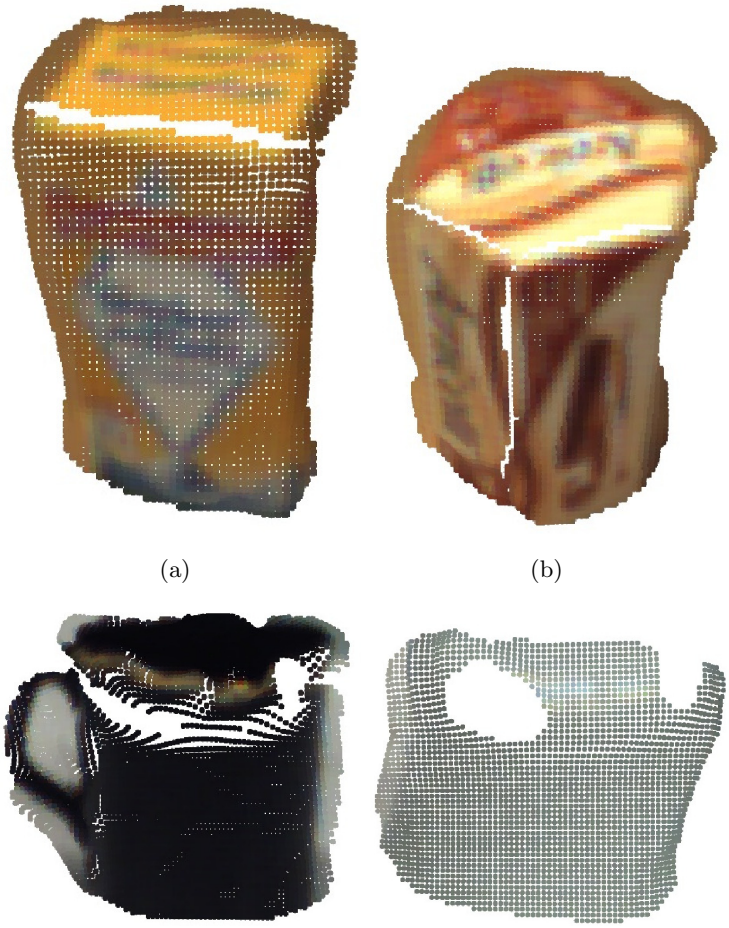
Improvement is especially notable in the GRASP 3 and GRASP 4 cases: this is due to the fact that the table contains only very few useful spatial features for extracting stereo information. Fig. 4.20 shows objects of interest (based on the segmented image) in the left stereo-image, while Fig. 4.21 displays 3D reconstructed point clouds for the same. Object recognition and grasping is based on the 3D reconstructed information.



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

Figure 4.20: Object of interest: (a) GRASP 1; (b) GRASP 2; (c) GRASP 3; (d) GRASP 4.

Fig. 4.21 shows 3D reconstructed point clouds for the objects of interest (seen in Fig. 4.20) based on the segmentation results.



(a) (b)

(c) (d)

Figure 4.21: 3D reconstruction of objects of interest: (a) GRASP 1; (b) GRASP 2; (c) GRASP 3; (d) GRASP 4.

4.5 Conclusions

We have demonstrated how spatial constraints, obtained automatically from the disparity data, can be used for enhancing coherency and quality of the resulting disparity estimations, therefore leading to better scene interpretation. The described framework is general enough to be used with other disparity estimation [57] and segmentation methods, apart from the ones described in this work. The framework effectively allows binding of low- and middle-level vision cues in a meaningful way for specific applications such as robotic vision. We feel that this is the right way to proceed and the obtained results back our thinking: instead of fine tuning the method used for obtaining the disparity estimations to the maximum, possibly leading to over fitting, we effectively use other visual cues for improving the overall quality, therefore making the scene interpretation easier. Obtained results, especially in the difficult cases, show that considerable enhancement of the disparity map is possible.

Same physical model was used for all the segments in the segmentation algorithm. Even better results could be obtained by assigning a different physical model for each segment, based either directly on the data, and/or other external cues. In order to avoid over segmentation, segments can be fused, if sufficiently similar, therefore minimising the overall energy.

In this study we have applied the HFVL cycle only once. An interesting possibility of future work would be to apply the HFL loop several times, iteratively, for different objects/surfaces. If the initial estimations are good enough, we expect that the model should converge to an even better solution.

5

Solving the Equations

5.1 Introduction

In this section, we describe how the previously introduced equations can be solved efficiently. This is achieved by solving the corresponding Euler-Lagrange equations using a *multigrid* approach. In the literature, depending on the field, multigrid is also known as *multilevel*, *multiscale* or *multiresolution* method. The PDEs to be solved in this work exhibit both *space-type*, and *time-type* characteristics, as described in [74]. The time-type is evolved with respect to a time variable t, until either a steady-state or a desired time is reached. On the other hand, a time-type problem, when discretised implicitly with respect to time, yields a discrete space-type problem which has to be solved at each step. On the contrary to the time-type problems, in the space-type problems the steady-state is searched directly. Related Matlab/MEX code is available at `http://atc.ugr.es/~jarnor/`. In Appendix C information related to the code is given.

In order to describe how the equations are solved, while trying to maintain a common notation in the field, we have used, and adapted where necessary, notation from the book Multigrid [74] and the PhD theses of Dr. Bruhn [17], Dr. Brox [13], and Dr. Javier Sánchez [63], therefore, avoiding 'reinventing the wheel'. The PhD thesis of Dr. Bruhn is one of the most thorough works covering the field of variational methods for optical-flow, while the book Multigrid [74] covers the subject of efficient solvers using multigrid techniques in a rigorous and understandable way. On the other hand, Dr. Brox's thesis covers both the optical-flow and segmentation using level-set based

formulation, obtaining very interesting results indeed. Last, but certainly not least, for the Spanish speaking population we can recommend the PhD thesis of Dr. Javier Sánchez. Amongst other things, it discusses a concept called *symmetric flow* that can be used to further improve the results obtained with the variational correspondence methods. There are other equally good studies about the subject of variational methods for optical-flow, but we have mentioned these, since these are the ones that we have mainly used.

We continue by introducing the used notation in Section 5.2. In Section 5.3, we briefly introduce concepts/solvers, such as, Successive Over Relaxation (SOR), Gauss-Seidel (GS), Alternating Line Relaxation (ALR) and other crucial techniques that we have used to solve the equations. In Section 5.4, for completeness' sake, we re-introduce the equations to be solved and discuss some of their properties. Discretisation of the equations is discussed in Section 5.5, while in sections (5.6.1) and (5.7) we show how to construct solvers for the optical-flow (both early- and late linearisation cases) and the level-set equations. In Appendix B extra information related to stencil notation is given.

5.2 A Word About the Used Notation

Before going any further, we need to introduce and explain some of the coming notation, conventions, and concepts, so that reading the rest of the section will be easier. We consider the image to be a continuous function/mapping with $I(x, y, k) : \mathbb{R}\times\mathbb{R} \to \mathbb{R}^{K+}$, where the domain of the image is $\Omega \subset \mathbb{R} \times \mathbb{R}$ and K defines the number of channels (e.g. $K = 1$ in the case of grey-level images and $K = 3$ in the case of RGB images). It is in this regular domain where our PDEs are defined. However, in order for us to solve the PDEs, they need to be discretised first. Also, the kind of images that we are dealing with are, in fact, discretised versions that we receive from the imaging devices, such as digital- or thermal cameras. We define a discretisation grid as given in (5.1):

$$G_h := \{(x, y) \mid x = x_i = ih_x,\, y = y_j = jh_y\,;\, i, j \in \mathbb{Z}\} \tag{5.1}$$

where $h = (h_x, h_y)$ is a discretisation parameter. With the discretisation grid, the domain of the discretised images can be defined as $\Omega_h = \Omega \cap G_h$. Instead of $I(x, y) =$

$I(ih_x, jh_y)$, we typically use $I_{i,j}$ when pointing to the pixels. Figure 5.1 depicts both the image pixels and the calculation grid.

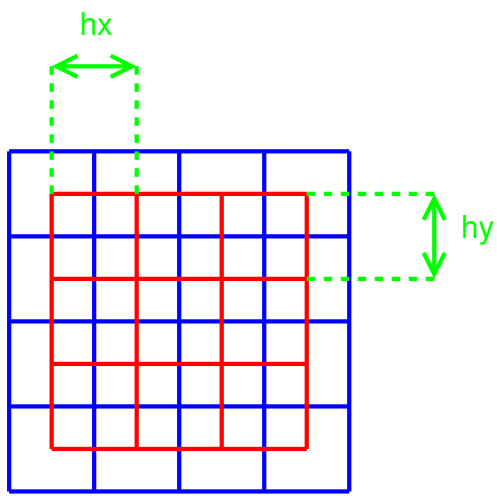


Figure 5.1: A Cartesian grid. The blue grid defines the image pixels while the red grid is the computational grid used for solving the PDEs. We assume that $h_x = h_y$. Origin of the grid is in the left upper corner and pixel positions are defined by subindices (i, j).

Figure 5.1 shows the grid used for discretising the equations. The blue cells can be thought of representing the physical image sensors (e.g. CCD cells), while the red grid shows the computational grid, where the nodes (points of interest) align with the centres of the image sensors. Due to Bayer alignment of the RGB image sensors, the above description is not accurate for most cameras, but an acceptable simplification.

Sometimes positions on a grid are given using both cardinal- and inter-cardinal directions, as defined in figure 5.2. The idea is to simplify the notation of the discretised versions of the equations which can be rather messy.

NW	N	NE
W	C	E
SW	S	SE

Figure 5.2: Directions on a grid. To simplify the notation, both cardinal- and inter-cardinal directions are used. Here W, N, E, S, and C refer to west, north, east, south, and centre, respectively.

5.2.1 Pixel Numbering Schemes

As it was already mentioned, we can refer to any pixel in the image by using $I_{i,j}$, where $0 \leq i \leq m$, $0 \leq j \leq n$. Here the discretisation parameter $h = (h_x, h_y)$ has been chosen so that the discretised domain is $\Omega_h : [1, m] \times [1, n]$. Another particularly useful way is to think of the discretised image as a vector $I \in \mathbb{R}^N$. Now, the components of the vector are $I_{\mathcal{I}}$ where $\mathcal{I} \in \{1, \ldots, N\}$ and N is the number of pixels in image. This second numbering scheme is particularly useful for algorithmic descriptions and in matrix notation, as will be shown later on.

Figure 5.3 depicts both column- and row-wise traversing order of pixels. For the element wise solvers we have used column wise ordering, while in the case of the block wise solver we use both column- and row wise traversing orders. We would like to point out that these are not the only possibilities. An interested reader is pointed to [74] for more information.

1	4	7
2	5	8
3	6	9

(a) Column wise.

1	2	3
4	5	6
7	8	9

(b) Row wise.

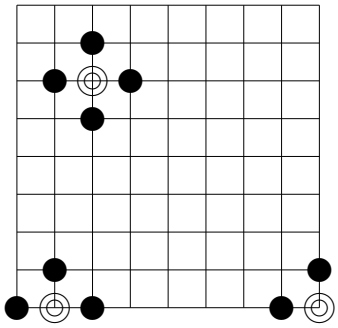
Figure 5.3: Column- and row-wise pixel orderings. Here, $\mathcal{I} \in \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

.

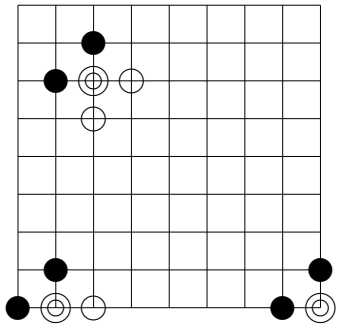
5.2.2 Pixel Neighbourhoods

In order to simplify the notation, for example in algorithmic descriptions, we define different kinds of pixel neighbourhoods. With pixel neighbourhood we mean image pixels $I_{\mathcal{J}}$ that are neighbours of a pixel of interest $I_{\mathcal{I}}$. The neighbourhoods are slightly different depending on if we are talking about a element- or a block-wise solver. By element wise solver we mean a Jacobi or a Gauss-Seidel type iterative solvers, that search for the solution for a single element at a time. On the other hand, block type solvers search for a solution for a group of elements (or a block). However, since the neighbourhoods have the same function in both the above mentioned cases, we use the same neighbourhood operator to denote the neighbours. It should be clear from the

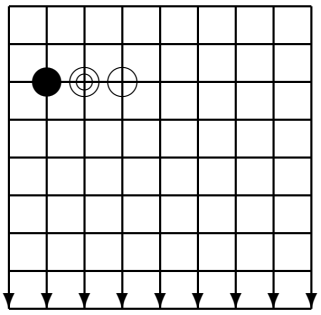
structure of the solver which kind of a neighbourhood is in question. $\mathcal{J} \in N(\mathcal{I})$ denotes the set of neighbours $\mathcal{J}$ of $\mathcal{I}$, as seen in Figure 5.4 (a).



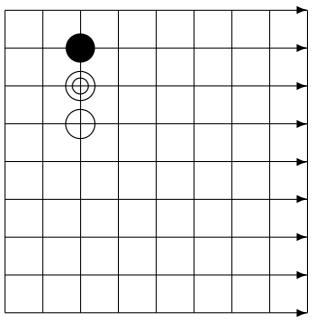
(a) 4-neighbourhood with eliminated boundary conditions.



(b) Element wise neighbourhood with eliminated boundary conditions.



(c) Block wise, column ordering.



(d) Block wise, row ordering.

Figure 5.4: Pixel neighbourhoods, where central pixel, $\mathcal{I}$, is denoted with a double circle. (a) Painted circles denote neighbouring pixels $\mathcal{J}$ that belong to the neighbourhood defined by $\mathcal{J} \in N(\mathcal{I})$. (b) Painted circles denote pixels $\mathcal{J}$ that belong to the neighbourhood defined by $\mathcal{J} \in N^{-}(\mathcal{I})$, while unpainted circles denote pixels $\mathcal{J}$ that belong to the neighbourhood defined by $\mathcal{J} \in N^{+}(\mathcal{I})$. (c)-(d) Painted circles denote pixels $\mathcal{J}$ that belong to the neighbourhood defined by $\mathcal{J} \in N^{-}(\mathcal{I})$, while unpainted circles denote pixels $\mathcal{J}$ that belong to the neighbourhood defined by $\mathcal{J} \in N^{+}(\mathcal{I})$. Processing order is defined by the arrows. Due to the eliminated boundary conditions 'scheme', the pixel neighbourhood operators only point to valid neighbours, as shown in (a) and (b).

Pixel wise. $\mathcal{J} \in N^{-}(\mathcal{I})$ denotes the set of neighbours $\mathcal{J}$ of $\mathcal{I}$ with $\mathcal{J} < \mathcal{I}$ (painted circles in Figure 5.4 (b)), and $\mathcal{J} \in N^{+}(\mathcal{I})$ denotes the neighbours $\mathcal{J}$ of $\mathcal{I}$ with $\mathcal{J} > \mathcal{I}$ (unpainted circles in Figure 5.4 (b)).

Block wise. $\mathcal{J} \in N^-(\mathcal{I})$ denotes the neighbours $\mathcal{J}$ of $\mathcal{I}$ with $\mathcal{J} < \mathcal{I}$ (painted circles in Figure 5.4 (c)-(d)). Since we run the block wise solver both column- and row-wise, depending on the direction, the neighbour(s) defined by this neighbourhood operator changes. $\mathcal{J} \in N^+(\mathcal{I})$ denotes the neighbours $\mathcal{J}$ of $\mathcal{I}$ with $\mathcal{J} > \mathcal{I}$ (painted circles in Figure 5.4 (c)-(d)). Again, the actual neighbours defined by this operator depends on the direction of the solver.

In a Gauss-Seidel (see Equation (5.14)) type solver, whether a point- of block-wise, a $N^-(\mathcal{I})$ denotes those neighbours that have a new solution (calculated at $l+1$), while $N^+(\mathcal{I})$ denotes those neighbours that still have an old solution (calculated at l).

5.3 Numerical Methods

In this section some of the used numerical techniques are briefly introduced in order to make this work more self-contained. We show, for example, how a linear system of equations $Ax = b$ can be solved for $x \in \mathbb{R}^N$, where $A \in M_{N\times N}(\mathbb{R})$ and $b \in \mathbb{R}^N$, using both element- and block wise iterative solvers. We denote this kind of an iterative solver by (5.2). Systems of equations can be solved directly by using methods such as Gaussian elimination or Cholesky decomposition. However, in our case the system matrix A is a highly sparse matrix (see Equations (5.40) and (5.41)), meaning that most of the entries contain zeros. Therefore, the above mentioned direct methods would be highly inefficient. Instead, we search for a solution using iterative schemes.

$$x^{l+iter} = SOLVER(A, b, x^l, iter) \tag{5.2}$$

where x^l as a solution at iteration cycle l and x^{l+iter} is the solution (approximation) after *iter* number of iterations. Later on, in Section (5.6.1), we show that it is not necessary to construct the complete system matrix A in order to solve the system using the iterative methods. This is highly beneficial in the case of sparse matrices where most of the entries are zeros.

In what follows, we introduce three different kinds of iterative solvers, namely Jacobi, Gauss-Seidel (GS) and Alternating Line-Relaxation (ALR). The first two are element wise solvers, while the last is a block wise solver. What comes to the convergence speed, ALR is the fastest, GS is the second and Jacobi method is the slowest. However, depending on the system architecture, where the system will be implemented, ALR or

GS might not necessary be the fastest options run-time wise. Data dependency , together with the system architecture, define what would be the most convenient option. An interested reader is pointed to [74] for more information on the subject.

5.3.1 SOR

Successive Over-Relaxation (SOR) [84] is a method for improving the convergence speed. It can be combined with different kind of solvers, such as Jacobi- or Gauss-Seidel methods. A SOR iteration cycle can be written as:

$$x^{l+1} = (1-\omega)x^l + \omega SOLVER(A, b, x^l, iter), \tag{5.3}$$

where ω is the relaxation factor. The choice of ω depends on the properties of the system matrix A, and finding an optimal or quasi optimal factor might not be simple.

5.3.2 Jacobi

The Jacobi method is guaranteed to converge if the matrix A is diagonally dominant. Equation (5.4) describes the linear system of equations in matrix/vector format.

$$Ax = b \tag{5.4}$$

where the matrix A and the vectors x and b are as follows:

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1N} \\ a_{21} & a_{22} & \dots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N1} & a_{N2} & \dots & a_{NN} \end{bmatrix}, x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}, b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_N \end{bmatrix} \tag{5.5}$$

The matrix A can be decomposed into D and R matrices containing the diagonal and the residual elements, as given in (5.6).

$$D = \begin{bmatrix} a_{11} & 0 & \dots & 0 \\ 0 & a_{22} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & a_{NN} \end{bmatrix}, R = \begin{bmatrix} 0 & a_{12} & \dots & a_{1N} \\ a_{21} & 0 & \dots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N1} & a_{N2} & \dots & 0 \end{bmatrix}, \tag{5.6}$$

Using $A = (D + R)$, we may rewrite the original system of equations as follows:

$$Dx = b - Rx \tag{5.7}$$

In a matrix/vector format, Jacobi iteration can be written as follows:

$$x^{l+1} = D^{-1}(b - Rx^l) \tag{5.8}$$

Since D is diagonal, its inverse is trivial to calculate. Entry-wise new approximations are obtained as given in (5.9).

$$x_i^{l+1} = \frac{1}{a_{ii}}\Big(b_i - \sum_{j \neq i} a_{ij} x_j^l\Big) \tag{5.9}$$

As it can be observed from (5.9), 'old' solutions (i.e. x^l) are used when approximating new solutions x^{l+1}. Due to this, the Jacobi method converges more slowly than the Gauss-Seidel method (that will be explained next). However, since there is no dependency of processing order between x^l and x^{l+1}, in certain parallel architectures, such as GPUs, parallel implementations of Jacobi method can be more efficient than corresponding Gauss-Seidel implementations.

5.3.3 Gauss-Seidel

Gauss-Seidel is an iterative method for solving a linear system of equations, which convergence is guaranteed only if the 'system' matrix is either diagonally dominant, or symmetric and positive definite. Equation (5.10) describes the linear system of equations in matrix/vector format.

$$Ax = b \tag{5.10}$$

where the matrix A and the vectors x and b are as follows:

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1N} \\ a_{21} & a_{22} & \dots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N1} & a_{N2} & \dots & a_{NN} \end{bmatrix}, \; x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}, \; b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_N \end{bmatrix} \tag{5.11}$$

By decomposing A into $D - L - U$, where D is a diagonal matrix, L is a strictly lower triangular matrix and U is a strictly upper triangular matrix, we can re-order the system as given in Equation (5.12).

$$Dx = b + (L + U)x \tag{5.12}$$

Since D is diagonal, its inverse is trivial to calculate. In matrix/vector format a Gauss-Seidel iteration can be defined as in (5.13).

$$x^{l+1} = D^{-1}b + D^{-1}(L+U)x^l \tag{5.13}$$

Entry-wise new approximations are obtained as given in (5.14).

$$x_i^{l+1} = \frac{1}{a_{ii}}\Big(b_i - \underbrace{\sum_{j>i} a_{ij}x_j^l}_{\text{old solutions}} - \underbrace{\sum_{j<i} a_{ij}x_j^{l+1}}_{\text{new solutions}}\Big) \tag{5.14}$$

From Equation (5.14) it can be noted that new solutions are used as they become available. Later on, when deriving Gauss-Seidel type solvers for the different equations, the fact that new- and old solutions are used, further complicates the notation, unfortunately. x^0 (guess on the first iteration) can be any vector, but the closer the initial guess is of the true solution, the faster the method will converge.

5.3.4 TDMA/ALR

TDMA. Tridiagonal matrix algorithm (TDMA) is a simplified form of Gaussian elimination, that can be used for solving tridiagonal systems of equations. In matrix/vector format this kind of a system can be written as in (5.15)

$$\underbrace{\begin{bmatrix} b_1 & c_1 & 0 & 0 & 0 \\ a_2 & b_2 & c_2 & 0 & 0 \\ 0 & a_3 & b_3 & \ddots & 0 \\ 0 & 0 & \ddots & \ddots & c_{N-1} \\ 0 & 0 & 0 & a_N & b_N \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_N \end{bmatrix}}_{x} = \underbrace{\begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ \vdots \\ d_N \end{bmatrix}}_{d} \tag{5.15}$$

The algorithm consists of two steps: the first (forward) sweep eliminates the a_i, while the second (backward) sweep calculates the solution. Equation (5.16) introduces the forward sweep, while Equation (5.17) shows the backward sweep.

$$
\begin{aligned}
c'_i &= \begin{cases} \dfrac{c_1}{b_1} & , i = 1 \\ \dfrac{c_i}{b_i - c'_{i-1} a_i} & , i = 2, 3, ..., N-1 \end{cases} \\
d'_i &= \begin{cases} \dfrac{d_1}{b_1} & , i = 1 \\ \dfrac{d_i - d'_{i-1} a_i}{b_i - c'_{i-1} a_i} & , i = 2, 3, ..., N \end{cases}
\end{aligned}
\tag{5.16}
$$

$$
\begin{aligned}
x_N &= d'_N \\
x_i &= d'_i - c'_i x_{i+1} \quad , i = N-1, N-2, ..., 1
\end{aligned}
\tag{5.17}
$$

Physical interpretation of the terms a_i and b_i is that they are diffusion weights, i.e. how much the neighbouring solutions are taken into account.

Alternating Line Relaxation. As can be understood from the tridiagonal system matrix A, this kind of a solver is used solving 1D problems, such as 1D Poisson equation. The system matrix A, in the 2D problems that we are interested in is, in fact, block tridigonal (see Equation (5.41)). In order to solve problems of higher dimensionality, we use a scheme called Alternating Line Relaxation (ALR) that is based on TDMA. In this kind of a solver TDMA is applied along each of the dimensions, by changing pixel ordering, while maintaining A tridiagonal (those elements that are not on the diagonals next to the main diagonal are transferred to d). In 2D case, using column wise ordering, we solve for new estimations by traversing the system column by column, while using row wise ordering ,we solve for new estimations by traversing the system row by row, as shown in Figure 5.5. For example, in the case of column wise ordering, we first apply forward- and backward sweeps on the first column and, therefore, obtain a new approximation for this column. Then we do the same for the rest of the columns. The basic idea behind ALR is similar to that of an Additive Operator Splitting (AOS) scheme [80].

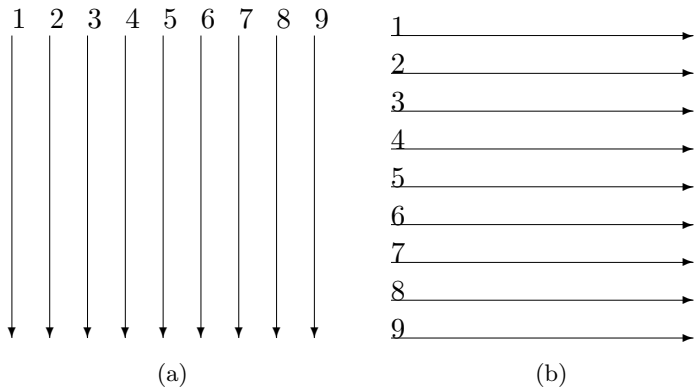


Figure 5.5: Column- and row-wise pixel ordering. In ALR we apply, for example, TDMA first along all the columns and then along all the rows. (a) Column wise ordering; (b) row wise ordering. In both the cases we show the processing order.

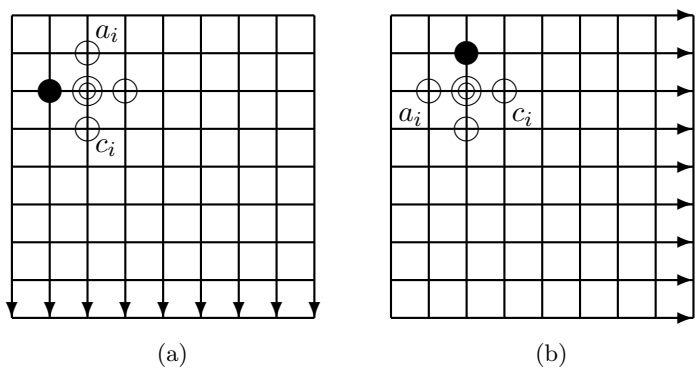


Figure 5.6: Pixel positions a_i and c_i of b_i, depending on the pixel ordering: (a) column wise ordering; (b) row wise ordering. Painted and unpainted circles denote the pixel neighbours, in Gauss-Seidel fashion, that are taken into account when calculating the solution: painted circle denotes a neighbour that contains a solution from the current iteration step, $l+1$, while unpainted circle denotes a neighbour that contains solution from the last iteration step l.

The effect of changing pixel ordering is that the positions a_i and c_i (b_i being the 'central' pixel) point to different pixels in the image, as seen in Figure 5.6. Algorithm 2 gives a basic description of an SOR-GS-ALR scheme (ω is a relaxation parameter like in Successive Over Relaxation (SOR)). While the TDMA gives an exact solution for 1D problems with tridiagonal matrix A, ALR is an iterative scheme. To be more

exact, as described here, it can be understood to be a block wise Gauss-Seidel method. Output from the algorithm is an approximation of x after *iter* iterations.

Algorithm 2 Calculate $x^{l+iter} = SOR\text{-}GS\text{-}ALR(A,\, d,\, x^l,\, \omega,\, iter)$

for $i \rightarrow iter$ **do**
  $x^l = (1-\omega)x^l + \omega TDMA_{col}(A,\, d,\, x^l)$
  $x^l = (1-\omega)x^l + \omega TDMA_{row}(A,\, d,\, x^l)$
end for
$x^{l+iter} = x^l$

5.3.5 Multigrid

In this section, without going too much into details, we will briefly explain the multigrid. Iterative solvers, such as the ones introduced previously, are efficient at 'smoothening' high frequency components of the error (here $error = b - Ax$). In other words, error of the approximation becomes smooth, but not necessarily small. On the other hand, a smooth error term can be approximated on a more coarse grid. This has two very clear benefits, (1) any operation on a coarse grid is computationally less expensive due to fewer grid points and (2) low frequency error components in fine grid become high frequency components on a coarse grid [74]. As it can be understood, these have positive effect on the computational efficiency and the convergence speed. In our case, there exists an additional reason, especially in the late linearisation case, why we want to use multigrid: in order to avoid physically irrelevant minimizers that are due to the fact, that the energy functional can be non-convex.

Basic multigrid. Let's suppose that we want to solve a linear system of equations, defined as follows:

$$Au = b, \text{ where } A \in M_{N\times N}(\mathbb{R}),\, b \in \mathbb{R}^N \tag{5.18}$$

Now, let's suppose that we have an approximation u^l of the real vector of u. Now we can calculate a defect:

$$d^l = b - Au^l \tag{5.19}$$

By using the defect defined in (5.19), Equation (5.18) can be written as follows (since $Av^l = b - Au^l$):

$$Av^l = d^l \tag{5.20}$$

By solving for the correction v^l term we obtain $u = u^l + v^l$. However, we can use an approximation $\hat{A}$ of A, so that the original defect equation is replaced with Equation (5.21)

$$\hat{A}v^l = d^l \tag{5.21}$$

Now, if this equation is easier/faster to solve than the original equation, then we have effectively come up with a more effective solver, and this is the basic idea behind the multigrid. Here, $\hat{A}$ would be the system matrix on a coarse grid. Naturally, the defect d^l needs to be transferred to the same grid. In what follows, we will introduce the needed transfer operators. Multigrid methods are based on two principles, namely those of the *smoothing principle* and the *coarse grid principle* [74]:

Smoothing principle. Many traditional iterative solvers, such as Gauss-Seidel, have strong smoothing effect on the error. Instead of making the error small, it first efficiently makes the error smooth.

Coarse grid principle. A smooth error term can be approximated well on a coarse grid. At the same time, any procedure on a coarse action takes less time due to reduced number of operations.

Related to the coarse grid principle, we need means to transfer defects/corrections between different grid sizes. With this in mind, we introduce two operators called the *restriction*- and the *prolongation* (interpolation) operators, that are used to transfer defects/corrections from the coarse-to-fine grid and vice-versa:

$$\begin{aligned} d_H^l &= O_h^H d_h^l \quad \text{, where } O_h^H \text{ is a restriction operator} \\ v_h^l &= O_H^h v_H^l \quad \text{, where } O_H^h \text{ is a prolongation operator} \end{aligned} \tag{5.22}$$

where the sub-index $\{h,\, H\}$ defines the grid where the defect/correction term is defined. An example of a fine- and a coarse grid are given in Figure 5.7.

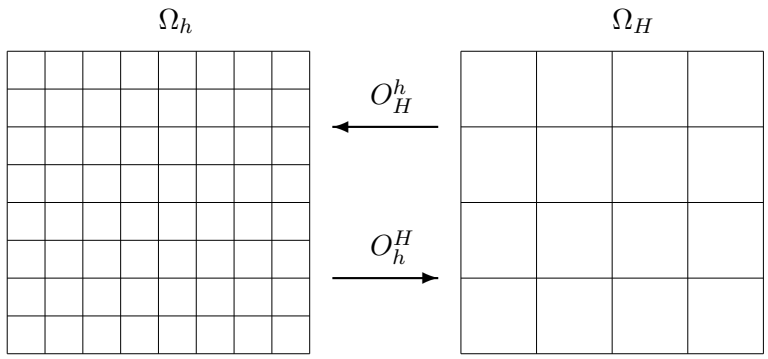


Figure 5.7: A fine (Ω_h) and a coarse (Ω_H) grid, in corresponding order.

Now that we have described all the necessary parts, and we suppose that we have an iterative solver (for example Gauss-Seidel as explained in Section 5.3.3), such that $u^l = SOLVER(A,\, b,\, u^0,\, l)$, where u^0 is an initial approximation and l is the number of iteration cycles, we move on to describing a two-grid cycle as follows:

Two-grid Cycle. Description of a two-grid cycle

1. Presmoothing
   - Start from an initial approximation u_h^0
   - Run solver for l iterations to obtain $u_h^l = SOLVER(A_h,\, b_h,\, u_h^0, l)$
2. Coarse Grid Correction (CGC)
   - Compute defect: $d_h^l = b_h - A_h u_h^l$
   - Restrict the defect: $d_H^l = O_h^H d_h^l$
   - Solve on Ω_H: $v_H^l = SOLVER(A_H,\, d_H^l,\, u_H^0, l)$
   - Interpolate the correction: $v_h^l = O_H^h v_H^l$
   - Corrected approximation $u_h^l = u_h^l + v_h^l$
3. Postsmoothing
   - Run solver for l iterations $u_h^{l+1} = SOLVER(A_h, b_h, u_h^l, l)$

The basic two-grid solver can be extended to run over several different grid sizes in varying manner. Figure 5.8 depicts different multigrid cycles, namely V- and W-cycles and a unidirectional Full MultiGrid (FMG) cycles.

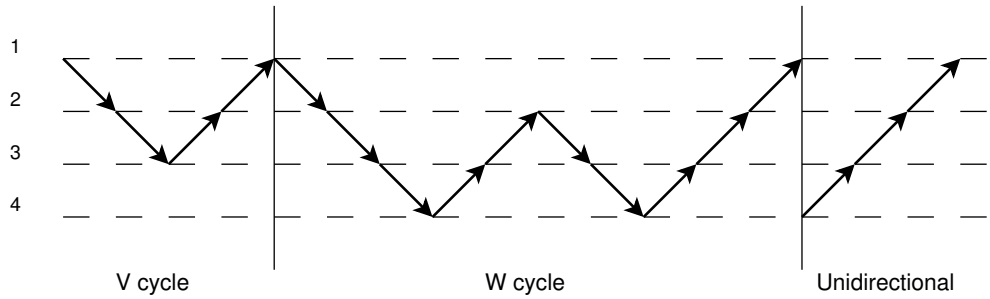
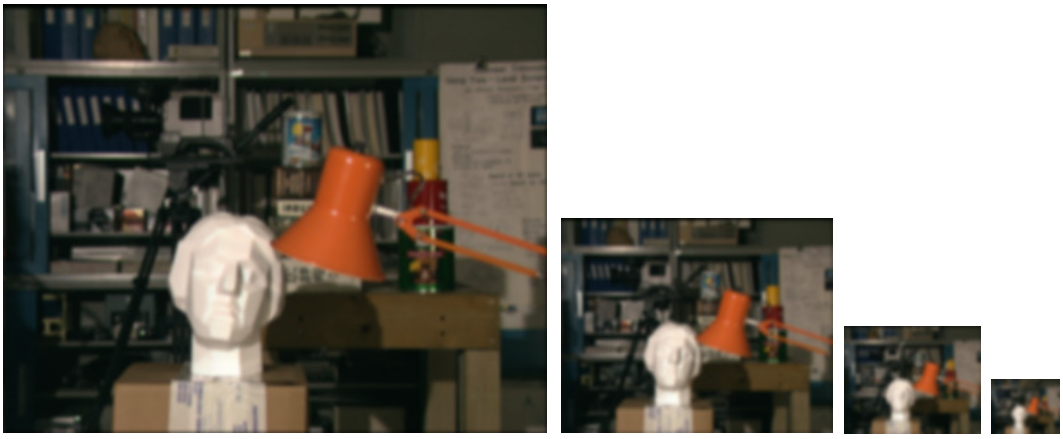


Figure 5.8: Multigrid V- and W-cycles and a unidirectional Full Multigrid (FMG) cycle. The unidirectional cycle is also known as coarse-to-fine scheme.

Multigrid schemes, where a solution from a coarse scale is used to initialise a finer scale, are also known as nested iteration or Full MultiGrid (FMG) schemes. Figure 5.8 depicts a unidirectional FMG cycle. Unidirectional means that the scheme advances towards finer scales without re-visiting the coarser scales again. The algorithms that we introduce in sections 5.6.1 and 5.6.2 for optical-flow are based on a unidirectional scheme. On the other hand, in the case of segmentation, see Section 5.7, we either perform a V- or a W-cycle. However, in our case we do not transfer the defect. Even in this case the convergence of the method benefits from using coarser discretisation of the coarser scales.

Our schemes are based on image pyramid, where the scale refers to a particular scale (size) of image in the image pyramid, as seen in Figure 5.9. In a multigrid framework, solving the optical-flow on a coarser (downscaled) image is equivalent to solving the problem using an approximation $\hat{A}x = b$ that is easier/faster to solve than the original problem $Ax = b$, as was explained earlier. Now, on the other hand, this kind of a coarse-to-fine processing is inherently connected with the linear scale-space framework (since we use linear filters to construct the pyramid). When we expect the displacement to be small, starting from an initial guess of $\mathbf{u}^0 = \mathbf{v}^0 = \mathbf{0}$ is fine. However, when the displacements are expected to be greater (e.g. in the case of the late linearization model), this initial guess might not be such a good one and we might get trapped in a local minima. By using a coarse-to-fine processing we initialise each finer scale with a solution from the previous coarse scale [3][2]. This way not only do we avoid getting trapped in a physically irrelevant minima, but also computational efficiency is greatly enhanced. A reader interested in FMG and/or other efficiency related issues in variational optical-flow calculation is pointed to [17][19][18].



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

Figure 5.9: Image pyramid of Tsukuba with a scale factor of 0.5: (a) scale 1; (b) scale 2; (c) scale 3; (d) scale 4.

5.4 Equations to be Solved

In order to make this section easier to follow, we re-introduce the Euler-Lagrange equations of the previously described models, therefore, hopefully making it unnecessary to jump back and forth. Specifically, we show how both the early- and the late-linearisation models for optical-flow, and the two-region level-set model can be solved. Since the stereo-disparity case is a simplified version of the optical-flow (i.e. displacements depending on one variable), we feel that it is sufficient to show how the optical-flow model can be solved, and an interested reader can work out the rest.

Optical-flow, early linearisation.
Equations (5.23) and (5.24) give the energy functional and the corresponding Euler-Lagrange equations for the early-linearisation version of the optical-flow (i.e. Horn&Schunck model [33]).

$$E(u,v) = \int_\Omega \sum_{k=1}^{K} \left(\underbrace{\left(\frac{\partial I_k}{\partial t} - \frac{\partial I_k}{\partial x} u - \frac{\partial I_k}{\partial y} v \right)^2}_{\text{data}} + \alpha \underbrace{\left(|\nabla u|^2 + |\nabla v|^2 \right)}_{\text{smoothness}} \right) \mathbf{dx} \tag{5.23}$$

where the sub-index k refers to the channels (e.g. R, G or B) of a vector valued image I_k, and α is the weight of the smoothness term.

$$\begin{aligned}\sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial t}-\frac{\partial I_k}{\partial x}u-\frac{\partial I_k}{\partial y}v\right)\frac{\partial I_k}{\partial x}+K\alpha DIV(\nabla u)=&0\\ \sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial t}-\frac{\partial I_k}{\partial x}u-\frac{\partial I_k}{\partial y}v\right)\frac{\partial I_k}{\partial y}+K\alpha DIV(\nabla v)=&0\end{aligned} \tag{5.24}$$

with reflecting boundary conditions $\partial_n u = 0$ and $\partial_n v = 0$, where n denotes the normal to the image boundary $\partial\Omega_h$. For more details, see Section 3.5.1.

Optical-flow, late linearisation.
Equations (5.25) and (5.26) give the energy functional and the corresponding Euler-Lagrange equations for the late-linearisation version of the optical-flow.

$$E(u,v)=\int_\Omega\sum_{k=1}^{K}\left(\underbrace{\Psi_D\left(\left(I_{k,1}-I^w_{k,0}\right)^2\right)}_{\text{data}}+\alpha\underbrace{\Psi_R\left(|\nabla u|^2+|\nabla v|^2\right)}_{\text{smoothness}}\right)\mathbf{dx} \tag{5.25}$$

where the sub-index k refers to the channels (e.g. R, G or B) of a vector valued image I_k, α is the weight of the smoothness term, $I_{k,t}=I(x,y,k,t)$ and $I^w_{k,t}=I(x+u,y+v,k,t)$ refers to a 'warped' image. Typical warping transformations are bilinear- or bicubic interpolation.

$$\begin{aligned}&(E_k)_D=(I_{k,1}-I^w_{k,0})^2=(I_{k,1}(x,y)-I_{k,0}(x+u,y+v))^2\\ &E_R=|\nabla u|^2+|\nabla v|^2\\ &\sum_{k=1}^{K}\Psi'_D\Big((E_k)_D\Big)\Big(I_{k,1}-I^w_{k,0}\Big)\frac{\partial I^w_{k,0}}{\partial x}+K\alpha DIV\Big(\Psi'_R(E_R)\nabla u\Big)=0\\ &\sum_{k=1}^{K}\Psi'_D\Big((E_k)_D\Big)\Big(I_{k,1}-I^w_{k,0}\Big)\frac{\partial I^w_{k,0}}{\partial y}+K\alpha DIV\Big(\Psi'_R(E_R)\nabla v\Big)=0\end{aligned} \tag{5.26}$$

Again, we use reflecting boundary conditions $\partial_n u = 0$ and $\partial_n v = 0$, where n denotes the normal to the image boundary $\partial\Omega_h$. Instead of writing $I_{k,0}(x+u,y+v)$, we typically use $I^w_{k,0}$ when referring to the 'warped' image. For more details, see Section 3.5.2.

Segmentation, 2-segment.

Equations (5.27) and (5.28) give the energy functional and the corresponding Euler-Lagrange equations for the level-set based two-region segmentation model [58]. For more details, see Section 4.4.4.5.

$$\begin{aligned} E(\Phi, \alpha_1) =& \mu \int_\Omega \delta(\Phi)|\nabla\Phi|\mathbf{dx} \\ &- \int_\Omega H(\Phi)\, log\, p_r\left(d|\alpha_1\right)\, \mathbf{dx} \\ &- \int_\Omega H(1-\Phi)\, log\, \overline{p_r}\left(d|\alpha_1\right)\, \mathbf{dx} \end{aligned} \tag{5.27}$$

where the first term is the the length of the boundary curve, and the second and the third terms are the cost of 'coding' the disparity values according to $p(d|\alpha_i)$ for each region.

$$\frac{\partial\Phi}{\partial t} = H'_\epsilon(\Phi)\left(log\, p_r\left(d|\alpha_1\right) - log\, \overline{p_r}\left(d|\alpha_1\right) + DIV\left(\frac{\nabla\Phi}{|\nabla\Phi|}\right)\right) \tag{5.28}$$

with reflecting boundary conditions $\partial_n\Phi = 0$ where n denotes the normal to the image boundary $\partial\Omega_h$.

From equations 5.24 and 5.26 it can be observed that the Euler-Lagrange equations for the optical-flow are of the space-type, i.e. we search directly for the steady-state. On the other hand, from Equation 5.28 it can be seen that the level-set formulation is of time-type. From the above formulations, it is clear that we need to approximate both first order derivatives and the divergence, DIV, operator in order to solve the equations.

5.5 Finite Difference Discretisation

Equations 5.24, 5.26, and 5.28 are continuous equations, and in order to be solve these, they need to be discretised first. To this end, we use finite-differences for approximating the derivatives. In other words, the continuous PDEs are replaced by their respective finite-difference discretised versions and we seek solutions to these. If the discretisation has been done 'carefully', then the solution to the discretised version approximates close enough a solution to the continuous version. On the other hand, incorrect discretisation can render the results completely useless.

5.5.1 Finite Difference Operators

Before going any further, we remind the reader of the 'standard' finite difference operators [49][51] which are used for approximating derivatives. Here, the function of interest, for which we want to find derivatives, is defined as $f(x, y) : \Omega \to \mathbb{R}$.

1. First order forward difference is given by:

$$D_x^+ f(x, y) = f_x^+(x, y) = \frac{f(x + \Delta x, y) - f(x, y)}{\Delta x} \tag{5.29}$$

2. First order backward difference is given by:

$$D_x^- f(x, y) = f_x^-(x, y) = \frac{f(x, y) - f(x - \Delta x, y)}{\Delta x} \tag{5.30}$$

3. First order central difference is given by:

$$D_x^0 f(x, y) = f_x^0(x, y) = \frac{f(x + \frac{1}{2}\Delta x, y) - f(x - \frac{1}{2}\Delta x, y)}{\Delta x} \tag{5.31}$$

4. Second order central difference is given by:

$$DD_x^0 f(x, y) = f_{xx}^0(x, y) = \frac{f(x + \Delta x, y) - 2f(x, y) + f(x - \Delta x, y)}{\Delta x^2} \tag{5.32}$$

where in f_x the sub-index (here x) indicates with respect to which variable the function has been differentiated. The above formulations can be simplified further if we assume a uniform grid with $\Delta x = \Delta y = 1$. The difference operators shown here approximate derivatives with respect to x. By switching the roles of x and y, corresponding difference operators for approximating derivatives with respect to y can be obtained easily. In the case of images containing several channels (e.g. RGB-images), derivatives are approximated separately for each channel.

Another way of describing the difference operators, is to think of them as being *correlation kernels* or *derivative filters* [66]. Correlation is an operation close to convolution, with the difference that in correlation the weight matrix (i.e. kernel) values are not reversed: the output of the operation is a weighted sum of the neighbourhood.

As it was already mentioned, choice of the difference operators has profound effect on the solution of the discretised PDEs and an inappropriate choice of these can render the results useless. In the case of the optical-flow and the stereo disparity, we have used

Simoncelli filters [66][24] to approximate both first-, and second order derivatives. In [65] and [66] Simoncelli shows that calculating derivatives of multi-dimensional signals using traditional forward-, backward- and central differences are often inaccurate, and proposes a bank of filters that produce more accurate results.

5.5.2 Discretization of DIV Operator

Now that we know how to approximate first and second order derivatives, we need a way to discretise the divergence, DIV, operator. This operator appears at each of the equations to be solved, i.e (5.24), (5.26), and (5.28). Conceptually we have two different cases, as given in (5.33).

$$\begin{aligned} &DIV\Big(\nabla f\Big) \\ &DIV\Big(g(x,y,t)\nabla f\Big) \end{aligned} \tag{5.33}$$

Here physical interpretation of the divergence is, in a sense, that of *diffusion* [53]. In the case of $DIV\,(\nabla f)$, diffusivity is the same in each direction, whereas in the case of $DIV\,(g(x,y,t)\,\nabla f)$, diffusivity is defined (or controlled) by the function g and is not necessarily the same in all the directions. Mathematically, for a differentiable vector function $F = Ui + Vj$, divergence operator is defined as in Equation (5.34).

$$DIV\Big(F\Big) = \frac{\partial U}{\partial x} + \frac{\partial V}{\partial y} \tag{5.34}$$

In other words, divergence is a sum of partial derivatives of a differentiable vector function. Therefore, in our case, we have the following.

$$\begin{aligned} DIV\Big(\nabla f\Big) &= \frac{\partial}{\partial x}(f_x) + \frac{\partial}{\partial y}(f_y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} = \Delta f \\ DIV\Big(g(x,y,t)\nabla f\Big) &= \frac{\partial}{\partial x}\big(g(x,y,t)f_x\big) + \frac{\partial}{\partial y}\big(g(x,y,t)f_y\big) = \nabla g \,\boldsymbol{.}\, \nabla f + g\Delta f \end{aligned} \tag{5.35}$$

Now, by simply using the finite differences introduced above, one way of discretising the divergence terms in (5.35) is using the central difference. First we apply the central difference and then the forward- and the backward differences for approximating the corresponding derivatives. The 'trick' here is to realise that $(f_x)(x + 0.5, y)$ is

actually the forward difference $D_x^+ f(x)$, while $(f_x)(x-0.5,y)$ is the backward difference $D_x^- f(x)$. Equations (5.36), and (5.37) show the discretisations for $DIV(\nabla f)$, and $DIV(g(x,y,t)\nabla f)$, respectively. This is the same discretisation as in the famous paper by Perona and Malik [53].

$$\begin{aligned}\frac{\partial}{\partial x}\Big(f_x\Big)(x,y)+\frac{\partial}{\partial y}\Big(f_y\Big)(x,y) =&\Big(f_x\Big)(x+0.5,y)-\Big(f_x\Big)(x-0.5,y)\\ &+\Big(f_y\Big)(x,y+0.5)-\Big(f_y\Big)(x,y-0.5)\\ =&f(x+1,y)-f(x,y)+f(x-1,y)-f(x,y)\\ &+f(x,y+1)-f(x,y)+f(x,y-1)-f(x,y)\\ =&\nabla_E f+\nabla_W f+\nabla_S f+\nabla_N f\end{aligned} \tag{5.36}$$

where $\nabla_{\{W,N,E,S\}} f$ denotes the difference in the directions given by W,N,E,S. As it was already mentioned, first we apply first order central difference on $f_x(x,y)$, and thus obtain $D_x^0 f_x(x,y)=\Big(f_x\Big)(x+0.5,y)-\Big(f_y\Big)(x-0.5,y)$. The rest should be clear.

$$\begin{aligned}\frac{\partial}{\partial x}\Big(gf_x\Big)(x,y)+\frac{\partial}{\partial y}\Big(gf_y\Big)(x,y) =&\Big(gf_x\Big)(x+0.5,y)-\Big(gf_x\Big)(x-0.5,y)\\ &+\Big(gf_y\Big)(x,y+0.5)-\Big(gf_y\Big)(x,y-0.5)\\ =&g(x+0.5,y)\Big(f(x+1,y)-f(x,y)\Big)\\ &+g(x-0.5,y)\Big(f(x-1,y)-f(x,y)\Big)\\ &+g(x,y+0.5)\Big(f(x,y+1)-f(x,y)\Big)\\ &+g(x,y-0.5)\Big(f(x,y-1)-f(x,y)\Big)\\ =&g_E\nabla_E f+g_W\nabla_W f+g_S\nabla_S f+g_N\nabla_N f\end{aligned} \tag{5.37}$$

where $g_{\{W,N,E,S\}}$ denotes the diffusivity in the directions given by W,N,E,S. As can be observed from Equation (5.37), we need to approximate the diffusivity between the pixels. A simple '2-point' approximation would be the average between neighbouring pixels, for example $g(x+0.5,y)=[g(x+1,y)+g(x,y)]/2$. In general, a more precise approximation leading to better results is a '6-point' approximation of Brox [13].

As it was already mentioned above, physical interpretation of the divergence is, in a sense, that of diffusion: the divergence operator introduces a 'connectivity' between the pixels. This simply means, as will be shown later on, that a solution at any position

(i, j) will depend on the solution at neighbouring positions. Because of this kind of a dependency of the solution between the adjacent positions, variational correspondence methods are said to be 'global'. This kind of connectivity is problematic at image borders, where we do not have neighbours anymore. In order to deal with this problem we use a scheme called eliminated boundary conditions, shown in Figure 5.10.

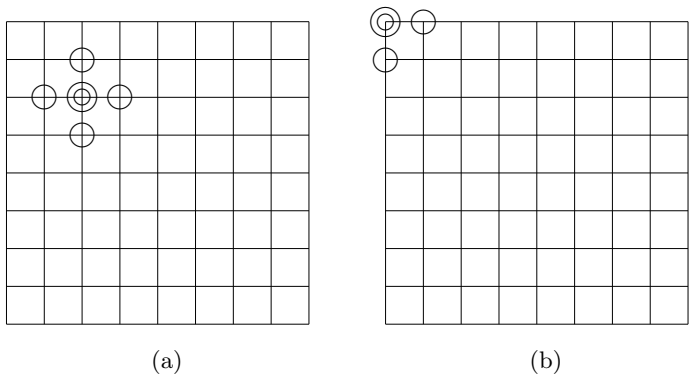


Figure 5.10: Double circle denotes the position of interest while simple circles are the neighbouring positions W,N,E,S; (b) shows the eliminated boundary conditions.

In the algorithmic descriptions that follow, $\Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}}$ denotes the diffusion weight between pixels $\mathcal{J}$ and $\mathcal{I}$.

5.6 Solving Optical-Flow

In this section we show how the optical-flow equations can be solved using a coarse-to-fine algorithm. We show how the related Euler-Lagrange equations are discretised, and linearised in the case of late linearisation, and derived into a form that allows constructing different solvers. We call this form, simply enough, the *solver form*. We introduce SOR prototype solvers, for both the cases, in programmatic form. The prototype solvers clearly indicate the needed input for solving the equations. From the corresponding solver forms we construct SOR-Jacobi, SOR-GS and ALR-GS solvers for both the early- and the late-linearisation cases.

5.6.1 Early Linearisation

In the early linearisation case we search for a solution at $l+1$ to the coupled PDEs given in Equation (5.38). For the coupled terms we use the last known solution (e.g. v^l when solving u^{l+1}).

$$\begin{aligned}\sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial t}-\frac{\partial I_k}{\partial x}u^{l+1}-\frac{\partial I_k}{\partial y}v^l\right)\frac{\partial I_k}{\partial x}+K\alpha\Delta u^{l+1}&=0\\ \sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial t}-\frac{\partial I_k}{\partial x}u^{l}-\frac{\partial I_k}{\partial y}v^{l+1}\right)\frac{\partial I_k}{\partial y}+K\alpha\Delta v^{l+1}&=0\end{aligned} \tag{5.38}$$

We plug-in the discretisation for the divergence term, use i,j to mark the pixel positions, and obtain the following:

$$\begin{aligned}&\sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial t}\frac{\partial I_k}{\partial x}\right)_{i,j}-\sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial x}\right)^2_{i,j}u^{l+1}_{i,j}-\sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial y}\frac{\partial I_k}{\partial x}\right)_{i,j}v^{l}_{i,j}\\ &\qquad+K\alpha\Big(u^{l+1}_{i+1,j}-2u^{l+1}_{i,j}+u^{l+1}_{i-1,j}+u^{l+1}_{i,j+1}-2u^{l+1}_{i,j}+u^{l+1}_{i,j-1}\Big)=0\\ &\sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial t}\frac{\partial I_k}{\partial y}\right)_{i,j}-\sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial x}\frac{\partial I_k}{\partial y}\right)_{i,j}u^{l}_{i,j}-\sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial y}\right)^2_{i,j}v^{l+1}_{i,j}\\ &\qquad+K\alpha\Big(v^{l+1}_{i+1,j}-2v^{l+1}_{i,j}+v^{l+1}_{i-1,j}+v^{l+1}_{i,j+1}-2v^{l+1}_{i,j}+v^{l+1}_{i,j-1}\Big)=0\end{aligned} \tag{5.39}$$

For each pixel position i,j we have equations similar to (5.39) that need to be solved.

Matrix format. While Equation (5.39) shows the discretised Euler-Lagrange equations for a pixel position i, j, we can write the system equations in matrix/vector format, covering the whole image, as given in (5.40). In Section 5.2 we mentioned that another useful way is to think of the discretised image as a vector $I \in \mathbb{R}^N$. Now, the components of the vector are $I_{\mathcal{I}}$ where $\mathcal{I} \in \{1, \ldots, N\}$ and N is the number of pixels in image. We use $\mathcal{I}, \mathcal{J}$ also to mark the positions in the system matrices $A_{\{1,2\}}$ given in (5.40). This is done in order to convey clearly the idea that the domains of the discretised images and the system matrices are different. If the domain of the discretised image is $\Omega_h : [1, m] \times [1, n]$ (discrete image with m columns and n rows), the system matrices $A_{\{1,2\}}$ are defined on $[1, m]^2 \times [1, n]^2$.

$$
\begin{aligned}
&\begin{cases} A_1 \mathbf{u}^{l+1} = \mathbf{b}_1 \\ A_2 \mathbf{v}^{l+1} = \mathbf{b}_2 \end{cases} \\
&A_1 = [a1_{\mathcal{I},\mathcal{J}}],\ A_2 = [a2_{\mathcal{I},\mathcal{J}}] \\
&a1_{\mathcal{I},\mathcal{J}} := \begin{cases} -K\alpha & [\mathcal{J} \in N(\mathcal{I})], \\ \sum\limits_{k=1}^{K} \left(\dfrac{\partial I_k}{\partial x} \right)_{\mathcal{I}}^2 + \sum\limits_{\substack{\mathcal{J} \in N^-(\mathcal{I}) \\ \mathcal{J} \in N^+(\mathcal{I})}} K\alpha & (\mathcal{J} = \mathcal{I}), \\ 0 & \text{(otherwise)} \end{cases} \\
&a2_{\mathcal{I},\mathcal{J}} := \begin{cases} -K\alpha & [\mathcal{J} \in N(\mathcal{I})], \\ \sum\limits_{k=1}^{K} \left(\dfrac{\partial I_k}{\partial y} \right)_{\mathcal{I}}^2 + \sum\limits_{\substack{\mathcal{J} \in N^-(\mathcal{I}) \\ \mathcal{J} \in N^+(\mathcal{I})}} K\alpha & (\mathcal{J} = \mathcal{I}), \\ 0 & \text{(otherwise)} \end{cases} \\
&\mathbf{b}_1 = \left[\sum_{k=1}^{K} \left(\frac{\partial I_k}{\partial t} \frac{\partial I_k}{\partial x} \right)_{\mathcal{I}} - \sum_{k=1}^{K} \left(\frac{\partial I_k}{\partial y} \frac{\partial I_k}{\partial x} \right)_{\mathcal{I}} v_{\mathcal{I}}^l \right]^T \\
&\mathbf{b}_2 = \left[\sum_{k=1}^{K} \left(\frac{\partial I_k}{\partial t} \frac{\partial I_k}{\partial y} \right)_{\mathcal{I}} - \sum_{k=1}^{K} \left(\frac{\partial I_k}{\partial x} \frac{\partial I_k}{\partial y} \right)_{\mathcal{I}} u_{\mathcal{I}}^l \right]^T
\end{aligned}
\tag{5.40}
$$

Equation (5.41) gives an example of how the system matrix A_1 would look like. Here C and N are block matrices that refer to the 'central' and the 'neighbouring' matrices, correspondingly.

$$
\begin{aligned}
A_1 &= \begin{bmatrix} C & N & 0 \\ N & C & N \\ 0 & N & C \end{bmatrix} \\
C &= \begin{bmatrix} \sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial x}\right)_{\mathcal{I}}^2 + \sum_{\substack{\mathcal{J}\in N^-(\mathcal{I}) \\ \mathcal{J}\in N^+(\mathcal{I})}} K\alpha & -K\alpha & 0 \\ -K\alpha & \sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial x}\right)_{\mathcal{I}}^2 + \sum_{\substack{\mathcal{J}\in N^-(\mathcal{I}) \\ \mathcal{J}\in N^+(\mathcal{I})}} K\alpha & -K\alpha \\ 0 & -K\alpha & \sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial x}\right)_{\mathcal{I}}^2 + \sum_{\substack{\mathcal{J}\in N^-(\mathcal{I}) \\ \mathcal{J}\in N^+(\mathcal{I})}} K\alpha \end{bmatrix} \\
N &= \begin{bmatrix} -K\alpha & 0 & 0 \\ 0 & -K\alpha & 0 \\ 0 & 0 & -K\alpha \end{bmatrix}
\end{aligned}
\tag{5.41}
$$

From (5.40) we can see how the the matrix A_1 looks like for a 3×3 size image: it is a block tridiagonal square matrix, of size 9×9, that has non-zero components only on the main diagonal and on the diagonals adjacent to this. Therefore, unless the image is very small, it is infeasible to solve the system by inverting $A_{\{1,2\}}$ directly. Instead, we search for a solution using iterative methods, such as those introduced in Section 5.3.

Solver form. We re-order Equation (5.39) so that $u_{i,j}^{l+1}$ and $v_{i,j}^{l+1}$ are on the left-hand side, while the rest of the terms are on the right-hand side, and obtain the following:

$$
\begin{aligned}
u_{i,j}^{l+1}\Big(\sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial x}\right)_{i,j}^2 + 4K\alpha\Big) &= \sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial t}\frac{\partial I_k}{\partial x}\right)_{i,j} - \sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial y}\frac{\partial I_k}{\partial x}\right)_{i,j} v_{i,j}^{l} \\
&\quad + K\alpha u_{i-1,j}^{l+1} + K\alpha u_{i+1,j}^{l+1} + K\alpha u_{i,j-1}^{l+1} + K\alpha u_{i+1,j}^{l+1} \\
v_{i,j}^{l+1}\Big(\sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial y}\right)_{i,j}^2 + 4K\alpha\Big) &= \sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial t}\frac{\partial I_k}{\partial y}\right)_{i,j} - \sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial x}\frac{\partial I_k}{\partial y}\right)_{i,j} u_{i,j}^{l} \\
&\quad + K\alpha v_{i-1,j}^{l+1} + K\alpha v_{i+1,j}^{l+1} + K\alpha v_{i,j-1}^{l+1} + K\alpha v_{i+1,j}^{l+1}
\end{aligned}
\tag{5.42}
$$

This is the 'basic' form from which different solvers can be constructed easily.

Prototype solver. An algorithmic description of the SOR with a generic solver for the early linearisation model is given in (5.43). A new approximation is a weighted combination of the previous and the new solution.

$$\begin{cases} u^{l+1} = (1-\omega)u^l + \omega SOLVER_u\left(u^l,\, v^l,\, \dfrac{\partial I}{\partial t},\, \dfrac{\partial I}{\partial x},\, \dfrac{\partial I}{\partial y}\right); \\ v^{l+1} = (1-\omega)v^l + \omega SOLVER_v\left(u^l,\, v^l,\, \dfrac{\partial I}{\partial t},\, \dfrac{\partial I}{\partial x},\, \dfrac{\partial I}{\partial y}\right); \end{cases} \tag{5.43}$$

where $SOLVER_{\{u,v\}}(.)$ can be any solver for linear system of equations that calculates a new solution at $l+1$. Subscript u, v defines which variable is solved. The ω is a parameter that defines how much weight is given to the old solution with respect to the new solution. As can be observed from the prototype solver, we do not need to construct the system matrices $A_{\{1,2\}}$ in order to solve the system of equations. In what follows, we give an algorithmic description of a coarse-to-fine algorithm that can be used for solving the equations given in (5.42).

5.6.1.1 Coarse-To-Fine Algorithm

Algorithm 3 gives a full description of a coarse-to-fine algorithm for the early linearisation case. Instead of iterating on each scale until a steady-state is reached, we run the solver a 'reasonable' number of times before changing the scale. In Table 5.1 we show typical values that can be used with the algorithm.

Algorithm 3 Coarse-to-Fine algorithm for early linearisation. Inputs are I_0 and I_1 (i.e. images at time $t = 0$ and $t = 1$), number of scales scl and scaling factor $sclFactor$.

INPUT : I_0, I_1, scl, $sclFactor$
OUTPUT : (u, v)
//Set u and v to zero
$u = 0$, $v = 0$;
//Create image pyramid
$[Iscl_0\{\}, Iscl_1\{\}] = pyramid(I_0, I_1, scl, sclFactor)$;
//This is the coarse-to-fine loop
while $s = scl : -1 : 1$ **do**
  $I_0 = Iscl_0\{s\}$, $I_1 = Iscl_1\{s\}$;
  Approximate derivatives for I_0 and I_1: $\frac{\partial I_k}{\partial t}, \frac{\partial I_{k,0}}{\partial x}, \frac{\partial I_{k,0}}{\partial y}$
  //Solve for new u and v
  $[u, v] = SOLVER(u, v, nLoops, \frac{\partial I_k}{\partial t}, \frac{\partial I_{k,0}}{\partial x}, \frac{\partial I_{k,0}}{\partial y})$;
  //Interpolate (prolongate) solution
  if $s - 1 > 0$ **then**
    $[u, v] = prolongate(u, v, sclFactor)$;
  end if
end while

For the actual solver we can use any of the solvers introduced in sections 5.6.1.2, 5.6.1.3 and 5.6.1.4.

Table 5.1: Typical parameters: coarse-to-fine algorithm for early linearisation (m and n refer to number of columns and rows).

Common	
scl	downscale until image size $m < 20$ or $n < 20$
$sclFactor$	0.5 - 0.75
ω	1.5 - 1.9
Solver iterations	
SOR-GS	50
SOR-GS-ALR	20

5.6.1.2 SOR-Jacobi

From Equation (5.44) we can calculate a new solution at step $l+1$ for position i, j.

$$
\begin{aligned}
u_{\mathcal{I}}^{l+1} =& (1-\omega) u_{\mathcal{I}}^{l} \\
&+ \omega \frac{\sum_{k=1}^{K} \left(\frac{\partial I_k}{\partial t} \frac{\partial I_k}{\partial x} \right)_{\mathcal{I}} - \sum_{k=1}^{K} \left(\frac{\partial I_k}{\partial y} \frac{\partial I_k}{\partial x} \right)_{\mathcal{I}} v_{\mathcal{I}}^{l} + K\alpha \left(\sum_{\mathcal{J} \in N^-(\mathcal{I})} u_{\mathcal{J}}^{l} + \sum_{\mathcal{J} \in N^+(\mathcal{I})} u_{\mathcal{J}}^{l} \right)}{\sum_{k=1}^{K} \left(\frac{\partial I_k}{\partial x} \right)_{\mathcal{I}}^{2} + \alpha \sum_{\substack{\mathcal{J} \in N^+(\mathcal{I}) \\ \mathcal{J} \in N^-(\mathcal{I})}} K} \\
v_{\mathcal{I}}^{l+1} =& (1-\omega) v_{\mathcal{I}}^{l} \\
&+ \omega \frac{\sum_{k=1}^{K} \left(\frac{\partial I_k}{\partial t} \frac{\partial I_k}{\partial y} \right)_{\mathcal{I}} - \sum_{k=1}^{K} \left(\frac{\partial I_k}{\partial x} \frac{\partial I_k}{\partial y} \right)_{\mathcal{I}} u_{\mathcal{I}}^{l} + K\alpha \left(\sum_{\mathcal{J} \in N^-(\mathcal{I})} v_{\mathcal{J}}^{l} + \sum_{\mathcal{J} \in N^+(\mathcal{I})} v_{\mathcal{J}}^{l} \right)}{\sum_{k=1}^{K} \left(\frac{\partial I_k}{\partial y} \right)_{\mathcal{I}}^{2} + \alpha \sum_{\substack{\mathcal{J} \in N^+(\mathcal{I}) \\ \mathcal{J} \in N^-(\mathcal{I})}} K}
\end{aligned}
\tag{5.44}
$$

5.6.1.3 SOR-GS

From Equation (5.45), we can calculate a new solution at step $l+1$ for position i, j. The Gauss-Seidel method advances so that new solutions are taken into account as soon they are available. Depending on what order the pixels are traversed (row- or column-wise), this affects which neighbourhood positions have a new solution (i.e. at $l+1$) and which have a solution from the last iteration cycle (i.e. at l) in the last term on the right-hand side.

$$
\begin{aligned}
u_{\mathcal{I}}^{l+1} =& (1-\omega)u_{\mathcal{I}}^{l} \\
&+\omega \frac{\sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial t}\frac{\partial I_k}{\partial x}\right)_{\mathcal{I}} - \sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial y}\frac{\partial I_k}{\partial x}\right)_{\mathcal{I}} v_{\mathcal{I}}^{l} + K\alpha\left(\sum_{\mathcal{J}\in N^-(\mathcal{I})} u_{\mathcal{J}}^{l+1} + \sum_{\mathcal{J}\in N^+(\mathcal{I})} u_{\mathcal{J}}^{l}\right)}{\sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial x}\right)_{\mathcal{I}}^{2} + \alpha \sum_{\substack{\mathcal{J}\in N^+(\mathcal{I})\\ \mathcal{J}\in N^-(\mathcal{I})}} K} \\
v_{\mathcal{I}}^{l+1} =& (1-\omega)v_{\mathcal{I}}^{l} \\
&+\omega \frac{\sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial t}\frac{\partial I_k}{\partial y}\right)_{\mathcal{I}} - \sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial x}\frac{\partial I_k}{\partial y}\right)_{\mathcal{I}} u_{\mathcal{I}}^{l} + K\alpha\left(\sum_{\mathcal{J}\in N^-(\mathcal{I})} v_{\mathcal{J}}^{l+1} + \sum_{\mathcal{J}\in N^+(\mathcal{I})} v_{\mathcal{J}}^{l}\right)}{\sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial y}\right)_{\mathcal{I}}^{2} + \alpha \sum_{\substack{\mathcal{J}\in N^+(\mathcal{I})\\ \mathcal{J}\in N^-(\mathcal{I})}} K}
\end{aligned}
\tag{5.45}
$$

5.6.1.4 GS-ALR

Here we show how the GS-ALR solver is constructed for the early linearisation case. As it was already mentioned in Section 5.3.4, the algorithm consists of two sweeps, namely the forward- and the backward-sweeps, as given in equations (5.46) and (5.47). The algorithm introduced here can be converted into SOR-GS-ALR easily, as is shown in Algorithm 2. The only thing that needs to be explained is what the $a_{\mathcal{I}}$, $b_{\mathcal{I}}$, $c_{\mathcal{I}}$ and $d_{\mathcal{I}}$ are.

$$
\begin{aligned}
c'_{\mathcal{I}} &= \begin{cases} \frac{c_1}{b_1} & , \mathcal{I}=1 \\ \frac{c_{\mathcal{I}}}{b_{\mathcal{I}} - c'_{\mathcal{I}-1}a_{\mathcal{I}}} & , \mathcal{I}=2,3,...,N-1 \end{cases} \\
d'_{\mathcal{I}} &= \begin{cases} \frac{d_1}{b_1} & , \mathcal{I}=1 \\ \frac{d_{\mathcal{I}} - d'_{\mathcal{I}-1}a_{\mathcal{I}}}{b_{\mathcal{I}} - c'_{\mathcal{I}-1}a_{\mathcal{I}}} & , \mathcal{I}=2,3,...,N \end{cases}
\end{aligned}
\tag{5.46}
$$

$$
\begin{aligned}
x_N &= d'_N \\
x_{\mathcal{I}} &= d'_{\mathcal{I}} - c'_{\mathcal{I}}x_{\mathcal{I}+1} \quad ,\mathcal{I}=N-1,N-2,...,1
\end{aligned}
\tag{5.47}
$$

Because in this case the diffusion weights are the same throughout the image, it is enough to show how the algorithm advances in column wise ordering. The only

difference in row wise ordering is the order of traversing the image. Equation (5.48) defines the elements when solving for u.

$$
\begin{aligned}
a_{\mathcal{I}} &= -K\alpha \\
b_{\mathcal{I}} &= \sum_{k=1}^{K} \left(\frac{\partial I_k}{\partial x} \right)_{\mathcal{I}}^2 + \alpha \sum_{\mathcal{J} \in N(\mathcal{I})} K \\
c_{\mathcal{I}} &= -K\alpha \\
d_{\mathcal{I}} &= \sum_{k=1}^{K} \left(\frac{\partial I_k}{\partial t} \frac{\partial I_k}{\partial x} \right)_{\mathcal{I}} - \sum_{k=1}^{K} \left(\frac{\partial I_k}{\partial y} \frac{\partial I_k}{\partial x} \right)_{\mathcal{I}} v_{\mathcal{I}}^l \\
&\quad + K\alpha \left(\sum_{\mathcal{J} \in N^-(\mathcal{I})} u_{\mathcal{J}}^{l+1} + \sum_{\mathcal{J} \in N^+(\mathcal{I})} u_{\mathcal{J}}^l \right)
\end{aligned}
\tag{5.48}
$$

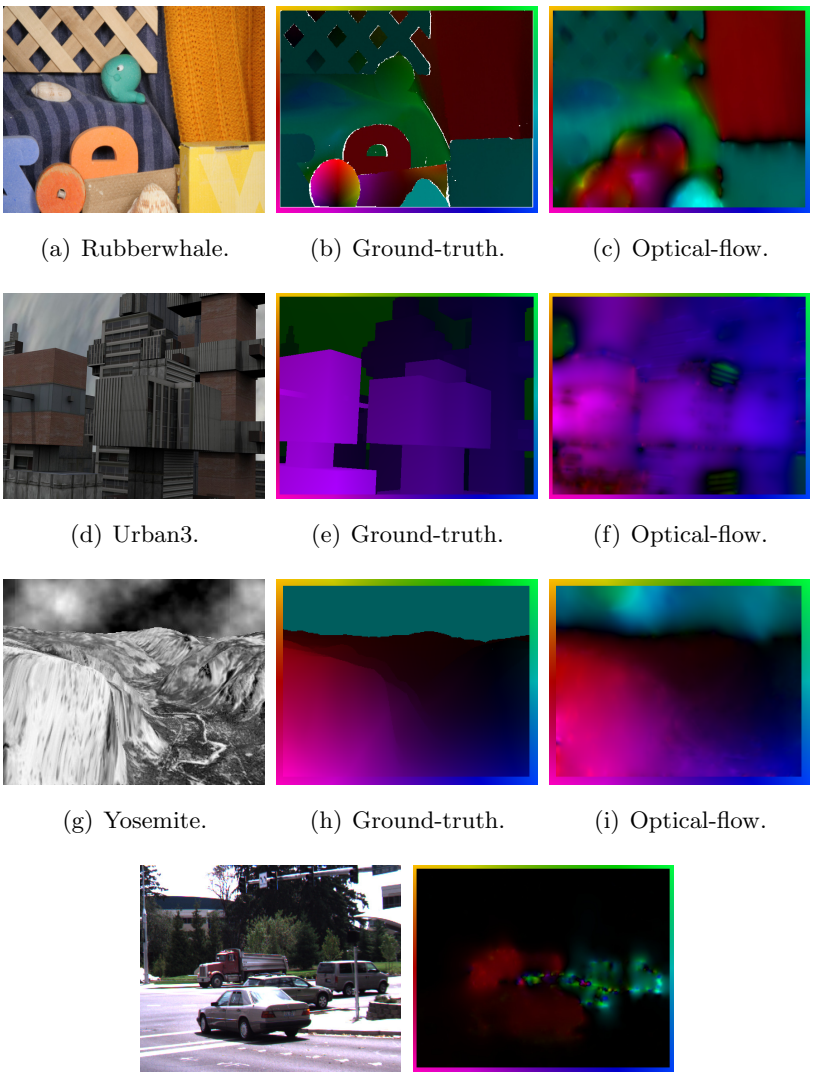
Equation (5.49) defines the elements when solving for v.

$$
\begin{aligned}
a_{\mathcal{I}} &= -K\alpha \\
b_{\mathcal{I}} &= \sum_{k=1}^{K} \left(\frac{\partial I_k}{\partial y} \right)_{\mathcal{I}}^2 + \alpha \sum_{\mathcal{J} \in N(\mathcal{I})} K \\
c_{\mathcal{I}} &= -K\alpha \\
d_{\mathcal{I}} &= \sum_{k=1}^{K} \left(\frac{\partial I_k}{\partial t} \frac{\partial I_k}{\partial x} \right)_{\mathcal{I}} - \sum_{k=1}^{K} \left(\frac{\partial I_k}{\partial y} \frac{\partial I_k}{\partial x} \right)_{\mathcal{I}} u_{\mathcal{I}}^l \\
&\quad + K\alpha \left(\sum_{\mathcal{J} \in N^-(\mathcal{I})} v_{\mathcal{J}}^{l+1} + \sum_{\mathcal{J} \in N^+(\mathcal{I})} v_{\mathcal{J}}^l \right)
\end{aligned}
\tag{5.49}
$$

5.6.1.5 Results

In the following we show some results for the early linearisation Horn&Schunck model (with homogeneous diffusion term) using test images from Middlebury[1]. We would like to remind the reader, that robust error functions can be incorporated also in the early linearisation model, therefore, improving the results.

[1] `http://vision.middlebury.edu/flow/data/`



(a) Rubberwhale. (b) Ground-truth. (c) Optical-flow.

(d) Urban3. (e) Ground-truth. (f) Optical-flow.

(g) Yosemite. (h) Ground-truth. (i) Optical-flow.

(j) Dumptruck. (k) Optical-flow.

Figure 5.11: Early linearisation results.

5.6.2 Late Linearisation

The late linearisation case with robust error function is basically similar to the early linearisation with some important differences. Firstly, there are two sources of non-linearities: the robust error function(s) and the constancy term itself. Secondly, the energy functional of the model, given in Equation (3.9), might be non-convex [3] due to the non-linear data term. This means, that the energy functional in question can be

multi-modal or, in other words, that several local minima might exist [3][2]. Therefore, unless we do an exhaustive search in the argument space of the energy functional, we cannot guarantee finding the global minimum, but might get trapped in a local minimum. In order to deal with the problem of the local minima, we use a continuation method similar to the Graduated Non-Convexity (GNC) [12]. We start searching for a possible solution in a simplified version of the original problem, and use this solution to initialise the search for a problem that is more alike with the original problem. By iteratively repeating this process, we hope to avoid getting stuck in a local minima and find a relevant minimiser. On the other hand, we deal with the non-linearities using two fixed point loops, as is explained in the following. We use sub-indices l and m to mark the iteration cycles but, unfortunately, this makes the notation somewhat cluttered. However, we give an algorithmic description of the solver later on.

First fixed point iteration loop. Purpose of the first fixed point iteration is to linearise the constancy assumption. We search for the new solution at $l+1$ to the coupled PDEs described by equations (5.50), and (5.51). However, this time the formulation is *semi-implicit*: we use the last known solution at l to approximate the partial derivatives $\frac{\partial}{\partial x} I_{k,0}^{w(l)}$ and $\frac{\partial}{\partial y} I_{k,0}^{w(l)}$. Therefore, this scheme is semi-implicit in the data term and fully implicit in the smoothness term. What we are actually trying to minimise is the difference between the $I_{k,1}$ and the warped image $I_{k,0}^{w(l+1)}$. In a sense, this difference is also the temporal derivative and, therefore, we mark it with $\frac{\partial}{\partial t} I_k^{l+1}$, thus making it easier to see the similarity between the late- and the early-linearisation versions.

$$\begin{aligned}
\left(E_k^{l+1}\right)_D &= \left(I_{k,1} - I_{k,0}^{w(l+1)}\right)^2 = \left(I_{k,1} - I_{k,0}\left(x + u^{l+1}, y + v^{l+1}\right)\right)^2 \\
E_R^{l+1} &= |\nabla u^{l+1}|^2 + |\nabla v^{l+1}|^2 \\
\frac{\partial I_k^{l+1}}{\partial t} &= \left(I_{k,1} - I_{k,0}^{w(l+1)}\right) = \left(I_{k,1}(x,y) - I_{k,0}(x + u^{l+1}, y + v^{l+1})\right)
\end{aligned} \tag{5.50}$$

$$\begin{aligned}
\sum_{k=1}^{K} \Psi'_D\left(\left(E_k^{l+1}\right)_D\right)\left(\frac{\partial I_k^{l+1}}{\partial t}\right)\left(\frac{\partial I_{k,0}^{w(l)}}{\partial x}\right) + K\alpha DIV\left(\Psi'_R(E_R^{l+1})\nabla u^{l+1}\right) = 0 \\
\sum_{k=1}^{K} \Psi'_D\left(\left(E_k^{l+1}\right)_D\right)\left(\frac{\partial I_k^{l+1}}{\partial t}\right)\left(\frac{\partial I_{k,0}^{w(l)}}{\partial y}\right) + K\alpha DIV\left(\Psi'_R(E_R^{l+1})\nabla v^{l+1}\right) = 0
\end{aligned} \tag{5.51}$$

We approximate the warped term $I_{k,0}^{w(l+1)}$ using first order Taylor expansion as given in Equation (5.52)

$$\begin{aligned} I_{k,0}^{w(l+1)} = I_{k,0}(x+u^{l+1}, y+v^{l+1}) =& I_{k,0}(x+u^l, y+v^l) \\ &+ \frac{\partial I_{k,0}\left(x+u^l, y+v^l\right)}{\partial x} du^l \\ &+ \frac{\partial I_{k,0}\left(x+u^l, y+v^l\right)}{\partial y} dv^l \\ =& I_{k,0}^{w(l)} + \frac{\partial I_{k,0}^{w(l)}}{\partial x} du^l + \frac{\partial I_{k,0}^{w(l)}}{\partial y} dv^l \end{aligned} \tag{5.52}$$

where the new solutions are $u^{l+1} = u^l + du^l$ and $v^{l+1} = v^l + dv^l$. We plug-in the linearised version of the warped term in $\frac{\partial I_k^{l+1}}{\partial t}$, and obtain the following:

$$\frac{\partial I_k^{l+1}}{\partial t} = \Big(\underbrace{I_{k,1} - I_{k,0}^{w(l)}}_{\frac{\partial I_k^l}{\partial t}} - \frac{\partial I_{k,0}^{w(l)}}{\partial x} du^l - \frac{\partial I_{k,0}^{w(l)}}{\partial y} dv^l \Big), \tag{5.53}$$

where we denote $\left(I_{k,1} - I_{k,0}^{w(l)}\right)$ with $\frac{\partial I_k^l}{\partial t}$ since, as previously, this is the temporal derivative, but also it is the difference that we are trying to minimise. With these in place, we can define the data- and smoothness terms as follows:

$$\begin{aligned} \left(E_k^l\right)_D &= \left(\frac{\partial I_k^l}{\partial t} - \frac{\partial I_{k,0}^{w(l)}}{\partial x} du^l - \frac{\partial I_{k,0}^{w(l)}}{\partial y} dv^l \right)^2 \\ E_R^l &= |\nabla(u^l + du^l)|^2 + |\nabla(v^l + dv^l)|^2 \\ \frac{\partial I_k^l}{\partial t} &= \left(I_{k,1} - I_{k,0}^{w(l)}\right). \end{aligned} \tag{5.54}$$

Euler-Lagrange equations with linearised constancy term are as follows:

$$
\begin{aligned}
\sum_{k=1}^{K} \Psi'_D\left(\left(E_k^l\right)_D\right)\left(\frac{\partial I_k^l}{\partial t} - \frac{\partial I_{k,0}^{w(l)}}{\partial x} du^l - \frac{\partial I_{k,0}^{w(l)}}{\partial y} dv^l\right)\frac{\partial I_{k,0}^{w(l)}}{\partial x} \\
+K\alpha DIV\left(\Psi'_R(E_R^l)\nabla(u^l + du^l)\right) = 0 \\
\sum_{k=1}^{K} \Psi'_D\left(\left(E_k^l\right)_D\right)\left(\frac{\partial I_k^l}{\partial t} - \frac{\partial I_{k,0}^{w(l)}}{\partial x} du^l - \frac{\partial I_{k,0}^{w(l)}}{\partial y} dv^l\right)\frac{\partial I_{k,0}^{w(l)}}{\partial y} \\
+K\alpha DIV\left(\Psi'_R(E_R^l)\nabla(v^l + dv^l)\right) = 0
\end{aligned}
\tag{5.55}
$$

As it was already mentioned, a new solution is obtained by solving for du and dv. For the warped terms we use the last known solution at l: first we warp $I_{k,0}^{w(l)}$ (using the known solution l) and then approximate the derivatives. By looking at Equation 5.55, we can see that the only nonlinear terms that we are left with are the influence/penalty functions $\Psi'_D\left(\left(E_k^l\right)_D\right)$ and $\Psi'_R\left(E_R^l\right)$.

Second fixed point iteration loop. In order to deal with the non-linearities arising from the robust error functions, we use the fixed point iteration method again. This time we use the last known solution for du and dv in the influence functions (i.e. du^m and dv^m), and search for a new solution at $m+1$. This scheme is also called lagged diffusivity fixed point method. The strategy to deal with the coupling is the same as in the early-linearisation case: we use the last known solution for the linked terms.

$$
\begin{aligned}
\left(E_k^{l,m}\right)_D &= \left(\frac{\partial I_k^l}{\partial t} - \frac{\partial I_{k,0}^{w(l)}}{\partial x} du^{l,m} - \frac{\partial I_{k,0}^{w(l)}}{\partial y} dv^{l,m}\right)^2 \\
E_R^{l,m} &= |\nabla(u^l + du^{l,m})|^2 + |\nabla(v^l + dv^{l,m})|^2 \\
\frac{\partial I_k^l}{\partial t} &= \left(I_{k,1} - I_{k,0}^{w(l)}\right)
\end{aligned}
\tag{5.56}
$$

$$
\begin{aligned}
&\sum_{k=1}^{K} \Psi'\Big((E_k^{l,m})_D\Big)\left(\frac{\partial I_k^l}{\partial t} - \frac{\partial I_{k,0}^{w(l)}}{\partial x} du_{i,j}^{l,m+1} - \frac{\partial I_{k,0}^{w(l)}}{\partial y} dv_{i,j}^{l,m}\right)\frac{\partial I_{k,0}^{w(l)}}{\partial x} \\
&\qquad + K\alpha DIV\Big(\Psi'\big(E_R^{l,m}\big)\nabla(u_{i,j}^l + du_{i,j}^{l,m+1})\Big) = 0 \\
&\sum_{k=1}^{K} \Psi'\Big((E_k^{l,m})_D\Big)\left(\frac{\partial I_k^l}{\partial t} - \frac{\partial I_{k,0}^{w(l)}}{\partial x} du_{i,j}^{l,m} - \frac{\partial I_{k,0}^{w(l)}}{\partial y} dv_{i,j}^{l,m+1}\right)\frac{\partial I_{k,0}^{w(l)}}{\partial y} \\
&\qquad + K\alpha DIV\Big(\Psi'\big(E_R^{l,m}\big)\nabla(v_{i,j}^l + dv_{i,j}^{l,m+1})\Big) = 0
\end{aligned}
\tag{5.57}
$$

Solver form. Now we plug-in the discretisation for the divergence term (here we use $\Psi'\big(E_R^{l,m}\big)_{\{W,N,E,S\}}$ to denote the diffusion weights), as given in Equation (5.37), and obtain the following:

$$
\begin{aligned}
&\sum_{k=1}^{K} \Psi'\Big((E_k^{l,m})_D\Big)\left(\frac{\partial I_k^l}{\partial t}\frac{\partial I_{k,0}^{w(l)}}{\partial x} - \Big(\frac{\partial I_{k,0}^{w(l)}}{\partial x}\Big)^2 du_{i,j}^{l,m+1} - \frac{\partial I_{k,0}^{w(l)}}{\partial y}\frac{\partial I_{k,0}^{w(l)}}{\partial x} dv_{i,j}^{l,m}\right) \\
&\qquad + K\alpha\Psi'\big(E_R^{l,m}\big)_N\Big(u_{i-1,j}^l - u_{i,j}^l + du_{i-1,j}^{l,m+1} - du_{i,j}^{l,m+1}\Big) \\
&\qquad + K\alpha\Psi'\big(E_R^{l,m}\big)_S\Big(u_{i+1,j}^l - u_{i,j}^l + du_{i+1,j}^{l,m+1} - du_{i,j}^{l,m+1}\Big) \\
&\qquad + K\alpha\Psi'\big(E_R^{l,m}\big)_W\Big(u_{i,j-1}^l - u_{i,j}^l + du_{i,j-1}^{l,m+1} - du_{i,j}^{l,m+1}\Big) \\
&\qquad + K\alpha\Psi'\big(E_R^{l,m}\big)_E\Big(u_{i,j+1}^l - u_{i,j}^l + du_{i,j+1}^{l,m+1} - du_{i,j}^{l,m+1}\Big) = 0 \\
&\sum_{k=1}^{K} \Psi'\Big((E_k^{l,m})_D\Big)\left(\frac{\partial I_k^l}{\partial t}\frac{\partial I_{k,0}^{w(l)}}{\partial y} - \frac{\partial I_{k,0}^{w(l)}}{\partial x}\frac{\partial I_{k,0}^{w(l)}}{\partial y} du_{i,j}^{l,m} - \Big(\frac{\partial I_{k,0}^{w(l)}}{\partial y}\Big)^2 dv_{i,j}^{l,m+1}\right) \\
&\qquad + K\alpha\Psi'\big(E_R^{l,m}\big)_N\Big(v_{i-1,j}^l - v_{i,j}^l + dv_{i-1,j}^{l,m+1} - dv_{i,j}^{l,m+1}\Big) \\
&\qquad + K\alpha\Psi'\big(E_R^{l,m}\big)_S\Big(v_{i+1,j}^l - v_{i,j}^l + dv_{i+1,j}^{l,m+1} - dv_{i,j}^{l,m+1}\Big) \\
&\qquad + K\alpha\Psi'\big(E_R^{l,m}\big)_W\Big(v_{i,j-1}^l - v_{i,j}^l + dv_{i,j-1}^{l,m+1} - dv_{i,j}^{l,m+1}\Big) \\
&\qquad + K\alpha\Psi'\big(E_R^{l,m}\big)_E\Big(v_{i,j+1}^l - v_{i,j}^l + dv_{i,j+1}^{l,m+1} - dv_{i,j}^{l,m+1}\Big) = 0
\end{aligned}
\tag{5.58}
$$

Equations in (5.58) are linear with respect to the arguments du and dv, and serve for the purpose of constructing different kinds of solvers, like was shown previously for the early linearisation.

Prototype solver. An algorithmic description of a SOR with a generic solver for the late linearisation scheme is given in 5.59. In order to construct an iterative solver, we

need to introduce a new iteration loop that we denote with the index n. The first two iteration loops are due to the linearisation of the data term (index l) and due to the non-linearities arising from the influence/penalty terms (index m). The question of how many iteration cycles per loop are needed has not been answered. In theory, for the lagged diffusivity fixed point method to converge, the solver should be iterated as many cycles as needed for the solution to be found (i.e. until a steady state is reached), before changing the fixed point (i.e. approximating the influence functions again). These steps would then be repeated until the solution would be not change anymore. However, we prefer to use a variant called inexact lagged diffusivity fixed point scheme, where we iterate the solver a few cycles and then change the fixed point. With this approach, we have obtained satisfactory results while considerably reducing the processing time.

$$\begin{cases} du^{l,m,n+1} = (1-\omega)du^{l,m,n} + \omega SOLVER_u\Bigg(\begin{array}{l} u^{l,m},\, v^{l,m},\, du^{l,m,n},\, dv^{l,m}, \\ \frac{\partial I_k^l}{\partial t},\, \frac{\partial I_{k,0}^{w(l)}}{\partial x},\, \frac{\partial I_{k,0}^{w(l)}}{\partial y}, \\ \Psi'\Big((E_k^{l,m})_D\Big), \\ \Psi'\left(E_R^{l,m}\right)_{\{W,N,E,S\}} \end{array} \Bigg); \\ dv^{l,m,n+1} = (1-\omega)dv^{l,m,n} + \omega SOLVER_v\Bigg(\begin{array}{l} u^{l,m},\, v^{l,m},\, du^{l,m,n},\, dv^{l,m}, \\ \frac{\partial I_k^l}{\partial t},\, \frac{\partial I_{k,0}^{w(l)}}{\partial x},\, \frac{\partial I_{k,0}^{w(l)}}{\partial y}, \\ \Psi'\Big((E_k^{l,m})_D\Big), \\ \Psi'\left(E_R^{l,m}\right)_{\{W,N,E,S\}} \end{array} \Bigg); \end{cases} \tag{5.59}$$

$SOLVER_{\{u,v\}}(.)$ calculates a new solution at $n+1$. As in the early linearisation case, we do not need to construct the system matrices in order to solve the system of equations.

5.6.2.1 Coarse-To-Fine Algorithm

Here we give a full description of a coarse-to-fine algorithm, with warping, for the late linearisation case. Table 5.2 shows typical parameters used with the algorithm.

Algorithm 4 Coarse-to-Fine algorithm for late linearisation. Inputs are I_0 and I_1 (i.e. images at time $t = 0$ and $t = 1$), number of scales scl and scaling factor $sclFactor$.

INPUT : I_0, I_1, scl, $sclFactor$
OUTPUT : (u, v)
//Set u and v to zero
$u = 0$, $v = 0$;
//Create image pyramid
$[Iscl_0\{\}\, Iscl_1\{\}] = pyramid(I_0, I_1, scl, sclFactor)$;
//Coarse-to-fine loop
while $s = scl : -1 : 1$ **do**
  $I_0 = Iscl_0\{s\}$, $I_1 = Iscl_1\{s\}$;
  //Warping loop
  while $fstLoop$ **do**
    //Warp image as per u and v
    $I^w_{k,0} = warp(I_{k,0}, u, v)$;
    Approximate derivatives for I^w_0 and I_1: $\frac{\partial I_k}{\partial t} = I_{k,1} - I^w_{k,0}$, $\frac{\partial I^w_{k,0}}{\partial x}$, $\frac{\partial I^w_{k,0}}{\partial y}$
    //Reset du and dv
    $du = 0$, $dv = 0$
    //Fixed-point loop due to the robust error functions
    while $sndLoop$ **do**
      //Calculate penalizer function values for data
      $\Psi'\Big((E_k)_D\Big)$, where $(E_k)_D = \left(\frac{\partial I_k}{\partial t} - \frac{\partial I^w_{k,0}}{\partial x}du - \frac{\partial I^w_{k,0}}{\partial y}dv\right)^2$;
      //Calculate the diffusion weights
      $\left[\Psi'\left(E^{l,m}_R\right)_W \Psi'\left(E^{l,m}_R\right)_N \Psi'\left(E^{l,m}_R\right)_E \Psi'\left(E^{l,m}_R\right)_S\right] = weights(u + du, v + dv)$;
      //Solve for new du and dv
      $[du\, dv] = SOLVER(\ u, v, du, dv, nLoops, \frac{\partial I_k}{\partial t}, \frac{\partial I^w_{k,0}}{\partial x}, \frac{\partial I^w_{k,0}}{\partial y}, \Psi'\Big((E_k)_D\Big), \Psi'\left(E^{l,m}_R\right)_W, \Psi'\left(E^{l,m}_R\right)_N, \Psi'\left(E^{l,m}_R\right)_S, \Psi'\left(E^{l,m}_R\right)_E\ )$;
    end while
    //Update u and v
    $u = u + du$, $v = v + dv$;
  end while
  //Interpolate (prolongate) solution
  if $s - 1 > 0$ **then**
    $[u\, v] = prolongate(u, v, sclFactor)$;
  end if
end while

Table 5.2: Typical parameters: coarse-to-fine algorithm for early linearisation (m and n refer to number of columns and rows).

Common	
scl	downscale until image size $m < 20$ or $n < 20$
$sclFactor$	0.5 - 0.75
$fstLoop$	4 - 6
$sndLoop$	4 - 8
ω	1.5 - 1.9
Solver iterations	
SOR-GS	50
SOR-GS-ALR	4

5.6.2.2 SOR-Jacobi

$$
\begin{aligned}
du_{\mathcal{I}}^{l,m,n+1} = (1-\omega) du_{\mathcal{I}}^{l,m,n} &+ \omega \frac{\sum_{k=1}^{K} \Psi'\left((E_k^{l,m})_D\right)_{\mathcal{I}} \left(\left(\frac{\partial I_k^l}{\partial t} \frac{\partial I_{k,0}^{w(l)}}{\partial x} \right)_{\mathcal{I}} - \left(\frac{\partial I_{k,0}^{w(l)}}{\partial y} \frac{\partial I_{k,0}^{w(l)}}{\partial x} \right)_{\mathcal{I}} du_{\mathcal{I}}^{l,m,n} \right)}{\sum_{k=1}^{K} \Psi'\left((E_k^{l,m})_D\right)_{\mathcal{I}} \left(\frac{\partial I_{k,0}^{w(l)}}{\partial x} \right)_{\mathcal{I}}^2 + K\alpha \sum_{\substack{\mathcal{J}\in N^-(\mathcal{I}) \\ \mathcal{J}\in N^+(\mathcal{I})}} \Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}}} \\
&+ \omega \frac{K\alpha \left(\sum_{\mathcal{J}\in N^-(\mathcal{I})} \Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}} \left(u_{\mathcal{J}}^l - u_{\mathcal{I}}^l + du_{\mathcal{J}}^{l,m,n} \right) \right)}{\sum_{k=1}^{K} \Psi'\left((E_k^{l,m})_D\right)_{\mathcal{I}} \left(\frac{\partial I_{k,0}^{w(l)}}{\partial x} \right)_{\mathcal{I}}^2 + K\alpha \sum_{\substack{\mathcal{J}\in N^-(\mathcal{I}) \\ \mathcal{J}\in N^+(\mathcal{I})}} \Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}}} \\
&+ \omega \frac{K\alpha \left(\sum_{\mathcal{J}\in N^+(\mathcal{I})} \Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}} \left(u_{\mathcal{J}}^l - u_{\mathcal{I}}^l + du_{\mathcal{J}}^{l,m,n} \right) \right)}{\sum_{k=1}^{K} \Psi'\left((E_k^{l,m})_D\right)_{\mathcal{I}} \left(\frac{\partial I_{k,0}^{w(l)}}{\partial x} \right)_{\mathcal{I}}^2 + K\alpha \sum_{\substack{\mathcal{J}\in N^-(\mathcal{I}) \\ \mathcal{J}\in N^+(\mathcal{I})}} \Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}}}
\end{aligned}
\tag{5.60}
$$

$$
\begin{aligned}
dv_{\mathcal{I}}^{l,m,n+1} = (1-\omega) dv_{\mathcal{I}}^{l,m,n} &+ \omega \frac{\sum_{k=1}^{K} \Psi'\left((E_k^{l,m})_D\right) \left(\left(\frac{\partial I_k^l}{\partial t} \frac{\partial I_{k,0}^{w(l)}}{\partial y} \right)_{\mathcal{I}} - \left(\frac{\partial I_{k,0}^{w(l)}}{\partial x} \frac{\partial I_{k,0}^{w(l)}}{\partial y} \right)_{\mathcal{I}} du_{\mathcal{I}}^{l,m,n} \right)}{\sum_{k=1}^{K} \Psi'\left((E_k^{l,m})_D\right)_{\mathcal{I}} \left(\frac{\partial I_{k,0}^{w(l)}}{\partial y} \right)_{\mathcal{I}}^2 + K\alpha \sum_{\substack{\mathcal{J}\in N^-(\mathcal{I}) \\ \mathcal{J}\in N^+(\mathcal{I})}} \Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}}} \\
&+ \omega \frac{K\alpha \left(\sum_{\mathcal{J}\in N^-(\mathcal{I})} \Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}} \left(v_{\mathcal{J}}^l - v_{\mathcal{I}}^l + dv_{\mathcal{J}}^{l,m,n} \right) \right)}{\sum_{k=1}^{K} \Psi'\left((E_k^{l,m})_D\right)_{\mathcal{I}} \left(\frac{\partial I_{k,0}^{w(l)}}{\partial y} \right)_{\mathcal{I}}^2 + K\alpha \sum_{\substack{\mathcal{J}\in N^-(\mathcal{I}) \\ \mathcal{J}\in N^+(\mathcal{I})}} \Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}}} \\
&+ \omega \frac{K\alpha \left(\sum_{\mathcal{J}\in N^+(\mathcal{I})} \Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}} \left(v_{\mathcal{J}}^l - v_{\mathcal{I}}^l + dv_{\mathcal{J}}^{l,m,n} \right) \right)}{\sum_{k=1}^{K} \Psi'\left((E_k^{l,m})_D\right)_{\mathcal{I}} \left(\frac{\partial I_{k,0}^{w(l)}}{\partial y} \right)_{\mathcal{I}}^2 + K\alpha \sum_{\substack{\mathcal{J}\in N^-(\mathcal{I}) \\ \mathcal{J}\in N^+(\mathcal{I})}} \Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}}}
\end{aligned}
\tag{5.61}
$$

Equations (5.60) and (5.61) (constructed from (5.58)) give formulation for a SOR-Jacobi type solver for late linearisation case (with warping). $\Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}}$ denotes the diffusion weight between pixels $\mathcal{J}$ and $\mathcal{I}$.

5.6.2.3 SOR-GS

$$
\begin{aligned}
du_{\mathcal{I}}^{l,m,n+1} = (1-\omega) du_{\mathcal{I}}^{l,m,n} &+ \omega \frac{\sum_{k=1}^{K} \Psi'\left((E_k^{l,m})_D\right)_{\mathcal{I}} \left(\left(\frac{\partial I_k^l}{\partial t} \frac{\partial I_{k,0}^{w(l)}}{\partial x} \right)_{\mathcal{I}} - \left(\frac{\partial I_{k,0}^{w(l)}}{\partial y} \frac{\partial I_{k,0}^{w(l)}}{\partial x} \right)_{\mathcal{I}} du_{\mathcal{I}}^{l,m,n} \right)}{\sum_{k=1}^{K} \Psi'\left((E_k^{l,m})_D\right)_{\mathcal{I}} \left(\frac{\partial I_{k,0}^{w(l)}}{\partial x} \right)_{\mathcal{I}}^2 + K\alpha \sum_{\substack{\mathcal{J}\in N^-(\mathcal{I}) \\ \mathcal{J}\in N^+(\mathcal{I})}} \Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}}} \\
&+ \omega \frac{K\alpha \left(\sum_{\mathcal{J}\in N^-(\mathcal{I})} \Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}} \left(u_{\mathcal{J}}^l - u_{\mathcal{I}}^l + du_{\mathcal{J}}^{l,m,n+1} \right) \right)}{\sum_{k=1}^{K} \Psi'\left((E_k^{l,m})_D\right)_{\mathcal{I}} \left(\frac{\partial I_{k,0}^{w(l)}}{\partial x} \right)_{\mathcal{I}}^2 + K\alpha \sum_{\substack{\mathcal{J}\in N^-(\mathcal{I}) \\ \mathcal{J}\in N^+(\mathcal{I})}} \Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}}} \\
&+ \omega \frac{K\alpha \left(\sum_{\mathcal{J}\in N^+(\mathcal{I})} \Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}} \left(u_{\mathcal{J}}^l - u_{\mathcal{I}}^l + du_{\mathcal{J}}^{l,m,n} \right) \right)}{\sum_{k=1}^{K} \Psi'\left((E_k^{l,m})_D\right)_{\mathcal{I}} \left(\frac{\partial I_{k,0}^{w(l)}}{\partial x} \right)_{\mathcal{I}}^2 + K\alpha \sum_{\substack{\mathcal{J}\in N^-(\mathcal{I}) \\ \mathcal{J}\in N^+(\mathcal{I})}} \Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}}}
\end{aligned}
\tag{5.62}
$$

$$
\begin{aligned}
dv_{\mathcal{I}}^{l,m,n+1} = (1-\omega) dv_{\mathcal{I}}^{l,m,n} &+ \omega \frac{\sum_{k=1}^{K} \Psi'\left((E_k^{l,m})_D\right) \left(\left(\frac{\partial I_k^l}{\partial t} \frac{\partial I_{k,0}^{w(l)}}{\partial y} \right)_{\mathcal{I}} - \left(\frac{\partial I_{k,0}^{w(l)}}{\partial x} \frac{\partial I_{k,0}^{w(l)}}{\partial y} \right)_{\mathcal{I}} du_{\mathcal{I}}^{l,m,n} \right)}{\sum_{k=1}^{K} \Psi'\left((E_k^{l,m})_D\right)_{\mathcal{I}} \left(\frac{\partial I_{k,0}^{w(l)}}{\partial y} \right)_{\mathcal{I}}^2 + K\alpha \sum_{\substack{\mathcal{J}\in N^-(\mathcal{I}) \\ \mathcal{J}\in N^+(\mathcal{I})}} \Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}}} \\
&+ \omega \frac{K\alpha \left(\sum_{\mathcal{J}\in N^-(\mathcal{I})} \Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}} \left(v_{\mathcal{J}}^l - v_{\mathcal{I}}^l + dv_{\mathcal{J}}^{l,m,n+1} \right) \right)}{\sum_{k=1}^{K} \Psi'\left((E_k^{l,m})_D\right)_{\mathcal{I}} \left(\frac{\partial I_{k,0}^{w(l)}}{\partial y} \right)_{\mathcal{I}}^2 + K\alpha \sum_{\substack{\mathcal{J}\in N^-(\mathcal{I}) \\ \mathcal{J}\in N^+(\mathcal{I})}} \Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}}} \\
&+ \omega \frac{K\alpha \left(\sum_{\mathcal{J}\in N^+(\mathcal{I})} \Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}} \left(v_{\mathcal{J}}^l - v_{\mathcal{I}}^l + dv_{\mathcal{J}}^{l,m,n} \right) \right)}{\sum_{k=1}^{K} \Psi'\left((E_k^{l,m})_D\right)_{\mathcal{I}} \left(\frac{\partial I_{k,0}^{w(l)}}{\partial y} \right)_{\mathcal{I}}^2 + K\alpha \sum_{\substack{\mathcal{J}\in N^-(\mathcal{I}) \\ \mathcal{J}\in N^+(\mathcal{I})}} \Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}}}
\end{aligned}
\tag{5.63}
$$

Equations (5.62) and (5.63) (constructed from (5.58)) give formulation for a SOR-GS type solver for late linearisation (with warping). $\Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}}$ denotes the diffusion weight between pixels $\mathcal{J}$ and $\mathcal{I}$.

5.6.2.4 GS-ALR

Here we show how the GS-ALR solver is constructed for the late linearisation case. As it was already mentioned in Section 5.3.4, the algorithm consists of two sweeps, namely the forward- and the backward-sweeps, as given in equations (5.64) and (5.65). The algorithm introduced here can be converted into SOR-GS-ALR easily, as is shown in Algorithm 2. The only thing that needs to be explained is what the $a_{\mathcal{I}}$, $b_{\mathcal{I}}$, $c_{\mathcal{I}}$ and $d_{\mathcal{I}}$ are.

$$
\begin{aligned}
c'_{\mathcal{I}} &= \begin{cases} \dfrac{c_1}{b_1} & , \mathcal{I} = 1 \\ \dfrac{c_{\mathcal{I}}}{b_{\mathcal{I}} - c'_{\mathcal{I}-1} a_{\mathcal{I}}} & , \mathcal{I} = 2, 3, ..., N-1 \end{cases} \\
d'_{\mathcal{I}} &= \begin{cases} \dfrac{d_1}{b_1} & , \mathcal{I} = 1 \\ \dfrac{d_{\mathcal{I}} - d'_{\mathcal{I}-1} a_{\mathcal{I}}}{b_{\mathcal{I}} - c'_{\mathcal{I}-1} a_{\mathcal{I}}} & , \mathcal{I} = 2, 3, ..., N \end{cases}
\end{aligned}
\tag{5.64}
$$

$$
\begin{aligned}
x_N &= d'_N \\
x_{\mathcal{I}} &= d'_{\mathcal{I}} - c'_{\mathcal{I}} x_{\mathcal{I}+1} \quad , \mathcal{I} = N-1, N-2, ..., 1
\end{aligned}
\tag{5.65}
$$

Here, however, since the diffusion weights change, we need to define the elements for both column- and row-wise traversing. The only difference is, apart from the traversing order, the diffusion weights denoted by $a_{\mathcal{I}}$:s and $c_{\mathcal{I}}$:s. $\Psi'\left(E_R^{l,m}\right)_{\{W,N,E,S\},\mathcal{I}}$ denotes the diffusivity as position $\mathcal{I}$, whereas $\{W, N, E, S\}$ is the diffusion 'direction' (i.e. West, North, East or South).

Column wise.

Equation (5.66) defines the elements when solving for du.

$$
\begin{aligned}
a_{\mathcal{I}} =& - K\alpha\Psi'\left(E_R^{l,m}\right)_{N,\mathcal{I}} \\
b_{\mathcal{I}} =& \sum_{k=1}^{K}\Psi'\left((E_k^{l,m})_D\right)_{\mathcal{I}}\left(\frac{\partial I_{k,0}^{w(l)}}{\partial x}\right)_{\mathcal{I}}^2 + K\alpha\sum_{\mathcal{J}\in N(\mathcal{I})}\Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}} \\
c_{\mathcal{I}} =& - K\alpha\Psi'\left(E_R^{l,m}\right)_{S,\mathcal{I}} \\
d_{\mathcal{I}} =& \sum_{k=1}^{K}\Psi'\left((E_k^{l,m})_D\right)_{\mathcal{I}}\left(\left(\frac{\partial I_k^l}{\partial t}\frac{\partial I_{k,0}^{w(l)}}{\partial y}\right)_{\mathcal{I}} - \left(\frac{\partial I_{k,0}^{w(l)}}{\partial x}\frac{\partial I_{k,0}^{w(l)}}{\partial y}\right)_{\mathcal{I}} dv_{\mathcal{I}}^{l,m,n}\right) \\
&+ K\alpha\left(\sum_{\mathcal{J}\in N^-(\mathcal{I})}\Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}}\left(u_{\mathcal{J}}^l - u_{\mathcal{I}}^l + du_{\mathcal{J}}^{l,m,n+1}\right)\right) \\
&+ K\alpha\left(\sum_{\mathcal{J}\in N^+(\mathcal{I})}\Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}}\left(u_{\mathcal{J}}^l - u_{\mathcal{I}}^l + du_{\mathcal{J}}^{l,m,n}\right)\right)
\end{aligned}
\tag{5.66}
$$

Equation (5.67) defines the elements when solving for dv.

$$
\begin{aligned}
a_{\mathcal{I}} =& - K\alpha\Psi'\left(E_R^{l,m}\right)_{N,\mathcal{I}} \\
b_{\mathcal{I}} =& \sum_{k=1}^{K}\Psi'\left((E_k^{l,m})_D\right)_{\mathcal{I}}\left(\frac{\partial I_{k,0}^{w(l)}}{\partial y}\right)_{\mathcal{I}}^2 + K\alpha\sum_{\mathcal{J}\in N(\mathcal{I})}\Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}} \\
c_{\mathcal{I}} =& - K\alpha\Psi'\left(E_R^{l,m}\right)_{S,\mathcal{I}} \\
d_{\mathcal{I}} =& \sum_{k=1}^{K}\Psi'\left((E_k^{l,m})_D\right)_{\mathcal{I}}\left(\left(\frac{\partial I_k^l}{\partial t}\frac{\partial I_{k,0}^{w(l)}}{\partial y}\right)_{\mathcal{I}} - \left(\frac{\partial I_{k,0}^{w(l)}}{\partial x}\frac{\partial I_{k,0}^{w(l)}}{\partial y}\right)_{\mathcal{I}} du_{\mathcal{I}}^{l,m,n}\right) \\
&+ K\alpha\left(\sum_{\mathcal{J}\in N^-(\mathcal{I})}\Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}}\left(v_{\mathcal{J}}^l - v_{\mathcal{I}}^l + dv_{\mathcal{J}}^{l,m,n+1}\right)\right) \\
&+ K\alpha\left(\sum_{\mathcal{J}\in N^+(\mathcal{I})}\Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}}\left(v_{\mathcal{J}}^l - v_{\mathcal{I}}^l + dv_{\mathcal{J}}^{l,m,n}\right)\right)
\end{aligned}
\tag{5.67}
$$

Row wise.

Equation (5.68) defines the elements when solving for du.

$$
\begin{aligned}
a_{\mathcal{I}} &= -K\alpha\Psi'\left(E_R^{l,m}\right)_{W,\mathcal{I}} \\
b_{\mathcal{I}} &= \sum_{k=1}^{K}\Psi'\left((E_k^{l,m})_D\right)_{\mathcal{I}}\left(\frac{\partial I_{k,0}^{w(l)}}{\partial x}\right)_{\mathcal{I}}^2 + K\alpha\sum_{\mathcal{J}\in N(\mathcal{I})}\Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}} \\
c_{\mathcal{I}} &= -K\alpha\Psi'\left(E_R^{l,m}\right)_{E,\mathcal{I}} \\
d_{\mathcal{I}} &= \sum_{k=1}^{K}\Psi'\left((E_k^{l,m})_D\right)_{\mathcal{I}}\left(\left(\frac{\partial I_k^l}{\partial t}\frac{\partial I_{k,0}^{w(l)}}{\partial y}\right)_{\mathcal{I}} - \left(\frac{\partial I_{k,0}^{w(l)}}{\partial x}\frac{\partial I_{k,0}^{w(l)}}{\partial y}\right)_{\mathcal{I}} du_{\mathcal{I}}^{l,m,n}\right) \\
&\quad + K\alpha\left(\sum_{\mathcal{J}\in N^-(\mathcal{I})}\Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}}\left(u_{\mathcal{J}}^l - u_{\mathcal{I}}^l + du_{\mathcal{J}}^{l,m,n+1}\right)\right) \\
&\quad + K\alpha\left(\sum_{\mathcal{J}\in N^+(\mathcal{I})}\Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}}\left(u_{\mathcal{J}}^l - u_{\mathcal{I}}^l + du_{\mathcal{J}}^{l,m,n}\right)\right)
\end{aligned}
\tag{5.68}
$$

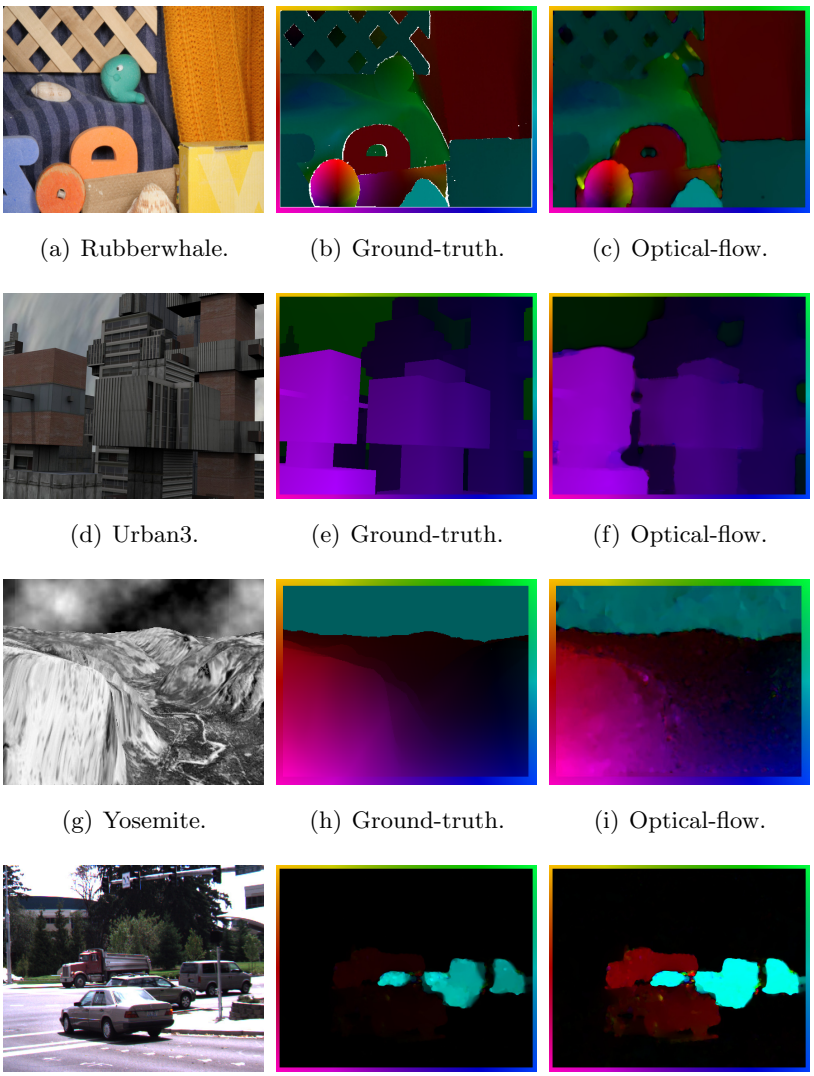
Equation (5.69) defines the elements when solving for dv.

$$
\begin{aligned}
a_{\mathcal{I}} &= -K\alpha\Psi'\left(E_R^{l,m}\right)_{W,\mathcal{I}} \\
b_{\mathcal{I}} &= \sum_{k=1}^{K}\Psi'\left((E_k^{l,m})_D\right)_{\mathcal{I}}\left(\frac{\partial I_{k,0}^{w(l)}}{\partial y}\right)_{\mathcal{I}}^2 + K\alpha\sum_{\mathcal{J}\in N(\mathcal{I})}\Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}} \\
c_{\mathcal{I}} &= -K\alpha\Psi'\left(E_R^{l,m}\right)_{E,\mathcal{I}} \\
d_{\mathcal{I}} &= \sum_{k=1}^{K}\Psi'\left((E_k^{l,m})_D\right)_{\mathcal{I}}\left(\left(\frac{\partial I_k^l}{\partial t}\frac{\partial I_{k,0}^{w(l)}}{\partial y}\right)_{\mathcal{I}} - \left(\frac{\partial I_{k,0}^{w(l)}}{\partial x}\frac{\partial I_{k,0}^{w(l)}}{\partial y}\right)_{\mathcal{I}} du_{\mathcal{I}}^{l,m,n}\right) \\
&\quad + K\alpha\left(\sum_{\mathcal{J}\in N^-(\mathcal{I})}\Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}}\left(v_{\mathcal{J}}^l - v_{\mathcal{I}}^l + dv_{\mathcal{J}}^{l,m,n+1}\right)\right) \\
&\quad + K\alpha\left(\sum_{\mathcal{J}\in N^+(\mathcal{I})}\Psi'\left(E_R^{l,m}\right)_{\mathcal{I}\sim\mathcal{J}}\left(v_{\mathcal{J}}^l - v_{\mathcal{I}}^l + dv_{\mathcal{J}}^{l,m,n}\right)\right)
\end{aligned}
\tag{5.69}
$$

5.6.2.5 Results

In the following we show some results for the late linearisation model with test images from Middlebury[1].

[1] `http://vision.middlebury.edu/flow/data/`



(a) Rubberwhale. (b) Ground-truth. (c) Optical-flow.

(d) Urban3. (e) Ground-truth. (f) Optical-flow.

(g) Yosemite. (h) Ground-truth. (i) Optical-flow.

(j) Dumptruck. (k) Optical-flow. (l) Optical-flow sat.

Figure 5.12: Late linearisation results. In the last row we show (k) optical-flow and (l) saturated optical-flow. With saturated we refer to how the flow is displayed. By limiting the maximum movement, also the objects with smaller displacements can be observed.

5.7 Solving Level-set Equation

Previously we have shown how both the early- and the late-linearisation models for optical-flow can be solved. As was explained earlier, these were defined as space-type problems, where we directly search for the steady-state (i.e. where the solution does

not change anymore). In what follows, we show how the segmentation method based on implicit curves can be solved. This model exhibits type-type behaviour. However, as it was already mentioned in the introduction of this chapter, a time-type problem, when discretised implicitly with respect to time, yields a space-type problem that is solved at each time step. Therefore, it is not a great surprise that the solver for the segmentation model is very similar to those seen previously. Naturally, this is a great advantage, since similar type of mathematical machinery can be used for conceptually different problems (optical-flow vs. segmentation). Equation 5.70 is the same as seen previously ((5.28) and (4.23)), with the difference that we use P to denote $log\, p_r\,(d|\alpha_1) - log\, \overline{p_r}\,(d|\alpha_1)$. This is done to simplify the notation.

$$\frac{\partial \Phi}{\partial t} = H'_\epsilon(\Phi)\Bigg(P + \alpha DIV\left(\frac{\nabla\Phi}{|\nabla\Phi|}\right)\Bigg) \tag{5.70}$$

We use a so called Euler forward method to discretise the time. Since we solve for Φ^{t+1} inside the DIV term, the complete scheme is known as Euler forward implicit scheme and is given as follows:

$$\begin{aligned} \frac{\Phi^{t+1} - \Phi^t}{\tau} &= H'_\epsilon(\Phi^t)\Bigg(P + \alpha DIV\left(\frac{\nabla\Phi^{t+1}}{|\nabla\Phi^t|}\right)\Bigg) \\ \Phi^{t+1} &= \Phi^t + \tau H'_\epsilon(\Phi^t)\Bigg(P + \alpha DIV\left(\frac{\nabla\Phi^{t+1}}{|\nabla\Phi^t|}\right)\Bigg) \end{aligned} \tag{5.71}$$

where τ is the step in time. Starting from Equation 5.71 we can discretise the DIV operator as seen previously. Instead of $DIV\left(\frac{\nabla\Phi^{t+1}}{|\nabla\Phi^t|}\right)$, we can write $DIV\left(\frac{1}{|\nabla\Phi^t|}\nabla\Phi^{t+1}\right)$ where $\frac{1}{|\nabla\Phi^t|}$ 'acts' like a diffusion weight. The only difference is that we approximate this term using a finite difference scheme with *harmonic averaging*[42][78], and, therefore, obtain the following:

$$
\begin{aligned}
\Phi_{i,j}^{t+1} = \Phi_{i,j}^{t} &+ \tau H'_{\epsilon}(\Phi_{i,j}^{t})P \\
&+ \tau\alpha H'_{\epsilon}(\Phi_{i,j}^{t})\left(\frac{2}{\left(|\nabla\Phi_{i,j}^{t}|\right)+\left(|\nabla\Phi_{i+1,j}^{t}|\right)}\right)\left(\Phi_{i+1,j}^{t+1}-\Phi_{i,j}^{t+1}\right) \\
&+ \tau\alpha H'_{\epsilon}(\Phi_{i,j}^{t})\left(\frac{2}{\left(|\nabla\Phi_{i,j}^{t}|\right)+\left(|\nabla\Phi_{i-1,j}^{t}|\right)}\right)\left(\Phi_{i-1,j}^{t+1}-\Phi_{i,j}^{t+1}\right) \\
&+ \tau\alpha H'_{\epsilon}(\Phi_{i,j}^{t})\left(\frac{2}{\left(|\nabla\Phi_{i,j}^{t}|\right)+\left(|\nabla\Phi_{i,j+1}^{t}|\right)}\right)\left(\Phi_{i,j+1}^{t+1}-\Phi_{i,j}^{t+1}\right) \\
&+ \tau\alpha H'_{\epsilon}(\Phi_{i,j}^{t})\left(\frac{2}{\left(|\nabla\Phi_{i,j}^{t}|\right)+\left(|\nabla\Phi_{i,j-1}^{t}|\right)}\right)(\Phi_{i,j-1}^{t+1}-\Phi_{i,j}^{t+1})
\end{aligned}
\tag{5.72}
$$

As we can see, Equation (5.72) is linear with respect to Φ^{t+1}, since in the derivative of the Heaviside function, $H'_{\epsilon}(\Phi_{i,j}^{t})$, and in the $\frac{1}{|\nabla\Phi^{t}|}$ we use the last known approximation of Φ. In the following we re-organise Equation (5.72) in order to facilitate the construction of concrete solvers.

$$
\begin{aligned}
\Phi_{i,j}^{t+1}\Bigg(1+\tau\alpha H'_{\epsilon}(\Phi_{i,j}^{t})\Big(&\frac{2}{\left(|\nabla\Phi_{i,j}^{t}|\right)+\left(|\nabla\Phi_{i+1,j}^{t}|\right)}+\frac{2}{\left(|\nabla\Phi_{i,j}^{t}|\right)+\left(|\nabla\Phi_{i-1,j}^{t}|\right)} \\
&+\frac{2}{\left(|\nabla\Phi_{i,j}^{t}|\right)+\left(|\nabla\Phi_{i,j+1}^{t}|\right)}+\frac{2}{\left(|\nabla\Phi_{i,j}^{t}|\right)+\left(|\nabla\Phi_{i,j-1}^{t}|\right)}\Big)\Bigg) = \\
&\Phi_{i,j}^{t} + \tau H'_{\epsilon}(\Phi_{i,j}^{t})P \\
&+ \tau\alpha H'_{\epsilon}(\Phi_{i,j}^{t})\left(\frac{2}{\left(|\nabla\Phi_{i,j}^{t}|\right)+\left(|\nabla\Phi_{i+1,j}^{t}|\right)}\right)\Phi_{i+1,j}^{t+1} \\
&+ \tau\alpha H'_{\epsilon}(\Phi_{i,j}^{t})\left(\frac{2}{\left(|\nabla\Phi_{i,j}^{t}|\right)+\left(|\nabla\Phi_{i-1,j}^{t}|\right)}\right)\Phi_{i-1,j}^{t+1} \\
&+ \tau\alpha H'_{\epsilon}(\Phi_{i,j}^{t})\left(\frac{2}{\left(|\nabla\Phi_{i,j}^{t}|\right)+\left(|\nabla\Phi_{i,j+1}^{t}|\right)}\right)\Phi_{i,j+1}^{t+1} \\
&+ \tau\alpha H'_{\epsilon}(\Phi_{i,j}^{t})\left(\frac{2}{\left(|\nabla\Phi_{i,j}^{t}|\right)+\left(|\nabla\Phi_{i,j-1}^{t}|\right)}\right)\Phi_{i,j-1}^{t+1}
\end{aligned}
\tag{5.73}
$$

An algorithmic description of the SOR with a generic solver is given in Equation 5.74. A new approximation is a weighted combination of the previous and the new solution.

$$\Phi^{t,l+1} = (1-\omega)\Phi^{t,l} + \omega SOLVER(\Phi^{t,l},\, P,\, |\nabla\Phi^t|,\, g_W^t,\, g_N^t,\, g_E^t,\, g_S^t) \tag{5.74}$$

where $SOLVER(\Phi^{t,l},\, P,\, |\nabla\Phi^t|,\, g_W^t,\, g_N^t,\, g_E^t,\, g_S^t)$ can be any solver for linear system of equations that calculates a new solution at $l+1$ for the input at l.

5.7.1 SOR-GS

Equation 5.75 gives an example of a SOR-GS type solver.

$$\begin{aligned}
\Phi_{\mathcal{I}}^{t,l+1} = (1-\omega)\Phi_{\mathcal{I}}^{t,l} &+ \omega \frac{\Phi_{\mathcal{I}}^{t,l} + \tau H'_\epsilon(\Phi_{\mathcal{I}}^t)P}{1 + \alpha\tau H'_\epsilon(\Phi_{\mathcal{I}}^t) \sum\limits_{\substack{\mathcal{J}\in N^-(\mathcal{I})\\ \mathcal{J}\in N^+(\mathcal{I})}} \frac{2}{(|\nabla\Phi^t|)_{\mathcal{I}} + (|\nabla\Phi^t|)_{\mathcal{J}}}} \\
&+ \omega \frac{\alpha\tau H'_\epsilon(\Phi_{\mathcal{I}}^t) \sum\limits_{\mathcal{J}\in N^-(\mathcal{I})} \frac{2}{(|\nabla\Phi^t|)_{\mathcal{I}} + (|\nabla\Phi^t|)_{\mathcal{J}}} \Phi_{\mathcal{J}}^{t,l+1}}{1 + \alpha\tau H'_\epsilon(\Phi_{\mathcal{I}}^t) \sum\limits_{\substack{\mathcal{J}\in N^-(\mathcal{I})\\ \mathcal{J}\in N^+(\mathcal{I})}} \frac{2}{(|\nabla\Phi^t|)_{\mathcal{I}} + (|\nabla\Phi^t|)_{\mathcal{J}}}} \\
&+ \omega \frac{\alpha\tau H'_\epsilon(\Phi_{\mathcal{I}}^t) \sum\limits_{\mathcal{J}\in N^+(\mathcal{I})} \frac{2}{(|\nabla\Phi^t|)_{\mathcal{I}} + (|\nabla\Phi^t|)_{\mathcal{J}}} \Phi_{\mathcal{J}}^{t,l}}{1 + \alpha\tau H'_\epsilon(\Phi_{\mathcal{I}}^t) \sum\limits_{\substack{\mathcal{J}\in N^-(\mathcal{I})\\ \mathcal{J}\in N^+(\mathcal{I})}} \frac{2}{(|\nabla\Phi^t|)_{\mathcal{I}} + (|\nabla\Phi^t|)_{\mathcal{J}}}}
\end{aligned} \tag{5.75}$$

5.8 Conclusions

In this chapter we have shown some techniques that can be used for solving the optical-flow and segmentation models, based on variational calculus, efficiently.

As it has been shown, the Euler-Lagrange equations related to the models lead, in fact, to systems of partial differential equations (PDEs) that need to be solved.

The subject of solving PDEs is so huge, that understandably we have only scratched surface here.

However, we feel confident that the descriptions (and algorithms) given in this chapter should be clear enough to encourage an interested reader to further research the subject.

6

Conclusions

6.1 Summary

Each of the sections related to the scientific goals established in Section 1.2 have introduced both the motivation for the problem at hand, and also the related conclusions, therefore hopefully making this document easier to follow. Relevant conclusions can be found in the following sections: robust data terms, Section 3.9.8; temporal- and spatial constraints in optical-flow and disparity calculation, Section 3.10.10; hypothesis-forming-validation loops, Section 4.5. Apart from the results shown in this work, more results are available at my homepage[1]. Following is a brief summary of the conclusions.

Our target has been improving robustness and, therefore, applicability, of the variational methods used for generating pixel correspondences, especially with real sequences. With real sequences image representation plays a crucial role when it comes down to establishing correct pixel-wise correspondences. As we have shown, improper image representation can render the results completely useless. Typically benchmarking of different algorithms is done using test images that do not contain relevant illumination errors and/or image noise. This is, of course, a good starting point for the evaluation, but does not convey enough information of how the algorithm in question will behave with sequences where lightning or related conditions cannot be controlled. Typical cases where lightning conditions cannot be controlled are outdoor sequences, like those in the DRIVSCO project. Therefore, it is difficult to choose an algorithm that would produce acceptable results for the task at hand. In Section 3.9.8 we have

[1] `http://atc.ugr.es/~jarnor/`

studied and found several representations to be considerably more robust than the typical RGB-representation. Especially image representations based on gradient have given very good results, both accuracy and robustness wise. In order to minimise 'human influence' upon the parameters related to each representation, we have used a genetic algorithm called differential evolution to search for the optimal (or pseudo-optimal) parameters. In order to improve generality of the results, we have used 5-fold cross validation.

Due to the local nature of low-level vision algorithms, these suffer from ambiguous interpretations. Even though the variational methods used in this work for establishing pixel correspondences are said to be global in nature, they are based on local interactions between neighbouring pixels. In other words, these methods do not try to reach a more 'global' interpretation of the scene. Vision process is said to be a creative process [38] where interpretations of the scene can change and, indeed, do change depending on the task at hand. Many optical tricks are based on this fact. Therefore, it is logical to use available a priori information to constrain the solution so that it would be 'coherent' with what we know of the world we live in. There exists evidence that, indeed, the primate vision system functions this way [38]: we are used to interpret depth cues in a certain way since either a) the vision system has evolved this way or b) our brains have learnt to interpret certain cues in a particular way due to the constraints of the world we live in. We have shown in Section (3.10) that considerable improvements are possible by using a priori information.

As it was already mentioned, there is consensus in the scientific community that visual perception is a creative process [38]. Thus scene interpretation depends on the context and the task at hand and, therefore, can change accordingly. In our hypothesis-forming-validation loop scheme we have used this idea to automatically extract and generate spatial constraints using segmentation, and then use these constraints to improve the results in a recurrent loop fashion. If the initial results, on which the segmentation is based on, are 'good' enough, then the model will converge towards a coherent scene interpretation. On the other hand, if the obtained constraints are not in-line with the data, then these simply get rejected [57][60][58] without influencing the final outcome from the system. As has been shown in Section 4.5, considerable improvements are possible, especially in the case of cases typically considered to be difficult.

We have added Chapter 5, related to how the equations can be solved, so that an interested reader could get acquainted with the variational methods. Even though we have shown how the equations related to the pixel correspondences (optical-flow and disparity) and level-sets can be solved, using the same numerical methods many other interesting problems, defined with calculus of variations, ca be solved.

6.2 Future Work

Future work consists of including more relevant cues in the segmentation in order to further improve the results. Also, temporal coherency of the resulting segments should be imposed in order to facilitate tracking of the segments temporally. Fusing segments based on (a) optical-flow and (b) segment neighbourhood will be studied: neighbouring segments that have similar optical-flow fields probably belong to the same objects and, therefore, can be either fused together or tracked together.

6.3 Publications

During the coarse of this thesis, related to the goals given in Section 1.2, we have studied the following: (1) how to propagate sparse disparity cues using a voting based methodology [56]; (2) how to disambiguate dense low-level disparity cues by using either more reliable cues or what is known on the solution beforehand [57]; (3) robustness of different image representations with respect to illumination errors and image noise in stereo-disparity [59]; (4) the influence of both spatial- and temporal constraints in both optical-flow and stereo disparity calculation [60] and (5) using constraints based on segmentation in disparity calculation [58]. Published papers (included those sent for publication but not published at the time of writing this thesis) are:

- F. Naveros, T. Díaz, **J. Ralli**, J. Díaz, and E. Ros, 'Flujo óptico variacional en plataformas paralelas GPU', submitted to a national (Spanish) conference.

- **J. Ralli**, F. Pelayo, and J. Díaz, 'Increasing Efficiency in Disparity Calculation', LNCS, Advances in Brain, Vision, and Artificial Intelligence, 2007, 298-307, url:`http://dx.doi.org/10.1007/978-3-540-75555-5_28`.

- **J. Ralli**, J. Díaz, and E. Ros, 'A Method for Sparse Disparity Densification Using Voting Mask Propagation', Journal of Visual Communication and Image Representation, 21(1):6774, 2009, url:`http://dx.doi.org/10.1016/j.jvcir.2009.08.005`.

- **J. Ralli**, J. Díaz, S. Kalkan, N. Krüger and E. Ros, 'Disparity Disambiguation by Fusion of Signal- and Symbolic-Level Information', Machine Vision and Applications, 2010, url:`http://dx.doi.org/10.1007/s00138-010-0266-z`.

- **J. Ralli**, J. Díaz, P. Guzmán and E. Ros, 'Complementary Image Representation Spaces in Variational Disparity Calculation', submitted for publication.

- **J. Ralli**, J. Díaz and E. Ros, 'Spatial and Temporal Constraints in Variational Correspondence Methods', Machine Vision and Applications, 2011, url:`http://dx.doi.org/10.1007/s00138-011-0360-x`.

- **J. Ralli**, J. Díaz, E. Ros, J. Ilonen, and V. Kyrki, 'External Constraints in Variational Disparity Calculation: Hypothesis-Forming-Validation-Loops and Segmentation', submitted for publication.

- N.R. Luque, J.A. Garrido, **J. Ralli**, J.J Laredo, and E. Ros, 'From Sensors to Spikes: Evolving Receptive Fields to Enhance Sensori-Motor Information in a Robot-Arm', submitted for publication.

- P. Guzmán, J. Díaz, **J. Ralli**, R. Agís, and E. Ros, 'Low-cost Sensor to Detect Overtaking Based on Optical-flow', accepted for publication in Machine Vision and Applications.

This represents a scientific production as follows: 5 journal papers with the author of this work as first author; 1 published conference paper; 2 collaborative journal papers and 1 collaborative conference paper.

6.4 Main Contributions

We now present the main contributions of this work.

- We have shown that image representation has a profound effect on the resulting disparity maps with real images containing both illumination errors and image noise.
- We have tested several image representations and ranked those with respect to the accuracy and the robustness of each representation. The best representations, both accuracy and robustness wise, are based on image gradient.
- We have come up with a 'sound' framework for finding pseudo-optimal parameters for different representation spaces using a genetic algorithm. This framework is general enough to be extended for other image processing problems as well.
- We have shown that using a priori information in establishing image correspondences can increase quality of the results considerably. Therefore, further image processing tasks based on the improved results, such as segmentation, will improve considerably.
- These a priori:s can be extracted automatically and imposed recurrently using a hypothesis-forming-validation loop.
- We have shown that segmenting an image into meaningful surfaces, where segmentation is based on calculated disparity maps, is possible.
- We have validated the variational approach as an efficient framework for solving many computer vision problems. The focus of this thesis was on developing methods capable of working with 'real-world' data and, therefore, robustness was one of the most important aspects.

7

Conclusiones en Castellano

7.1 Sumario

Cada una de las secciones de este trabajo, relacionados a los 'retos' científicos introducidos en la Sección 1.2, introducen tanto la motivación como las conclusiones de los resultados obtenidos. De esta manera cada una de las secciones pretenden ser 'auto-contenidas' y, supuestamente, mas fáciles de leer. Las conclusiones más relevantes se encuentran en las siguientes secciones: términos de datos robustos, Sección 3.9.8; restricciones espaciales y temporales en el cálculo de flujo-óptico y disparidad, 3.10.10; 'hypothesis-forming-validation loops' (bucles de formación y convalidación de hipótesis), Sección 4.5. Aparte de los resultados mostrados en esta tesis, más resultados están disponibles en mi pgina web[1]. A continuación un resumen corto de las conclusiones.

Nuestro objetivo ha sido mejorar la robustez y, por tanto, la aplicabilidad de los métodos variacionales utilizados para la generación de correspondencias de píxeles, especialmente con secuencias reales. Utilizando secuencias 'reales', la representación de la imagen tiene un papel muy importante cuando se trata de establecer correspondencias correctas entre imágenes temporales (flujo óptico) o estéreo (disparidad). Como hemos comprobado, una representación inadecuada puede dejar los resultados completamente inservibles. Típicamente las secuencias/imágenes que se usan para comparación cualitativa no contienen errores relevantes de iluminación y/o ruido. Este, obviamente, es un punto de partida muy válido para tal comparación, pero lamentablemente no provee

[1] `http://atc.ugr.es/~jarnor/`

información clave de como se portaría el algoritmo en cuestión en situaciones donde las condiciones de la iluminación no se pueden controlar. Casos típicos donde no se pueden controlar las condiciones son las de 'aire libre', tales como las del proyecto DRIVSCO. Por lo tanto, es difícil escoger un algoritmo apropiado, capaz de producir resultados aceptables para la tarea que se pretende a resolver. En Sección 3.9.8 hemos estudiado varias representaciones diferentes y hemos encontrado varias que son considerablemente más robustas que la de RGB. Sobre todo representaciones basados en gradiente han dado muy buenos resultados en cuanto a la precisión y la robustez. Para minimizar la influencia humana relacionada a cada representación, se ha utilizado un algoritmo genético llamado evolución diferencial (ingl. differential evolution) en la búsqueda los parámetros. Con el fin de mejorar la generalidad de los resultados, se han utilizado validación cruzada , dividiendo la muestra en cinco conjuntos (ingl. 5-fold cross validation)

Por la naturaleza local de los algoritmos de nivel bajo, estos sufren de interpretaciones ambiguas. Aunque los métodos variacionales, utilizados para establecer correspondencias, supuestamente son globales, estos están basados en interacciones locales entre los pixeles más próximos. En otras palabras, estos modelos no tratan de llegar a una interpretación más global de la escena. El proceso visual es un proceso creativo [38], en el cual la interpretación de la escena se puede cambiar debido a la tarea que se está realizando. Varios trucos 'ópticos' empleados por los magos están basados en este hecho. Por lo tanto, es natural usar información a priori para restringir la solución para que esta sea coherente con lo que sabemos del mundo donde vivimos. Existe evidencia que la visión en los primates funciona de esta manera [38]: interpretamos rasgos visuales de cierta manera ya que (a) el sistema visual ha evolucionado en esta dirección o (b) hemos aprendido a interpretar dichos rasgos de cierta manera debido a las restricciones del mundo donde vivimos. Hemos comprobado en la Sección (3.10) que mejoras considerables son posibles utilizando información a priori.

Tal cual como hemos indicado en capítulos/párrafos anteriores, existe un consenso en la comunidad científica que la percepción visual es un proceso creativo [38]. Esto quiere decir que la interpretación de la escena depende tanto del contexto como del problema que estamos tratando de resolver y, por lo tanto, la interpretación se puede cambiar. Hemos utilizado la idea de 'hypothesis-forming-validation loop' (bucles de formación y confirmación de hipótesis) para generar restricciones espaciales utilizando

segmentación de la escena. Si los resultados iniciales son suficientemente buenos, es este caso el sistema converge hacia una interpretación más coherente de la escena. Por otro lado, si las restricciones obtenidos no están de acuerdo con la evidencia (datos), en este caso simplemente se rechaza dicha restricción [57][60][58] sin que esta influya en el resultado. Tal cual como hemos demostrado en la Sección 4.5, incluso en casos difíciles, se pueden extraer hipótesis útiles que luego se usan para mejorar los resultados. Son justamente estos casos difíciles donde se ven mejoras considerables.

Hemos agregado Capítulo 5, donde se demuestra como se pueden resolver las ecuaciones numéricamente, para que un lector interesado pueda conocer mejor los métodos variacionales. La misma maquinaria matemática es suficientemente genérica para ser utilizada en otros casos, aparte de calculo de flujo-óptico o disparidad, relacionados a tratamiento de imágenes con métodos variacionales.

7.2 Trabajo Futuro

El trabajo futuro consiste en incluir otros rasgos relevantes en la segmentación para mejorar los resultados mas allá de los obtenidos. Además, se pretende investigar como se podría imponer coherencia temporal en la segmentación para facilitar el seguimiento (ingl. tracking) de los segmentos. También se pretende estudiar el fusionado de los segmentos basado en (a) flujo-óptico como (b) vecindad de los segmentos: segmentos vecinos con flujo-óptico parecido probablemente pertenecen al mismo objeto y, por lo tanto, se podrían fusionar.

7.3 Publicaciones

Durante la creación de esta tesis, relacionado a las metas introducidas en la Sección 1.2, hemos investigado lo siguiente: (1) como propagar estimaciones dispersas de disparidad utilizando filtros de votación [56]; (2) como se pueden desambiguar aproximaciones densas de disparidad utilizando aproximaciones mas confiables, pero menos densos a la vez [57]; (3) la robustez de varias representaciones de imágenes con respecto a errores de iluminación y ruido [59]; (4) la influencia de restricciones espaciales y temporales en el cálculo de flujo-óptico y estéreo [60] y (5) como obtener dichas restricciones espaciales

automáticamente utilizando segmentación [58]. Los artículos escritos (incluyendo los que están en proceso de publicación) son:

- F. Naveros, T. Díaz, **J. Ralli**, J. Díaz, and E. Ros, 'Flujo óptico variacional en plataformas paralelas GPU', enviado a conferencia nacional.
- **J. Ralli**, F. Pelayo, and J. Díaz, 'Increasing Efficiency in Disparity Calculation', LNCS, Advances in Brain, Vision, and Artificial Intelligence, 2007, 298-307, url:`http://dx.doi.org/10.1007/978-3-540-75555-5_28`.
- **J. Ralli**, J. Díaz, and E. Ros, 'A Method for Sparse Disparity Densification Using Voting Mask Propagation', Journal of Visual Communication and Image Representation, 21(1):6774, 2009, url:`http://dx.doi.org/10.1016/j.jvcir.2009.08.005`.
- **J. Ralli**, J. Díaz, S. Kalkan, N. Krüger and E. Ros, 'Disparity Disambiguation by Fusion of Signal- and Symbolic-Level Information', Machine Vision and Applications, 2010, url:`http://dx.doi.org/10.1007/s00138-010-0266-z`.
- **J. Ralli**, J. Díaz, P. Guzmán and E. Ros, 'Complementary Image Representation Spaces in Variational Disparity Calculation', enviado para publicación.
- **J. Ralli**, J. Díaz and E. Ros, 'Spatial and Temporal Constraints in Variational Correspondence Methods', Machine Vision and Applications, 2011, url:`http://dx.doi.org/10.1007/s00138-011-0360-x`.
- **J. Ralli**, J. Díaz, E. Ros, J. Ilonen, and V. Kyrki, 'External Constraints in Variational Disparity Calculation: Hypothesis-Forming-Validation-Loops and Segmentation', enviado para publicación.
- N.R. Luque, J.A. Garrido, **J. Ralli**, J.J Laredo, and E. Ros, 'From Sensors to Spikes: Evolving Receptive Fields to Enhance Sensori-Motor Information in a Robot-Arm', enviado para publicación.
- P. Guzmán, J. Díaz, **J. Ralli**, R. Agís, and E. Ros, 'Low-cost Sensor to Detect Overtaking Based on Optical-flow', aceptado para publicación en Machine Vision and Applications.

Esto representa una producción científica de cinco publicaciones, enviadas a revistas científicas relevantes, donde el doctorando es el primer autor, un articulo relacionado a un congreso internacional, dos colaboraciones en publicaciones enviadas a revistas científicas y una colaboración en una publicacin enviada a un congreso nacional.

7.4 Contribuciones Principales

A continuación se presentan las principales aportaciones de este trabajo.

- Hemos demostrado que la representación de la imagen influye considerablemente en la calidad de los mapas de disparidad.
- Hemos probado varias representaciones distintas y hemos clasificado estas de acuerdo a los resultados obtenidos. Las mejores representaciones, según nuestro estudio, están basadas en la gradiente de la imagen.
- Hemos creado una metodología para obtener parámetros pseudo-óptimos relacionados a diferentes representaciones en el cálculo de disparidad. La metodología es suficientemente genérica para ser usada en otros problemas de tratamiento de imágenes.
- Hemos demostrado que utilizando información a priori se puede mejorar, considerablemente, los resultados en los cálculos de flujo-óptico y/o disparidad. Por lo tanto, el procesamiento de imágenes basado en los resultados mejorados, por ejemplo segmentación, también se mejorará considerablemente.
- Esta información a priori se puede extraer e imponer automáticamente utilizando 'hypothesis-forming-validation loop' (bucles de formación y convalidación de hipótesis).
- Hemos demostrado que la segmentación de mapas de disparidad en superficies que representan objetos (o partes de objetos) es posible.
- Hemos validado el método variacional como un marco eficaz para la solución de varios problemas de visión artificial. El objetivo de esta tesis fué el de desarrollar métodos capaces de trabajar con los datos de 'mundo real', y por tanto, la robustez fué uno de los aspectos más importantes.

References

[1] R. Acar and C. R. Vogel. Analysis of bounded variation penalty methods for ill-posed problems. In *Inverse Problems*, pages 1217–1229, 1994. 37, 79

[2] L. Alvarez, J. Escalarín, M. Lefébure, and J. Sánchez. A pde model for computing the optical flow. In *CEDYA XVI, Universidad de Las Palmas de Gran Canaria*, pages 1349–1356, 1999. 79, 139, 156

[3] L. Alvarez, J. Weickert, and J. Sánchez. Reliable estimation of dense optical flow fields with large displacements. *Int. J. Comput. Vision*, 39(1):41–56, 2000. 8, 20, 36, 37, 38, 77, 78, 100, 101, 139, 155, 156

[4] L. Alvarez, R. Deriche, Théo Papadopoulo, and J. Sanchez. Symmetrical dense optical flow estimation with occlusion detection. In *In ECCV*, pages 721–735, 2002. 46, 47

[5] Luis Alvarez, Rachid Deriche, Javier Snchez, and Joachim Weickert. Dense disparity map estimation respecting image discontinuities: A pde and scale-space based approach. *JOURNAL OF VISUAL COMMUNICATION AND IMAGE REPRESENTATION*, 13:3–21, 2000. 8, 20, 40

[6] S. Baker, D. Scharstein, J.P. Lewis, S. Roth, M.J. Black, and R. Szeliski. A database and evaluation methodology for optical flow. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8, 2007. 82, 114

[7] J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12:43–77, 1994. 83

[8] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (surf). *Comput. Vis. Image Underst.*, 110:346–359, June 2008. 50

[9] M. Björkman and D. Kragic. Active 3d segmentation through fixation of previously unseen objects. In *Proceedings of the British Machine Vision Conference*, pages 119.1–119.11. BMVA Press, 2010. ISBN 1-901725-40-5. doi:10.5244/C.24.119. 48

[10] M. Black. Recursive non-linear estimation of discontinuous flow fields. In *In Third European Conference on Computer Vision*, pages 138–145, 1994. 74, 98

[11] M. Black and P. Anandan. Robust dynamic motion estimation over time. In *Proc. Computer Vision and Pattern Recognition*, pages 296–302, 1991. 74, 75, 98

[12] A. Blake and A. Zisserman. *Visual Reconstruction.* The MIT Press, 1987. 39, 156

[13] T. Brox. *From Pixels to Regions: Partial Differential Equations in Image Analysis.* PhD thesis, Saarland University, Saarbrücken, Germany, 2005. 8, 20, 36, 38, 50, 79, 103, 108, 125, 145

[14] T. Brox and J. Weickert. Level set segmentation with multiple regions. *IEEE Transactions on Image Processing*, 15(10):3213–3218, October 2006. 103, 108

[15] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In *ECCV04*, pages 25–36, 2004. 36, 38, 77, 79, 100, 101

[16] T. Brox, A. Bruhn, and J. Weickert. Variational motion segmentation with level sets. In *ECCV06*, pages 471–483, 2006. 99

[17] A. Bruhn. *Variational Optic Flow Computation: Accurate Modelling and Efficient Numerics.* PhD thesis, Saarland University, Saarbrücken, Germany, 2006. 32, 50, 100, 125, 139

[18] A. Bruhn, J. Weickert, C. Feddern, T. Kohlberger, and C. Schnorr. Variational optical flow computation in real time. *IEEE Transactions on Image Processing*, 14(5):608–615, 2005. 8, 20, 32, 139

[19] A. Bruhn, J. Weickert, T. Kohlberger, and C. Schnörr. A multigrid platform for real-time motion computation with discontinuity-preserving variational methods. *Int. J. Comput. Vision*, 70(3):257–277, 2006. 32, 108, 139

[20] T.F. Chan and L.A. Vese. Active contours without edges. *IEEE Transactions on Image Processing*, 10(2):266–277, 2001. 102, 103, 110

[21] B. Dellen and F. Wörgötter. Disparity from stereo-segment silhouettes of weakly-textured images. In *BMVC09*, pages 1–11, 2009. 99

[22] P. A. Devijver and J. Kittler. *Pattern recognition: a statistical approach.* Prentice Hall, 1982. 60

[23] M. G. Epitropakis, V. P. Plagianakos, and M. N. Vrahatis. Hardware-friendly higher-order neural network training using distributed evolutionary algorithms. *Appl. Soft Comput.*, 10:398–408, March 2010. 53

[24] H. Farid and E. P. Simoncelli. Differentiation of discrete multidimensional signals. *IEEE Transactions on Image Processing*, 13:496–508, 2004. 144

[25] O.D. Faugeras. What can be seen in three dimensions with an uncalibrated stereo rig. In *ECCV '92: Proceedings of the Second European Conference on Computer Vision*, pages 563–578, 1992. 81

[26] M.A. Fischler and R.C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, 1981. 109

[27] D.J. Fleet and A.D. Jepson. Phase-based disparity measurement. *Computer Vision Graphics and Image Processing*, 53(2):198–210, 1991. 5, 17, 56, 57

[28] D.J. Fleet and A.D. Jepson. Stability of phase information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(12):1253–1268, 1993. 5, 17, 56, 57

[29] R.C Gonzalez, R.E Woods, and S.L Eddins. *Digital Image Processing Using MATLAB(R).* Gatesmark Publishing, 2003. 61

[30] R. I. Hartley. Estimation of relative camera positions for uncalibrated cameras. In *ECCV '92: Proceedings of the Second European Conference on Computer Vision*, pages 579–587, 1992. 81

[31] R.I Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004. 28, 41

[32] H. Hirschmuller and D. Scharstein. Evaluation of cost functions for stereo matching. In *CVPR07*, pages 1–8, 2007. 82, 114

[33] Berthold K. P. Horn and Brian G. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981. 34, 37, 48, 140

[34] Yingping Huang and Ken Young. Binocular image sequence analysis: Integration of stereo disparity and optic flow for improved obstacle detection and tracking. *EURASIP Journal on Advances in Signal Processing*, page 10, 2008. doi: 10.1155/2008/843232. 48

[35] D.H. Hubel and T.N. Wiesel. Anatomical demonstration of columns in the monkey striate cortex. *Nature*, 221:747–750, 1969. 5, 17, 57

[36] L. I.Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Phys. D*, 60:259–268, November 1992. 38, 42

[37] S. Kalkan, S. Yan, V. Krüger, F. Wörgötter, and N. Krüger. A signal-symbol loop mechanism for enhanced edge extraction. In *International Conference on Computer Vision Theory and Applications VISAPP'08*, pages 214–221, 2008. 74

[38] E. R. Kandell, J.H. Schwartz, and T.M. Jessell. *Principles of Neural Science*. Elsevier, 1991. 5, 7, 17, 19, 97, 176, 182

[39] A. Klaus, M. Sormann, and K. Karner. Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure. In *ICPR '06: Proceedings of the 18th International Conference on Pattern Recognition*, pages 15–18, 2006. 99

[40] R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, pages 1137–1143. Morgan Kaufmann, 1995. 60

[41] N. Krüger. Three dilemmas of signal- and symbol-based representations in computer vision. In *Proceedings of the workshop Brain, Vision and Artificial Intelligence*, volume 3704, pages 167–176, 2005. 74

[42] G. Kühne, J. Weickert, M. Beier, and W. Effelsberg. Fast implicit active contour models. In *Proceedings of the 24th DAGM Symposium on Pattern Recognition*, pages 133–140. Springer-Verlag, 2002. 111, 171

[43] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60:91–110, November 2004. 50

[44] N.R. Luque, J.A. Garrido, R.R. Carrillo, O.J.D. Coenen, and E. Ros. Cerebellar input configuration toward object model abstraction in manipulation tasks. *Neural Networks, IEEE Transactions on*, 22(8):1321 –1328, aug. 2011. 5, 17

[45] N.R. Luque, J.A. Garrido, R.R. Carrillo, and S. Toluand E. Ros. Adaptive cerebellar spiking model embedded in the control loop: context switching and robustness against noise. *Int. J. Neural Syst*, 21(5):385–401, 2011. 5, 17

[46] I. Markelić, , A. Kjær-Nielsen, K. Pauwels, L. Baunegaard With Jensen, N. Chumerin, A. Vidugiriene, M. Tamosiunaite, A. Rotter, M. Van Hulle, N. Krüger, and F. Wörgötter. The driving school system: Learning basic driving skills from a teacher in a real car. *Intelligent Transportation Systems, IEEE Transactions on*, PP(99):1 –12, 2011. doi: 10.1109/TITS.2011.2157690. 3, 15

[47] B.A. Maxwell, R.M. Friedhoff, and C.A. Smith. A bi-illuminant dichromatic reflection model for understanding images. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, June 2008. 49

[48] Y. Mileva, A. Bruhn, and J. Weickert. Illumination-robust variational optical flow with photometric invariants. In *DAGM07*, pages 152–162, 2007. 32, 48, 50, 53

[49] K.W. Morton and D.F. Mayers. *Numerical Solution of Partial Differential Equations*. Cambridge University Press, 2005. 143

[50] H.H. Nagel and W. Enkelmann. An investigation of smoothness constraints for the estimation of displacement vector fields from image sequences. *PAMI*, 8(5): 565–593, September 1986. 36, 37, 78

[51] S. Osher and R. Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces*. Springer-Verlag, 2003. ISBN 978 0 387 95482 0. 91, 94, 95, 143

[52] S. Osher and J. A. Sethian. Fronts propagating with curvature dependent speed: Algorithms based on hamilton-jacobi formulations. *JOURNAL OF COMPUTATIONAL PHYSICS*, 79(1):12–49, 1988. 95

[53] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:629–639, 1990. 35, 42, 79, 144, 145

[54] Vassilis P. Plagianakos and Michael N. Vrahatis. Parallel evolutionary training algorithms for hardware-friendly neural networks. *Natural Computing*, 1:307–322, 2002. 53

[55] N. Pugeault, F. Wörgötter, and N. Krüger. Multi-modal scene reconstruction using perceptual grouping constraints. In *Proceedings of the 5th IEEE Computer Society Workshop on Perceptual Organization in Computer Vision,New York City June 22, 2006 (in conjunction with IEEE CVPR 2006)*, 2006. 8, 20

[56] J. Ralli, J. Díaz, and E. Ros. A method for sparse disparity densification using voting mask propagation. *Journal of Visual Communication and Image Representation*, 21(1):67–74, 2009. URL `http://dx.doi.org/10.1016/j.jvcir.2009.08.005`. 177, 183

[57] J. Ralli, J. Díaz, S. Kalkan, N. Krüger, and E. Ros. Disparity disambiguation by fusion of signal- and symbolic-level information. *Machine Vision and Applications*, 2010. doi: 10.1007/s00138-010-0266-z. URL `http://dx.doi.org/10.1007/s00138-010-0266-z`. 74, 81, 114, 124, 176, 177, 183

[58] J. Ralli, J. Díaz, E. Ros, J. Ilonen, and V. Kyrki. External constraints in variational disparity calculation: Hypothesis-forming-validation-loops and segmentation. *submitted for publication*, 2010. 60, 88, 142, 176, 177, 183, 184

[59] J. Ralli, J. Díaz, P. Guzmán, and E. Ros. Complementary image representation spaces in variational disparity calculation. *submitted for publication*, 2011. 32, 37, 78, 100, 177, 183

[60] J. Ralli, J. Díaz, and E. Ros. Spatial and temporal constraints in variational correspondence methods. *Machine Vision and Applications*, 2011. URL `http://dx.doi.org/10.1007/s00138-011-0360-x`. 98, 99, 100, 176, 177, 183

[61] S.P. Sabatini, G. Gastaldi, F. Solari, K. Pauwels, M.M. Van Hulle, J. Díaz, E. Ros, N. Pugeault, and N. Krüger. A compact harmonic code for early vision based on anisotropic frequency channels. *Comput. Vis. Image Underst.*, 114:681–699, 2010. 56

[62] A. Salgado and J. Sánchez. Temporal constraints in large optical flow estimation. In *EUROCAST*, pages 709–716, 2007. 74, 98

[63] J. Sánchez. *Estimación Del Flujo Óptico en Secuencias de Imágenes y de la Carta de Disparidad en Pars Estéreo: Aplicación a la Reconstrucción Tridimensional.* PhD thesis, Universidad de Las Palmas de Gran Canaria, Spain, 2001. 8, 20, 125

[64] S.A. Shafer. Using color to separate reflection components. Technical report, 1984. TR 136, Computer Science Department, University of Rochester. 49

[65] E P Simoncelli. *Distributed Analysis and Representation of Visual Motion.* PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, January 1993. Also available as MIT Media Laboratory Vision and Modeling Technical Report #209. 144

[66] E. P. Simoncelli. Design of multi-dimensional derivative filters. In *In First International Conference on Image Processing*, pages 790–793, 1994. 143, 144

[67] N. Slesareva, A. Bruhn, and J. Weickert. Optic flow goes stereo: A variational method for estimating discontinuity-preserving dense disparity maps. In *DAGM05*, pages 33–40, 2005. 40

[68] F. Steinbruecker, T. Pock, and D. Cremers. Large displacement optical flow computation without warping. 2009. 45

[69] C.V Stewart. Robust parameter estimation in computer vision. *SIAM Rev.*, 41 (3):513–537, 1999. 109

[70] R. Storn and K. Price. Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces. Technical report, 1995. TR-95-012, ICSI. 52

[71] R. Storn and K. Price. Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11(4): 341–359, 1997. 52

[72] D. K. Tasoulis, N.G. Pavlidis, V. P. Plagianakos, and M. N. Vrahatis. Parallel differential evolution. In *In IEEE Congress on Evolutionary Computation (CEC*, pages 1–6, 2004. 53

[73] A.N. Tikhonov and V.Y. Arsenin. *Solutions of Ill-Posed Problems.* V. H. Winston & Sons, Washington, D.C.: John Wiley & Sons, New York,, 1977. 35

[74] U. Trottenberg, C. Oosterlee, and A. Schüller. *Multigrid.* Academic Press, 2001. 38, 108, 125, 128, 131, 136, 137, 199

[75] S. O. Unverdi and G. Tryggvason. A front-tracking method for viscous, incompressible, multi-fluid flows. *J. Comput. Phys.*, 100:25–37, 1992. 94

[76] L. Valgaerts, A. Bruhn, and J. Weickert. A variational model for the joint recovery of the fundamental matrix and the optical flow. In *Proceedings of the 30th DAGM symposium on Pattern Recognition*, pages 314–324, 2008. 81

[77] A. Wedel, T. Pock, J. Braun, U. Franke, and D. Cremers. Duality tv-l1 flow with fundamental matrix prior. In *IVCNZ08*, pages 1–6, 2008. 81

[78] J. Weickert. Application of nonlinear diffusion in image processing and computer vision. *Acta Mathematica Universitatis Comenianae*, 70(1):33–50, 2001. 74, 171

[79] J. Weickert and C. Schnörr. A theoretical framework for convex regularizers in pde-based computation of image motion. *Int. J. Comput. Vision*, 45(3):245–264, 2001. 98

[80] J. Weickert, B. M. Ter Haar Romeny, and M. A. Viergever. Efficient and reliable schemes for nonlinear diffusion filtering. *IEEE Transactions on Image Processing*, 7:398–410, 1998. 111, 134

[81] M. Werlberger, W. Trobin, T. Pock, A. Wedel, D. Cremers, and H. Bischof. Anisotropic huber-l1 optical flow. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2009. 74, 98

[82] C. Wöhler and P. d'Angelo. Stereo image analysis of non-lambertian surfaces. *International Journal of Computer Vision*, 81(2):172–190, 2009. 50

[83] Q. Yang, C. Engels, and A. Akbarzadeh. Near real-time stereo for weakly-textured scenes. In *BMVC*, pages 80–87, 2008. 97, 99

[84] D. M. Young. *Iterative Methods for Solving Partial DIfference Equations of Elliptical Type*. PhD thesis, Harvard Universidad, 1995. 131

[85] Y. L. Zhang, K. S. Yeo, B. C. Khoo, and C. Wang. 3d impact and toroidal bubbles. *J. Comput. Phys.*, 166:336–360, 2001. 94

[86] S.C. Zhu and A. Yuille. Region competition: Unifying snakes, region growing, and bayes/mdl for multiband image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(9):884–900, 1996. 102, 103, 109

[87] H. Zimmer, A. Bruhn, J. Weickert, L. Valgaerts, A. Salgado, B. Rosenhahn, and H.P. Seidel. Complementary optic flow. In *EMMCVPR*, volume 5681 of *Lecture Notes in Computer Science*, pages 207–220, 2009. 37, 42, 79

Appendix A

Euler-Lagrange Equations

A.1 Temporal Constraint for Optical-Flow

Here, we show a derivation of the Euler-Lagrange equations for a given optical-flow formulation. The energy functional has two data terms, gradient direction and magnitude of the gradient, a temporal constraint, and a flow-based regularisation term. As can be observed, non-linearised constancy assumptions are used in the data terms. One of the implications of the non-linearised constancy terms is that the model copes better with large displacements. Firstly, the energy functional is given in (A.1) and the related Euler-Lagrange equations are given in (A.3).

A.1.1 Energy Functional

$$\begin{aligned} E(u,v) = \int_{\Omega} \sum_{k=1}^{K} \Bigg\{ & b_1 \Psi_D \left((\frac{\partial I_{1,k}}{\partial x} - \frac{\partial I^w_{0,k}}{\partial x})^2 \right) + b_1 \Psi_D \left((\frac{\partial I_{1,k}}{\partial y} - \frac{\partial I^w_{0,k}}{\partial y})^2 \right) \\ & + b_2 \Psi_D \left((\frac{\partial I_{1,k}}{\partial x} - \frac{\partial I^w_{0,k}}{\partial x})^2 + (\frac{\partial I_{1,k}}{\partial y} - \frac{\partial I^w_{0,k}}{\partial y})^2 \right) \\ & + \gamma_t \Psi_{CT} \left((u_{tc} - u)^2 \right) + \gamma_t \Psi_{CT} \left((v_{tc} - v)^2 \right) \\ & + \alpha \Psi_R \left(|\nabla u|^2 + |\nabla v|^2 \right) \Bigg\} \mathbf{dx} \end{aligned} \tag{A.1}$$

where $I_{1,k} = I(x, y, k, t = 1)$, $I^w_{0,k} = I(x + u, y + v, k, t = 0)$, and u_{tc} and v_{tc} are the constraints. The robust functions are as explained in Section 3.10.7. In other words, we are looking for a transformation defined by (u, v) that warps the image from $t = 0$ to $t = 1$. Typically, bilinear- or bicubic interpolation is used for the transformation.

The curly brackets have no special meaning and are simply used in order to make the equation more readable.

A.1.2 Related Euler-Lagrange Equations

Firstly, we introduce some auxiliary variables in order to simplify the notation.

$$\begin{aligned} E_{D1} &= \left(\frac{\partial I_{1,k}}{\partial x} - \frac{\partial I^w_{0,k}}{\partial x}\right)^2 \\ E_{D2} &= \left(\frac{\partial I_{1,k}}{\partial y} - \frac{\partial I^w_{0,k}}{\partial y}\right)^2 \\ E_{D3} &= \left(\frac{\partial I_{1,k}}{\partial x} - \frac{\partial I^w_{0,k}}{\partial x}\right)^2 + \left(\frac{\partial I_{1,k}}{\partial y} - \frac{\partial I^w_{0,k}}{\partial y}\right)^2 \end{aligned} \tag{A.2}$$

Using the above mentioned auxiliary variables, the Euler-Lagrange equations can be written as follows:

$$\begin{aligned} & b_1 \sum_{k=1}^{K} \Psi'_D\left(E_{D1}\right)\left(\frac{\partial I_{1,k}}{\partial x} - \frac{\partial I^w_{0,k}}{\partial x}\right)\left(\frac{\partial^2 I_{1,k}}{\partial x^2}\right) + b_1 \sum_{k=1}^{K} \Psi'_D\left(E_{D2}\right)\left(\frac{\partial I_{1,k}}{\partial y} - \frac{\partial I^w_{0,k}}{\partial y}\right)\left(\frac{\partial^2 I_{1,k}}{\partial y \partial x}\right) \\ & + b_2 \sum_{k=1}^{K} \Psi'_D\left(E_{D3}\right)\left(\frac{\partial I_{1,k}}{\partial x} - \frac{\partial I^w_{0,k}}{\partial x}\right)\left(\frac{\partial^2 I_{1,k}}{\partial x^2}\right) + b_2 \sum_{k=1}^{K} \Psi'_D\left(E_{D3}\right)\left(\frac{\partial I_{1,k}}{\partial y} - \frac{\partial I^w_{0,k}}{\partial y}\right)\left(\frac{\partial^2 I_{1,k}}{\partial y \partial x}\right) \\ & + K\gamma_t \Psi'_{CT}\left((u_{tc} - u)^2\right)(u_{tc} - u) + K\alpha DIV\Big(\Psi'_R\left(|\nabla u|^2 + |\nabla v|^2\right)\nabla u\Big) = 0 \\ & b_1 \sum_{k=1}^{K} \Psi'_D\left(E_{D1}\right)\left(\frac{\partial I_{1,k}}{\partial x} - \frac{\partial I^w_{0,k}}{\partial x}\right)\left(\frac{\partial^2 I_{1,k}}{\partial x \partial y}\right) + b_1 \sum_{k=1}^{K} \Psi'_D\left(E_{D2}\right)\left(\frac{\partial I_{1,k}}{\partial y} - \frac{\partial I^w_{0,k}}{\partial y}\right)\left(\frac{\partial^2 I_{1,k}}{\partial y^2}\right) \\ & + b_2 \sum_{k=1}^{K} \Psi'_D\left(E_{D3}\right)\left(\frac{\partial I_{1,k}}{\partial x} - \frac{\partial I^w_{0,k}}{\partial x}\right)\left(\frac{\partial^2 I_{1,k}}{\partial x \partial y}\right) + b_2 \sum_{k=1}^{K} \Psi'_D\left(E_{D3}\right)\left(\frac{\partial I_{1,k}}{\partial y} - \frac{\partial I^w_{0,k}}{\partial y}\right)\left(\frac{\partial^2 I_{1,k}}{\partial y^2}\right) \\ & + K\gamma_t \Psi'_{CT}\left((v_{tc} - v)^2\right)(v_{tc} - v) + K\alpha DIV\Big(\Psi'_R\left(|\nabla u|^2 + |\nabla v|^2\right)\nabla v\Big) = 0 \end{aligned} \tag{A.3}$$

At a first look, it might seem that the gradient and the gradient magnitude data terms contain redundant information, but this is certainly not the case: gradient contains directional information (implicitly also magnitude), while gradient magnitude term contains the magnitude explicitly without directional information. This is clear from the robust functions as can be observed from E_{D1}, E_{D2}, and E_{D3}.

Appendix B

Solvers

B.1 Optical-flow

B.1.1 Discrete Differential Operator

We have added this appendix in order to link the notation typically used in the machine vision literature with the notation/terminology used typically with PDEs. Hopefully it will further clarify construction of the solvers as given in Section 5. As it was shown in Section 5.6.1, using matrix/vector notation is not always the most convenient way of describing PDEs. From discretisation point of view it is more convenient to talk about *discrete differential operators* rather than matrices, and of *grid functions* rather than vectors.

For the definition of discrete differential operators, L_h, we use a 'stencil' notation as in [74]. A general stencil

$$[s_{k_1k_2}]_h = \begin{bmatrix} & \vdots & \vdots & \vdots & \\ \cdots & s_{-1,-1} & s_{0,-1} & s_{1,-1} & \cdots \\ \cdots & s_{-1,0} & s_{0,0} & s_{1,0} & \cdots \\ \cdots & s_{-1,1} & s_{0,1} & s_{1,1} & \cdots \\ & \vdots & \vdots & \vdots & \end{bmatrix}_h \quad (s_{k_1k_2} \in \mathbb{R}) \tag{B.1}$$

defines an operator on the grid functions (in our case u, v, du or dv) as defined by Equation B.2

$$[s_{k_1k_2}]_h \, w(i,j) = \sum_{k_1,k_2} s_{k_1k_2} w(i + k_1h_x, j + k_2h_y) \tag{B.2}$$

where $w(i,j)$ denotes the grid function. The kind of stencil that we use in this thesis is a so called *5-point* stencil, as defined next.

$$\begin{bmatrix} & s_{0,-1} & \\ s_{-1,0} & s_{0,0} & s_{1,0} \\ & s_{0,1} & \end{bmatrix}_h \tag{B.3}$$

Other stencils are, for example, *9-point* stencil and *27-point* stencil (in 3D neighbourhoods).

B.1.1.1 Early Linearisation

From Equation 5.39 we can easily detect the discrete differential operators, as given in Equation B.4.

$$\begin{aligned}
&\underbrace{K\alpha\Big(u^{l+1}_{i+1,j} - 2u^{l+1}_{i,j} + u^{l+1}_{i-1,j} + u^{l+1}_{i,j+1} - 2u^{l+1}_{i,j} + u^{l+1}_{i,j-1}\Big) - \sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial x}\right)^2_{i,j} u^{l+1}_{i,j}}_{L_h u} = \\
&\qquad \underbrace{\sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial y}\frac{\partial I_k}{\partial x}\right)_{i,j} v^{l}_{i,j} - \sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial t}\frac{\partial I_k}{\partial x}\right)_{i,j}}_{f} \\
&\underbrace{K\alpha\Big(v^{l+1}_{i+1,j} - 2v^{l+1}_{i,j} + v^{l+1}_{i-1,j} + v^{l+1}_{i,j+1} - 2v^{l+1}_{i,j} + v^{l+1}_{i,j-1}\Big) - \sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial y}\right)^2_{i,j} v^{l+1}_{i,j}}_{L_h v} = \\
&\qquad \underbrace{\sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial x}\frac{\partial I_k}{\partial y}\right)_{i,j} u^{l}_{i,j} - \sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial t}\frac{\partial I_k}{\partial y}\right)_{i,j}}_{f}
\end{aligned} \tag{B.4}$$

Now we can construct the discrete differential stencils as given in equations B.5 and B.6.

$$\begin{bmatrix} & K\alpha & \\ K\alpha & -4 - \sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial x}\right)^2_{i,j} & K\alpha \\ & K\alpha & \end{bmatrix}_h u^{l+1}_{i,j} = \sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial y}\frac{\partial I_k}{\partial x}\right)_{i,j} v^{l}_{i,j} - \sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial t}\frac{\partial I_k}{\partial x}\right)_{i,j} \tag{B.5}$$

$$\begin{bmatrix} & K\alpha & \\ K\alpha & -4-\sum\limits_{k=1}^{K}\left(\frac{\partial I_k}{\partial y}\right)^2_{i,j} & K\alpha \\ & K\alpha & \end{bmatrix}_h v_{i,j}^{l+1} = \sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial x}\frac{\partial I_k}{\partial y}\right)_{i,j} u_{i,j}^{l} - \sum_{k=1}^{K}\left(\frac{\partial I_k}{\partial t}\frac{\partial I_k}{\partial y}\right)_{i,j} \tag{B.6}$$

Figure B.1 visualises the correspondence between the stencil notation and the grid.

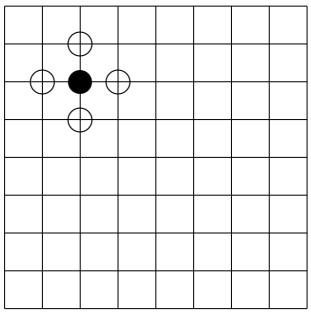


Figure B.1

B.1.1.2 Late Linearisation

Here we only identify the discrete differential operators from Equation 5.58. Constructing the stencil(s) is similar than in the early linearisation case.

$$\begin{aligned}
&\underbrace{\begin{aligned}
&K\alpha\Psi'\left(E_S^{l,m}\right)_N\left(u_{i-1,j}^{l}-u_{i,j}^{l}+du_{i-1,j}^{l,m+1}-du_{i,j}^{l,m+1}\right)\\
&+K\alpha\Psi'\left(E_S^{l,m}\right)_S\left(u_{i+1,j}^{l}-u_{i,j}^{l}+du_{i+1,j}^{l,m+1}-du_{i,j}^{l,m+1}\right)\\
&+K\alpha\Psi'\left(E_S^{l,m}\right)_W\left(u_{i,j-1}^{l}-u_{i,j}^{l}+du_{i,j-1}^{l,m+1}-du_{i,j}^{l,m+1}\right)\\
&+K\alpha\Psi'\left(E_S^{l,m}\right)_E\left(u_{i,j+1}^{l}-u_{i,j}^{l}+du_{i,j+1}^{l,m+1}-du_{i,j}^{l,m+1}\right)\\
&-\sum_{k=1}^{K}\Psi'\left((E_k^{l,m})_D\right)_{i,j}\left(\frac{\partial I_{k,0}^{w(l)}}{\partial x}\right)^2_{i,j} du_{i,j}^{l,m+1}
\end{aligned}}_{L_h du}\\
&\qquad\underbrace{=\sum_{k=1}^{K}\Psi'\left((E_k^{l,m})_D\right)_{i,j}\left(\left(\frac{\partial I_{k,0}^{w(l)}}{\partial y}\frac{\partial I_{k,0}^{w(l)}}{\partial x}\right)_{i,j} dv_{i,j}^{l,m}-\left(\frac{\partial I_k^{l}}{\partial t}\frac{\partial I_{k,0}^{w(l)}}{\partial x}\right)_{i,j}\right)}_{f}
\end{aligned} \tag{B.7}$$

$$
\underbrace{\begin{aligned}
&K\alpha\Psi'\left(E_S^{l,m}\right)_N\left(v_{i-1,j}^l - v_{i,j}^l + dv_{i-1,j}^{l,m+1} - dv_{i,j}^{l,m+1}\right)\\
&+K\alpha\Psi'\left(E_S^{l,m}\right)_S\left(v_{i+1,j}^l - v_{i,j}^l + dv_{i+1,j}^{l,m+1} - dv_{i,j}^{l,m+1}\right)\\
&+K\alpha\Psi'\left(E_S^{l,m}\right)_W\left(v_{i,j-1}^l - v_{i,j}^l + dv_{i,j-1}^{l,m+1} - dv_{i,j}^{l,m+1}\right)\\
&+K\alpha\Psi'\left(E_S^{l,m}\right)_E\left(v_{i,j+1}^l - v_{i,j}^l + dv_{i,j+1}^{l,m+1} - dv_{i,j}^{l,m+1}\right)\\
&-\sum_{k=1}^{K}\Psi'\left((E_k^{l,m})_D\right)_{i,j}\left(\frac{\partial I_{k,0}^{w(l)}}{\partial y}\right)_{i,j}^2 dv_{i,j}^{l,m+1}
\end{aligned}}_{L_h dv}
$$

$$
\underbrace{=\sum_{k=1}^{K}\Psi'\left((E_k^{l,m})_D\right)_{i,j}\left(\left(\frac{\partial I_{k,0}^{w(l)}}{\partial x}\frac{\partial I_{k,0}^{w(l)}}{\partial y}\right)_{i,j} du_{i,j}^{l,m} - \left(\frac{\partial I_k^l}{\partial t}\frac{\partial I_{k,0}^{w(l)}}{\partial y}\right)_{i,j}\right)}_{f} \tag{B.8}
$$

Appendix C

Matlab/MEX Code

C.1 Introduction

Code related to the early- and late linearisation cases for optical-flow and disparity calculation, related to this thesis, can be downloaded from `http://atc.ugr.es/~jarnor/`. The code is a combination of Matlab and MEX (programmed in C) code. Parts of the code were implemented in C in order to improve the computational efficiency. The implementations are based on a coarse-to-fine strategy (see sections 5.6.1.1 and 5.6.2.1 for more information). With the explanations given in this thesis (along with the comments in the code), an interested reader should be able to understand the code without major complications.

The code uses only minimal, if any, error checking. The main purpose of 'scientific' computer programs is to validate the concepts that they are related to. This is the case with this code as well. Also, the code was written in such a way, that it should be clear what each part does, at the same time sacrificing some computational efficiency.

C.2 Notation Used in the Matlab Code

In order to write the programs in Matlab, we need to simplify the notation. This is done as follows.

C.2.1 Early Linearisation

We denote the terms as given in (C.1).

$$
\begin{aligned}
M_{i,j} &= \left(\frac{\partial I_k}{\partial y}\frac{\partial I_k}{\partial x}\right)_{i,j} \\
Cu_{i,j} &= \left(\frac{\partial I_k}{\partial t}\frac{\partial I_k}{\partial x}\right)_{i,j} \\
Cv_{i,j} &= \left(\frac{\partial I_k}{\partial t}\frac{\partial I_k}{\partial y}\right)_{i,j} \\
Du_{i,j} &= \left(\frac{\partial I_k}{\partial x}\right)^2_{i,j} \\
Dv_{i,j} &= \left(\frac{\partial I_k}{\partial y}\right)^2_{i,j}
\end{aligned}
\tag{C.1}
$$

With the above definitions, we can write the discretised, early linearisation, equations for u and v as follows:

$$
\begin{aligned}
K\alpha\Big(u^{l+1}_{i+1,j} - 2u^{l+1}_{i,j} + u^{l+1}_{i-1,j} + u^{l+1}_{i,j+1} - 2u^{l+1}_{i,j} + u^{l+1}_{i,j-1}\Big) - \sum_{k=1}^{K} Du_{i,j}u^{l+1}_{i,j} = \\
\sum_{k=1}^{K} M_{i,j}v^{l}_{i,j} - \sum_{k=1}^{K} Cu_{i,j} \\
K\alpha\Big(v^{l+1}_{i+1,j} - 2v^{l+1}_{i,j} + v^{l+1}_{i-1,j} + v^{l+1}_{i,j+1} - 2v^{l+1}_{i,j} + v^{l+1}_{i,j-1}\Big) - \sum_{k=1}^{K} Dv_{i,j}v^{l+1}_{i,j} = \\
\sum_{k=1}^{K} M_{i,j}u^{l}_{i,j} - \sum_{k=1}^{K} Cv_{i,j}
\end{aligned}
\tag{C.2}
$$

C.2.1.1 Solver

The solver for the early linearisation case is called *opsor_elin_2d*. It is a SOR-type solver with two inbuilt solvers: GS and ALR. The used solver (GS or ALR) is chosen with the parameter *solver* (1=GS, 2=ALR). As it can be seen from the function definition below, the solver actually works with diffusion weights (*wW, wN, wE and wS*) and with robust error function for the data term. In Horn&Schunck case, diffusivity is 1 in all the directions and the influence term, *Gd* (for explanation, see late linearisation case below), is simply omitted or assumed to be 1.

$$
\begin{aligned}
[U^{l+iter}, V^{l+iter}] = \text{opsor_elin4_2d}\Big(& U^l, \\
& V^l, \\
& MGd, \\
& CuGd, \\
& CvGd, \\
& DuGd, \\
& DvGd, \\
& wW, \quad \text{//diffusion weights, West} \\
& wN, \quad \text{//diffusion weights, North} \\
& wE, \quad \text{//diffusion weights, East} \\
& wS, \quad \text{//diffusion weights, South} \\
& iter, \quad \text{//number of iterations} \\
& omega, \quad \text{//SOR parameter } \omega \\
& solver \quad \text{//type of solver to use}\Big);
\end{aligned}
\tag{C.3}
$$

where U^l, V^l, MGd, $CuGd$, $CvGd$, $DuGd$, $DvGd$, wW, wN, wE, $wS \in \mathbb{R}^N$ (N is the number of pixels in the input image).

C.2.2 Late Linearisation

We denote the terms and the diffusion weights as given in (C.4)

$$
\begin{aligned}
Gd_{i,j} &= \sum_{k=1}^{K} \Psi'\Big((E_k^{l,m})_D\Big)_{i,j} \\
M_{i,j} &= \left(\frac{\partial I_{k,0}^{w(l)}}{\partial x}\frac{\partial I_{k,0}^{w(l)}}{\partial y}\right)_{i,j} \\
Cu_{i,j} &= \left(\frac{\partial I_k^l}{\partial t}\frac{\partial I_{k,0}^{w(l)}}{\partial x}\right)_{i,j} \\
Cv_{i,j} &= \left(\frac{\partial I_k^l}{\partial t}\frac{\partial I_{k,0}^{w(l)}}{\partial y}\right)_{i,j} \\
Du_{i,j} &= \left(\frac{\partial I_{k,0}^{w(l)}}{\partial x}\right)^2_{i,j} \\
Dv_{i,j} &= \left(\frac{\partial I_{k,0}^{w(l)}}{\partial y}\right)^2_{i,j} \\
W_{\{W,N,E,S\}} &= \Psi'\big(E_R^{l,m}\big)_{\{W,N,E,S\}}
\end{aligned}
\tag{C.4}
$$

With the above definitions, we can write the discretised, late linearisation, equations for du and dv as follows:

$$
\begin{aligned}
&\big(Gd_{i,j}Cu_{i,j}\big) - \big(Gd_{i,j}Du_{i,j}\big)du_{i,j}^{l,m+1} - \big(Gd_{i,j}M_{i,j}\big)dv_{i,j}^{l,m} \\
&\quad +K\alpha W_N\Big(u_{i-1,j}^l - u_{i,j}^l + du_{i-1,j}^{l,m+1} - du_{i,j}^{l,m+1}\Big) \\
&\quad +K\alpha W_S\Big(u_{i+1,j}^l - u_{i,j}^l + du_{i+1,j}^{l,m+1} - du_{i,j}^{l,m+1}\Big) \\
&\quad +K\alpha W_W\Big(u_{i,j-1}^l - u_{i,j}^l + du_{i,j-1}^{l,m+1} - du_{i,j}^{l,m+1}\Big) \\
&\quad +K\alpha W_E\Big(u_{i,j+1}^l - u_{i,j}^l + du_{i,j+1}^{l,m+1} - du_{i,j}^{l,m+1}\Big) = 0 \\
&\big(Gd_{i,j}Cv_{i,j}\big) - \big(Gd_{i,j}Dv_{i,j}\big)dv_{i,j}^{l,m+1} - \big(Gd_{i,j}M_{i,j}\big)du_{i,j}^{l,m} \\
&\quad +K\alpha W_N\Big(v_{i-1,j}^l - v_{i,j}^l + dv_{i-1,j}^{l,m+1} - dv_{i,j}^{l,m+1}\Big) \\
&\quad +K\alpha W_S\Big(v_{i+1,j}^l - v_{i,j}^l + dv_{i+1,j}^{l,m+1} - dv_{i,j}^{l,m+1}\Big) \\
&\quad +K\alpha W_W\Big(v_{i,j-1}^l - v_{i,j}^l + dv_{i,j-1}^{l,m+1} - dv_{i,j}^{l,m+1}\Big) \\
&\quad +K\alpha W_E\Big(v_{i,j+1}^l - v_{i,j}^l + dv_{i,j+1}^{l,m+1} - dv_{i,j}^{l,m+1}\Big) = 0
\end{aligned}
\tag{C.5}
$$

C.2.2.1 Solver

The solver for the late linearisation case is called *opsor_llin4_2d*. It is a SOR-type solver with two inbuilt solvers: GS and ALR. The used solver (GS or ALR) is chosen with the parameter *solver* (1=GS, 2=ALR).

$$
\begin{aligned}
[dU^{n+1},\, dV^{n+1}] = \text{opsor_llin4_2d}\Big(\; & U^m, \\
& V^m, \\
& dU^n, \\
& dV^n, \\
& MGd, \\
& CuGd, \\
& CvGd, \\
& DuGd, \\
& DvGd, \\
& wW, \quad \text{//diffusion weights, West} \\
& wN, \quad \text{//diffusion weights, North} \\
& wE, \quad \text{//diffusion weights, East} \\
& wS, \quad \text{//diffusion weights, South} \\
& iter, \quad \text{//number of iterations} \\
& omega, \quad \text{//SOR parameter } \omega \\
& solver \quad \text{//type of solver to use}\Big);
\end{aligned}
\tag{C.6}
$$

where U^m, V^m, dU^n, dV^n, MGd, $CuGd$, $CvGd$, $DuGd$, $DvGd$, wW, wN, wE, wS $\in \mathbb{R}^N$ (N is the number of pixels in the input image).