Living in a Dynamic World: Semantic Segmentation of Large Scale 3D Environments

Ondrej Miksik
St Catherine's College

DPhil Thesis

Supervised by Prof Philip H. S. Torr
Dr Patrick Pérez

Robotics Research Group
Department of Engineering Science
University of Oxford

Trinity Term 2017

This thesis is submitted to the Department of Engineering Science, University of Oxford, for the degree of Doctor of Philosophy. This thesis is entirely my own work, and, except where otherwise indicated, describes my own research.

Ondrej Miksik
St Catherine's College
University of Oxford

Doctor of Philosophy
Trinity Term
2017

Living in a Dynamic World: Semantic Segmentation of Large Scale 3D Environments

Abstract

As we navigate the world, for example when driving a car from our home to the work place, we continuously perceive the 3D structure of our surroundings and intuitively recognise the objects we see. Such capabilities help us in our everyday lives and enable free and accurate movement even in completely unfamiliar places. We largely take these abilities for granted, but for robots, the task of understanding large outdoor scenes remains extremely challenging.

In this thesis, I develop novel algorithms for (near) real-time dense 3D reconstruction and semantic segmentation of large-scale outdoor scenes from passive cameras. Motivated by "smart glasses" for partially sighted users, I show how such modeling can be integrated into an interactive augmented reality system which puts the user in the loop and allows her to physically interact with the world to learn personalized semantically segmented dense 3D models. In the next part, I show how sparse but very accurate 3D measurements can be incorporated directly into the dense depth estimation process and propose a probabilistic model for incremental dense scene reconstruction. To relax the assumption of a stereo camera, I address dense 3D reconstruction in its monocular form and show how the local model can be improved by joint optimization over depth and pose.

The world around us is not stationary. However, reconstructing dynamically moving and potentially non-rigidly deforming texture-less objects typically require "contour correspondences" for shape-from-silhouettes. Hence, I propose a video segmentation model which encodes a single object instance as a closed curve, maintains correspondences across time and provide very accurate segmentation close to object boundaries.

Finally, instead of evaluating the performance in an isolated setup (IoU scores) which does not measure the impact on decision-making, I show how semantic 3D reconstruction can be incorporated into standard Deep Q-learning to improve decision-making of agents navigating complex 3D environments.

Acknowledgements

Submitting a DPhil thesis forces me to look back and realize how much fortunate I have been over the years.

I would first like to thank to Phil and Patrick. Their initial faith towards me undertaking a DPhil is something I will always be grateful for. Working with Phil has always resembled a roller-coaster ride – his positive "craziness", energy, passion, vision and creativity are just infectious! And Patrick, despite being in the country of wine, cheese and baguettes has always been around with his open mind whenever I needed him. Being your student, I do not regret a single day and would like to thank *you* from bottom of my heart.

I would also like to thank Andrew Fitzgibbon and David Murray for serving as my examiners. They provided many great comments on the manuscript and the viva itself turned out to be a very fun and enjoyable afternoon.

I have always been fortunate to have around myself someone I could count on, no matter how well things progressed. I am forever in debt to Dan Munoz, Vibhav Vineet and Pawan Kumar. I have learnt a lot from *you* and who knows where I would have ended up without *you* guys!

I could not have made it here without critical help of many great researchers and at the same time great mentors who have influenced me significantly. Krystian Mikolajczyk has helped to start my research career many years ago and taught me that computer vision is an experimental science. Martial Hebert and Drew Bagnell hosted me at CMU and showed me how to build robust algorithms from very imperfect ingredients. Shahram Izadi showed me how determined and efficient one could be. And my special thanks goes to "George" Matas who has helped to shape my future numerous times and introduced me to an exciting world of *moving objects*.

One of the best perks of being a student at Oxford is having Andrew Zisserman and Andrea Vedaldi at every single seminar and researchers from all around the world such as Richard Hartley, Ramin Zabih or Carsten Rother regularly returning to our lab and always being keen to any discussion. But this perk is nowhere near having a chance to share the "office" with everyone from TVG/VGG/AVL and OVAL groups. I am not going to name you guys here as that would be a huuuge list. But it is *you* who make this place so special and enjoyable, thanks for that!

I would like to thank Eric, Sunando, Paul, Mike and everyone from OUCKC for making my Oxford experience more fruitful also outside of the lab.

And last but not least, I would like to thank my family and Marketa for never-ending support, encouragement and love over the years.

Contents

1 Introduction **1**

1.1 Motivation and Objective 1

1.2 Challenges 7

1.3 Approach 11

1.4 Contributions 13

1.5 Thesis Outline 16

1.6 Publications 19

2 Preliminaries **21**

2.1 Probabilistic Graphical Models 21

2.1.1 Computer Vision as Labelling Problems 22

2.1.2 Inference in Graphical Models 25

2.2 Parameter Learning 38

2.2.1 Supervised Learning / Empirical Risk Minimization 38

2.2.2 Linear Models 39

2.2.3 Structured Prediction 41

2.3 Multiview Geometry in Computer Vision 43

2.3.1 High-level Overview 44

2.3.2 Geometry of a Monocular Camera 46

2.3.3 Epipolar Geometry 47

2.3.4 Dense Depth Estimation 52

2.3.5 Bundle Adjustment / SLAM 55

2.4 Computer vision tools 57

3 Incremental Dense Semantic Stereo Fusion **63**
3.1 Introduction . . . 63
3.2 Related Work . . . 67
3.2.1 Reconstruction . . . 67
3.2.2 Semantic Segmentation . . . 68
3.3 Large-scale outdoor reconstruction . . . 69
3.3.1 Camera Calibration . . . 69
3.3.2 Depth Estimation . . . 70
3.3.3 Camera Pose Estimation . . . 70
3.3.4 Large-Scale Fusion . . . 70
3.4 Semantic Fusion . . . 72
3.5 Volumetric CRF and Mean-field inference . . . 73
3.5.1 Model . . . 73
3.5.2 Efficient Volumetric Filtering-based Mean-Field Inference . . . 74
3.5.3 Online mean-field . . . 75
3.6 Experiments . . . 76
3.6.1 Qualitative KITTI results. . . . 79
3.6.2 Quantitative KITTI Results . . . 80
3.6.3 Other Qualitative Results . . . 82
3.7 Conclusion . . . 83

4 The Semantic Paintbrush **85**
4.1 Introduction . . . 86
4.2 Related Work . . . 88
4.3 System Overview . . . 90
4.3.1 Hardware . . . 90
4.3.2 User Interface and Interaction . . . 90
4.4 Software Pipeline . . . 91
4.4.1 The Laser Paintbrush . . . 91
4.4.2 Interactive Improvement of Disparity . . . 94
4.4.3 Interactive Learning of Semantic Labels . . . 95
4.5 Augmented Visualization . . . 96

4.5.1 Interactive Reduction of Visual Clutter 96
4.5.2 Map Sharing 97
4.6 Experimental Results 98
4.6.1 Qualitative Results 98
4.6.2 Quantitative Results 98
4.6.3 Computational Efficiency 99
4.7 Applications 100
4.8 Semantic Maps for the Visually Impaired 101
4.9 Other Applications for Semantic Maps 103
4.10 Conclusions 105

5 Incremental Dense Multi-modal 3D Scene Reconstruction 106
5.1 Introduction 106
5.2 Related work 110
5.3 Dense Multi-modal Depth-Map Estimation 111
5.3.1 Setting the stage 111
5.3.2 Pivots 112
5.3.3 Model 113
5.3.4 Unary Potential Function 113
5.3.5 Pairwise Potentials Function 116
5.3.6 Efficient inference 116
5.3.7 Temporal Sequences of Images 117
5.4 Experiments 117
5.4.1 Implementation details 117
5.4.2 Dataset and baselines 118
5.4.3 Qualitative results 119
5.4.4 Quantitative results 120
5.4.5 Limitations 120
5.5 Conclusion 120

6 Dense Monocular 3D Reconstruction 122
6.1 Introduction 123
6.1.1 Contributions 125

6.2 Proposed Energy for Monocular Depth Estimation 126
6.2.1 Photometric Energy 127
6.2.2 Local Spatial Plane Regularizer 128
6.3 Optimization Strategy 129
6.3.1 Constrained Depth Map Updates 130
6.3.2 Simultaneous Pose and Depth Estimation 131
6.3.3 CPU Computation in Realtime 133
6.4 Results 133
6.4.1 Quantitative Evaluation on the TUM Dataset 134
6.4.2 Qualitative Results 137
6.5 Conclusion 138

7 ROAM for Rotoscoping 139
7.1 Introduction 140
7.2 Related work and motivation 141
7.2.1 Rotoscoping and curve-based approaches 141
7.2.2 Masks and region-based approaches 142
7.3 Introducing ROAM 143
7.3.1 Curve-based modelling: E^C 144
7.3.2 Landmark-based modelling: E^L 147
7.3.3 Curve-landmarks interaction: E^J 147
7.4 Using ROAM 147
7.5 Results 151
7.6 Conclusion 161

8 Semantic SLAM for Deep Reinforcement Learning 162
8.1 Introduction 163
8.2 Related work 165
8.2.1 Deep Reinforcement Learning 165
8.2.2 Object detection 166
8.3 Semantic Mapping 167
8.4 Recognition and Reconstruction 168
8.4.1 Object detection 169

8.4.2 Camera pose estimation 169
8.4.3 Mapping 170
8.5 Implementation details 170
8.6 Experiments 172
8.6.1 Oracle Semantic Maps (OSM) 173
8.6.2 Noisy Oracle Semantic Maps (NOSM) 174
8.6.3 Reconstructed Semantic Maps (RSM) 175
8.6.4 Prioritized Duel DQN 176
8.6.5 Mean Run Length 176
8.7 Discussion and Conclusion 177

9 Conclusions **178**
9.1 Summary 178
9.2 Future Directions and Open Questions 179
9.2.1 Object Tracking as Question Answering 179
9.2.2 Modelling (Intuitive) Physics 181
9.2.3 SLAM with Dynamically Moving Objects 182
9.2.4 Future Directions for Semantic Scene Understanding 183
9.2.5 Models with Memory 187
9.3 Concluding Remarks 189

List of Figures

1.1 Atari and Alpha Go 2
1.2 Intermediate representation used by Google Car 3
1.3 Scene understanding 4
1.4 Motivation for this thesis 5
1.5 Sparse vs. dense representations 6
1.6 Instances in Pascal VOC 2012 dataset 8
1.7 Context and co-occurence 8
1.8 Intrinsic scene decomposition 9
1.9 Difficult examples for object recognition 10
1.10 Temporal inconsistency 12
1.11 Dense incremental 3D reconstruction and semantic segmentation 13
1.12 The Semantic Paintbrush 14
1.13 ROAM for video object segmentation 15
1.14 SLAM-Augmented Deep Reinforcement Learning 15
1.15 Dense multi-modal and monocular 3D reconstruction 17

2.1 Computer vision as labelling problems 22
2.2 Grid Random Field 23
2.3 Pairwise MRF and CRF models with 4-neighborhood 24
2.4 Chains, Tree-structured models and Simple Loops 26
2.5 Dynamic programming on a tree 27
2.6 Examples of open-chain, tree-structured and closed chain graphical models 28
2.7 Foreground/background segmentation 30
2.8 α-expansion 31
2.9 Pairwise models 32

2.10 Dense CRF 32
2.11 Dense CRF II 33
2.12 Mean-field approximation 34
2.13 Mean-field factorizations 35
2.14 Polytopes 36
2.15 KL divergence of the mean-field approximation 37
2.16 Influence of pairwise kernel parameters 37
2.17 Regularization 39
2.18 Overview of SfM and SLAM 43
2.19 LIDAR vs. camera 44
2.20 SLAM success stories 45
2.21 Monocular camera, calibrated stereo rig, RGB-D camera, LIDAR 46
2.22 Camera geometry 47
2.23 Epipolar plane 48
2.24 Epipolar constraint 49
2.25 Monocular vs. stereo camera 50
2.26 Pose transformations between three different reference frames 51
2.27 Historical approaches to dense disparity estimation 52
2.28 Limitations of models with 1st order prior 53
2.29 Modern approaches to dense disparity estimation 54
2.30 SLAM motivation 55
2.31 SLAM formulations 56
2.32 Harris corner detector 58
2.33 FAST detector 59
2.34 Local feature descriptors 60
2.35 KD tree. 62

3.1 Incremental reconstruction and semantic segmentation 64
3.2 Overview of our system 66
3.3 Labelled mesh 71
3.4 Voxel hashing 72
3.5 Surface normals 75

3.6 Results 76
3.7 Moving objects 77
3.8 Overlayed segmentation 78
3.9 A close-up view of a semantic model 79
3.10 Quantitative results for depth evaluation 82
3.11 Final labelling surfaces for four reconstructed sequences 82
3.12 Towards CRF-as-RNN 84

4.1 The Semantic Paintbrush 87
4.2 State-of-the-art technology for helping the visually impaired 89
4.3 The main hardware components of our AR glasses 90
4.4 Overview of the Semantic Paintbrush 92
4.5 Laser pointer tracker 93
4.6 Disparity correction 95
4.7 Final mean-field inference 97
4.8 Label propagation 99
4.9 A potential application area for semantic 3D maps 102

5.1 LIDAR vs camera 108
5.2 Overview of our system 109
5.3 Pivots 112
5.4 Piecewise-slanted prior 114
5.5 Quantitative results 118
5.6 Qualitative results 119

6.1 Overview 124
6.2 Cost functions 128
6.3 Smoothing 129
6.4 Median SCE 135
6.5 Inverse depth 136
6.6 Trajectories 137
6.7 Depth completeness 137

7.1 ROAM for video object segmentation 139

7.2 Graphical model of ROAM 144
7.3 ROAMingredients 145
7.4 Structure and notations of proposed model 146
7.5 Graph-cut proposals based 149
7.6 Qualitative results on the DAVIS dataset 154
7.7 Evolution of IoU on DAVIS dataset 155
7.8 Qualitative results on the ROTO++ dataset 156
7.9 Benefit of landmarks-based modeling 157
7.10 More qualitative results on the DAVIS dataset 158
7.11 Energy vs. number of iterations 159
7.12 More qualitative results 160

8.1 Motivation 164
8.2 System overview 168
8.3 Average reward 174
8.4 Results 175
8.5 Results II 176

9.1 Modelling intuitive physics 181
9.2 Dynamic SLAM 183
9.3 Narrow and variable baseline matching 184
9.4 Sparse long-range potentials 185
9.5 Trajectory forecasting 186
9.6 DNC 188

List of Tables

3.1 Comparison with some related work 69
3.2 Quantitative results for on the KITTI dataset 81

4.1 Global accuracies 100
4.2 Approximate system timing 100

6.1 Median Scale Corrected Error 134

7.1 Quantitative evaluation on CPC dataset 152
7.2 Quantitative comparisons on DAVIS dataset 153
7.3 Different types of contour warping 157
7.4 Timing details for full configuration of ROAM 159

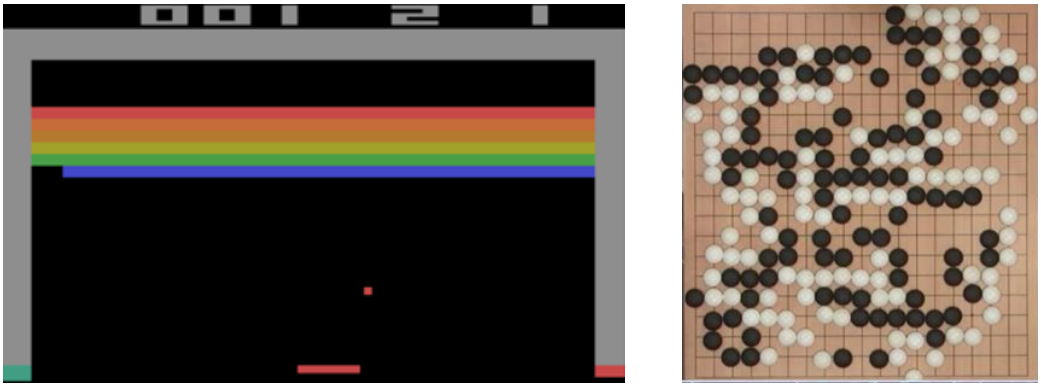
8.1 Best mean test rewards for the different frameworks run 173

1 Introduction

As we navigate the world, for example when driving a car from our home to the work place, we continuously perceive the 3D structure of our surroundings and intuitively recognise the objects we see. Such capabilities help us in our everyday lives and enable free and accurate movement even in unfamiliar places. We largely take these abilities for granted, but for robots, the task of understanding large scenes remains extremely challenging. In this thesis, I develop novel algorithms for (near) real-time dense 3D reconstruction and semantic segmentation of large-scale outdoor scenes from passive cameras and show how such intermediate representations (abstractions) improve decision making of agents navigating complex 3D environments.

1.1 Motivation and Objective

Recently, there has been an emerging trend towards learning direct mappings from raw (image) pixels to decisions without any intermediate representation such as object detection or image segmentation (Mnih *et al.*, 2015). This is a very appealing paradigm since it completely removes handcrafting of all intermediate steps and directly optimizes an objective function corresponding to the ultimate goal, *i.e.* decision making of an agent navigating the environment and interacting with the objects in its vicinity. While this has been a very successful strategy for playing Atari games (Mnih *et al.*, 2013) and more importantly Alpha GO (Silver *et al.*, 2016), it should be noted that such games represent only very simple visual worlds (*cf.* Fig. 1.1). It is not only the fact that these games use very simple rendering engines which dramatically reduces difficulty of the recognition process, but also the fact that the agents can directly observe either the whole environment, or at least a significant part of it, (typically) from a top-down view.



(a) (b)

Figure 1.1: Atari and Alpha Go have fully observable and rather simple visual worlds.

This is not the case in a real-world. No matter what type of agent we consider, all agents navigating the complex 3D world are able to observe only a very limited part of it at each time instant. Also, they typically do not have a direct access to the top-down view "summarizing" the state of the environment around the agent, however, they need to build it from "first-person" views.

One line of research attempts to push the idea that this can happen implicitly (Jaderberg *et al.*, 2016; Chen *et al.*, 2015a), while more traditional approaches extract as much information as possible by means of various intermediate representations or abstractions (Urmson *et al.*, 2008). One might argue that the former should always be preferred since it focuses directly on the ultimate goal. However, there will always be a trade-off between how much information we want to encode explicitly and how much the machine learning model is able to capture implicitly.

Balancing this trade-off is common to all applications, ranging from relatively simple tasks such as semantic segmentation, where we can see that some approaches explicitly enforce structural constraints (Zheng *et al.*, 2015; Chen *et al.*, 2015b) while others do not (Shelhamer *et al.*, 2017), to complex industrial projects such as autonomous driving cars. For instance, it is well-known that perhaps the most advanced autonomous car which has been being built by GoogleX (Waymo)[1] relies heavily on very detailed maps (*cf.* Fig. 1.2), object detection and tracking (Urmson, 2015). On the other hand of the spectra, we can see start-ups such as AutoX[2] claiming that mapping is not necessary and that the whole problem can be entirely solved by learning mappings from raw signals directly to decisions.

[1] https://waymo.com
[2] http://autox.ai

(a) (b)

Figure 1.2: Google (Waymo) self-driving car relies heavily on intermediate representations. It projects all processed data onto a common metric 3D map to provide situation awareness to the decision making subsystem. This map contains detected and tracked cars, pedestrians, signs and obstacles among others.

Although the concept of learning direct mappings to decisions is very appealing, I have focused on developing novel algorithms for extracting intermediate representations (abstractions) throughout this thesis for the following reasons:

- **Heavy-tailed data** – data typically follow heavy-tailed distributions. Ideally, we would like to capture all "rare" events simultaneously (*e.g.* rarely observed actions *and* rare appearances of the environment), however, this is much more difficult even with synthetic data than using the most convenient data source for each of the tasks separately and composing them together.

- **Interpretability** – when it comes down to safety, ability to interpret and analyze models in detail is invaluable for handling corner cases. Despite some attempts to train end-to-end models in an interpretable way, the vast majority are black-boxes that transform input signals to outputs, however, it remains difficult to understand what exactly is happening inside.

- **Other applications** – intermediate representations can often be used for other applications. For instance 3D reconstruction pipelines are often used in graphics and (semantic) video segmentation can be used for movie editing.

It is fair to say that back in 2013 both the computer vision and robotics communities were still much more skeptical about learning of direct mappings to decisions. I would also like to highlight that intermediate representations do not forbid use of an appealing framework of a single computational graph and end-to-end training; expressing all tasks within a single graph is usually a relatively straightforward step. Hence, it all boils down to how much we want to *guide* the model.

Approaches relying on intermediate (or auxiliary) representations require each agent to answer the following three questions to successfully and safely navigate throughout the environment and to interact with it (Borenstein *et al.*, 1996):

- Where am I?
- What surrounds me?
- What should I do next?

Hence, the intermediate representations (abstractions) should encode information about agents' pose, distances to obstacles and other agents, recognized objects and the scene itself. Ideally, we should not process each frame independently but rely on causality and be able to do the data association. This should allow us to understand which objects are moving or remain static, their trajectories and physical relationships among them, and ideally forecast their goals and intentions or even explain causes and effects. In computer vision, we refer to such abilities as *scene understanding* (*cf.* Fig. 1.3) and it has been one of the central topics for almost 40 years (Barrow and Tenenbaum, 1981).

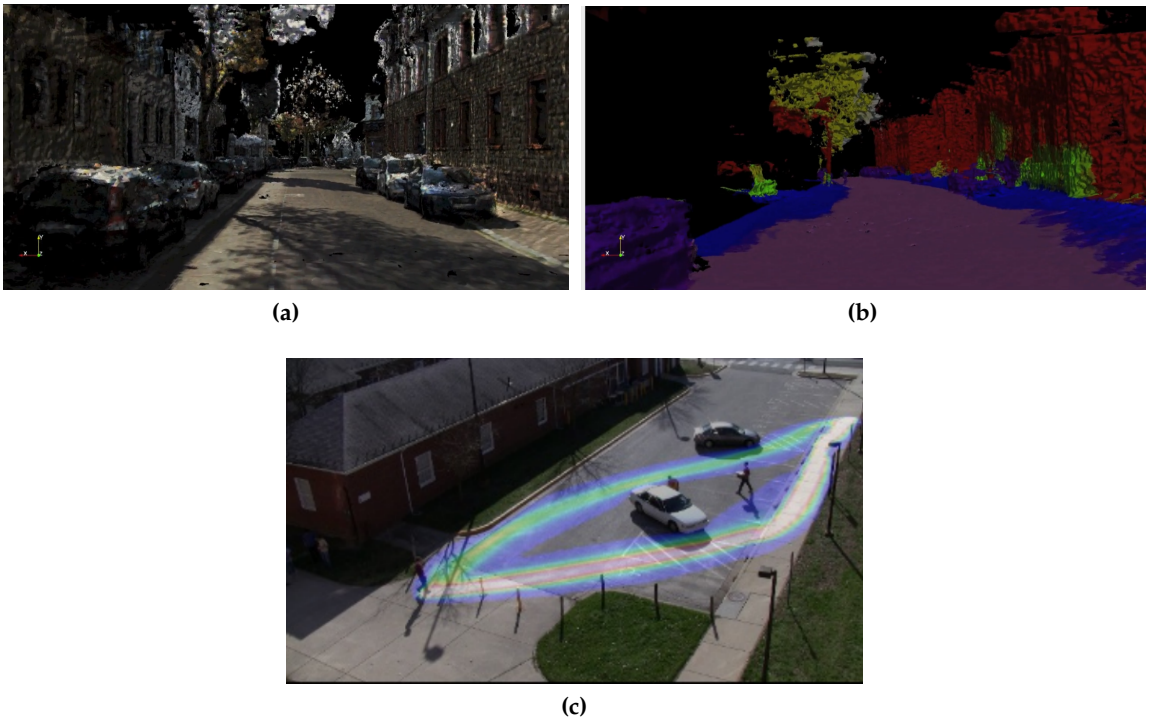


Figure 1.3: Scene understanding involves various tasks such as (a) dense 3D reconstruction, (b) semantic segmentation or (c) goal (intention) forecasting (Kitani *et al.*, 2012).

(a) (b) (c)

Figure 1.4: Motivation: (a) augmented reality glasses for partially sighted, (b) perception for self-driving cars, (c) rotoscoping (video editing).

Throughout this thesis, I am motivated by three distinct applications that despite being very different have much in common (*cf.* Fig. 1.4):

- **Perception for partially sighted** – There are more than 285 million people in the world living with sight loss, which has a significant impact on their daily lives. Over 85% of these individuals have some remaining vision. Recently, there has been an interest in developing smart glasses, which seek to provide additional information from the nearby environment through stimulation of the residual vision. The aim is to increase the information level regarding the close environment using scene understanding.

- **Perception for self-driving cars** – Without any doubt, the most important motivation is accident reduction since the leading cause of most automobile accidents today is driver error (NHTSA, 2008). But there is much more such as increased highway capacity, eliminated hunting for parking or car sharing.

- **Video editing** – Outlining accurately one or several scene elements in each frame of a shot represents a key operation for video editing tasks such as compositing, colour grading and new view synthesis among others.

My goal is to develop suitable intermediate representations and demonstrate their efficacy not just with standard metrics for 3D reconstruction or semantic video segmentation but also in actual decision making.

For these reasons, I focus on (near) real-time understanding of large-scale outdoor scenes (mostly) from passive cameras. I do not process each frame independently, however concentrate on sequential video processing. On one hand, this makes the whole problem more challenging, since we need to make sure the predictions are consistent across time. On the other hand, if we would not just compensate camera and object motions, video

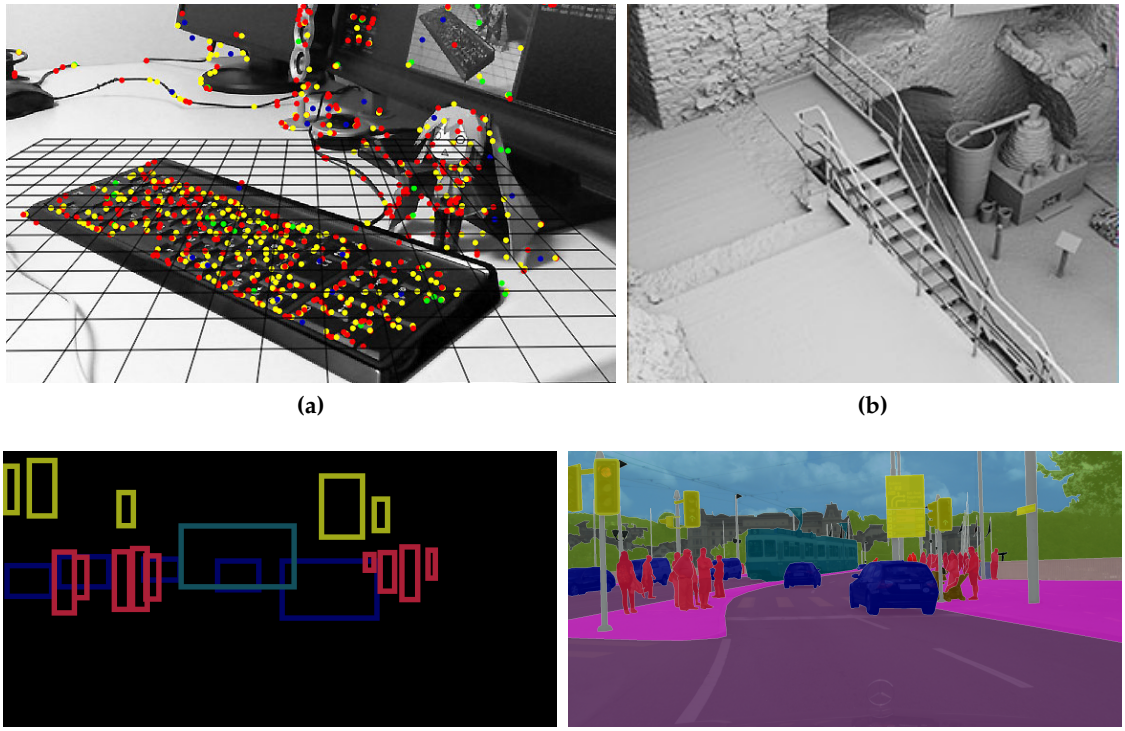


Figure 1.5: Sparse vs. dense representations: (a) only few sparse 3D points are reconstructed, hence provide very limited information about the scene (Klein and Murray, 2007), (b) dense 3D reconstruction (Capturing Reality, 2017), (c) bounding box representation, (d) per-pixel semantic segmentation of the same scene (Cordts *et al.*, 2016).

processing brings the advantage that we can observe the same or very similar scene and objects multiple times, hence we can and we should benefit from motion[3]. For instance, an object may not be well visible or recognizable from one view, however, if the camera moves and/or if we collect more statistics over time, the problem usually becomes simpler.

I focus on dense representations (*cf.* Fig. 1.5). Although there has been lot of progress and some good use-cases of sparse representations, they would never provide as much information as the dense counterpart. For instance, sparse 3D reconstruction is useful for camera pose estimation but would never provide enough information for detailed understanding of the environment. Similarly, while bounding boxes are useful for object detection, they would never provide enough information about spatial extend of objects or about "stuff" classes such as road and grass.

[3]One might argue that if we solved the problem of recognition or segmentation from single still images, they would be temporally consistent by default. However, unless this happens (and if ever), considering multiple views provide more information and makes recognition simpler.

1.2 Challenges

While we largely take understanding of large-scale outdoor scenes for granted, this task is extremely challenging for machines. This is witnessed by performance of state-of-the-art models on various benchmarks. For instance, performance on the Pascal VOC segmentation challenge (Everingham *et al.*, 2010) was for a long time saturated around 50% average precision. This has changed with the deep learning models, however on more modern benchmarks such as CityScapes (Cordts *et al.*, 2016), the best models do not achieve better scores than 59% when a stricter instance-level intersection-over-union metric is used. This raises a very natural question: *Why is scene understanding so difficult for machines?*

Data and dataset bias. Although we often use terms such as "artificial intelligence", the state-of-the-art machine learning models are nothing more than very efficient (nonlinear) function approximators. Thus, data has become an integral part of machine learning models and significantly influences quality of the models. This is well witnessed even with modern deep learning models. Although we nowadays use some extra tricks such as ReLUs and massive compute power of modern GPUs, the state-of-the-art models are not fundamentally different from models we had back in 80s (Rumelhart *et al.*, 1988; LeCun *et al.*, 1990). There is no doubt that the main difference is in the amount of training data we are able to process.

However, this poses several challenges. First of all, it is often very difficult and time consuming to construct such large-scale datasets. Even in late 2017, we have large-scale datasets only for a few, relatively simple problems such as recognition or detection for which it is easy to obtain ground-truth data (Russakovsky *et al.*, 2015). In fact, for many problems (*e.g.* intrinsic scene decomposition) it is so difficult to create large-scale datasets efficiently that we are forced to use synthetically generated data. This brings us to the second issue. Although we pay a lot of attention to balancing the datasets and making sure they reflect the test scenarious well, all of them are *biased* in some way. Datasets typically follow heavy-tailed distribution (Fig. 1.6) because we are simply unable to capture enough samples of "rare" events (*e.g.* a car in the lake). It also often happens that the intra-class variability is simply too large, the datasets capture only a small subset of possible visual appearances and consequently the model fails on unseen data. This causes the biggest challenge with deploying machine learning algorithms in the real-world, where we want to solve some problem and not just do relative comparison of different models on a benchmark.

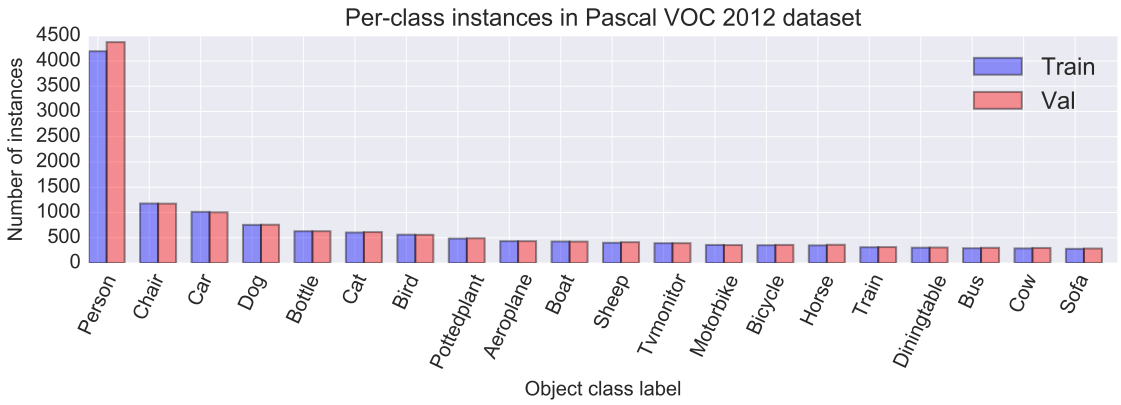


Figure 1.6: Number of instances in Pascal VOC 2012 dataset.

Context, priors, understanding of purpose, causes and effects. Objects in natural scenes never occur isolated. In fact, they always co-vary with other objects and particular environments (*cf.* Fig. 1.7). This introduces lot of clutter and occlusions. Humans have remarkable (and somewhat surprising) ability to exploit contextual information on multiple levels, including semantic co-occurrence (*e.g.* table and chairs are often present in the same scene), spatial configuration (*e.g.* cup is expected to be on the desk), pose (*e.g.* chairs are not oriented upside-down) and ability to understand and explain purpose, causes and effects (Oliva and Torralba, 2007). This enables humans to quickly guide their attention and eyes to regions of interest in natural scenes and gives them the ability to quickly recognize thousands of object categories in cluttered scenes, despite variability in pose, changes in illumination and occlusions. Most computer vision algorithms are still unable to exploit such contextual information efficiently and hence often fail in such situations.

(a) (b)

Figure 1.7: (a) Objects in natural scenes co-vary, *e.g.* chairs, screens and desks appear together in a specific spatial configuration. (b) Ability to explain purpose and geometry improves recognition rates and vice versa (Fouhey *et al.*, 2014).

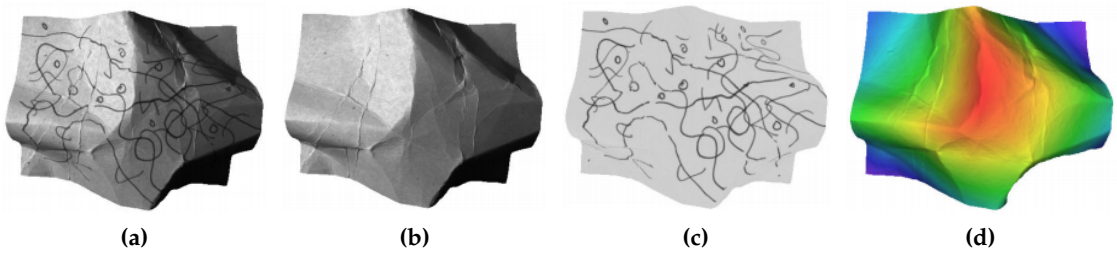


Figure 1.8: Intrinsic scene decomposition: (a) input image, (b) shading, (c) albedo and (d) 3D structure (Barron and Malik, 2012).

Scene de-rendering is an ill-posed problem. Recognition of natural scenes is complicated by variations in illumination, pose and viewpoint, *etc.* One might be tempted to invert the scene formation (rendering) process to disentangle scene structure, illumination, albedo and shading (intrinsic scene decomposition), hoping that the recognition process would become simpler if we managed to separate structure from appearance artifacts. Ideally, we would like to go even beyond that and project objects into their canonical views before we start the recognition. Unfortunately, intrinsic scene decomposition even on the per-pixel (not object) level is an ill-posed problem full of ambiguities and as such is typically more difficult that the recognition itself (Barrow and Tenenbaum, 1978; Barron and Malik, 2012).

Large intra-class variability. Supervised machine learning models also suffer from the "label bottleneck" (Efros, 2017). For instance, we want to recognize *animals*, however, such label is semantically very coarse since it contains visually very dissimilar animals such as *cows*, *birds* or *dolphins*. This phenomena is referred as large intra-class variation. One might argue that the solution is to be more careful when constructing datasets and ground-truth labelling, however, this phenomena occurs even with fairly constrained classes such as *chairs*. This suggest, that recognition based solely on visual appearance and language ground-truth labels is a very difficult problem and we probably should move beyond that on a semantic level.

Geometric deformations and illumination variations. Since the objects and scenes are not observed in their canonical views, we need to deal with various geometric deformations. This includes changes in scale, pose, texture and illumination or non-rigid deformations.

Detection vs. segmentation, things vs. stuff. While we often tackle object detection and segmentation in a very similar way (features, *etc.*), these tasks are substantially different. It turns out that for object detection, it is usually enough to learn just some sufficiently

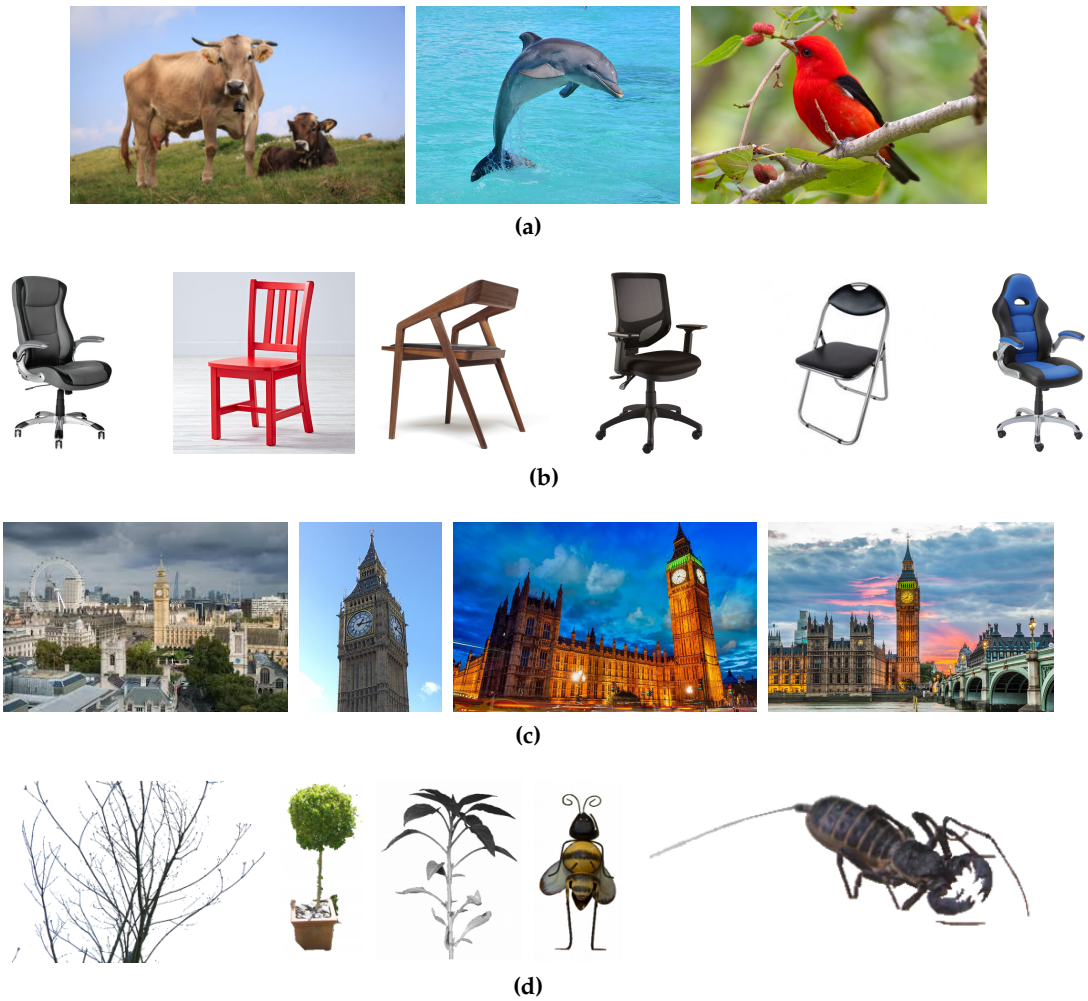


Figure 1.9: Difficult examples for recognition: (a) a single label (animal) used for visually dissimilar objects (label bottleneck), (b) large intra-class variability, (c) pose, scale and illumination changes, (d) thin and elongated structures (Jegelka and Bilmes, 2011).

unique and discriminative part of an object which acts as a *supporter* or *anchor* of its spatial extent (Grabner *et al.*, 2010). This is fundamentally different for segmentation, where the model has to learn how to segment all object parts. Segmenting thin and/or elongated object parts is a very difficult task which is highlighted by benchmarks that are not dominated by *stuff* classes (*e.g.* road, sky, grass) – while most methods achieve relatively high scores on stuff classes, their performance on objects is often rather poor. This could be (among others) explained also by the "label bottleneck"; stuff classes are typically visually uniform and differ significantly between each other, however objects consist of many parts (*e.g.* car consists of visually dissimilar bonnet and wheels) whose appearance is often shared by parts of different objects. Examples are shown in Fig. 1.9.

1.3 Approach

The above-mentioned challenges raise a question of how to tackle such problems efficiently. Throughout this thesis, I use four principles outlined below that allow me to build semantically annotated dense large-scale 3D models.

Visual world is structured. One of the core ideas behind modern computer vision is that the visual world is highly structured. In fact, it is highly structured on multiple levels. At low-level, we can observe that object textures form repetitive patterns and neighboring pixels typically vary smoothly. This is widely explored in image de-noising or inpainting tasks. At the object level, we can observe that (man-made) objects are often symmetric and certain properties such as object label or surface normals vary slowly and smoothly. We can also observe, that semantic labels are often highly correlated with geometry, for instance *road* tend to be flat. At the scene-level, we observe that certain objects co-occur and similar scenes typically share the same layout.

Usually, contextual knowledge is observable at the "global" level and cannot be extracted solely from local features. At the same time, we need to be able to deal with uncertainty of our predictions. I tackle both challenges through the elegant framework of probabilistic graphical models, which allows me to encode such structural constraints and at the same time encode uncertainty in predictions in a principled way (Koller and Friedman, 2009).

Visual world is causal. I do not work with photographs downloaded from the internet but with video sequences. I could process each video frame completely independently and hope that this would lead to optimal performance, however, this usually only highlights failures of computer vision algorithms (*cf.* Fig. 1.10). Many approaches only "compensate" such artifacts, typically caused by motion of camera and objects within the scene. I believe that we should "benefit" from motion since often, an object that is difficult to recognize from one view becomes trivially recognizable from another view, or from statistics collected over time. In this thesis, I show how treating video data as temporal sequences instead of independent predictions improves accuracy of semantic segmentation.

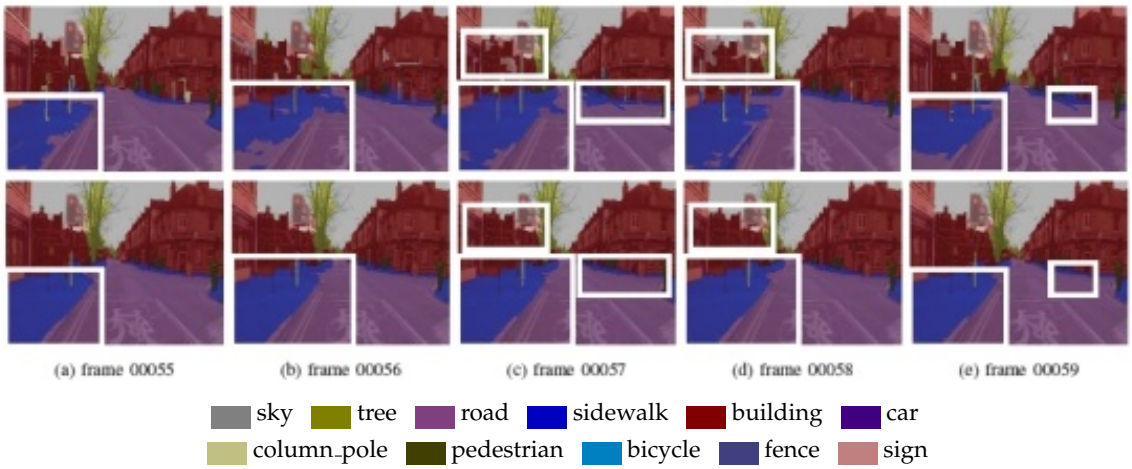


Figure 1.10: Temporal inconsistency. Top row: per-frame predictions, bottom: temporally consistent predictions (Miksik *et al.*, 2013).

Visual world is three-dimensional (3D). I explicitly exploit 3D geometry since it provides powerful constraints. I could process the input visual stream directly in the image space without any geometric cues, hoping that the model would be powerful enough. Instead, I use geometry whenever it is possible. This provides at least two important advantages: i) geometrical constraints allow more robust and faster feature matching (*e.g.* 2D search is reduced to 1D search along the epipolar line) and ii) depth is a cue that allows extraction of powerful features for recognition. In addition to that, I show that if we are able to map 2D images to 3D structure, we can associate predictions with respective 3D voxels and hence efficiently overcome issues with temporal consistency. Another advantage of having dense 3D map is that it can be used to measure distances.

Interactivity and personalisation. Although we usually spend significant effort by constructing datasets that reflect the test scenarios well, all of them are biased in some way. In fact, even if we managed to train models with millions of classes that would have perfect accuracy in test scenarios, this "one-size-fits-all" approach would never be enough to provide personalised experience and satisfy demands such as recognizing "my favorite cup". Both issues can be addressed with interactivity. Putting an agent in the loop allows me to acquire ground-truth labels and adapt the model during deployment. While I show this concept with a "human" in the loop, this approach could be extended to fully autonomous agents which could provide a form of feedback that would help with "domain adaptation".

1.4 Contributions

This thesis consists of five main contributions detailed in respective chapters. I would like to emphasize that my primary interest is in building and understanding the representations itself and not the particular tools I have used. Since computer vision and machine learning have changed dramatically over the past few years, I discuss how each of the contributions could be updated with the state-of-the-art tools at the end of each chapter.

Dense Large Scale Semantic 3D Reconstruction. I propose a robust approach to dense 3D reconstruction and semantic segmentation of large-scale outdoor environments from passive stereo-cameras (Fig. 1.11). To the best of my knowledge, this is the first end-to-end system for (near) real-time dense reconstruction and recognition of large-scale outdoor environments from passive cameras.

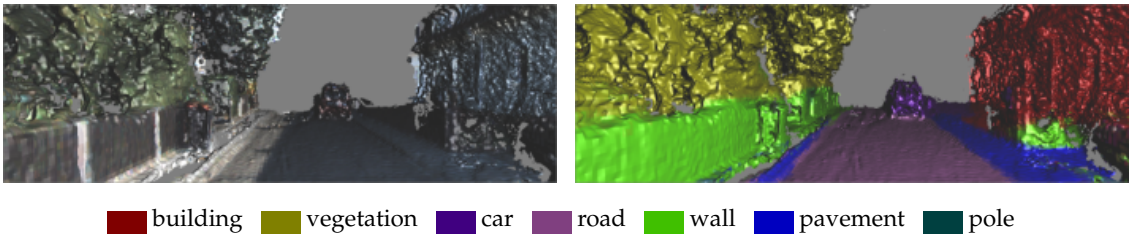


Figure 1.11: Dense incremental 3D reconstruction (left) and semantic segmentation (right) from the proposed system, as seen from a moving platform on-the-fly (*i.e.* not a final mesh).

Interactive Large Scale Scene Understanding. Most semantic segmentation models suffer from large discrepancy between training and test data, since obtaining ground-truth data for per-pixel semantic segmentation is very time consuming and hence the amount of labelled data is in orders of magnitude smaller than for classification or detection tasks.

I propose an augmented reality system for user-friendly interactive 3D reconstruction and labelling of large scale outdoor scenes (Fig. 1.12). This system puts the user in the loop and allows her to physically interact with the world and collect ground-truth data in the target environment to learn better semantic segmentation models. The main motivation were "smart glasses" for partially sighted users and to the best of my knowledge, this was the first system that managed to increase the information level regarding the close environment by semantic labelling and not just by depth or enhanced edges.

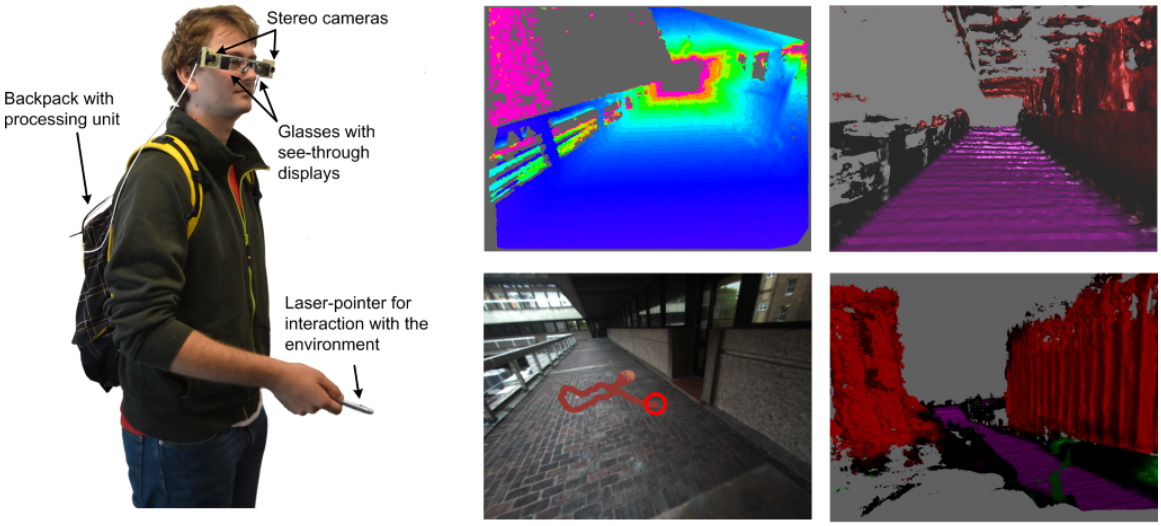


Figure 1.12: The Semantic Paintbrush comprises of an off-the-shelf pair of optical see-through glasses, with additional stereo RGB-Infrared cameras, and an additional handheld infrared/visible light laser. The passive stereo cameras are used for depth estimation. The user can see these reconstructions immediately using the heads-up display, and can use a laser pointer to draw onto the 3D world to semantically segment objects (once segmented these labels will propagate to new parts of the scene). The laser pointer can also be triangulated precisely in the stereo infrared images allowing for interactive "cleaning up" of the model during capture. Final output is the dense semantic 3D map of the scene.

Dense 3D Reconstruction. I propose two approaches to improve dense 3D reconstruction. The first one performs multi-modal sensor fusion and uses a small set of sparse but more accurate depth measurements to guide the dense stereo matching algorithm. The second one improves local model for dense monocular 3D reconstruction by using joint optimization over depth and pose.

Video Segmentation. I propose a system for interactive segmentation of objects in videos. It explicitly encodes a single object instance by closed curve, maintains correspondences across time and provides very accurate segmentation close to object boundaries. While I demonstrate the efficacy on "rotoscoping" task (detailed delineation of scene elements through a video shot, starting from an initial outline provided by the user), it can also be used in fully automatic setups as a prerequisite for shape-from-silhouettes (Fig. 1.13).

Figure 1.13: ROAM for video object segmentation. Designed to help *rotoscoping*, the proposed object appearance model allows the automatic delineation of a complex object in a shot, starting from an initial outline provided by the user.

SLAM-Augmented Deep Reinforcement Learning. While many papers on semantic segmentation claim that such representation is a necessary prerequisite for any decision making (such as in self-driving cars), they do not evaluate their impact on the this task. Instead, they typically use an isolated setup and intersection over union (IoU) or similar scores to evaluate only the quality of the segmentation. While this is useful for improving segmentation itself, it does not measure the impact on the ultimate goal. In this thesis, I show how such intermediate representation improves decision making of agents navigating complex 3D environments (Fig. 1.14).

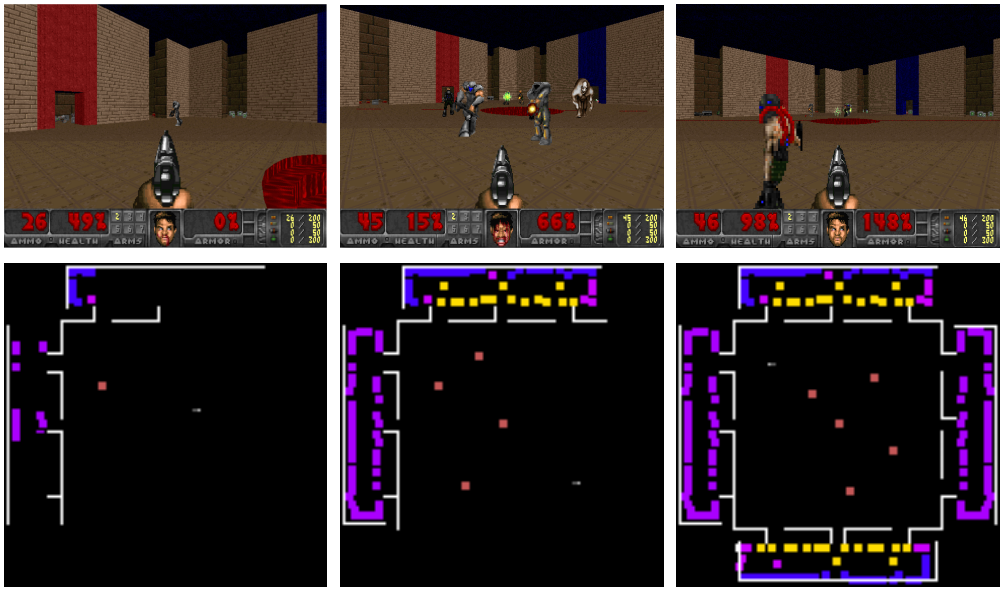


Figure 1.14: SLAM-Augmented Deep Reinforcement Learning: As the agent explores the environment, the first-person-view (top) only sees a restricted portion of the scene, whereas in the semantic map (bottom), the effect of exploration is cumulative, indicating both semantic labels and poses.

1.5 Thesis Outline

Chapter 2. In Chapter 2, I show that many computer vision tasks can be formulated as labelling problems. Hence I cover the basics of probabilistic graphical models. Specifically, I discuss the Markov Random Field (MRF) and Conditional Random Field (CRF) models, inference methods and parameter learning. Next, I cover basics of monocular camera multiview geometry and conclude with an overview of low-level computer vision tools such as feature extractors.

Chapter 3. In Chapter 3, I propose a robust approach to dense 3D reconstruction and segmentation of large-scale outdoor scenes from passive stereo camera. At the core of this system is a hash-based fusion approach for dense 3D reconstruction with standard sparse visual odometry for camera pose estimation and an efficient mean-field approach to volumetric semantic segmentation. This system exploits synergy between the reconstruction and recognition tasks since it uses 3D data to improve recognition and semantic labelling to improve 3D reconstruction of moving objects. I show high quality dense reconstruction and labelling of the scenes and demonstrate the effectiveness of this approach on the KITTI dataset (Geiger *et al.*, 2012). To the best of my knowledge, this is the first end-to-end system for (near) real-time dense large-scale 3D reconstruction and semantic segmentation.

Chapter 4. I develop a system that makes interactive 3D reconstruction and recognition of large-scale outdoor scenes fast, simple and user-friendly. A user is equipped with augmented reality glasses, wearable stereo camera and a laser pointer. As the user navigates the environment, she is able to use the laser pointer to label previously unobserved and unlabelled region of space. Such ground-truth labels are used to adapt a classifier in a background thread, which is then used to generalize and recognize previously unseen regions of the world. In addition to that, unique correspondences provided by the laser pointer are used to "guide" the dense disparity matching and hence to improve the resulting 3D reconstruction. This is a direct extension of a system for dense large-scale 3D reconstruction and recognition presented in Chapter 3 and SemanticPaint (Valentin *et al.*, 2015).

Chapter 5. In Chapter 5, I push the idea of using a sparse but very accurate and confident 3D measurements to guide the dense depth matching further. I propose to combine a calibrated stereo camera with LIDAR, which is able to provide much more accurate measurements. While combining these two sensors together is a common strategy, measurements

from both are typically processed completely independently and only the final solutions (*e.g.* recognized objects) are merged or fused together in a hope that the complementarity of the two would improve overall results. In contrast to this strategy, I show how sparse but more accurate LIDAR measurements can be incorporated directly into the dense depth estimation and propose a probabilistic model for incremental dense scene reconstruction. Effectiveness of this system is demonstrated on the KITTI dataset (Geiger *et al.*, 2012), where we show that using even a very small number of LIDAR measurements leads to substantial improvement in dense depth estimation. This is an important result since LIDAR sensors capable of providing dense point clouds are still very expensive, which is a major issue for their deployment in consumer-grade applications.

Chapter 6. In Chapter 6, I relax the assumption of using a calibrated stereo camera and focus on dense monocular reconstruction. Recently, direct methods to simultaneous localization and mapping using the whole image data have become popular since they remove the need of feature extraction and matching (Stühmer *et al.*, 2010; Newcombe *et al.*, 2011b; Engel *et al.*, 2014a). However, many of these approaches alternate between pose estimation and computing (semi-)dense depth maps. I propose a framework for dense monocular SLAM, and its local model in particular, which optimizes over depth and pose simultaneously. Importance of joint optimization is demonstrated on the TUM dataset (Sturm *et al.*, 2012).

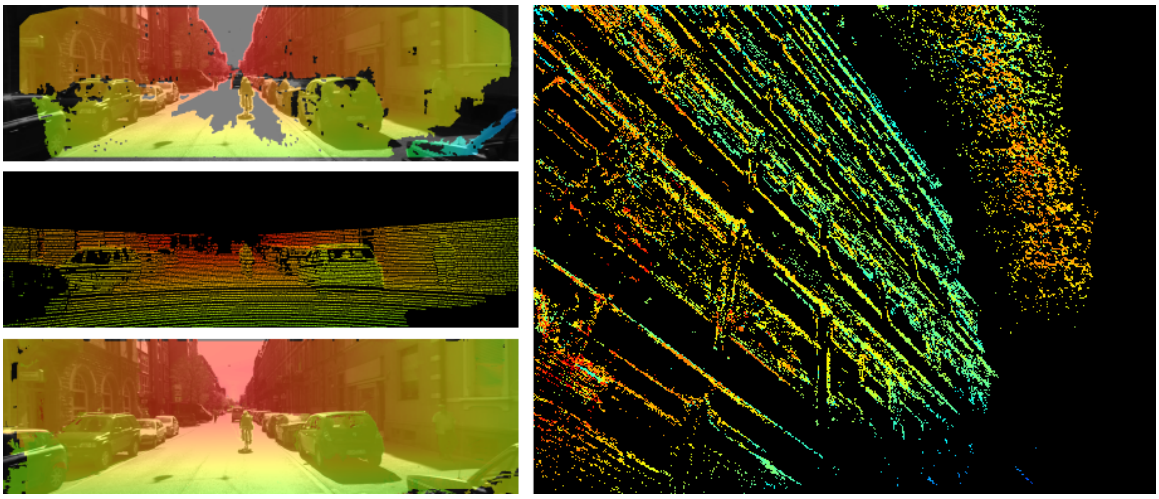


Figure 1.15: Dense multi-modal (left) and monocular 3D reconstruction (right).

Chapter 7. So far, I have assumed that the world is mostly static. Recently, there has been lot of progress in 3D reconstruction of dynamic scenes, however, such methods typically assume a pre-segmented object of interest and depth provided by Kinect-like cameras (Newcombe *et al.*, 2015; Dou *et al.*, 2015). Dynamic objects are typically textureless, blurred and suffer from non-rigid deformations. This makes dense feature matching extremely challenging and is often addressed by 3D reconstruction known as *shape-from-silhouettes* (Cashman and Fitzgibbon, 2013; Nurutdinova and Fitzgibbon, 2015). However, such approaches typically assume the *contour correspondences* are given to the algorithm.

In Chapter 7, I propose a system for efficient video segmentation which represents objects by closed contours. At a high level, this model can be seen as a combination of old-fashioned "Snakes" (Amini *et al.*, 1990) equipped with much more powerful local cues and a pictorial structure that allows to control rigidity and handles large displacements. As such, it is able to provide long-term contour correspondences and a very accurate segmentation close to object boundaries. For inference, this model uses an efficient block-coordinate descent with two alternating blocks that are solved exactly with dynamic programming. I demonstrate the efficacy of this approach on "rotoscoping" task (detailed delineation of scene elements through a video shot, starting from an initial outline provided by the user) using the DAVIS (Perazzi *et al.*, 2016), CPC (Lu *et al.*, 2016) and Video SnapCut (Bai *et al.*, 2009) datasets.

Chapter 8. Most papers on semantic segmentation claim that it is a necessary prerequisite for autonomous driving. However, these papers always evaluate only intersection over union (IoU) scores of semantic segmentation and take the fact that such representation helps to decision making for granted. In Chapter 8, I show how semantically labelled 3D maps can be incorporated into standard Deep Q-learning approach to improve agents' decision making in complex 3D environments by providing more complete overview of the environment. To the best of my knowledge, this is the first time it has been shown how such intermediate representation improves agents' decision making based on standard Deep Q-learning approaches.

Chapter 9. I discuss open questions, future directions and conclude the thesis.

1.6 Publications

The work detailed in this thesis has been published in following papers:

Vineet V.*, Miksik O.*, Lidegaard M., Niener M., Golodetz S., Prisacariu V.A., Khler O., Murray D.W., Izadi S., Pérez P. and Torr P.H.S., *Incremental Dense Semantic Stereo Fusion for Large-Scale Semantic Scene Reconstruction.* In IEEE International Conference on Robotics and Automation (ICRA) 2015.
IEEE ICRA 2015 Best Robotic Vision Paper Award Finalist
* Joint first authors

Miksik O.*, Vineet V.*, Lidegaard M., Prasaath R., Niener M., Golodetz S., Hicks S.L., Pérez P., Izadi S. and Torr P.H.S., *The Semantic Paintbrush: Interactive 3D Mapping and Recognition in Large Outdoor Spaces.* In 33nd annual ACM conference on Human factors in computing systems (CHI) 2015.
* Joint first authors

Miksik O., Amar Y., Vineet V., Pérez P. and Torr P.H.S., *Incremental Dense Multi-modal 3D Scene Reconstruction.* In IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) 2015.

Liwicki S., Zach C., Miksik O. and Torr P.H.S., *Coarse-to-fine Planar Regularization for Dense Monocular Depth Estimation.* In European Conference on Computer Vision (ECCV) 2016.

Miksik O,*, Pérez-Rúa J-M.*, Torr P.H.S and Pérez P., *ROAM: a Rich Object Appearance Model with Application to Rotoscoping.* In IEEE International Conference on Computer Vision and Pattern Recognition (CVPR) 2017.
* Joint first authors

Bhatti S., Desmaison A., Miksik O., Nardelli N. Siddharth N. and Torr P.H.S., *Playing Doom with SLAM-Augmented Deep Reinforcement Learning.* arXiv preprint arXiv:1612.00380

Other publications:

Bertinetto L., Valmadre J., Golodetz S., Miksik O. and Torr P.H.S., *Staple: Complementary Learners for Real-Time Tracking.* In IEEE International Conference on Computer Vision and Pattern Recognition (CVPR) 2016.

Arnab A., Sapienza M., Golodetz S., Valentin J., Miksik O., Izadi S. and Torr P.H.S., *Joint Object-Material Category Segmentation from Audio-Visual Cues.* In British Machine Vision Conference (BMVC) 2015.

Miksik O., Vineet V., Pérez P. and Torr P.H.S., *Distributed Non-Convex ADMM-inference in Large-scale Random Fields.* In British Machine Vision Conference (BMVC) 2014.

Miksik O., Munoz D., Bagnell, J. A. and Hebert M., *Efficient Temporal Consistency for Streaming Video Scene Analysis.* In IEEE International Conference on Robotics and Automation (ICRA) 2013.

The VOT Challenges:

Kristan *et al.* , *The Visual Object Tracking VOT2015 challenge results.* In IEEE International Conference on Computer Vision (ICCV) Workshops 2015.

Felsberg *et al.* , *The Thermal Infrared Visual Object Tracking VOT-TIR2015 Challenge Results.* In IEEE International Conference on Computer Vision (ICCV) Workshops 2015.

Kristan *et al.* , *The Visual Object Tracking VOT2016 challenge results.* In European Conference on Computer Vision (ECCV) Workshops 2016.

Felsberg *et al.* , *The Thermal Infrared Visual Object Tracking VOT-TIR2016 Challenge Results.* In European Conference on Computer Vision (ECCV) Workshops 2016.

2 Preliminaries

In an attempt to make this thesis as self-contained as possible, I present a number of mathematical concepts upon which this thesis is based. First, I show that many computer vision problems, including those addressed in this thesis, can be formulated as labelling problems. Hence, I provide a brief overview of probabilistic graphical models and inference methods commonly used to tackle them. Next, I focus on parameter estimation, introduce the concept of supervised learning and discuss the standard loss functions used in solving it. The third part covers fundamentals of multi-view geometry and visual simultaneous localization and mapping (SLAM). Finally, the last part of this chapter provides an overview of basic computer vision tools such as local feature detectors and descriptor extractors that are widely used for correspondence matching in geometry or as image statistics for probabilistic graphical models.

2.1 Probabilistic Graphical Models

In most computer vision problems, using evidence based solely on local features and a winner-takes-all strategy is not enough since they do not encode context sufficiently. For instance, in semantic segmentation or dense depth estimation, local evidence is usually very noisy which leads to suboptimal solutions (*cf.* Fig. 2.4 (a)). This can be overcome by adding pairwise or higher order constraints that enforce priors such as spatial smoothness (Boykov *et al.*, 2001; Szeliski *et al.*, 2008) or even some higher-order consistency constraints such as co-occurrence (Kohli *et al.*, 2009; Ladicky *et al.*, 2014). To this end, we usually formulate such tasks as labelling problems with sets of random variables in the elegant framework of probabilistic graphical models which allows to encode structural constraints and at the same time encode prediction uncertainty in a principled way (Koller and Friedman, 2009).

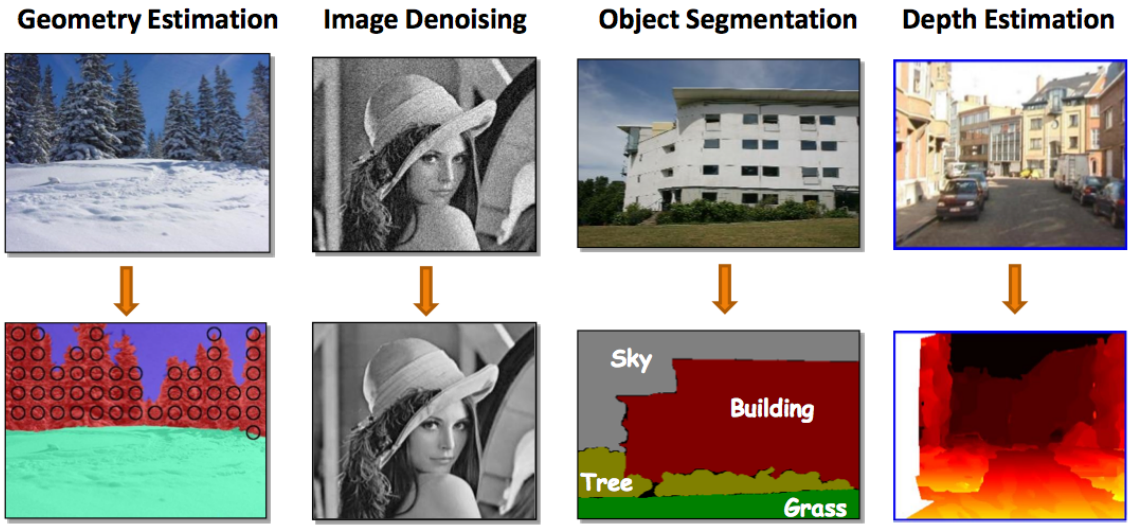


Figure 2.1: Typical instances of computer vision tasks formulated as labelling problems include geometry estimation (surface normal prediction), image denoising, semantic segmentation or dense depth estimation (Ladicky, 2012).

2.1.1 Computer Vision as Labelling Problems

In this framework, we define a set of discrete random variables $\mathcal{X} = \{X_1, \ldots, X_N\}$ associated with a lattice $\mathcal{V} \in \{1, \ldots, N\}$, which typically corresponds to image pixels or 3D voxels. Each discrete random variable X_i then takes a label l from a finite label set $\mathcal{L} = \{l_1, \ldots, l_L\}$ based on the observation $\mathbf{D}$. These labels are always determined by the application itself. For instance, they can correspond to various object classes such as *car*, *building* or *road* in the case of semantic segmentation, disparity levels in case of dense depth estimation or pixel intensities in case of image denoising (*cf.* Fig. 2.1).

Probabilistic graphical models form joint probability distribution $P(\mathbf{x}, \mathbf{D})$ or conditional probability distribution $P(\mathbf{x}|\mathbf{D})$ over these variables. We refer to any possible assignment of labels to the variables as a labelling or configuration $\mathbf{x} = (x_1, \ldots, x_N)$, where x_i denotes a particular label for the i-th variable (Fig. 2.2). Our goal is to infer the best possible labelling $\mathbf{x}^* \in L^N$. The maximum a posterior (MAP) solution $\mathbf{x}^*$ corresponds to

$$\mathbf{x}^* = \operatorname*{argmax}_{\mathbf{x}} P(\mathbf{x}|\mathbf{D}). \tag{2.1}$$

Similarly, we may also be interested in estimation of marginal distributions

$$P(x_i) = \sum_{\mathbf{x} \setminus x_i} P(\mathbf{x}|\mathbf{D}) \qquad \forall i \in \mathcal{V}, \tag{2.2}$$

where $\mathbf{x} \setminus x_i$ denotes all possible configurations of all variables except the i-th variable. In general, these problems are NP-hard. To demonstrate the difficulty, consider that our goal

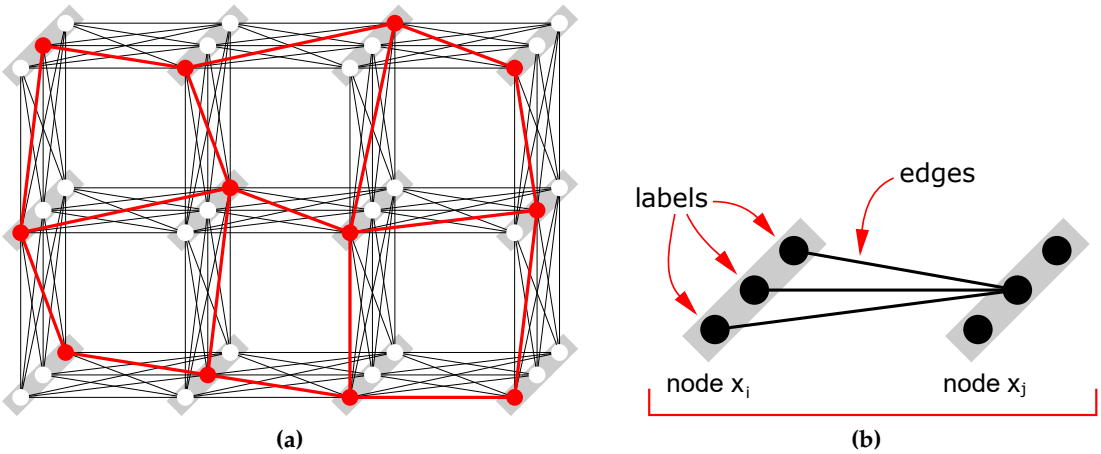


Figure 2.2: Grid Random Field with 3×4 nodes and label set $\mathcal{L}$ consisting of 3 labels. A possible labelling $\mathbf{x}$ is shown in red (a). Two nodes X_i and X_j are linked by pairwise edges (b); we show only edges for the second label of node X_j to avoid clutter (Werner, 2007).

is to infer the best possible labelling $\mathbf{x}^*$ from L^N possible configurations. In other words, this complexity scales exponentially with number of variables, which makes exact inference intractable in many cases. For instance, there are usually hundreds of labels L in the case of dense disparity estimation and images typically have thousands of pixels N. However, a number of approximate inference methods have been developed and successfully applied over the years for special cases and we will briefly look into some examples that are widely used in computer vision. A more detailed overview of probabilistic graphical models can be found in (Koller and Friedman, 2009; Barber, 2012; Hartley *et al.*, 2018).

Markov Random Field

The Markov Random Field (MRF) models a joint probability distribution of the random field configuration $\mathbf{x}$ and the data $\mathbf{D}$ as

$$P(\mathbf{x}, \mathbf{D}) = P(\mathbf{D}|\mathbf{x})P(\mathbf{x}), \tag{2.3}$$

where $P(\mathbf{x})$ is a prior on the label configuration and $P(\mathbf{D}|\mathbf{x})$ is the data likelihood. The probability distribution $P(\mathbf{x}, \mathbf{D})$ of the Markov Random Field has to fulfill the positivity condition

$$P(\mathbf{x}, \mathbf{D}) \geq 0. \tag{2.4}$$

While the positivity condition seems to be restricting, it is required for the Hammersley-Clifford theorem (Hammersley and Clifford, 1971), which is integral to the theory of MRFs and characterizes the probability distribution for the MRF. In addition to that, every MRF

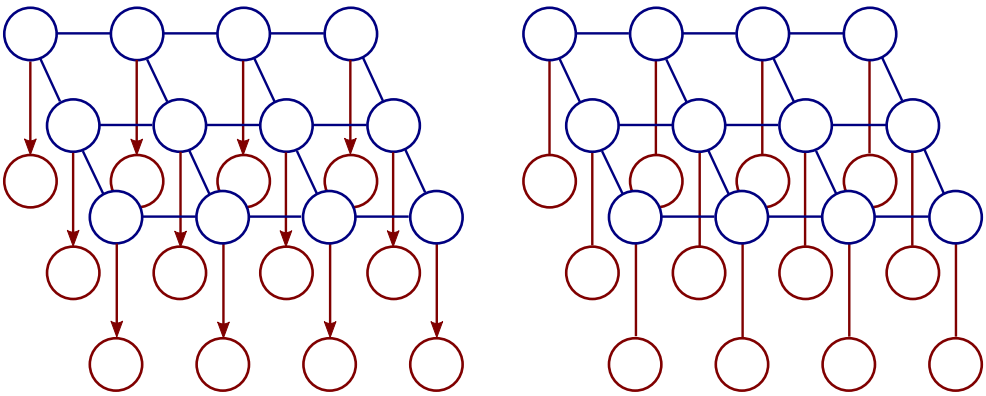


Figure 2.3: Pairwise MRF and CRF models with 4-neighborhood (Prince, 2012).

has to fulfil the Markovian property

$$P(x_i|x_j : j \in \mathcal{V} - \{i\}) = P(x_i|x_j : j \in \mathcal{N}_i), \qquad \forall i \in \mathcal{V}, \tag{2.5}$$

which says that each variable X_i is conditionally independent from all other variables given its neighbours $X_{\mathcal{N}_i}$. This allows us to define a set of cliques $c \in \mathcal{C}$. Each clique c denotes a set of random variables X_c which are conditionally dependent on each other. We also define non-negative potential functions $\psi_c(\mathbf{x}_c, \mathbf{D}_c)$ for each clique $c \in \mathcal{C}$, where $\mathbf{D}_c$ corresponds to observed variables in clique c (*cf.* Fig. 2.3).

Then, using the Hammersley-Clifford theorem, we can write the probability distribution $P(\mathbf{x}, \mathbf{D})$ of an MRF as a product of potential functions ψ_c

$$P(\mathbf{x}, \mathbf{D}) = \frac{1}{Z} \prod_{c \in \mathcal{C}} \exp\left(-\psi_c(\mathbf{x}_c, \mathbf{D}_c)\right) = \frac{1}{Z} \exp\left(-\sum_{c \in \mathcal{C}} \psi_c(\mathbf{x}_c, \mathbf{D}_c)\right), \tag{2.6}$$

where $Z = \sum_{\mathbf{x}} \sum_{\mathbf{D}} \prod_{c \in \mathcal{C}} \exp(-\psi_c(\mathbf{x_c}, \mathbf{D_c})) = \sum_{\mathbf{x}} \sum_{\mathbf{D}} \exp(-\sum_{c \in \mathcal{C}} \psi_c(\mathbf{x_c}, \mathbf{D_c}))$ is the partition function which ensures the probability distribution is normalized.

Conditional Random Field

In most applications, we do not need the joint probability $P(\mathbf{x}, \mathbf{D})$ which models both the hidden and observed variables. Usually, we are interested in predicting labelling $\mathbf{x}$ given the observed data $\mathbf{D}$. Conditional Random Field (CRF) directly models this conditional probability $P(\mathbf{x}|\mathbf{D})$. Similarly to MRFs, we can express the CRF as a product of potential functions defined over the cliques c

$$P(\mathbf{x}|\mathbf{D}) = \frac{1}{Z(\mathbf{D})} \prod_{c \in \mathcal{C}} \exp\left(-\psi_c(\mathbf{x}_c|\mathbf{D}_c)\right) = \frac{1}{Z(\mathbf{D})} \exp\left(-\sum_{c \in \mathcal{C}} \psi_c(\mathbf{x}_c|\mathbf{D}_c)\right). \tag{2.7}$$

However, the partition function $Z(\mathbf{D}) = \sum_{\mathbf{x}} \prod_{c \in \mathcal{C}} \exp(-\psi_c(\mathbf{x_c}|\mathbf{D_c}))$ is a function of the observed data $\mathbf{D}$ and summation is only over the possible label configurations.

2.1.2 Inference in Graphical Models

Now we briefly review particular CRF factorizations and related inference methods. Since random fields have to satisfy the strict positivity condition, we can take the negative logarithm to express these models in the form of energy E

$$P(\mathbf{x}|\mathbf{D}) = \frac{1}{Z(\mathbf{D})} \exp(-E(\mathbf{x}|\mathbf{D}))$$
$$E(\mathbf{x}|\mathbf{D}) = \sum_{c \in \mathcal{C}} \psi_c(\mathbf{x}_c),$$

which is the negative log-likelihood of the conditional probability and fully describes the model. The energy function $E(\mathbf{x}|\mathbf{D})$ is often referred as the Gibbs energy function. Throughout the rest of this thesis, we describe probabilistic models through their energy function E and hence refer to inference as *energy minimization*

$$\mathbf{x}^* = \operatorname*{argmax}_{\mathbf{x}} P(\mathbf{x}|\mathbf{D}) = \operatorname*{argmin}_{\mathbf{x}} E(\mathbf{x}|\mathbf{D}). \tag{2.8}$$

In general, this problem is intractable. However, there are many special instances that can be solved exactly, and many special cases that can be solved at least approximately with relatively strong theoretical guarantees. We discuss such examples in the following part.

Chains, Trees and Simple Loops

Open chains and tree-structured models. Chain and tree-structured graphs are distinct models since they can be solved exactly with dynamic programming. We show the basic concepts with the simpler model – an open chain with pairwise cliques (edges). Interestingly, undirected and directed chain graphical models are equivalent. Such models consist of N nodes that are linked by $N-1$ edges, *i.e.* the chain does not form any loop and remains open. Nodes are characterized by unary potential functions $\psi_u(x_i)$ which capture the correlation between the state (assigned label) of the unobserved variable and observed data D_i. Edges are characterized by pairwise potential functions $\psi_p(x_i, x_{i+1})$ and encode priors. Typically, pairwise terms enforce smoothness constraints by encouraging neighbouring variables to take the same labels. Hence, the energy function $E(\mathbf{x}|\mathbf{D})$ is defined as

$$E(\mathbf{x}|\mathbf{D}) = \sum_{i=1}^{N} \psi_u(x_i) + \sum_{i=1}^{N-1} \psi_p(x_i, x_{i+1}). \tag{2.9}$$

At this point, we do not constrain the pairwise potential function ψ_p to some particular form since the exact solution (minimum of energy *Eq.* 2.9) can be found with dynamic programming in two sweeps (forward and backward) in $\mathcal{O}(NL^2)$. This represents huge

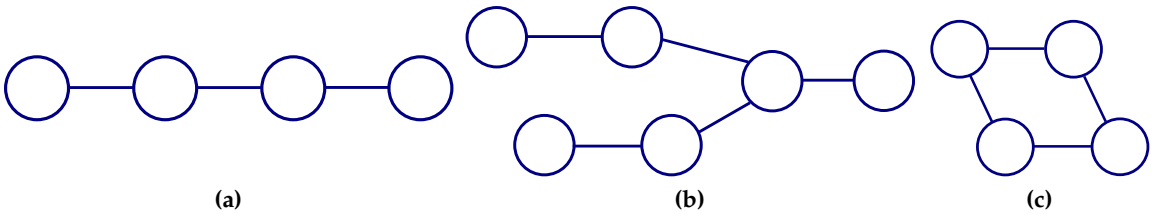


Figure 2.4: (a) Open chains, (b) tree-structured models and (c) simple loops (closed chains) are instances of graphical models which can be solved exactly with dynamic programming.

savings with respect to the brute-force approach which would require $\mathcal{O}(L^N)$ operations. Let us note that the computational complexity can further be reduced to $\mathcal{O}(NL)$ for certain classes of pairwise potentials (Felzenszwalb and Huttenlocher, 2006).

Inference for tree-structured models is very similar; we just need to order the messages in a particular way. In the forward pass, we first designate an arbitrary variable node as the root node and then we proceed from leaves to the root of the tree. The min-cost path is then extracted during the reverse pass. The computational complexity for tree-structured models is larger than for open chains since we must minimize over multiple variables at the junctions in the tree (*cf.* Fig. 2.5).

Closed chain model. Unfortunately, models for most computer vision tasks (semantic segmentation, dense depth estimation, ...) are typically defined over grid lattices with loopy clique structure. Thus, we cannot use dynamic programming naively for such models. While we will discuss suitable inference methods for such models in detail in the following part, a special case exists for which dynamic programming approach is very suitable – a closed chain model (*cf.* Fig. 2.6).

Such model consists of N nodes and N edges, *i.e.* the last edge closes the chain by linking the last and the first node (to simplify notation, we use $X_{N+1} = X_1$)

$$E(\mathbf{x}|\mathbf{D}) = \sum_{i=1}^{N} \left[\psi_u(x_i) + \psi_p(x_i, x_{i+1})\right], \tag{2.10}$$

Exact inference for this model is achieved by "fixing" one node, and computing the min-cost path with standard dynamic programming for each fixed label $l \in \mathcal{L}$. The actual labelling is then extracted as a solution corresponding to the lowest energy. Although this procedure provides an exact solution, it is only tractable when the label space L is small[1] and is not directly applicable for graphs with multiple loops.

[1]In practice, each node can have different label space $\mathcal{L}$. In such case, we can fix the node with the smallest number of labels for computational efficiency.

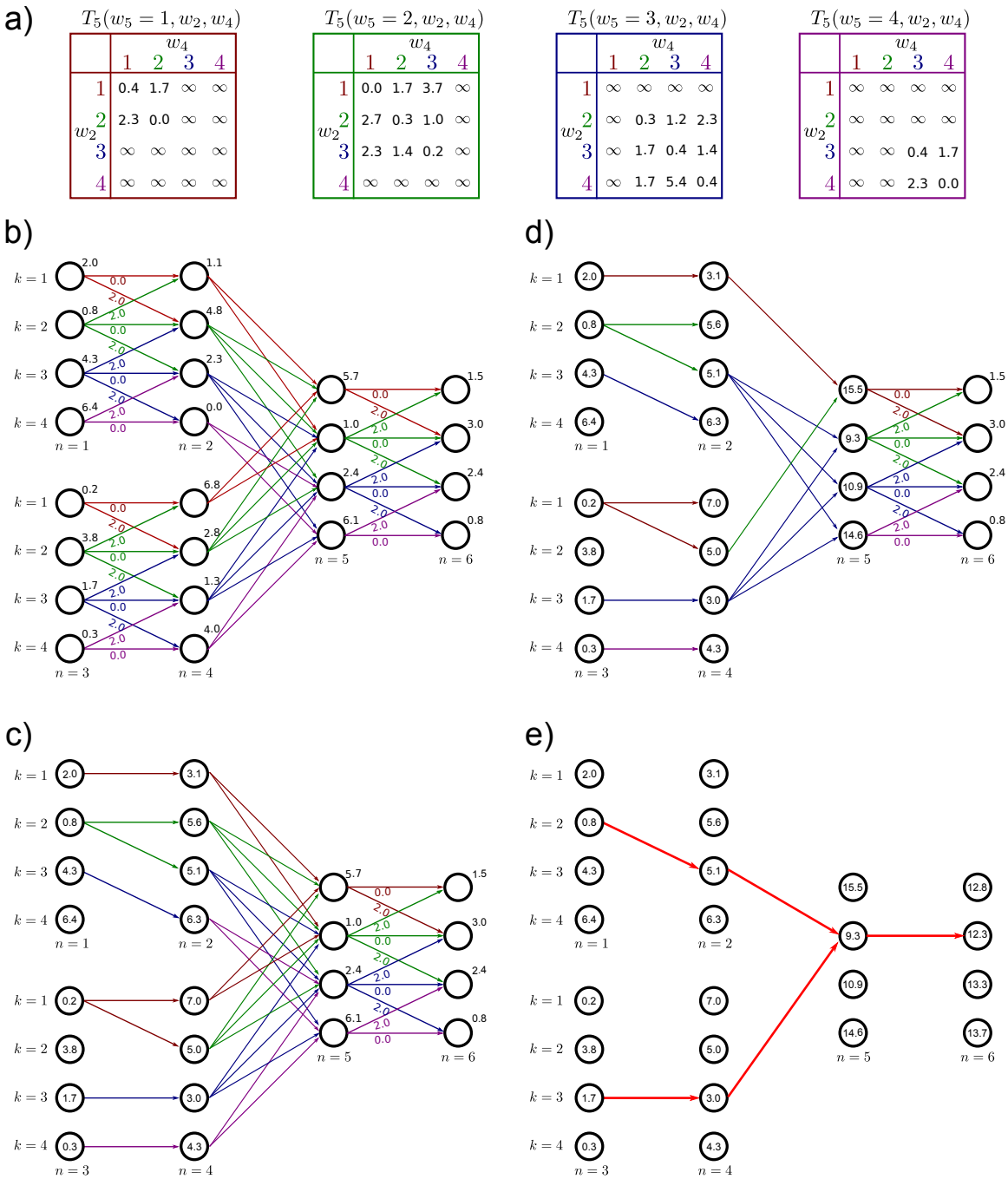
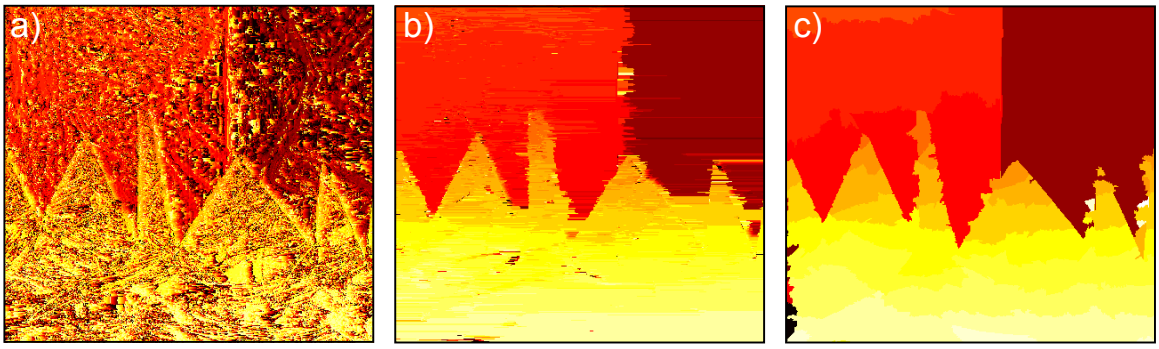
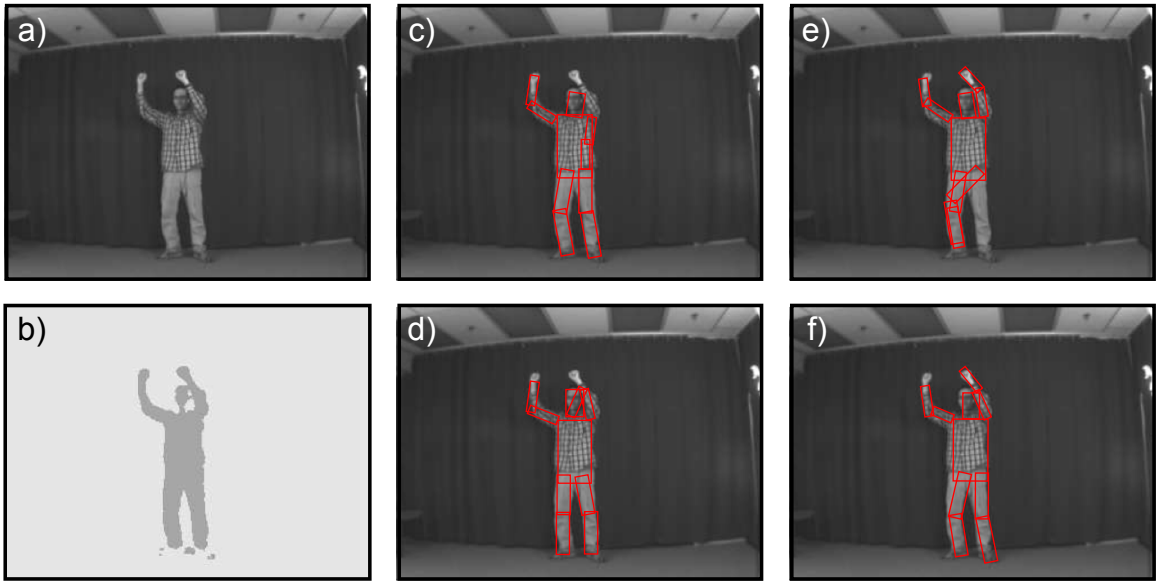


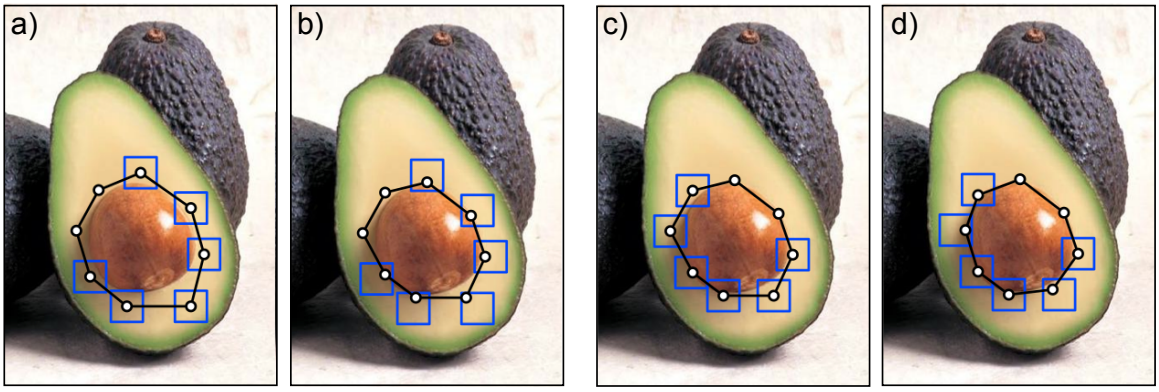
Figure 2.5: Dynamic programming on a tree-structured model from Fig. 2.4: (a) costs for the junction potential, (b) unary and pairwise potentials, (c) we start from the leaves and proceed to the branch, (d) when we reach the branch, we find the minimal cost considering every combination of the incoming states, (e) we continue until we reach the root and find the minimum cost. During the path reconstruction, we need to split correctly at the junction according to which pair of states was chosen (Prince, 2012).



(a) Dense depth estimation: a) winner-takes-all (unary potentials), b) each horizontal scanline solved independently (open chain model), c) tree-structured model (Veksler, 2005)



(b) Human pose estimation with pictorial structures (Felzenszwalb and Huttenlocher, 2005)



(c) Snake-like object segmentation (Felzenszwalb and Zabih, 2011)

Figure 2.6: Examples of open-chain, tree-structured and closed chain graphical models.

Let us note that a common heuristic for closed chain models is to use an iterative inference algorithm; in each iteration, we fix two neighbouring nodes and optimize over the remaining part of the curve (which now forms an open chain). Then we keep iterating until convergence and in each iteration we fix different nodes. While such an approach guarantees convergence only to local minimum (it can be seen as a block coordinate descent with exactly solved blocks), this approach is often prefered due to its speed (Felzenszwalb and Zabih, 2011).

Cyclic Graphs

The expressiveness of chain and tree-structured models is quite limited (*cf.* Fig. 2.6). Usually, we need models with richer interactions between the random variables and hence we use loopy pairwise grid graphs. Such models consists of N nodes which are typically linked to all other nodes in some neighbourhood $\mathcal{N}_i$. In contrast to simpler chain or tree-structured models, they are able to model more complex interdependencies between the random variables and as such, generally provide "smoother" solutions. A classic example is preventing of artifacts in dense depth estimation (*cf.* Fig. 2.6). However, inference in pairwise CRFs is generally NP-hard and we need to constrain the model to a particular structure and class of pairwise potentials to allow efficient inference.

Binary (submodular) problems. We start with the classic 4/8-neighbourhood model, *i.e.* the random variables are associated with a grid-structured lattice and each random variable X_i is connected to all four or eight neighbouring nodes

$$E(\mathbf{x}|\mathbf{D}) = \sum_{i=1}^{N} \left(\psi_u(x_i) + \sum_{j \in \mathcal{N}_i} \psi_p(x_i, x_j) \right). \tag{2.11}$$

If the label space is binary ($L = 2$) and all pairwise potentials fulfil

$$\psi_{ij}(0,0) + \psi_{ij}(1,1) \leq \psi_{ij}(0,1) + \psi_{ij}(1,0), \tag{2.12}$$

we say that such function is regular and it can be exactly minimized by minimizing an equivalent submodular function over the nodes. This is implemented as a mincut problem over a graph (Kolmogorov and Zabin, 2004). Note that an efficient variant for dynamically changing graphs also exists (Kohli and Torr, 2007).

A classic example of such model is foreground/background segmentation (*cf.* Fig. 2.7). Unary potentials typically measure how well the pixel appearance is represented by a Gaussian Mixture Model classifier and pairwise potentials enforce smoothness. One class of

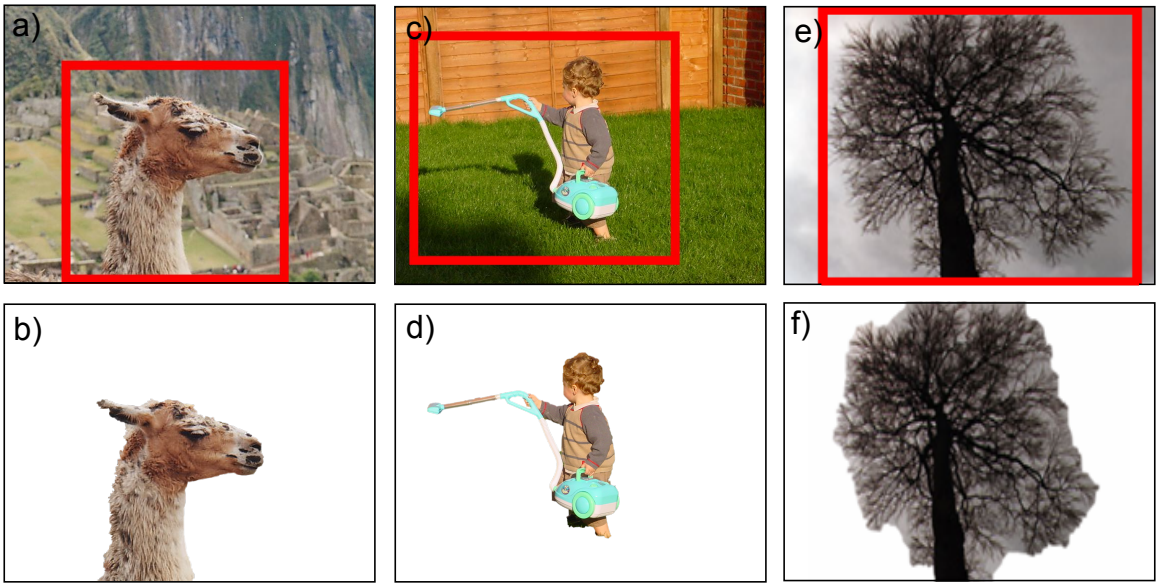


Figure 2.7: Interactive foreground/background segmentation. A user defines only the region of interest (top) which is used to learn the GMM classifier. This classifier is used to compute unary potentials for CRF model with contrast sensitive Potts pairwise potentials. Next, GrabCut segments the foreground by alternating between adapting the classifiers and segmenting the image. User can provide extra guidance to the algorithm by adding extra foreground/background strokes after each iteration (Rother *et al.*, 2004).

pairwise potentials that fulfills the condition *Eq.* 2.12 is the (contrast sensitive) Potts model which takes the following form

$$\psi_p(x_i, x_j) = \begin{cases} 0, & \text{if } x_i = x_j \\ \kappa(D_i, D_j), & \text{otherwise.} \end{cases} \tag{2.13}$$

This cost is zero if two random variables take the same label but adds a data-dependent penalty $\kappa(D_i, D_j)$ for any other assignment. One way of setting the penalty $\kappa(D_i, D_j)$ is to use a mixture of Gaussian kernels over appearance features I and location features p

$$\kappa(D_i, D_j) = w_1 \exp\left(-\frac{|p_i - p_j|^2}{2\theta_\alpha^2} - \frac{|I_i - I_j|^2}{2\theta_\beta^2}\right) + w_2 \exp\left(-\frac{|p_i - p_j|^2}{2\theta_\gamma^2}\right), \tag{2.14}$$

where w_1 and w_2 are relative weights controlling strength of these kernels and θs define their bandwidth. The first kernel ensures that pixels with similar appearance would get the same label. Similarly, the second kernel enforces spatial smoothness by removing isolated (noisy) regions.

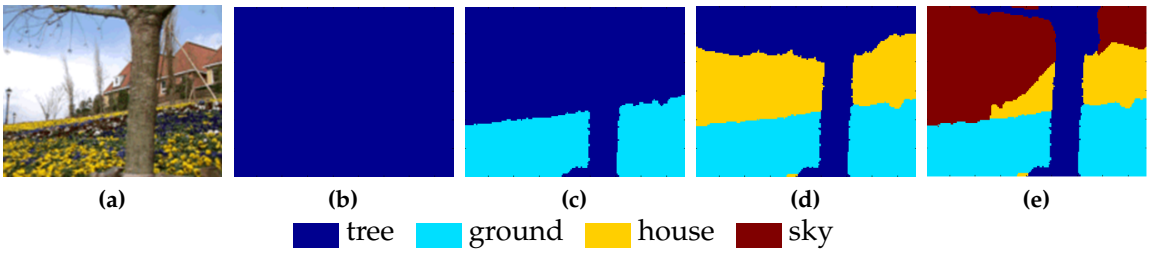


Figure 2.8: α-expansion iterations. We initialize with label *tree* and then subsequently expand labels *ground*, *house* and *sky*. In each iteration we either preserve the current solution or accept the expanded label if the new energy is lower (Kumar and Kohli, 2008).

Multi-label problems. Most computer vision problems are not binary but often involve tens or even hundreds of labels. This problem is NP-hard, however, an approximate solution can be found using the α-expansion algorithm. This procedure assumes a similar constraint on pairwise potentials

$$\psi_{ij}(\alpha, \alpha) + \psi_{ij}(x_i, x_j) \leq \psi_{ij}(\alpha, x_j) + \psi_{ij}(x_i, \alpha), \qquad \forall \alpha, x_i, x_j \in \mathcal{L}, \tag{2.15}$$

which can be viewed as transforming a multi-label problem into a series of binary problems where value 1 indicates the current assignment x_i and value 0 the proposed move α. In other words, we use a block coordinate descent algorithm which is guaranteed to converge to a local minimum since each block is solved exactly (Fig. 2.8).

Although we have focused on the graph-cut method, there is a variety of alternative methods based on linear programs (Wainwright *et al.*, 2002; Werner, 2007; Kumar *et al.*, 2009), primal-dual formulations (Komodakis *et al.*, 2011; Jojic *et al.*, 2010) or message passing (Yedidia *et al.*, 2003; Wainwright and Jordan, 2008).

Densely-connected Pairwise Graphs

Grid pairwise CRF models with 4/8 neighbourhood have formed a long-standing basis for many computer vision problems, however, their expressiveness is quite limited since they are not able to model long-range interactions. This is addressed by densely-connected pairwise models, where each node X_i is connected with all remaining random variables

$$E(\mathbf{x}|\mathbf{D}) = \sum_{i=1}^{N} \psi_u(x_i) + \sum_{i<j} \psi_p(x_i, x_j). \tag{2.16}$$

This is very important since locally-connected pairwise models are able to propagate information only over parts of the graph which is sufficiently "uncertain". In other words, if there is a region isolated by globally optimal partial assignment of variables, the model

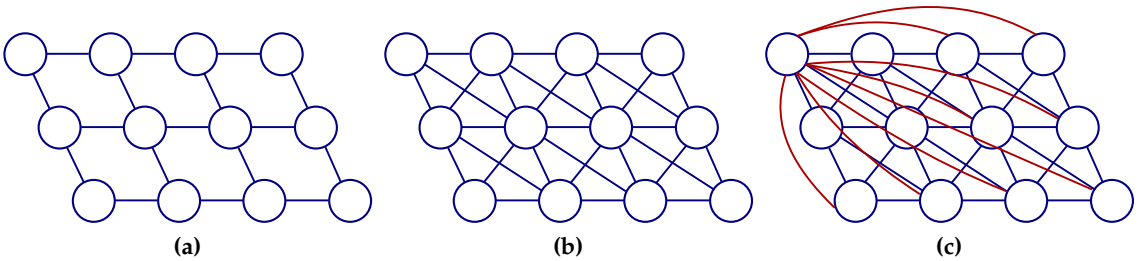


Figure 2.9: Pairwise models: (a) 4-neighborhood, (b) 8-neighborhood and (c) densely connected model in which each variable is connected with all remaining nodes. The difference between densely connected and 8-neighborhood models is highlighted by red edges. Extra edges are shown only for the first node to avoid clutter.

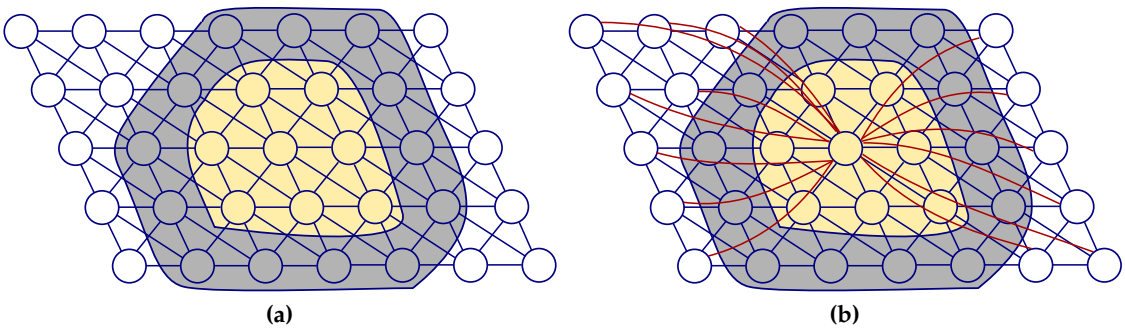


Figure 2.10: If the gray area has globally optimal partial assignment of variables, standard 4/8-neighborhood CRF models cannot propagate information between the isolated parts (white and yellow). In contrast, the dense CRF model allows to "leap over" such areas and hence propagates information more efficiently. This property is very useful in practice since it exploits *e.g.* common repetitive patterns within an image more efficiently.

(despite being global) is not able to propagate any information to such regions. Nowadays, deep learning models often provide such strong potentials; the densely-connected CRF models allow to "leap over" areas that have globally optimal partial assignment of variables and propagate information more efficiently (*cf.* Fig. 2.10) (Shekhovtsov, 2014). Moreover, richer expressiveness of model allows to better capture finer details along object boundaries.

The main issue with densely-connected model is prohibitively large run-time for graph-cut or MCMC inference, which is in order of tens of hours for a single image. However, for models with pairwise potentials constrained to the form of a mixture of Gaussian kernels (*Eq.* 2.14), fast run-times are achievable using the efficient permutohedral lattice-based filtering method formulated either in the mean-field framework (Krähenbühl and Koltun, 2011) or as recently proposed quadratic or linear relaxations of integer program (Desmaison *et al.*, 2016; Ajanthan *et al.*, 2017). Next, we provide a brief overview of variational filter-based mean-field approach, which is a specific case of a generic message passing algorithm.

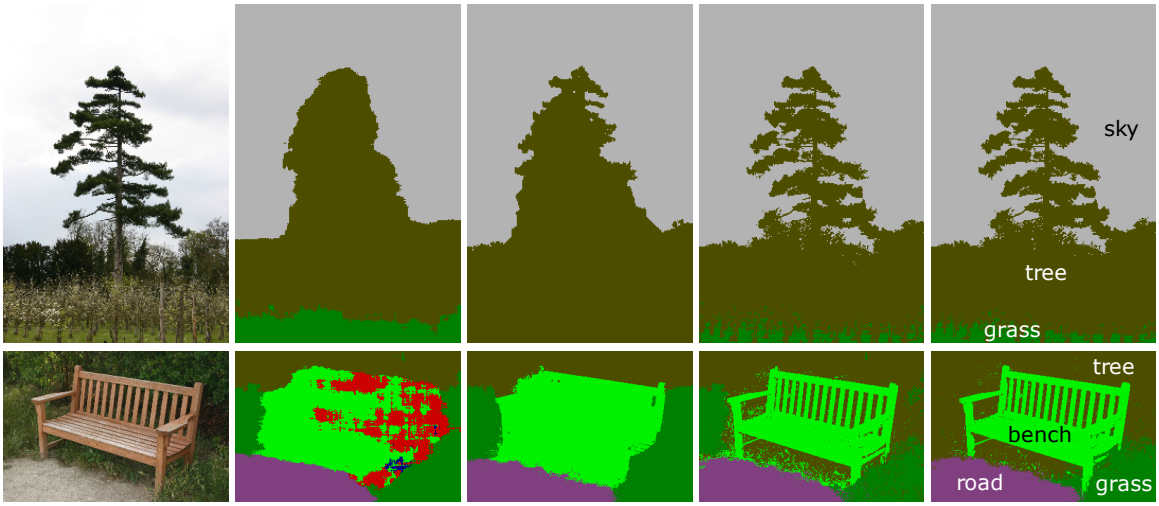


Figure 2.11: Efficiency of Dense CRF. From left to right: input image, unary potentials, 8-neighborhood pairwise model, dense CRF, ground-truth (Krähenbühl and Koltun, 2011).

Filter-based mean-field inference. The key idea behind the mean-field method is to approximate the complex probability distribution that is intractable (in the sense of finding its maximum or mode) by a simpler one that can be solved efficiently. Clearly, to make such an approximation useful, we cannot pick an arbitrary distribution, however, the simpler distribution has to resemble the more complex one as closely as possible subject to some measure of similarity between the two models. Thus, the key parts of the mean-field approach involve:

1. Definition of a measure that allows to compare similarity between intractable distribution P and tractable approximation Q.

2. Specification of a class of probability distributions in which we want to find a similar distribution Q.

3. Finding approximation Q such that it minimizes the similarity measure to the intractable distribution P.

4. Solving the maximization problem for the approximate distribution Q.

A natural measure between probability distributions are α-divergences. The mean-field method uses a specific case, called Kullback-Leibler (KL) divergence, in the form of

$$D_{KL}(Q||P) = \sum_{\mathbf{x}} Q(\mathbf{x}) \log \frac{Q(\mathbf{x})}{P(\mathbf{x})} = -\sum_{\mathbf{x}} Q(\mathbf{x}) \log P(\mathbf{x}) + \sum_{\mathbf{x}} Q(\mathbf{x}) \log Q(\mathbf{x}). \tag{2.17}$$

This divergence measure is convex with respect to both P and Q. As such, it satisfies the basic properties of an error measure, *i.e.* $D_{KL}(Q||P) \geq 0$ for all P, Q and $D_{KL}(Q||P) = 0$ if

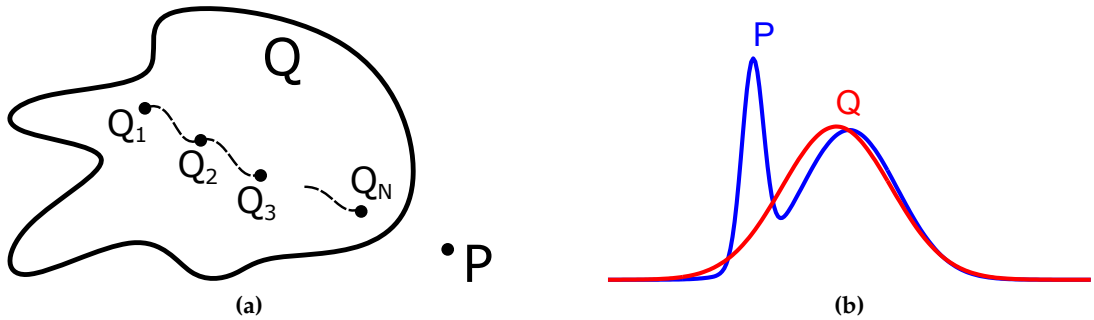


Figure 2.12: (a) The mean-field approach is an iterative algorithm which approximates the true distribution P by an approximate but tractable distribution Q. Each iteration leads to a better approximation. (b) Kullback-Leibler divergence $D_{KL}(Q||P)$ is exclusive, *i.e.* if two identical Gaussians are separated enough it prefers to represent only one of them.

and only if $P = Q$. However it is not a metric since it is not commutative ($D_{KL}(P||Q) \neq D_{KL}(Q||P)$) and does not satisfy the triangle inequality. We also say that $D_{KL}(Q||P)$ is an *exclusive* divergence (Fig. 2.12). This means that if two identical Gaussians are separated enough, an exclusive divergence prefers to represent only one of them (Minka, 2005).

Plugging the Gibbs distribution into the Kullback-Leibler $D_{KL}(Q||P)$ similarity measure (*Eq.* 2.17), we obtain

$$\begin{aligned} D_{KL}(Q||P) &= -\sum_{\mathbf{x}} Q(\mathbf{x}) \log\left(\frac{1}{Z}\exp(-E(\mathbf{x}))\right) + \sum_{\mathbf{x}} Q(\mathbf{x}) \log Q(\mathbf{x}) \\ &= \sum_{\mathbf{x}} Q(\mathbf{x})E(\mathbf{x}) + \log Z + \sum_{\mathbf{x}} Q(\mathbf{x}) \log Q(\mathbf{x}), \end{aligned} \tag{2.18}$$

where we have used the fact that $\sum_{\mathbf{x}} Q(\mathbf{x}) = 1$. Since $\log Z$ is a constant, it does not influence optimization and hence minimization of KL divergence $D_{KL}(Q||P)$ is equivalent to minimization of the following functional

$$F(Q) = \sum_{\mathbf{x}} Q(\mathbf{x})E(\mathbf{x}) + \sum_{\mathbf{x}} Q(\mathbf{x}) \log Q(\mathbf{x}). \tag{2.19}$$

The first term is the expected value of the energy $E(\mathbf{x})$ under probability distribution Q and second term is the negative entropy of probability distribution Q.

By expanding the first term and rearranging the order of summations we obtain

$$\sum_{\mathbf{x}} Q(\mathbf{x})E(\mathbf{x}) = \sum_{i\in\mathcal{V}} \sum_{x_i} Q(x_i)\psi_u(x_i) + \sum_{i,j} \sum_{x_i,x_j} Q(x_i)Q(x_j)\psi_p(x_i,x_j). \tag{2.20}$$

Thus the expected value of the energy under distribution Q is equal to the sum of the expected clique energies.

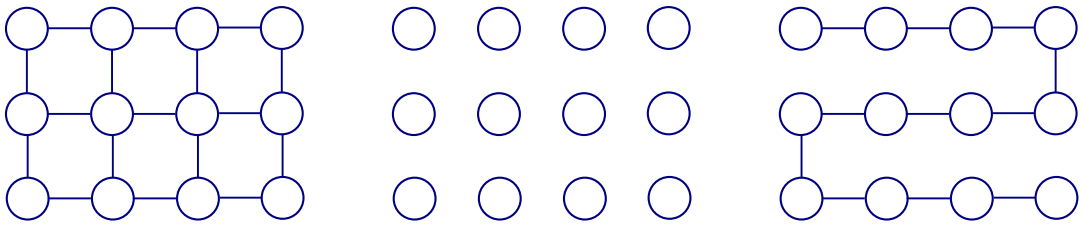


Figure 2.13: (a) Naive mean-field approximation approximates the true distribution P by simpler distribution Q (b) which is modelled as a product of independent distributions, each defined on a single random variable X_i. Such approximation often gets stuck in local minima. (c) One way how this can be improved is using structured mean-field approximation, which splits the original graph into a set of tractable distributions.

Now, we discuss the family of approximate distributions Q. The simplest choice is *naive mean-field* approximation (Fig. 2.13), which assumes that Q is a product of independent distributions, each defined on a single random variable X_i

$$Q(\mathbf{x}) = \prod_{i \in \mathcal{V}} Q(x_i). \tag{2.21}$$

An advantage of choosing such a simple approximation is that the negative entropy term in *Eq.* 2.19 decomposes into a sum of entropies of the individual probability distributions Q_i

$$\sum_{\mathbf{x}} Q(\mathbf{x}) \log Q(\mathbf{x}) = \sum_i \sum_{x_i} Q(x_i) \log Q(x_i). \tag{2.22}$$

Putting this together, the mean-field functional for a pairwise random field takes the form of

$$F(Q) = \sum_{i \in \mathcal{V}} \sum_{x_i} Q(x_i)\psi_u(x_i) + \sum_{i,j} \sum_{x_i,x_j} Q(x_i)Q(x_j)\psi_p(x_i, x_j) + \sum_i \sum_{x_i} Q(x_i) \log Q(x_i). \tag{2.23}$$

Let us note that the naive mean-field approach uses very simple approximation which leads to poor convergence properties. This can be overcome by using structured mean-field approximation which provides better accuracy and faster convergence (Koller and Friedman, 2009; Wainwright and Jordan, 2008).

Next, we need to find distribution Q from the defined family that is close to the intractable distribution P. This is cast as a constrained minimization problem

$$\begin{aligned} \underset{Q(x)}{\text{minimize}} \quad & F(Q) \\ \text{subject to} \quad & \sum_{x_i} Q(x_i) = 1, \ \forall i \in \mathcal{V}. \end{aligned} \tag{2.24}$$

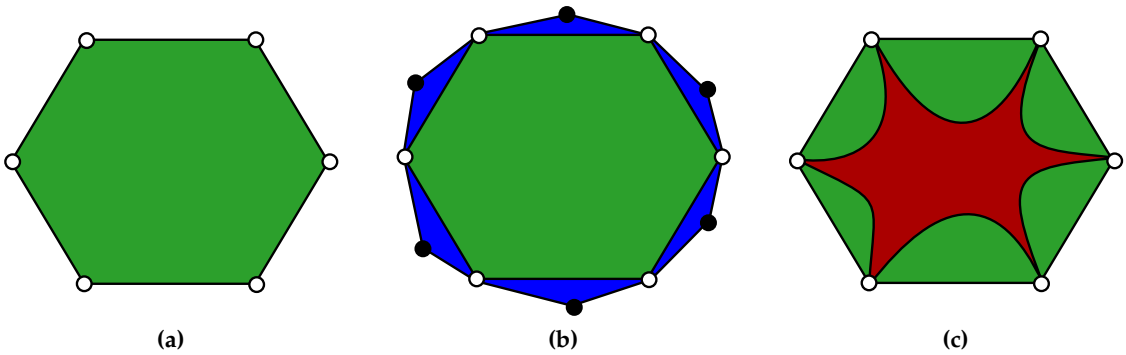


Figure 2.14: (a) Marginal polytope (green), (b) its outer approximation called local polytope (blue) and (c) mean-field approximation (red) (Wainwright and Jordan, 2008).

which can be approached using Lagrange multipliers

$$L(Q, \lambda) = F(Q) + \sum_{i \in \mathcal{V}} \lambda_i \left(\sum_{x_i} Q(x_i) - 1 \right). \tag{2.25}$$

Taking the derivative of $L(Q, \lambda)$ with respect to $Q(x_i)$, setting it to zero, re-arranging the terms and re-normalizing leads to the mean-field update for $Q(x_i)$

$$Q(x_i = l) = \frac{1}{Z_i} \exp \left(-\psi_i(x_i = l) - \sum_{l' \in \mathcal{L}} \sum_{j \neq i} Q(x_j = l') \psi_{ij}(x_i, x_j) \right), \tag{2.26}$$

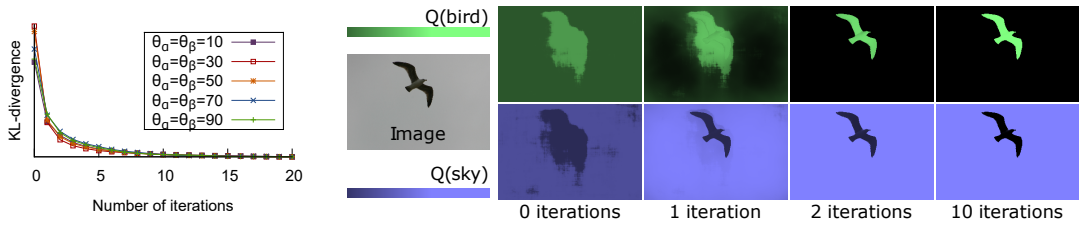
where $Z_i = \sum_{x_i = l \in \mathcal{L}} \exp \left(-\psi_i(x_i) - \sum_{l' \in \mathcal{L}} \sum_{j \neq i} Q(x_j = l') \psi_{ij}(x_i, x_j) \right)$ is a constant which normalizes the marginal at node i.

A naive mean-field algorithm for densely connected graphs has a quadratic computational complexity in the number of variables since each update $Q(x_i)$ involves summation over all remaining variables. This is prohibitively large, however, Krähenbühl and Koltun (2011) showed that it can be reduced to linear complexity by interpreting the message-passing step in mean-field updates as high-dimensional low-pass filtering in the Q space, which can be performed very efficiently using permutohedral lattice. This algorithm is summarized in Algorithm 2.1.

Although the filter-based mean-field inference for densely connected CRFs has been shown to be very efficient in practice, it fails to provide strong theoretical guarantees on the quality of its solutions. The problem itself is non-convex and as such is sensitive to initialization and leads to many significant computational challenges (multiple local minima, *etc.*). To address this deficiency, it has recently been shown that it is possible to use the same filtering approach to speed-up the optimisation of convex quadratic programming (QP) and linear programming (LP) relaxations (Desmaison *et al.*, 2016; Ajanthan *et al.*, 2017).

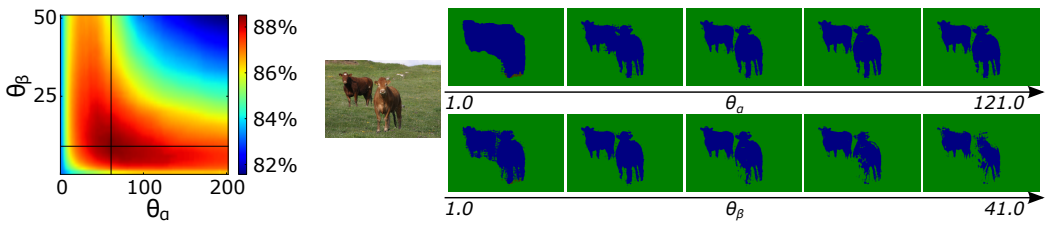
Algorithm 2.1 Mean-field algorithm in densely connected CRFs

1: Initialize Q ▷ $Q_i(x_i) \leftarrow \frac{1}{Z_i} \exp\{-\psi_u(x_i)\}$
2: **while** not converged **do**
3: $\quad \widetilde{Q}_i^{(m)}(l) \leftarrow \sum_{j \neq i} k^{(m)}(D_i, D_j) Q_j(l)$ for all m ▷ **Message passing** from all X_j to all X_i
4: $\quad \widehat{Q}_i(x_i) \leftarrow \sum_{l \in \mathcal{L}} \mu^{(m)}(x_i, l) \sum_m w^{(m)} \widetilde{Q}_i^{(m)}(l)$ ▷ **Compatibility transform**
5: $\quad Q_i(x_i) \leftarrow \exp\{-\psi_u(x_i) - \widehat{Q}_i(x_i)\}$ ▷ **Local update**
6: $\quad$ normalize $Q_i(x_i)$
7: **end while**
8: **return** Q



(a) **(b)**

Figure 2.15: (a) KL divergence of the mean-field approximation during successive iterations for different values of θs parameters. (b) Visualization of the Q values across first 10 iterations for bird and sky labels (Krähenbühl and Koltun, 2011).



(a) **(b)**

Figure 2.16: (a) Influence of pairwise kernel parameters θ_α and θ_β on accuracy. (b) Qualitative results for one image (Krähenbühl and Koltun, 2011).

2.2 Parameter Learning

In the previous section we have described how we formulate computer vision problems through an elegant framework of probabilistic graphical models. Now we discuss how we learn parameters of these models.

2.2.1 Supervised Learning / Empirical Risk Minimization

Throughout this thesis we mostly focus on semantic segmentation of 3D environments. In other words, we want to assign a semantically meaningful label (*e.g.* tree, road, ...) from a predefined label set to each pixel in an image or voxel in a 3D space. To this end, we need need to *learn* a model h which is able to *predict* some hidden property y from output set $\mathcal{Y}$ of the observed data $x \in \mathcal{X}$. In the *supervised learning* case, we are given labeled training data $\mathcal{D} = \{(x_1, y_1), \ldots, (x_N, y_N)\} \in (\mathcal{X} \times \mathcal{Y})^N$ consisting of N *examples* (input-output pairs) and we wish to learn a mapping $h : \mathcal{X} \to \mathcal{Y}$. We assume that there is a an underlying joint probability distribution $P(X, Y)$ that governs the generation of data, and that the samples of training data $\mathcal{D}$ are drawn independently and identically distributed.

An important question is how to pick the hypothesis h from a hypothesis class $\mathcal{H} \subseteq \{h : \mathcal{X} \to \mathcal{Y}\}$. We wish a model that predicts outputs $y^* = h(x)$ close enough to expected output y on test data (*i.e.* data for which only x is observed). This is formalized through the concepts of *expected* and *empirical* risks (details can be found in, *e.g.* Bishop (2006)). Loosely speaking, we wish to minimize a non-negative cost function $L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$ which vanishes on the diagonal (*i.e.* $L(y, y) = 0$). An example of such a cost function is the $0/1$ cost $L_{0/1}(y^*, y) = [y^* \neq y]$ ($[\cdot]$ is the Iverson bracket). Intuitively, we want as little cost as possible on unseen data.

The *expected risk* of a model h with respect to loss function L is

$$\text{risk}(h; L) := \mathbb{E}_{X,Y}\left[L(h(X), Y)\right], \tag{2.27}$$

where $\mathbb{E}_{X,Y}$ denotes the expectation operator under $P(X, Y)$. Ideally, we would like to use model $h^* \in \mathcal{H}$ which minimizes the expected risk; however we do not have an access to the true distribution $P(X, Y)$ and hence the best we can do without further assumptions on $P(X, Y)$ is to minimize the *empirical risk* over N data samples

$$\text{risk}_{\mathcal{D}}(h; L) := \mathbb{E}_{\mathcal{D}}\left[L(h(X), Y)\right] = \frac{1}{N}\sum_{i=1}^{N} L(h(x_i), y_i). \tag{2.28}$$

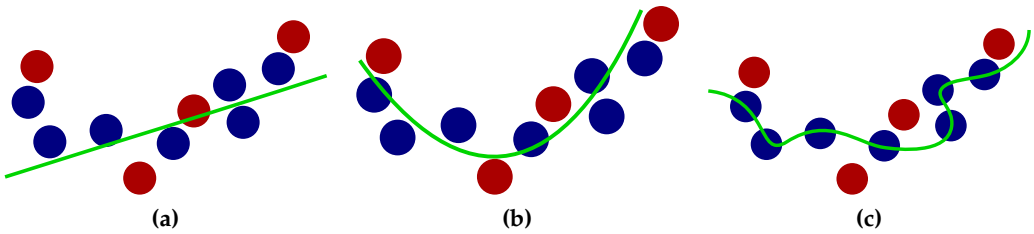


Figure 2.17: A regression problem with polynomial models. The training samples are blue circles and test examples are depicted in red. (a) Model is not complex enough to represent the data well. (c) With increased complexity of the model, we can reduce the empirical risk, however this increases the danger of overfitting the training data. (b) Our aim is to balance these two extremes.

Unfortunately, optimizing empirical risk directly often leads to two well-known problems:

- For many loss functions, such optimization represents a difficult combinatorial problem which is intractable even for relatively simple models,
- If the training set $\mathcal{D}$ is too small, the classifier might easily *overfit* the data which leads to poor generalization.

The first issue can be overcome by the *surrogate loss* and the second avoided by introduction of a *regularizer* which penalizes the complexity of the model.

2.2.2 Linear Models

In this review, we focus on the most commonly used linear discriminative models, in particular: logistic regression, SVM and their structured counterparts. Linear models can be formulated as convex programs, which is a desired property since they can be efficiently optimized using the well-studied machinery of convex optimization and we do not need to worry about local minima. In the rest of this section, we consider linear models

$$h_w(x) = \operatorname*{argmax}_{y \in Y} w \cdot f(x, y) \tag{2.29}$$

where $(\cdot)$ is the inner product, *i.e.* $w \cdot f(x, y) = \sum_{d=1}^{D} w_d f_d(x, y)$ and $w \in \mathbb{R}^D$ is a parameter vector. For binary or multi-class classification, this *inference* problem is relatively easily solved by simply enumerating over all the labels $y \in Y$ and picking the one with the highest score. However, this is a completely different story in case of structured prediction as we have already seen in section 2.1. The problem of *learning* the parameter vector w is

formalized as an optimization problem

$$\begin{aligned} \text{minimize} \quad & \Omega(w) + \frac{1}{N}\sum_{n=1}^{N} L(w, x_n, y_n) \\ w.r.t. \quad & w \in \mathbb{R}^D \end{aligned} \tag{2.30}$$

where $\Omega : \mathbb{R}^D \to \mathbb{R}$ is a *regularizer* and $L : \mathbb{R}^D \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ is a *loss function*.

Regularization

A natural choice for the regularizer Ω is a norm since it is always guaranteed to be convex and penalizes coefficients of the weight vector w. This behaviour can be interpreted as preference for simpler model (Occam's razor). The two most widely used regularizers are

- L_2-regularization: $\omega_\lambda^{L_2} := \frac{\lambda}{2}\|w\|_2^2$
- L_1-regularization: $\omega_\tau^{L_1} := \tau\|w\|_1$

where λ and τ are non-negative hyper-parameters that trades off between model complexity and minimizing the loss function. These hyper-parameters are typically set through cross-validation.

While the L_2 regularization penalizes coefficients of the weight vector w, the resulting vector remains dense since the coefficients typically remain non-zero. In contrast, the L_1 regularization induces sparsity since some of the coefficients of the weight vector w become exactly zero. This leads to more compact models, since unused features can be omitted. It should be noted that in many applications (*e.g.* 3D geometry), penalizing coefficients independently does not make much sense. In such situations, we often use structured sparsity regularization methods that allow to impose structure *e.g.* according to predefined groups (Bouaziz *et al.*, 2013).

Loss Functions

Next, we focus on loss function L and discuss two typical linear models before we move to structured prediction.

Multinomial Logistic Regression. A log-linear probabilistic model that generalizes logistic regression to multiclass problems

$$\begin{aligned} P(y|x;w) &= \frac{1}{Z(x,w)} \exp\left(w \cdot f(x,y)\right) \\ Z(x,w) &= \sum_{y' \in Y} \exp\left(w \cdot f(x,y')\right) \end{aligned} \tag{2.31}$$

This model can be fit to the data by maximizing the conditional log-likelihood which leads to the multinomial logistic loss function

$$L_{\text{MLR}}(w, x_n, y_n) = -w \cdot f(x_n, y_n) + \log \sum_{y' \in \mathcal{Y}} \exp\left(w \cdot f(x_n, y')\right). \tag{2.32}$$

Support Vector Machines. While multinomial logistic regression provides a notion of confidence via probability, SVM does it via the margin. SVM attempts to score the true label over other labels with maximum margin

$$\begin{aligned} \text{minimize} \quad & \frac{\lambda}{2}\|w\|_2^2 + \frac{1}{N}\sum_{n=1}^{N} \xi_n \\ \text{s.t.} \quad & w \cdot f(x_n, y_n) \geq \max_{y' \in \mathcal{Y}/y_n} \left(w \cdot f(x_n, y') + 1\right) - \xi_n \qquad \forall n \in \{1 \dots N\} \end{aligned} \tag{2.33}$$

where ξ_n are slack variables that relax the margin constraints in the case the data is not linearly separable. With $\xi_n = 0$, the parameter vector w is such that the score of the true output y_n is always greater than of any other labelling $y' \in Y/y_n$ at least by 1 (or other suitable cost function). If the constraint is violated, *i.e.* $\xi_n > 0$, they pay a linear penalty.

We can rewrite this optimization problem in the form of *Eq.* 2.30 by setting the regularization term to $\frac{\lambda}{2}\|w\|_2^2$ and rewriting the loss in its unconstrained form as

$$L_{\text{SVM}}(w, x_n, y_n) = -w \cdot f(x, y) + \max_{y' \in \mathcal{Y}/y_n} \left(w \cdot f(x_n, y') + 1\right). \tag{2.34}$$

By comparing SVM with multinomial logistic regression we observe that SVM uses the hinge loss function which replaces soft-max by max and includes the cost function in the scope of maximization.

2.2.3 Structured Prediction

As we have already seen in Section §2.1, a natural assumption in computer vision is interdependency among the output variables, often with sequential or graphical structure. Structured models allow interactions between a large number of variables, which leads to rich models that are able to represent complex relationships existing between the data. In other words, we learn a mapping h from an input domain $\mathcal{X}$ to a *structured output* domain $\mathcal{Y}$.

As we have seen, there are three notable properties that make structured prediction powerful and at the same time more difficult:

- Input dependent admissible outputs: for input $x \in \mathcal{X}$, some outputs may not be structurally possible and hence not part of the structured output space.
- Global coupling: the problem is not *trivially* decomposable into a set of completely independent (and easy to solve) problems.
- Difficulty / scale of optimization: common computer vision problems are large scale (both, in number of nodes and labels) and the typical inference problems (resp. their unconstrained / general variants) are known to be NP-complete or NP-hard, whereas we usually need to solve in real-time.

Typical structured loss functions are:

Conditional Random Field. CRF is a generalization of a logistic regression model for structured prediction (Lafferty *et al.*, 2001). Instead of matching statistics over indidual random variables, CRF model couples them into groups of random variables

$$\begin{aligned} P(\mathbf{y}|x;w) &:= \frac{1}{Z(w,x)} \exp\left(\sum_{c\in\mathcal{C}} w \cdot f_c(x,\mathbf{y}_c)\right) \\ Z(w,x) &:= \sum_{\mathbf{y}'\in\mathcal{Y}} \exp\left(\sum_{c\in\mathcal{C}} w \cdot f(x,\mathbf{y}'_c)\right) \end{aligned} \tag{2.35}$$

where $\mathcal{C}$ is the set of all such groupings and $c \in \mathcal{C}$ is a particular grouping of random variables, *i.e.* $\mathbf{y}_c = \{\mathbf{y}_i | i \in c\}$. The structured loss function of CRF then corresponds to the negative conditional log-likehood which is written as a soft-max function.

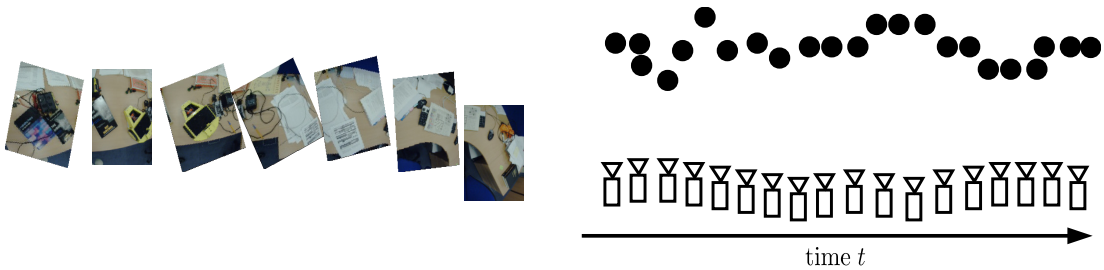
Structured Support Vector Machine (SSVM) . While CRF model is a generalization of multinomial logistic regression for structured problems, SSVM is the structured prediction analog of an SVM (Taskar *et al.*, 2003; Tsochantaridis *et al.*, 2004). This approach is popular for learning of random fields since it does not require evaluation of a partition function which represents a major challenge to maximum likelihood based approaches for CRF learning. Similarly to standard SVM, the structured loss requires the score of the ground-truth labelling $\mathbf{y}$ to be greater than any other hypothesis labelling $\mathbf{y}'$ by a margin. The structured hinge loss replaces the soft-max function by max and includes the cost function in the scope of the maximization. More details about structured prediction can be found in Koller and Friedman (2009); Nowozin and Lampert (2011).

2.3 Multiview Geometry in Computer Vision

As agents navigate throughout the environment, they need to perceive the 3D structure. Reconstructing 3D worlds is relatively straightforward with active sensors such as Kinect or various LIDARs because they directly perceive dense depth. This is significantly more challenging task with passive cameras since we completely lose information about 3D structure during the data acquisition process. In this section, we provide an overview of the complete 3D reconstruction pipeline.

Dense 3D reconstruction has been studied separately for a long time in photogrametry and mobile robotics, however most concepts are shared. The main differences between the two are that photogrametry typically assumes off-line processing of large scale scenes (Snavely *et al.*, 2006; Agarwal *et al.*, 2011) and does not impose any constraints on captured data (*i.e.* images do not need to be sequentially ordered), whereas robotics requires incremental processing of perceived data at real-time rates, however, all captured data is sequentially ordered (Davison *et al.*, 2007; Klein and Murray, 2007; Newcombe *et al.*, 2011b).

(a) Offline Structure-from-Motion (SfM) (Snavely *et al.*, 2006).



(b) Online Simultaneous Localization and Mapping (SLAM) (Strasdat, 2012).

Figure 2.18: Offline SfM does not impose any constraints on the data acquisition process, however such high-quality 3D reconstructions often take hours to compute. In contrast, online SLAM assumes sequentially ordered data however requires real-time processing.

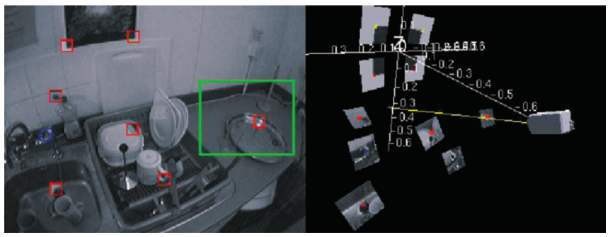
2.3.1 High-level Overview

Let us assume a camera freely moving throughout the environment and capturing a stream of image data. We also assume that all captured video frames are sequentially ordered, *i.e.* we do not deal with images captured at random time instants and locations, however camera moves along a smooth trajectory. Intuitively, if we knew the camera motion (trajectory) and depth of perceived scene, we could project these points into a common reference frame and hence we would reconstruct the 3D scene. The problem is that we need to know the scene structure in order to estimate the camera trajectory and vice versa. This is a classic chicken-and-egg problem dubbed Simultaneous Localization and Mapping (SLAM) in mobile robotics community or structure-from-motion (SfM) in photogrametry and computer vision (Triggs *et al.*, 1999; Hartley and Zisserman, 2003; Thrun *et al.*, 2005).

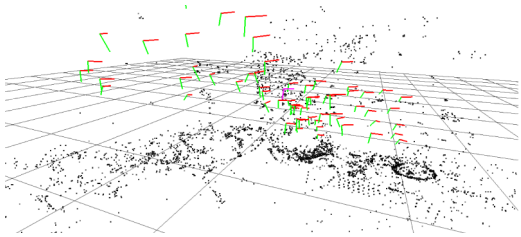
In mobile robotics, we usually use active sensors such as LIDARs or Kinect-like cameras that directly perceive the 3D structure of the environment. In that case, data association and loop closures represent relatively simpler problems since we directly perceive the 3D structure and SLAM more or less boils down to well-studied back-end optimization (bundle adjustment) that jointly optimizes over all observations and camera poses. This is not the case for visual SLAM with passive cameras since we completely lose information about depth during the data acquisition process (we do not sense directly the 3D points, however only their 2D projections) and we need to estimate the missing coordinate (depth).

In the following part, we describe the camera as a measurement device. Then, we discuss epipolar geometry which describes how multiple views are related. Since we are mostly concerned with dense 3D reconstruction in this thesis, we provide a short overview of most common methods for dense depth estimation in Section §2.3.4. Finally, Section §2.3.5 briefly discusses SLAM back-end (bundle adjustment), which jointly optimizes over camera poses and scene geometry.

Figure 2.19: LIDARs directly sense the 3D structure while cameras only its 2D projection.

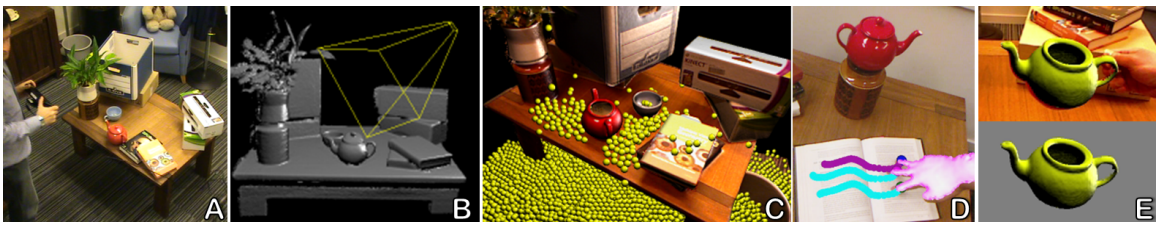


(a) MonoSLAM (Davison *et al.*, 2007)



(b) PTAM (Klein and Murray, 2007)

(c) DTAM (Newcombe *et al.*, 2011b; Stühmer *et al.*, 2010)



(d) KinectFusion (Newcombe *et al.*, 2011a; Izadi *et al.*, 2011; Nießner *et al.*, 2013)

(e) DynamicFusion (Newcombe *et al.*, 2015)

Figure 2.20: (a, b) Sparse SLAM methods are useful for camera pose estimation however they provide only very limited information about environment. (c) This is addressed by dense 3D reconstruction which provide complete information about the scene. Real-time dense 3D reconstruction from passive camera is a difficult and unsolved problem. (d) KinectFusion simplified this problem by using an active camera (Kinect) that directly senses depth and demonstrated high-quality 3D reconstruction. While this approach is limited to indoor environments, it shows how dense 3D reconstruction can be used in interactive scenarios. (e) The main limitation of these approaches is that they assume a static scene. This deficiency is addressed by DynamicFusion, which is able to handle moving and deforming objects, however it assumes pre-segmented moving objects and it must be possible to map the deformed object through non-rigid warp to a canonical view.

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

Figure 2.21: (a) Monocular camera, (b) calibrated stereo rig, (c) RGB-D camera, (d) LIDAR.

2.3.2 Geometry of a Monocular Camera

We start with a short description of a pinhole camera. Consider a point in the 3D space with $\mathbf{X} = (X, Y, Z)^\top$ coordinates which we want to map to the point $\mathbf{x} = (x, y)^\top$ on the image plane, where a line joining the point $\mathbf{X}$ with the camera center intersects the image plane. Let us also assume that the camera poses are represented by rigid body transformations $\mathcal{T}_i \in \mathbf{SE}(3)$, composed of the rotation matrix $\mathbf{R}_i \in \mathbf{SO}(3)$ and a translation vector $\mathbf{t}_i$. Using this transformation, we first project the point $\mathbf{X}$ from world-centric to the camera-centric coordinates. In the next step, we apply the pinhole camera model and project the point to the camera image plane. Using homogeneous coordinates, this non-linear projection can be expressed as a linear mapping, which can be concisely written as

$$\mathbf{x} = \mathbf{K}[\mathbf{R}_i|\mathbf{t}_i]\mathbf{X} = \mathbf{P}_i\mathbf{X}. \tag{2.36}$$

Here, $\mathbf{K}$ represents *intrinsic camera parameters* and $\mathbf{P}_i$ is referred as a *camera matrix*. The intrinsic camera calibration matrix $\mathbf{K}$ has the form

$$\mathbf{K} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.37}$$

Here, f is the focal length and $(p_x, p_y)^\top$ is the principal point which allows us to move the origin of the camera coordinate frame (*cf.* Fig. 2.22). We do not model any lens distortions and assume the images were undistorted during the pre-processing stage. Camera intrinsic parameters $\mathbf{K}$ can be obtained using a known object with standard calibration tools (Bouguet, 2000; Furgale *et al.*, 2015; Liu *et al.*, 2016).

The reverse mapping is not straightforward. Due to the projective nature of a monocular camera, we can only back-project the image point $\mathbf{x}$ to a ray passing through the camera center. However, to recover the original 3D point $\mathbf{X}$, we need to know its depth d.

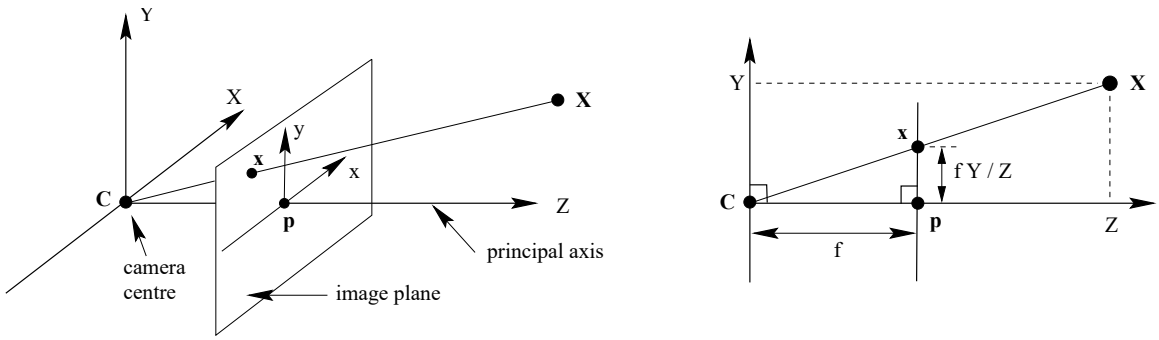


Figure 2.22: Any 3D point $\mathbf{X}$ is imagined on the image plane as point $\mathbf{x}$ at the intersection of the ray connecting point $\mathbf{X}$ and a camera center C with image plane. Principal point $\mathbf{p}$ defines origin of the camera coordinate system and focal length f influences "size" of measured objects and camera field of view (Hartley and Zisserman, 2003).

2.3.3 Epipolar Geometry

To recover the original 3D scene, we will use multiple cameras, respectively multiple camera views. There are many variants of this task however most concepts remain the same for all of them. For instance, in case of multi-view stereo, we have hundreds or even thousands of pre-captured images of the environment and we process the scene off-line. However, we usually do not know anything about the cameras since the camera matrices are unknown and images are typically unordered[2]. On the other hand of the spectrum is monocular 3D reconstruction. In this case, we have only a single camera (Fig. 2.21), however, this camera is typically calibrated (we know the camera intrinsic parameters), images are captured in a causal manner (sequentially ordered) and we need to reconstruct the scene at real-time rates. These two tasks represent probably the two most extreme cases and there are many variants in-between, however, all of them rely on the very same fundamental properties of multi-view geometry.

In the following, we assume that a set of image correspondences between two camera views $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ is given. The reconstruction task is to find the original 3D points $\mathbf{X}_i$ and camera matrices such that

$$\mathbf{x}_i = \mathbf{P}\mathbf{X}_i, \qquad \mathbf{x}'_i = \mathbf{P}'\mathbf{X}_i, \quad \forall i. \tag{2.38}$$

The relation between the image correspondences is formalized through epipolar geometry. The epipolar geometry is the projective geometry between two views, which describes how a 3D point $\mathbf{X}$ is imaged in each view. It is independent of scene structure and depends only on cameras' relative poses and intrinsic parameters.

[2]Often downloaded from internet services such as Flickr, *etc.*

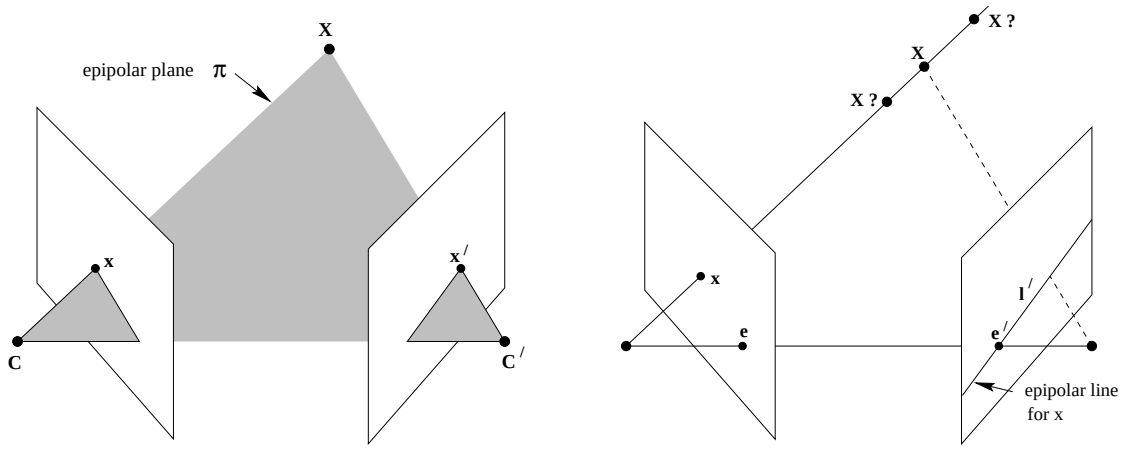


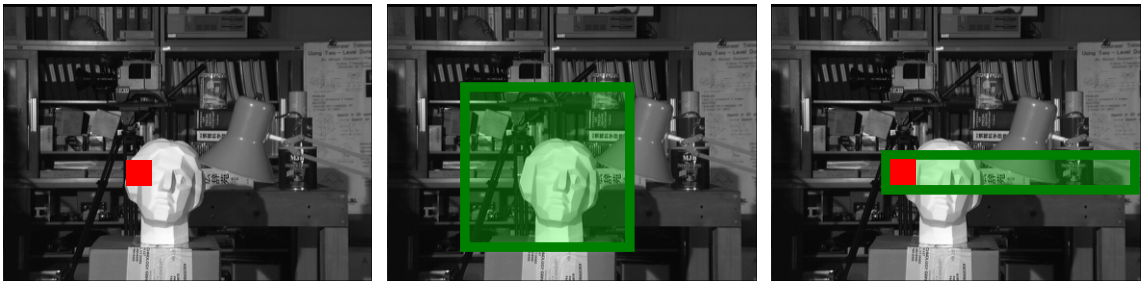
Figure 2.23: Camera centers C, C', the 3D point $\mathbf{X}$ and its projections $\mathbf{x}$, $\mathbf{x}'$ on camera planes are all co-planar and form an epipolar plane (Hartley and Zisserman, 2003).

Fundamental and Essential Matrices

Let us consider a 3D point $\mathbf{X}_i$ which is imagined in two views as points $\mathbf{x}_i$ and $\mathbf{x}'_i$, respectively. In the previous section, we have seen that any 3D point $\mathbf{X}_i$ projects into the point $\mathbf{x}_i$ on the image plane at an intersection with a ray defined by the 3D point itself and the camera center C. Let us now consider the second camera observing the same 3D point $\mathbf{X}_i$. Now, it is projected into the point $\mathbf{x}'_i$, again at the intersection of the camera plane and a ray, however, this time defined by the 3D point itself and camera center C' of the second view. Putting this together, one can see that the 3D point $\mathbf{X}_i$, camera centers C, C' and image correspondences $\mathbf{x}_i, \mathbf{x}'_i$ are co-planar, forming an *epipolar plane* (*cf.* Fig. 2.23).

Knowing only the point $\mathbf{x}_i$, a natural question is whether and how the corresponding point $\mathbf{x}'_i$ in the second camera is constrained. If we again consider the epipolar plane, we see that this plane is fully determined by point $\mathbf{x}_i$ and camera centers C, C'. Thus the corresponding point $\mathbf{x}'_i$ has to lie on the *epipolar line* $\mathbf{l}'$ which intersects the epipolar plane with the image plane (*cf.* Fig. 2.23). This is a very important result, since in most computer vision applications, we are not given a set of ground-truth image correspondences, however our goal is to establish them. Having such constraint allows us to shrink the search space for each point $\mathbf{x}_i$ from the entire image plane (or a local 2D subregion established around coordinates of point $\mathbf{x}_i$) to the epipolar line $\mathbf{l}'$. This not only significantly speedups the matching process but also makes it much more robust (Fig. 2.24).

Such mapping is formalized through the *fundamental matrix*. For any pair of images captured by cameras with non-coincident camera centers, the fundamental matrix $\mathbf{F}$ is a



(a) (b) (c)

Figure 2.24: For a feature point shown in (a), epipolar constraint simplifies correspondence matching from 2D region (b) to 1D matching along the epipolar line (c).

unique 3×3 rank 2 matrix which satisfies

$$\mathbf{x}'^{\top}\mathbf{F}\mathbf{x} = 0 \tag{2.39}$$

for all corresponding points $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$. This constraint is a necessary condition for points to corresponds. It has 7 degrees of freedom and in general can be estimated from at least 7 correspondences. The epipolar line $\mathbf{l}'$ in the second image corresponding to any point $\mathbf{x}$ in the first image can be computed as $\mathbf{l}' = \mathbf{F}\mathbf{x}$. Similarly, if we reverse the order of cameras, the epipolar line $\mathbf{l}$ for image point $\mathbf{x}'$ is defined by $\mathbf{l} = \mathbf{F}^{\top}\mathbf{x}'$.

The cameras we use in visual SLAM are usually calibrated, *i.e.* we know the intrinsic camera parameters $\mathbf{K}$. In that case, we can use the *essential matrix* $\mathbf{E}$, which is a specialization of the fundamental matrix. Since we know the camera matrix $\mathbf{K}$, we can use its inverse to express image points $\mathbf{x}$ in *normalized coordinates* as $\hat{\mathbf{x}} = \mathbf{K}^{-1}\mathbf{x}$. This can be seen as using camera $\mathbf{P}$ with identity $\mathbf{I}$ as a calibration matrix, which removes the effect of known calibration matrix, *i.e.* $\hat{\mathbf{x}} = [\mathbf{R}|\mathbf{t}]\mathbf{X}$. All corresponding points $\hat{\mathbf{x}}_i \leftrightarrow \hat{\mathbf{x}}'_i$ in normalized image coordinates satisfy similar condition as we have seen before for the uncalibrated case

$$\hat{\mathbf{x}}'^{\top}\mathbf{E}\hat{\mathbf{x}} = 0. \tag{2.40}$$

If we substitute image correspondence expressed in normalized coordinates $\hat{\mathbf{x}}$, we obtain $\mathbf{x}'^{\top}\mathbf{K}'^{-\top}\mathbf{E}\mathbf{K}^{-1}\mathbf{x} = 0$. We can compare it with conditions for fundamental matrix $\mathbf{x}'^{\top}\mathbf{F}\mathbf{x} = 0$ and we see that the relationship between essential and fundamental matrix is given by

$$\mathbf{E} = \mathbf{K}'^{\top}\mathbf{F}\mathbf{K}. \tag{2.41}$$

Essential matrix $\mathbf{E}$ has only five degrees of freedom. More details about epipolar geometry can be found in (Hartley and Zisserman, 2003).

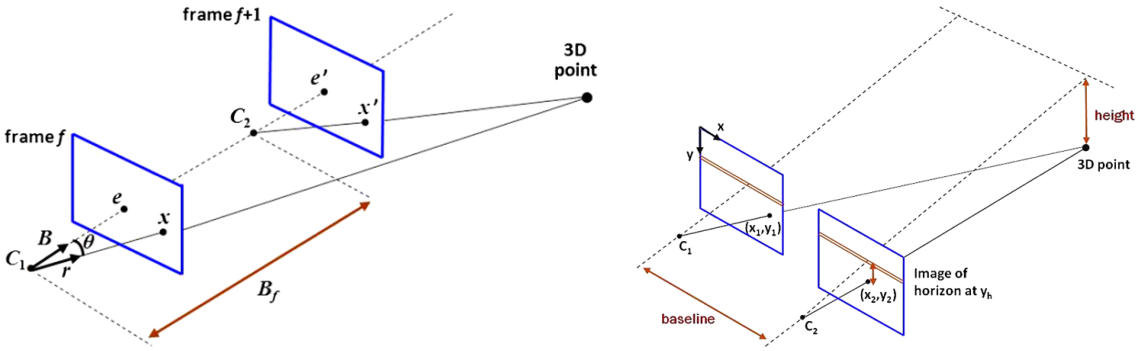


Figure 2.25: In order to estimate depth of point $\mathbf{X}$, we need to now relative camera pose between frames t and $t+1$ and vice versa. This is a chicken-and-egg problem in case of monocular camera (a). Relative camera pose between left and right camera is known and remains constant over time for calibrated stereo rig. Adopted from (Ladický *et al.*, 2012).

Camera Pose Estimation

We assume a calibrated camera, hence we decompose the essential matrix. In contrast to decomposing the fundamental matrix where there is a projective ambiguity, the camera matrices from decomposition of the essential matrix are only up to scale and four-fold ambiguity in solutions. In other words, we retrieve four possible solutions and the only unknown is the overall scale which cannot be determined.

We set the first camera matrix to $\mathbf{P} = [\mathbf{I}|\mathbf{0}]$. The essential matrix has the form of $\mathbf{E} = [\mathbf{t}]_\times \mathbf{R}$, where $[\cdot]_\times$ denotes the cross product matrix. The essential matrix can be factored into the product of a skew-symmetric matrix and a rotation matrix. Hence, it can be shown that using the SVD of $\mathbf{E} = \mathbf{U}\mathrm{diag}(1,1,0)\mathbf{V}^\top$, the four possible choices of the second camera $\mathbf{P}'$ are

$$\mathbf{P}' = [\mathbf{U}\mathbf{W}\mathbf{V}^\top | +\mathbf{u}_3] \text{ or } [\mathbf{U}\mathbf{W}\mathbf{V}^\top | -\mathbf{u}_3] \text{ or } [\mathbf{U}\mathbf{W}^\top\mathbf{V}^\top | +\mathbf{u}_3] \text{ or } [\mathbf{U}\mathbf{W}^\top\mathbf{V}^\top | -\mathbf{u}_3], \quad (2.42)$$

where $\mathbf{u}_3$ is the last column of $\mathbf{U}$, *i.e.* $\mathbf{u}_3 = \mathbf{U}(0,0,1)^\top$ and $\mathbf{W}$ is orthogonal matrix

$$\mathbf{W} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \quad (2.43)$$

The first two solutions for $\mathbf{P}'$ differ only by the direction of the translation vector. The first and the third solutions are rotated by $180°$. Points $\mathbf{X}$ lie in front of both cameras $\mathbf{P}$ and $\mathbf{P}'$ only in one of these four solutions. Hence, it is enough to check for which solution of camera $\mathbf{P}'$ the points $\mathbf{X}$ are in front of cameras using cheirality constraint to decide between the four options (Hartley and Zisserman, 2003).

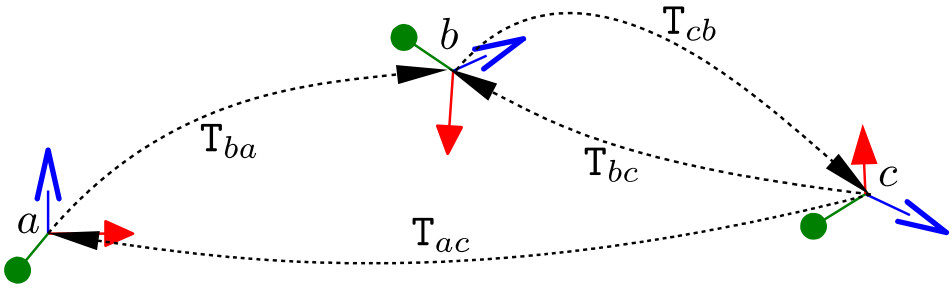


Figure 2.26: Pose transformations between three different reference frames (Strasdat, 2012).

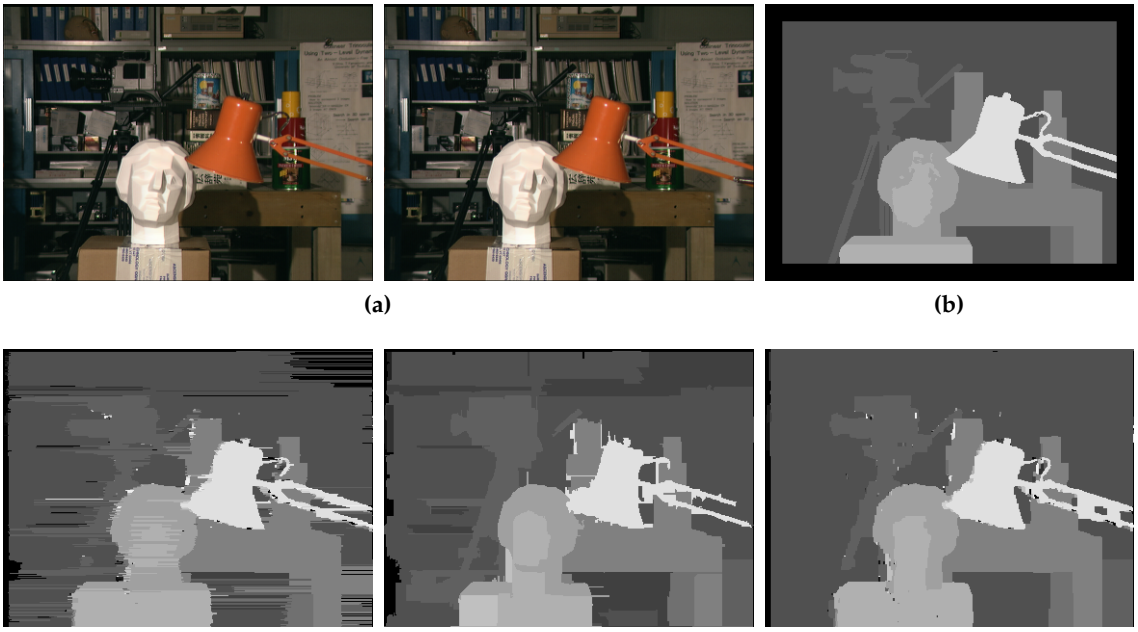
Visual odometry with a calibrated stereo rig. The scale ambiguity present in monocular setup can be eliminated using a calibrated stereo rig. A calibrated stereo-rig consists of a pair of synchronized cameras $\mathbf{P} = \mathbf{K}[\mathbf{I}|\mathbf{0}]$ and $\mathbf{P}' = \mathbf{K}'[\mathbf{R}|\mathbf{t}]$ for which both intrinsic parameters $\mathbf{K}, \mathbf{K}'$ and extrinsic parameters $[\mathbf{R}|\mathbf{t}]$ are known and remain constant over time. These parameters can again be estimated using the standard calibration tools (Bouguet, 2000; Furgale *et al.*, 2015; Liu *et al.*, 2016).

Such a setup simplifies reconstruction of 3D points $\mathbf{X}$ significantly. At each time instant t, we capture images with both cameras. Since we know how these cameras are oriented with respect to each other, estimation of 3D points $\mathbf{X}$ simplifies to correspondence matching. It should be noted that the calibrated stereo rig makes correspondence estimation problem significantly more robust since it enables so called *circular matching*. All these properties can efficiently be used also for visual odometry, in which we want to estimate relative pose between frames captured at time t and $t+1$.

The geometry of three cameras is described by the *trifocal tensor*. However, using it directly for real-time systems is somewhat problematic since the resulting models require inverting matrices that grow linearly with the number of matched features (Kitt *et al.*, 2010). Instead, we can project image correspondences $\mathbf{x}_i^{(t)} \leftrightarrow \mathbf{x}_i'^{(t)}$ matched in frame t into 3D points $\mathbf{X}_i$ and then simply minimize the the reprojection error in both views of the current frame $t+1$

$$\text{minimize}_{\mathbf{R},\mathbf{t}} \quad \sum_{i=1}^{N} \|\mathbf{x}_i^{(t+1)} - \pi(\mathbf{X}_i; \mathbf{R}, \mathbf{t})\|_\tau + \sum_{i=1}^{N} \|\mathbf{x}_i'^{(t+1)} - \pi'(\mathbf{X}_i; \mathbf{R}, \mathbf{t})\|_\tau \tag{2.44}$$

where $\|\|_\tau$ is a suitable error function and $\pi(\mathbf{X}_i; \mathbf{R}, \mathbf{t})$ is projection of 3D points $\mathbf{X}_i$ on image plane of the first camera. Similarly, $\pi'(\mathbf{X}_i; \mathbf{R}, \mathbf{t})$ denotes projection on image plane of the second camera. In practice, this cost function is optimized with the Gauss-Newton method, typically wrapped into RANSAC to improve robustness (Geiger *et al.*, 2011).



(a) (b)

(c) (d) (e)

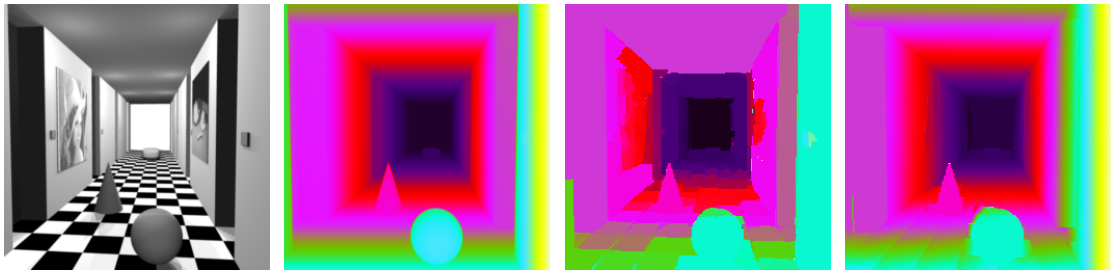
Figure 2.27: Historical approaches to dense disparity estimation demonstrate importance of structural constraints: (a) input rectified stereo image pair, (b) ground-truth, (c) each scan-line solved independently, (d) tree-structured model, (e) loopy graph.

2.3.4 Dense Depth Estimation

Throughout this thesis, we are concerned with *incremental* localization and mapping. That means, the captured images are sequentially ordered which simplifies matching since we can assume sufficient overlap between the neighboring frames. On the other hand, we need to process the data efficiently, typically, at (near) real-time rates. The two basic setups are monocular camera and a calibrated stereo rig. Leaving the scale ambiguity of a monocular setup aside and focusing solely on the matching problem, a synchronized calibrated stereo rig still offers a few advantages:

- The entire scene is imaged as *rigid*, including dynamically moving objects.
- The transformation between the cameras forming a calibrated rig is known.

Using synchronized cameras allows us to capture both images at the same time. Thus, the entire scene, even if it contains dynamically moving objects, is imaged as *rigid*. This is very important since correspondence matching reduces to 1D search along the epipolar lines for all image points, no matter whether they correspond to dynamically moving objects or not. For monocular camera, this is not the case since the two frames we consider are



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

Figure 2.28: Limitations of models with 1st order prior in natural scenes: (a) input image, (b) ground-truth, (c) 1st-order and (d) 2nd-order prior. Although there is a significant qualitative improvement, such optimization is rather slow (Woodford *et al.*, 2009). Thus, modern methods typically encode 2nd order prior in more efficient way (Geiger *et al.*, 2010; Sinha *et al.*, 2014; Yamaguchi *et al.*, 2014).

captured at different times instants. Hence, if we do not know which parts of the scene are moving, we cannot use the camera matrices to constrain the correspondence search and we need to consider a 2D subregion instead since such camera matrices are estimated with respect to the static background and do not model dynamically moving objects. This makes the matching step both slower and less robust and is typically addressed by joint motion segmentation and scene flow estimation. Recently, Engel *et al.* (2015) showed the advantages of combination of static and dynamic baselines; while static stereo baseline effectively removes scale as a free parameter, the temporal stereo allows depth estimation from baselines beyond the narrow baseline of a fixed stereo rig.

Image rectification. Although the epipolar constraint significantly reduces the search space, we also need to consider the computational efficiency of a practical implementation. If we were about to use un-rectified images, we would waste a lot of computational resources by evaluating line equations to iterate over the corresponding pixels. Instead, we rectify the images during the preprocessing step, *i.e.* we map the epipole to infinity. This step makes the epipolar lines perfectly parallel and we simply keep evaluating the search space along a single axis (typically the x coordinate). It should be noted that mapping of the epipole to infinity is significantly harder if the epipole lies within the image (produces very large images), however, it can be overcome with log-polar rectification (Pollefeys *et al.*, 1999).

Dense correspondence matching. Even with constrained search space, dense correspondence matching represents one of the most challenging problems in computer vision. The simplest approach is to use local evidence (small pixel neighbourhood) and "winner-takes-all" strategy. It is very fast and fully-parallel, however, the results are typically very noisy.

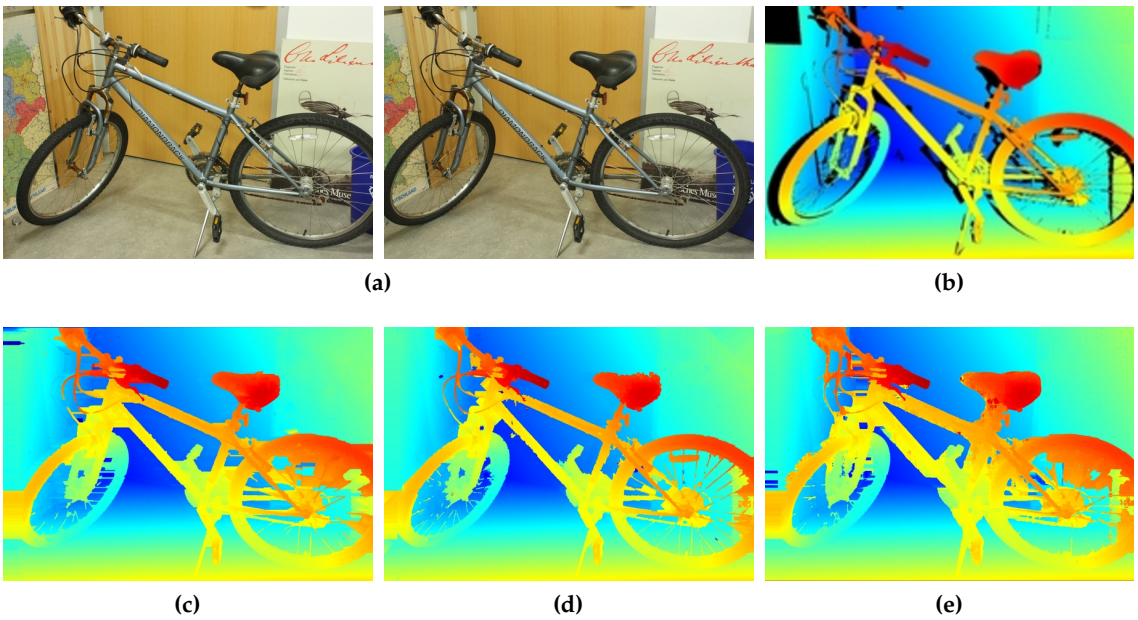
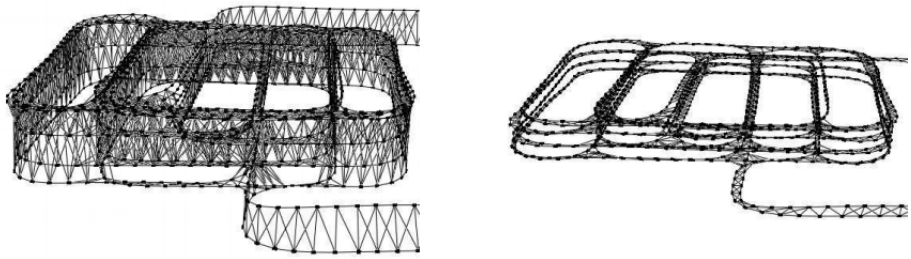


Figure 2.29: Modern (and computationally efficient) approaches to dense disparity estimation: (a) input rectified stereo image pair, (b) ground-truth, (c) ELAS (Geiger *et al.*, 2010), (d) semi-global matching (Hirschmüller, 2005), (e) local plane sweeps (Sinha *et al.*, 2014).

Hence, the more advanced methods attempt to use some prior knowledge and efficient optimization. Probabilistic graphical models and global discrete optimization methods represent a very elegant approach to dense depth estimation, however, they are typically slow since the label space consists of hundreds of labels and it is difficult to make them run in parallel if inference relies on graph-cuts (Boykov *et al.*, 2001). Moreover, most methods typically enforce only a 1st order label consistency (Fig. 2.28), which introduces a strong fronto-parallel bias (Woodford *et al.*, 2009). Semi-global methods (Hirschmüller, 2005) overcome the computational complexity bottleneck by splitting the original problem into a set of subproblems. However, in contrast to dual decomposition formulations, they rely on block-coordinate descent without explicitly enforced consensus through the dual variables. Despite this simplification, they achieve good accuracy and are attractive due to relatively fast run-time and possibility to run in parallel. Local methods with slanted-plane prior have recently become popular since they offer attractive real-time rates and slanted-plane prior turns out to be a very good approximation for many real-world scenes (Geiger *et al.*, 2010; Sinha *et al.*, 2014).



(a) (b)

Figure 2.30: (a) All measurements come with some uncertainty. As we keep integrating camera poses over time, we also integrate this uncertainty. Hence all (visual) odometry methods are prone to drift, which becomes apparent if we visit some area multiple times. (b) The goal of bundle adjustment is to produce a globally consistent joint estimate over the scene structure and camera poses (figure adopted from (Kuemmerle *et al.*, 2011)).

2.3.5 Bundle Adjustment / SLAM

Let us assume a camera freely moving in an environment and recording a sequence of images $\{I_1, I_2, \ldots, I_N\}$. Our goal is to estimate corresponding camera poses $\{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_N\}$ and 3D structure represented by discrete points $\{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_j\}$. Even if we assume that the data association problem has been solved, the problem is that all measurements are *uncertain* (*cf.* Fig. 2.30). Bundle adjustment attempts to produce a solution, which would be jointly optimal with respect to both 3D structure and camera poses by minimizing the reprojection error between point prediction and its measurement (Triggs *et al.*, 1999).

In general, it can be represented as a Bayesian network (Fig. 2.31 (a)). In this case, variables $\mathcal{T}_i$ represent all historic positions of a moving camera and $\mathbf{X}_j$ are stationary 3D points, linked by image observations $\mathbf{x}_j$. However as the camera moves throughout the environment, the number of parameters represented by this network continuously keeps constantly growing at each time step. This represents a major limitation for real-time SLAM since the computational cost is unbounded. In robotics, we need a constant-time inference algorithm, hence we need to restrict the number of variables represented within a graph.

Typical solutions to this problem are filtering and keyframe optimization. Filtering methods marginalise out all poses other than the current one after every frame and summarise the information gained over time with a probability distribution (Fig. 2.31 (c)). This means that the resulting graph is relatively compact, however, it quickly becomes fully inter-connected since every elimination of a past pose variable causes fill-in with new links between every pair of feature variables to which it was joined. The computational cost of propagating joint distributions scales poorly with the number of variables involved, and

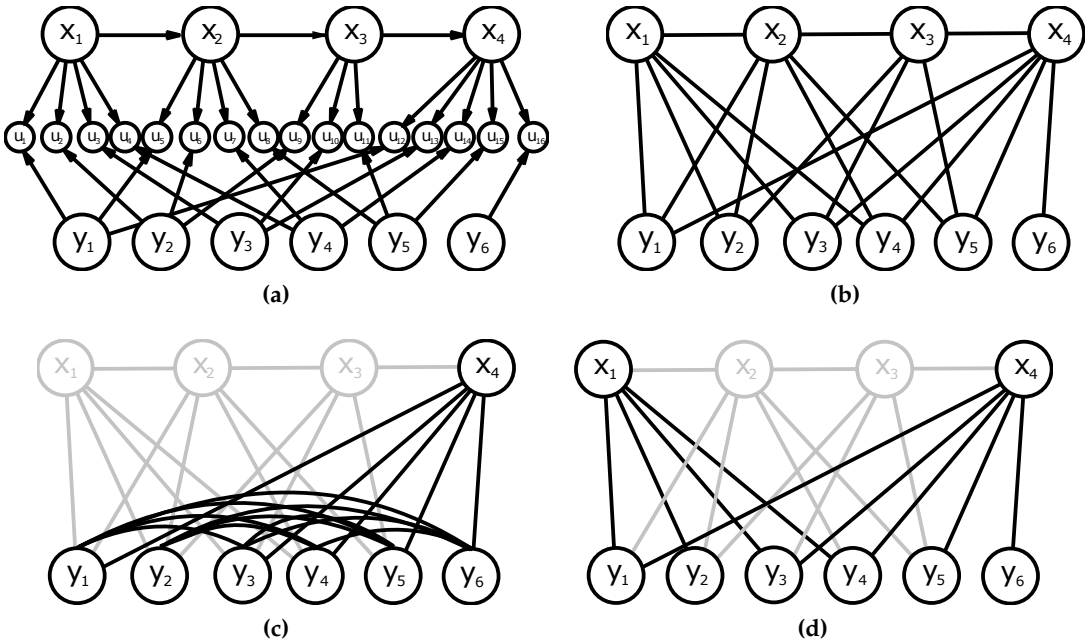


Figure 2.31: (a) Bayesian network for SLAM/SFM. (b) SLAM/SFM as MRF without representing the measurements explicitly. (c) and (d) visualise how inference progressed in a filter and with keyframe-based optimisation (Strasdat *et al.*, 2010).

this is the main drawback of filtering. The other option is to solve the graph from scratch every time, but to sparsify it by removing all but a small subset of heuristically chosen keyframes (Fig. 2.31 (d)). Compared to filtering, such graph would have a larger number of elements, however they will remain sparsely inter-connected which is important for efficient inference. These algorithms are widely available in standard software packages such as iSAM2 (Kaess *et al.*, 2012), g2o (Kuemmerle *et al.*, 2011) or Ceres Solver (Agarwal *et al.*, 2010). More details about SLAM back-ends can be found in (Triggs *et al.*, 1999; Strasdat, 2012; Hartley and Zisserman, 2003; Zach, 2014; Kaess *et al.*, 2012).

2.4 Computer vision tools

We have discussed probabilistic graphical models, inference, learning and geometry. All these tools use feature detectors, descriptors and many other basic computer vision concepts that we discuss in this section.

Local Feature Detectors

As we have seen in Section §2.3, detection of interest points or regions has become essential for many applications in computer vision, and in particular for 3D reconstruction and visual navigation. The requirement for detectors of regions covariant with a class of transformations is that their shape and appearance are not fixed but automatically adapt, based on the underlying 3D surface. Intuitively, we wish to detect regions that correspond to the same 3D patch in different images; *i.e.* these regions are related by a geometric and/or photometric transformation induced by the viewpoint change. Regions detected after the viewpoint change should be the same, modulo noise, as the transformed versions of the regions detected in the original image-image transformation.

Harris and Hessian detectors. We start with description of two related methods detecting interest regions covariant with scale change or affine transformation. These methods first localize features in a spatial domain with the Harris or Hessian detector and apply the scale-selection step based on the Laplacian. Finally, they can be combined with iterative estimate of an affine shape.

The Harris "corner" detector uses the autocorrelation matrix describing local image structures. The eigenvalues of this matrix represent two principal signal changes in a neighbourhood of the point. This property enables an extraction of (corner-like) points, for which both curvatures are significant; that is the signal change is significant in orthogonal directions. Such points are stable in arbitrary lighting conditions and are representative of an image (Fig. 2.32).

Similar approach uses the Hessian detector. The second derivatives, which are used in the Hessian matrix give strong responses on blobs and ridges. Detected regions are similar to those detected by Laplacian operator but determinant of the Hessian matrix penalizes very long structures for which the second derivative in one particular orientation is very small. A blob structure is detected as a local maximum of the determinant (Tuytelaars and Mikolajczyk, 2008).

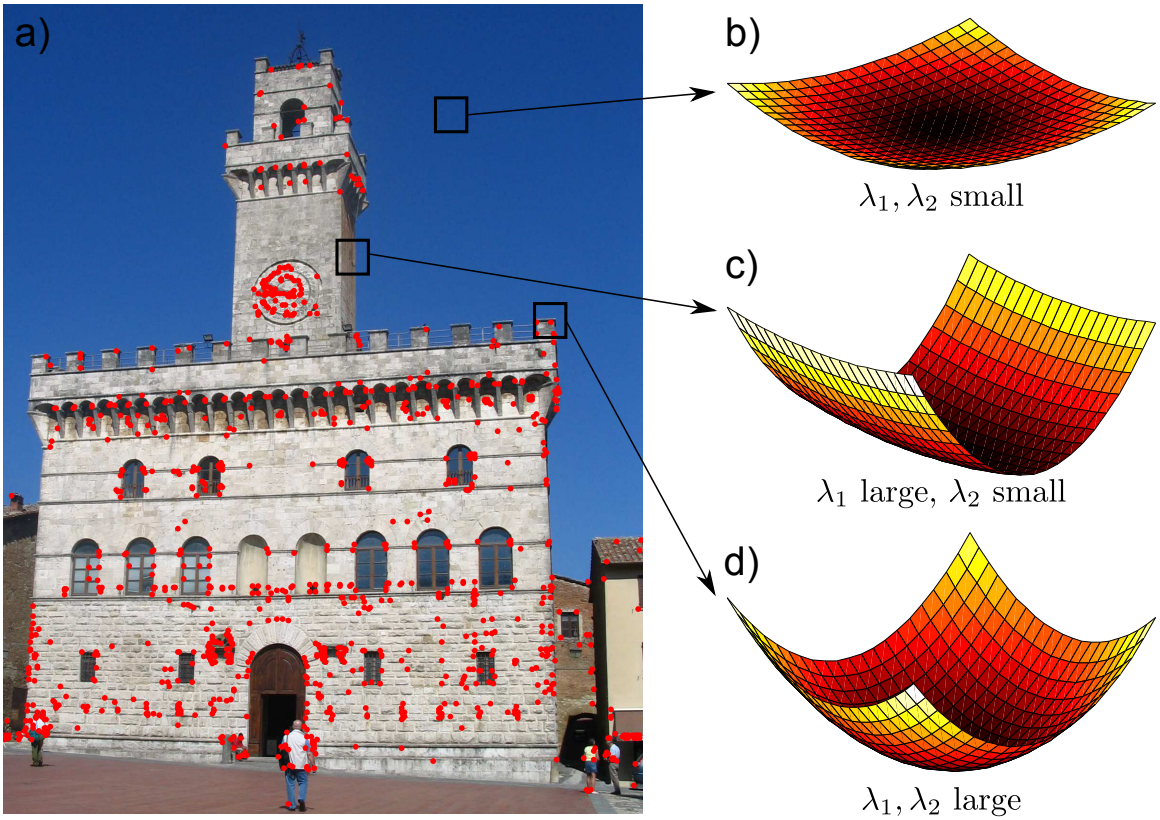


Figure 2.32: Harris corner detector. A local feature is detected if and only if both eigenvalues are large (adopted from (Prince, 2012)).

Efficient approximations. In many situations (*e.g.* controlled lighting, visual navigation, ...), efficiency is more important than the robustness since the higher the frame-rate, the more similar the input frames are. The Difference of Gaussians (DoG) detector (used also in SIFT) (Lowe, 2004) constructs an image pyramid with Gaussian filters at different scales which can be seen as a discrete approximation to the Laplacian. An efficient approximation of the Hessian detector is proposed by SURF (Fast-Hessian) (Bay *et al.*, 2008), which approximates Hessian matrix and gradients by a set of box-type filters and integral images.

The most efficient detectors are based on simple intensity comparisons. The FAST detector is based on an efficient segment test algorithm, which compares pixels on a ring centered at a feature point (Fig. 2.33). The detector is actually a decision tree which classifies the pixel as a feature or non-feature. ORB (Rublee *et al.*, 2011) extends FAST by efficiently computed orientations based on the intensity centroid moment. Similarly, BRISK (Leutenegger *et al.*, 2011) extends FAST by searching for maxima in a 3D scale-space.

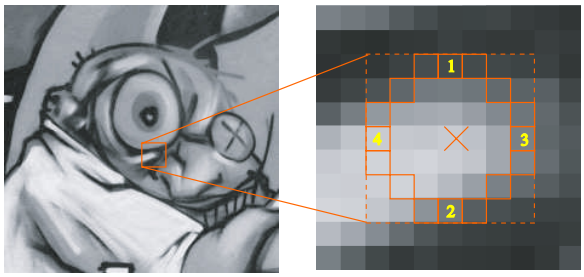


Figure 2.33: FAST detector efficiently compares pixels on a ring centered at a feature point.

Region detectors. Corner-like regions are not always stable enough; typical examples include tracking of object that exhibits lack of texture and/or large motion blur. In such cases, region detectors offer a more robust alternative.

MSER detector (Matas *et al.*, 2002) uses an efficient watershed segmentation algorithm to extract the *extremal* regions by testing stability of connected components obtained by all possible image thresholds. The enumeration of the set of extremal regions is very efficient, almost linear in the number of image pixels. The set of detected regions is unaffected by a monotonic change of image intensities or geometric transformations. Hence detected regions are geometrically and photometrically affine covariant.

Local Features Descriptors

In order to perform various tasks such as matching or recognition, we need to associate pixels with descriptors summarizing their local neighbourhoods. A basic feature can be each pixel's intensity or colour channel values, either in RGB (red, green and blue) or any non-linear colour space such as HSV/HSL or Lab/Luv. Colour features capture only a very local information. It has been reported in the literature that edges are important for visual perception of mammals. Edges correspond to sudden changes in the input signal, hence we convolve the image with various (often derivative) filters and aggregate these responses into a vector. Since we need to describe also variations in scale, rotation, *etc.* it is rather desired to use filter banks consisting of multiple filters (Fig. 2.34 (a)). Examples include the Gabor filter bank or Leung-Malik filter bank (Varma and Zisserman, 2003).

SIFT-like descriptors. One of the most influential papers in computer vision is the seminal work on SIFT (Lowe, 2004). The SIFT feature descriptor is based on the gradient distribution in the detected region. The descriptor is a 3D histogram of gradient locations and orientations, where location is quantized into 4×4 grid and angle is quantized into 8 orientations with soft-assignment, resulting in a 128 dimensional real-valued vector

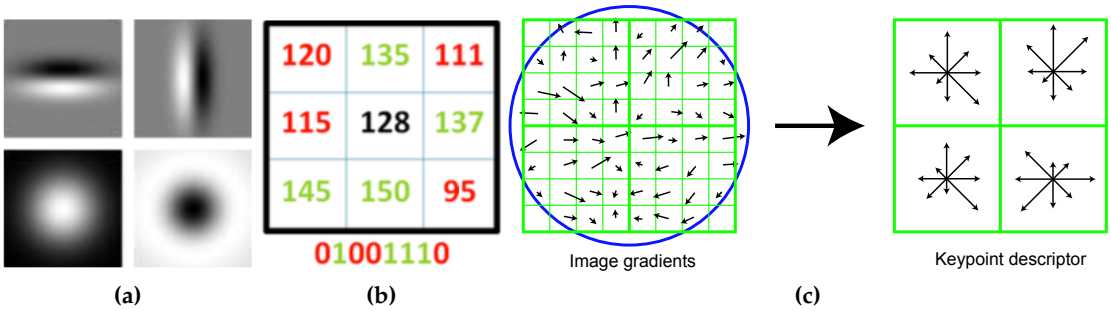


Figure 2.34: Derivative filter bank (a), local binary pattern (b), SIFT (c).

which makes the descriptor robust to small geometric distortions and small errors in the measurement region (Fig. 2.34 (c)). The key step for rotation invariance is estimation of a dominant orientation of an image patch, based on histogram of gradients. The idea of using histograms of gradients has been further explored in numerous variants, such as GLOH (Mikolajczyk and Schmid, 2005), HOG (Dalal and Triggs, 2005), C-SIFT (Abdel-Hakim and Farag, 2006) or PCA-SIFT (Ke and Sukthankar, 2004).

LBP descriptors. Although the SIFT descriptor has stood the test of time and has been widely used in various applications including panorama stitching, object recognition, image retrieval or visual navigation, the high dimensional descriptor suffers from a computational complexity, which makes it unsuitable for time-constrained applications such as SLAM, object tracking, real-time recognition (Miksik and Mikolajczyk, 2012).

A different approach is to use binary descriptors (Calonder *et al.*, 2010; Leutenegger *et al.*, 2011; Rublee *et al.*, 2011). The main advantage of local binary patterns (LBP) is computational efficiency. In contrast to other descriptors, an expensive computing of gradient distributions is replaced by a set of simple binary tests (pairwise intensity comparisons) in a fixed sampling pattern resulting in a binary string encoding relative order of discontinuities (Fig. 2.34 (b)). Another advantage of binary descriptors is that the Hamming distance can be implemented with `XOR` and `POPCNT` which is efficiently implemented on modern CPUs (SSE instructions).

Rotation invariance. One of the main issues causing major errors with rotation invariant descriptors is estimation of dominant orientation of a local patch. Descriptors such as SIFT, SURF (Lowe, 2004; Bay *et al.*, 2008), *etc.* are robust since they rely on local histograms, however, if the dominant orientation is incorrect, the whole descriptor itself becomes useless. In fact, in many applications in which domain knowledge can be utilized (visual odometry

for self-driving cars, ...), it is often better to use descriptors without rotation invariance such as upright-SURF. This issue is addressed by MROGH (Fan *et al.*, 2012) and LIOP (Wang *et al.*, 2011) descriptors which overcome the need for orientation estimation by pooling the intensity orders that are implicitly invariant to rotation or monotonic intensity change.

Feature Encoders

It has been shown that a better performance can be achieved if raw features are replaced by their encoded counterparts. Encoders can be decomposed into two steps: 1) the *embedding step* which maps extracted features into a high-dimensional space and 2) *aggregating step* that produces a single vector from a set of mapped vectors, typically using sum/max pooling. Since we are mainly interested in per-pixel labelling, we omit the aggregation step and focus purely on embedding.

The bag-of-visual-words (BOW) (Sivic and Zisserman, 2003) encoding trains a visual codebook/dictionary from training set by (approximate nearest neighbour) k-means clustering (Bishop, 2006) and maps an image descriptor into a D-dimensional vector having a single element equal to one and others zero. The position of the non-zero element is determined by the nearest neighbour assignment rule. Other variants include multiple assignments, where several components are set to one or soft-assignment which gives different weights to a few components based on their distance to the centroids.

The BOW encodes only 0-th order statistics; the Fisher Vectors (FV) (Perronnin *et al.*, 2006, 2010) extend the BOW by encoding higher order statistics (first and second order). The FVs are trained as a Gaussian Mixture Model (GMM) consisting of K Gaussians using Maximum Likelihood estimation (Bishop, 2006). The FVs give the direction, in parameter space, into which the learned distribution should be modified to better fit the observed data. The resulting size of the FV is $2DK$ (for gradients w.r.t. both, mean and variance). The vector of locally aggregated descriptors (VLAD) (Jégou *et al.*, 2012) is a non-probabilistic version of the FV as it is trained by k-means; *i.e.* the weights are uniform and covariance matrices are isotropic. Other encoders include Locality-constrained linear encoding (Wang *et al.*, 2010b) or triangulation embedding (Jégou and Zisserman, 2014).

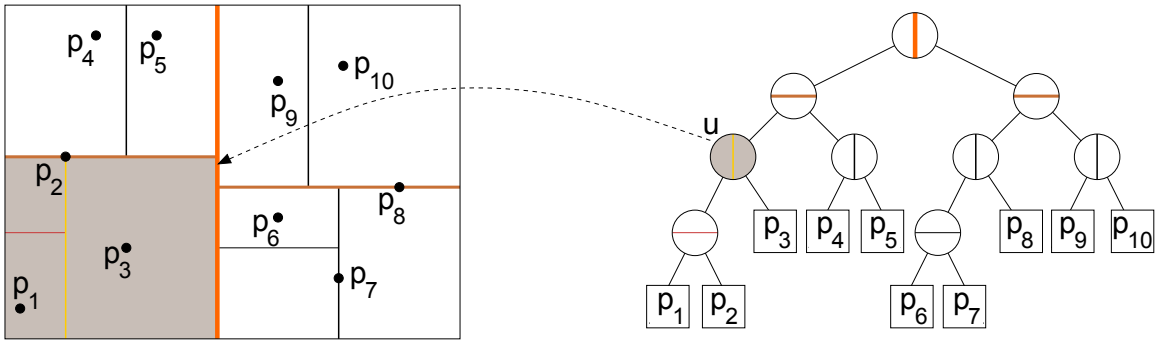


Figure 2.35: KD tree.

Efficient Data Structures

Much research is focused on improving the efficiency of feature matching via nearest neighbour (NN) search. Widely used algorithms in computer vision applications are kd-trees (Arya and Mount, 1993; Lowe, 2004; Muja and Lowe, 2009) and hashing methods (Wang *et al.*, 2010a). Hashing is a fast NN search approach that relies on projection functions that map similar data points into the same buckets that can be efficiently accessed in Hamming space. The kd-tree belongs to a category of a geometric data structures, which is based on iterative partitioning of individual dimensions. Its issues with high dimensional nearest neighbor searching may be overcome by an ϵ-*approximate* nearest neighbor (ϵ-ANN) search (Arya *et al.*, 1998), where search is terminated if a certain condition is satisfied *e.g.* maximum number of leaves visited or a termination parameter which guarantees that the distance to ANN found so far is smaller than distance to the true NN multiplied by a constant i.e. $(\epsilon + 1)d_{NN} \geq d_{ANN}$.

3 Incremental Dense Semantic Stereo Fusion for Large-Scale Semantic Scene Reconstruction

Our abilities in scene understanding, which allow us to perceive the 3D structure of our surroundings and intuitively recognise the objects we see, are things that we largely take for granted. But for robots, the task of understanding large-scale scenes remains extremely challenging. Recently, scene understanding approaches based on 3D reconstruction and semantic segmentation have become popular, but existing methods either do not scale, fail outdoors, provide only sparse reconstructions or are rather slow. In this chapter, we build on a recent hash-based technique for large-scale fusion and an efficient mean-field inference algorithm for densely-connected CRFs to present what to our knowledge is the first system that can perform dense, large-scale, outdoor semantic reconstruction of a scene in (near) real time. We also present a 'semantic fusion' approach that allows us to handle dynamic objects more effectively than previous approaches. We demonstrate the effectiveness of our approach on the KITTI dataset, and provide qualitative and quantitative results showing high-quality dense reconstruction and labelling of a number of scenes.

3.1 Introduction

As we navigate the world, for example when driving a car from our home to the workplace, we constantly perceive the 3D structure of the environment around us and recognise objects within it. Such capabilities help us in our everyday lives and allow us free and accurate movement even in unfamiliar places.

Building a system that can automatically perform incremental real-time dense large-scale reconstruction and semantic segmentation, as illustrated in Fig. 3.1, is a crucial prerequisite for a variety of applications, including robot navigation (Dahlkamp *et al.*, 2006; Urmson

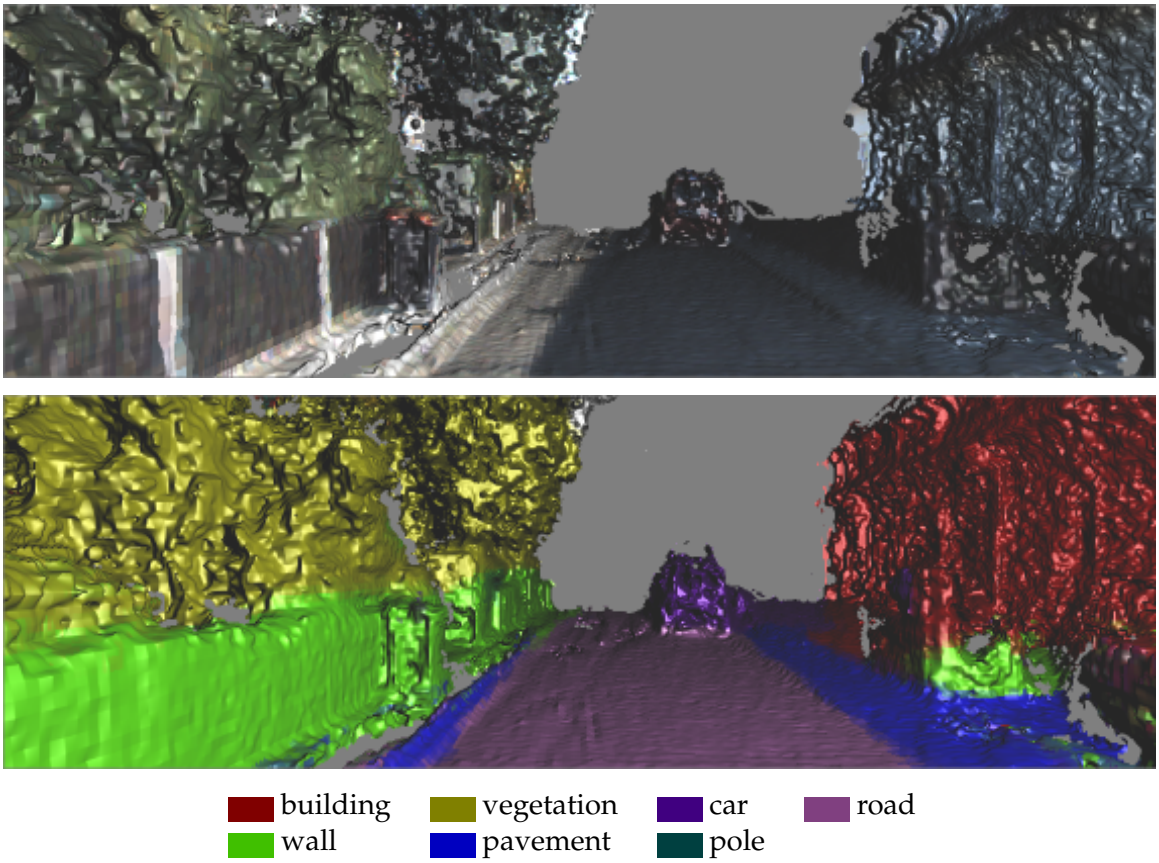


Figure 3.1: Incremental reconstruction (top) and semantic segmentation (bottom) from our system, as seen from a moving platform on-the-fly (*i.e.* not a final mesh).

et al., 2008), semantic mapping (Sengupta *et al.*, 2012, 2013), wearable and/or assistive technology (Google, 2014; Hicks *et al.*, 2013), and change detection (Taneja *et al.*, 2013). However, despite the large body of literature motivated by such applications (Sengupta *et al.*, 2012; Häne *et al.*, 2013; Hermans *et al.*, 2014; Koppula *et al.*, 2011; Kundu *et al.*, 2014; Sengupta *et al.*, 2013; Valentin *et al.*, 2013), most existing approaches suffer from a variety of limitations. Offline reconstruction methods can achieve impressive results at city scale (Agarwal *et al.*, 2011) and beyond, but cannot be used in a real-time setting. Sparse online reconstructions (Davison *et al.*, 2007; Klein and Murray, 2007; Huang *et al.*, 2011; Forster *et al.*, 2014) were historically favoured over dense ones due to their lower computational requirements and the difficulties of acquiring adequate input for dense methods, but sparse maps are not guaranteed to contain objects of interest (*e.g.* traffic lights, signs). Dense reconstructions working on a regular voxel grid (Stühmer *et al.*, 2010; Newcombe *et al.*, 2011b,a) are limited to small volumes due to memory requirements. This has been addressed by approaches that use scalable data structures and stream data between GPU and CPU memory (Chen

et al., 2013; Nießner *et al.*, 2013), but they use Kinect-like cameras that only work indoors (Hermans *et al.*, 2014; Koppula *et al.*, 2011). Approaches working outdoors usually take significant time to run (Häne *et al.*, 2013; Sengupta *et al.*, 2013; Kundu *et al.*, 2014; Floros and Leibe, 2012), do not work incrementally (Valentin *et al.*, 2013) or rely on LIDAR data (Munoz *et al.*, 2009). Existing systems also do not cope well with moving objects. Ideally, we believe a method should

1. be able to incrementally build a dense semantic 3D map of any indoor or outdoor environment at any scale;
2. perform both tasks on-the-fly at real-time rates;
3. be amenable to handling moving objects.

In this chapter, we propose an end-to-end system that can process the data incrementally and perform real-time dense stereo reconstruction and semantic segmentation of unbounded outdoor environments. The system outputs a per-voxel probability distribution instead of a single label (soft predictions are desirable in robotics, as the vision output is usually fed as input into other subsystems). Our system is also able to handle moving objects more effectively than prior approaches by incorporating knowledge of object classes into the reconstruction process. In order to achieve fast test times, we extensively use the computational power of modern GPUs.

Our goal is to *incrementally* build dense large-scale semantic *outdoor* maps. We emphasise the *incremental* nature of our approach, as many methods employ post-processing techniques such as surface densification, texture mapping and tone matting, *etc.* to produce high-quality or visually-plausible meshes. However, in most robotics settings it is the actual output produced on-the-fly that matters (Fig. 3.1). This consideration motivates both our reconstruction pipeline and the system as a whole.

At the core of our system (Fig. 3.2) is a scalable fusion approach (Nießner *et al.*, 2013) that allows the reconstruction of high-quality surfaces in virtually unbounded scenes. It replaces the fixed dense 3D volumetric representation of the standard formulations (Stühmer *et al.*, 2010; Newcombe *et al.*, 2011b,a) with a hash-table-driven counterpart that ignores unoccupied space in the target environment. Furthermore, whilst the standard formulations are limited by the available GPU memory, (Nießner *et al.*, 2013) swaps/streams map data between device and host memories as needed. This is key for scalable dense reconstruction, and to our knowledge has thus far only been used in *indoor* environments.

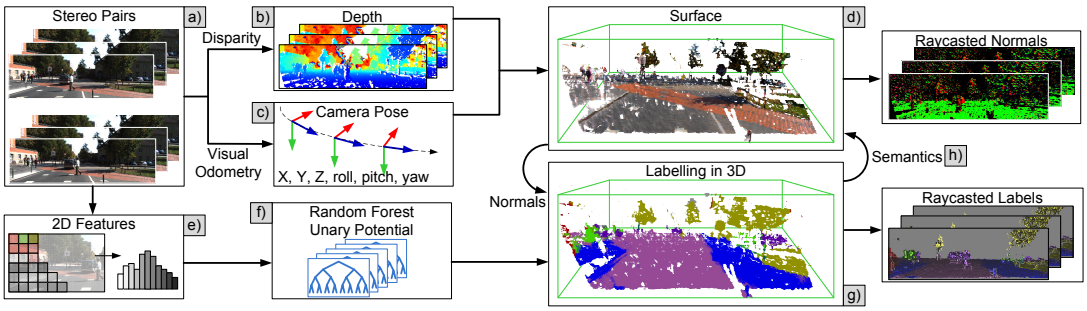


Figure 3.2: Overview of our system: (a) given stereo image pairs, we (b) generate depth and (c) estimate 6 DoF camera pose using visual odometry in parallel. Next, we (d) fuse the depth into a common 3D map. We also (e) extract features, (f) evaluate unary potentials for each voxel and (g) perform inference over a densely-connected pairwise 3D random field to generate a high-quality labelling, which (h) controls fusion weights.

Outdoor scenes present several challenges: 1) Kinect-like cameras are less effective outdoors, whilst LIDARs are often too large for "wearable robotics" or produce overly sparse point-clouds: we thus prefer to rely on stereo, which is suitable for both large robots and wearable glasses/headsets; 2) as a result, the estimated depth (Geiger *et al.*, 2010) is usually more noisy; 3) the depth range is much larger and 4) dynamically moving objects are much more common and the camera itself may move significantly between consecutive frames (*e.g.* if mounted on a car, *etc.*). All of this makes data association for ICP camera pose estimation (as used in (Newcombe *et al.*, 2011a; Nießner *et al.*, 2013)) harder, so we replaced it with a more reliable visual odometry method (Huang *et al.*, 2011).

Our semantic segmentation pipeline extracts 2D features and evaluates unary potentials based on random forest classifier predictions. It transfers these into the 3D volume, where we define a densely-connected CRF. Volumetric CRFs reduce the computational burden, since multiple pixels usually correspond to the same voxel, and enforce temporal consistency, since we label actual 3D surfaces. In order to efficiently infer the approximate maximum posterior marginal (MPM) solution, we propose an online volumetric mean-field inference technique that incrementally refines the marginals of a voxel across iterations, and design a volumetric filter that is suitable for parallel implementation. This allows us to run inference each frame (a single mean-field update takes 2-6ms), so our dynamic energy landscape changes slowly and only a few mean-field update iterations are required at each time step. We use our semantic labels to reinforce the weights in the fusion step, thereby allowing us to handle moving objects more effectively than prior approaches.

Our system is implemented on a GPU, except for visual odometry and disparity estimation, but both are easily parallelisable and can hence be switched to the GPU.

3.2 Related Work

3.2.1 Reconstruction

In the past years, there have been rapid developments in algorithms and systems for indoor and outdoor mapping, at varying scales. Offline Structure-from-Motion (SfM) methods work directly on photos taken of the same scene from different viewpoints (potentially from online repositories and heterogeneous sets of cameras). These systems typically utilize computationally expensive feature matching and bundle adjustment and require minutes, hours or even days to create 3D models. The output can be sparse (Triggs *et al.*, 1999) or dense point clouds (Agarwal *et al.*, 2011), or even detailed and connected surface models (Furukawa *et al.*, 2009).

Algorithms based on Simultaneous Localization and Mapping (SLAM) instead perform real-time mapping using a single monocular camera. Early on, they represented the world by a small number of reconstructed 3D points (Davison *et al.*, 2007; Klein and Murray, 2007). With the advent of *dense* real-time methods (Stühmer *et al.*, 2010; Newcombe *et al.*, 2011b), they have moved to reconstructing detailed surfaces, but their use of a regular voxel grid limited reconstruction to very small environments due to memory requirements.

Recently LSD-SLAM (Engel *et al.*, 2014a) demonstrated large-scale semi-dense point cloud reconstruction using only a monocular mobile phone camera. The method is based on a variant of semi-dense whole image alignment for camera tracking (Engel *et al.*, 2013). Other notable systems focusing on city-scale reconstructions use passive cameras (*e.g.* Taneja *et al.* (2013)). Chen *et al.* (2011) localize landmarks at city-scale on mobile devices. Geiger *et al.* (2011) use stereo camera input to build a dense 3D reconstruction of scene in real-time.

KinectFusion (Newcombe *et al.*, 2011a) directly sensed depth using active sensors such as structured light or time-of-flight systems and thus efficiently replaced the challenges of depth estimation using passive cameras. The ability to compute (noisy) real-time depth maps cheaply at high frame-rate allows to fuse noisy depth measurements of the perceived scene over time to recover high-quality surfaces in real-time, however suffers from the same scalability issue. This drawback has since been removed by scalable approaches that use either a voxel hierarchy (Chen *et al.*, 2013) or voxel block hashing (Nießner *et al.*, 2013) to avoid storing unnecessary data for free space, and stream individual trees in the hierarchy or voxel blocks between the GPU and CPU to allow scaling to unbounded scenes. The hashing approach has the advantage of supporting constant-time lookups of voxel blocks, whereas lookups even in a balanced hierarchy are logarithmic in the number of blocks.

However, these systems rely on active sensors, which limits their use outdoors (*i.e.* in direct sunlight or at extended sensing ranges). The ability of such systems to reconstruct objects such as buildings at long-range is thus limited.

Whilst these systems have demonstrated impressive 3D mapping results, they stop purely at geometry level. Recognition of scene objects is another important area that is not addressed by these approaches.

3.2.2 Semantic Segmentation

A great deal of work has focused on developing efficient and accurate algorithms predicting object labels at the pixel level. Examples include the models of Ladicky *et al.* (2014) or Munoz *et al.* (2010). Recently many others have focused on labeling *voxels* or other 3D representations. Some of them focus on indoor scenes (Koppula *et al.*, 2011; Valentin *et al.*, 2013; Hermans *et al.*, 2014), and others on outdoor scenes (Xiong *et al.*, 2011; Floros and Leibe, 2012; Sengupta *et al.*, 2013). Some other recent works have also tried to jointly optimize for both the tasks of reconstruction and recognition, and hence incorporate the synergy effects between these two high level vision tasks (Häne *et al.*, 2013; Kundu *et al.*, 2014). A summary of the most relevant papers for outdoor large-scale reconstruction is provided in Tab. 3.1.

Hermans *et al.* (2014) use a random forest classifier and a dense 2D CRF, transfer the resulting marginals into 3D and solve a 3D CRF to refine the predictions. Other shortcomings aside (see Tab. 3.1), a CPU implementation requires heuristic scheduling (frame-skipping, *etc.*) to maintain a near-real-time frame rate. Sengupta *et al.* (2013) proposed an offline method, which uses label transfer from 2D to 3D with sampling in a reversed order, which is computationally very expensive. They support streaming from RAM (CPU implementation), but not back again, *i.e.* they always start from scratch. Similarly, Valentin *et al.* (2013) define a CRF over a reconstructed mesh, leading to faster inference. However, their method is not incremental, *i.e.* they need to reconstruct the whole scene first and then label it. Kundu *et al.* (2014) proposed an offline method (based on personal communication) to integrate sparse (monocular) reconstruction with 2D semantic labels into a CRF model to determine the structure and labelling of a scene. Whilst their results are visually appealing, they do appear slightly voxelated when viewed at close range. Other methods (Stühmer *et al.*, 2010; Newcombe *et al.*, 2011b; Floros and Leibe, 2012; Häne *et al.*, 2013) share similar issues, whilst Hu *et al.* (2013) relies on LIDAR data. In contrast to (Floros and Leibe, 2012; Sengupta *et al.*, 2013; Valentin *et al.*, 2013; Kundu *et al.*, 2014), our method provides soft predictions.

Table 3.1: Comparison with some related work: O = outdoor, C = camera only, I = incremental, SDT = sparse data structures, S = host-device streaming, RT = real-time, MV = moving objects

Method	O	C	I	SDT	S	RT	MV
Sengupta *et al.* (2013)	✓	✓			out-of-device only		
Valentin *et al.* (2013)	✓	✓					
Häne *et al.* (2013)	✓	✓				N/A	
Kundu *et al.* (2014)	✓	✓	✓	✓			
Hermans *et al.* (2014)			✓			✓	
Hu *et al.* (2013)	✓		✓	✓		✓	
Ours	✓	✓	✓	✓	✓	✓	✓

3.3 Large-scale outdoor reconstruction

Our system relies on passive stereo cameras, so we need to estimate the depth data. Even with the state of the art models, the estimated depth maps are generally noisy. Hence, we follow the scalable hash-based fusion approach of Nießner *et al.* (2013) in order to generate high-quality surfaces. The key property of this approach is that it is able to generate high-quality surfaces of large-scale indoor scenes by fusing noisy depth data measured over time. However, there are two main drawbacks of this system: i) it is fully dependent on Kinect data, hence it fails to work in an outdoor environment and ii) the method depends on the ICP approach for camera pose estimation which generally fails with noisy depth from stereo cameras.

Our first key contribution is to adapt this scalable hashing to work with outdoor scenes given stereo pairs and also to solve the issues associated with camera tracking. The following subsections describe the three parts of our reconstruction system (depth estimation, camera pose estimation and large-scale fusion) in more detail.

3.3.1 Camera Calibration

First an offline calibration process is performed on the two cameras. This comprises of: 1) *intrinsic calibration* to compute the geometric parameters of each camera lens (focal length, principal point, radial and tangential distortion); 2) *stereo calibration* to compute the geometric relationship between the two cameras, expressed as a rotation matrix and translation vector; 3) *stereo rectification* to correct the camera image planes to ensure they are scanline-aligned to simplify disparity computation. For more details see Hartley and Zisserman (2003).

3.3.2 Depth Estimation

To estimate depth from each stereo pair, we first estimate disparity and then convert it to depth through standard disparity-to-depth mapping. For disparity estimation, we use the approach of Geiger *et al.* (2010), which forms a triangulation on a set of support points that can be robustly matched. This reduces matching ambiguities and allows efficient exploitation of the disparity via constraints on the search space without requiring any global optimization. As a result, the method can be easily parallelised.

3.3.3 Camera Pose Estimation

To estimate camera pose, we use the FOVIS feature-based visual odometry method (Huang *et al.*, 2011). First, an input pair of images is preprocessed using a Gaussian smoothing filter and a three-level image pyramid is built (each level corresponds to one octave in scale space). Then, a set of sparse local features is extracted by using a FAST corner detector (Rosten and Drummond, 2006) with an adaptively-chosen threshold to detect a sufficient number of features. The feature extraction step is usually "biased" using bucketing to ensure that features are uniformly distributed across space and scale.

To constrain the feature matching stage to local search windows, an initial rotation of the image plane is estimated to deal with small motions in 3D. The matching stage associates the extracted features with descriptors and features are matched using a mutual-consistency check. A robust estimate is performed either by finding a maximal clique in the graph or using RANSAC, and the final transformation is estimated on the inliers. Robustness is further increased by using "keyframes", which reduces drift when the camera viewpoint does not change significantly. This can be further improved by using a full SLAM with loop closures, but this is beyond the scope of this chapter.

3.3.4 Large-Scale Fusion

Traditionally, KinectFusion-based approaches have fused depth inside a full, dense, volumetric 3D representation, which severely limits the size of reconstruction that can be handled. However, in real-world scenarios, a large part of this volume only contains free space, which does not need to be densely stored. By focusing the representation on the useful parts of the scene, we can use memory much more efficiently, which in turn enables to reconstruct much larger environments. This insight has acted as a catalyst for the hash-based method of (Nießner *et al.*, 2013) and the octree technique of (Chen *et al.*, 2013).

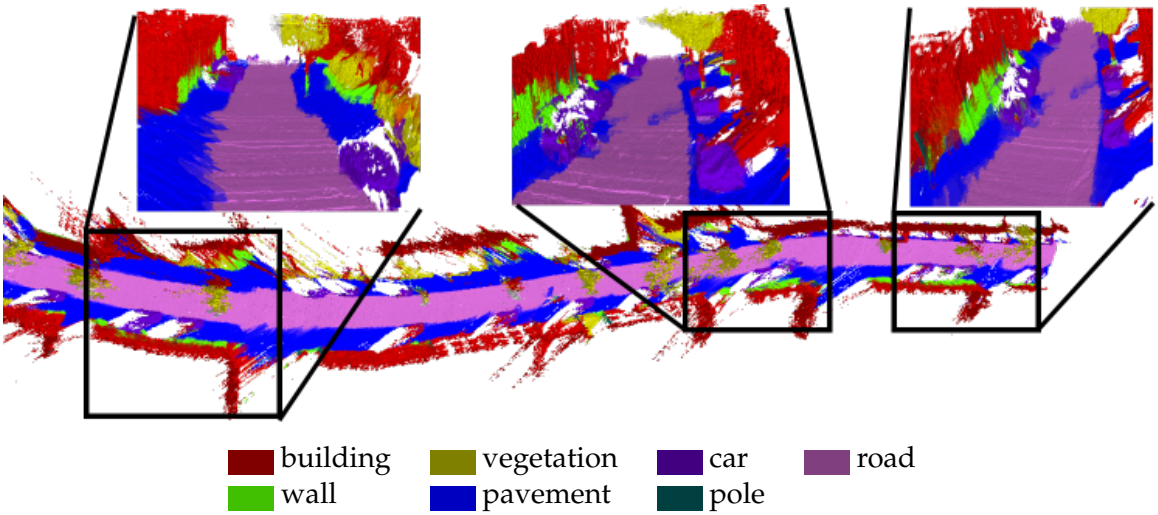


Figure 3.3: Labelled mesh (output of our algorithm) for sequence 95 from the KITTI residential dataset, consisting of 268 stereo pairs. The close-up views show snapshots of the scene at several places along the route.

We adopt the hash-based fusion method (Nießner *et al.*, 2013), which allocates space for only those voxels that fall within a small distance of the perceived surfaces in the scene. This space is organised into small voxel blocks. As with other depth fusion approaches, the dense areas are represented using an approximate truncated signed distance function (TSDF) (Curless and Levoy, 1996). Access to individual voxel blocks is mediated by a hash table. Given a known camera pose (§3.3.3), we use the following fusion pipeline:

Allocation. We ensure that voxel blocks are allocated for each voxel visible in the depth image. This is done by (i) back-projecting all visible voxels to voxel block world coordinates; (ii) looking up each unique voxel block in the hash table to determine whether or not it is currently allocated and (iii) allocating all blocks that are currently unallocated.

Integration. We integrate the current depth estimates and RGB frames into the volumetric data structure, using the standard sliding-average technique of (Curless and Levoy, 1996).

Host-device streaming. Although current GPUs have several GB of device memory, it is generally not enough to store a full large-scale reconstruction. To this end, data is streamed between the device and host. We only keep parts that are in or near the frustum. To implement this approach, we actively swap parts of the map between device and host memory as they move in and out of view. Note that the scale of the reconstructions we can handle is still limited by host RAM in the current implementation. However, it is simple to

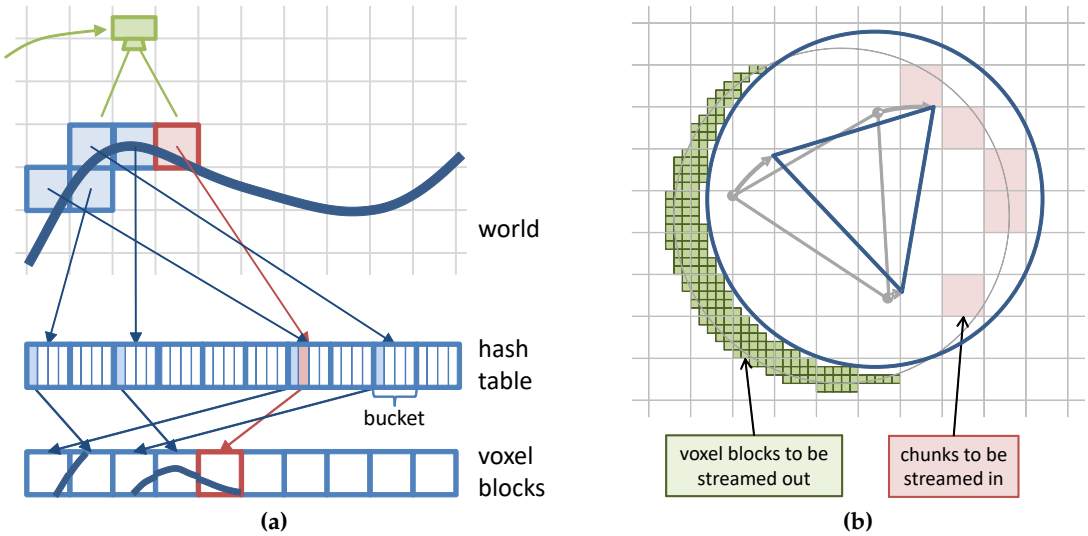


Figure 3.4: (a) Voxel hashing approach maps world coordinates to the hash buckets which store a small array of pointers to regular grid voxel blocks. Each voxel block contains a grid of SDF values. (b) As camera moves throughout the environment, the voxel blocks leaving the camera frustum are streamed out (green) and previously observed blocks (red) are streamed in (Nießner *et al.*, 2013).

use the "swapping in and out" strategy between RAM and disk storage to achieve virtually unbounded reconstructions.

Raycasting. The fused map is rendered from the current camera position in each frame.

3.4 Semantic Fusion

In the standard fusion approach, each voxel i stores TSDF and colour measurements $\hat{T}_i^t$ and $\hat{C}_i^t$ at time t, together with weights $\hat{w}_{T,i}^t$ and $\hat{w}_{C,i}^t$ that capture our confidence in these measurements. These values are updated over time using the corresponding live TDSF and colour measurements T_i^t and C_i^t, and some live weights $w_{T,i}^t$ and $w_{C,i}^t$ that can often be set to 1 to give simple running averages, *e.g.*:

$$\begin{aligned} \hat{w}_{T,i}^t &= \hat{w}_{T,i}^{t-1} + w_{T,i}^t \\ \hat{T}_i^t &= (\hat{w}_{T,i}^{t-1}\hat{T}_i^{t-1} + w_{T,i}^t T_i^t)/(\hat{w}_{T,i}^{t-1} + w_{T,i}^t) \end{aligned} \tag{3.1}$$

This fusion step generally fails when there are moving objects in the scene, since static objects can become corrupted when we fuse in depth data from moving objects. This effect can be reduced by basing the live weights $w_{T,i}^t$ and $w_{C,i}^t$ on object class: by using higher weights for voxels that are labelled with moving object classes (*e.g.* car, pedestrian, *etc.*),

we can speed up the process of fusing new data into our TSDF in places where the scene is more likely to be changing rapidly, which allows us to avoid being left with incorrect surfaces in places that briefly contained moving objects (note that the weights for voxels increase as we fuse in moving object data, and take some time to decrease again after the objects leave the voxels again). We call this adaptation of the original scheme "semantic fusion", and update our measurements using

$$\begin{aligned} \hat{w}_{T,i}^{t} &= \hat{w}_{T,i}^{t-1} + w_{\ell_i^t} \\ \hat{T}_i^t &= (\hat{w}_{T,i}^{t-1}\hat{T}_i^{t-1} + w_{\ell_i^t}T_i^t)/(\hat{w}_{T,i}^{t-1} + w_{\ell_i^t}), \end{aligned} \tag{3.2}$$

where $w_{\ell_i^t}$ is a per-class fixed weight corresponding to the semantic label of voxel i at time t.

This approach temporarily decreases the smoothness of the surface of affected voxels, but it allows us to preserve moving objects in a scene and avoids corruption of static objects. An example showing the way in which our semantic fusion approach is able to handle dynamically-moving objects is shown in Figure 3.7.

3.5 Volumetric CRF and Mean-field inference

3.5.1 Model

We begin by defining a random field over random variables $\mathcal{X} = \{X_1, ..., X_N\}$, conditioned on the 3D surface $\mathbf{D}$. We assume that each discrete random variable X_i is associated with a voxel $\mathcal{V} \in \{1, ..., N\}$ in the 3D reconstruction volume and takes a label l_i from a finite label set $\mathcal{L} = \{l_1, ..., l_L\}$, corresponding to different object classes such as car, building or road. We formulate the problem of assigning object labels to the voxels as one of solving a volumetric, densely-connected, pairwise Conditional Random Field (CRF).

Since our volumetric reconstruction is dynamically changing as new observations are captured, we have to deal with a dynamic energy function that keeps on changing in each iteration. We define this CRF over the voxels in the current view frustum as

$$\begin{aligned} P(\mathbf{x}|\mathbf{D}) &= \frac{1}{Z(\mathbf{D})}\exp(-E(\mathbf{x}|\mathbf{D})) \\ E(\mathbf{x}|\mathbf{D}) &= \sum_{i\in\mathcal{V}} \psi_u(X_i) + \sum_{i<j\in\mathcal{V}} \psi_p(X_i, X_j), \end{aligned} \tag{3.3}$$

in which $E(\mathbf{x}|\mathbf{D})$ is the energy associated with a configuration $\mathbf{x}$, conditioned on the volumetric data $\mathbf{D}$, $Z(\mathbf{D}) = \sum_{\mathbf{X}'} \exp(-E(\mathbf{x}'|\mathbf{D}))$ is the (data-dependent) partition function and $\psi_u(\cdot)$ and $\psi_p(\cdot,\cdot)$ are the unary potential and pairwise potential functions, respectively, both implicitly conditioned on the data $\mathbf{D}$.

Unary potentials. Unary potential terms $\psi_u(\cdot)$ correspond to the cost of voxel i taking an object label $l \in \mathcal{L}$. In order to evaluate the per-voxel unary potentials, we first train per-pixel object class models derived from TextonForest (Shotton *et al.*, 2008) using a set of per-pixel ground truth training images (Sengupta *et al.*, 2013). We use the 17-dimensional filter bank suggested by Shotton *et al.* (2008), and follow Ladicky *et al.* (2014) by adding colour, histogram of oriented gradients (HOG), and pixel location features. At test time, we evaluate unary potentials in the image domain and then project them onto the voxels using the current camera pose and average them over time.

Pairwise potentials. The pairwise potential function $\psi_p(\cdot,\cdot)$ enforces consistency over pairs of random variables and thus generally leads to a smooth output. In our application, we use the weighted Potts model, which takes the form $\psi_{ij}(l,l') = \lambda_{ij}(\mathbf{f}_i,\mathbf{f}_j)[l \neq l']$, where $[.]$ is the Iverson bracket (1 iff the condition in the square bracket is satisfied and 0 otherwise) and $\mathbf{f}_i, \mathbf{f}_j$ are the 3D features extracted from data $\mathbf{D}$ at the ith and j^{th} voxels (respectively).

In the 2D segmentation domain, the cost λ_{ij} of assigning different labels to neighbouring pixels is generally chosen such that it preserves image edges. Inspired by these edge-preserving smoothness costs, we make λ_{ij} a weighted combination of Gaussian kernels (with unit covariance matrix) that depend on appearance and depth features:

$$\lambda_{ij} = \sum_{m=1}^{M} \theta^m \lambda_{ij}^m(\mathbf{f}_i,\mathbf{f}_j) = \theta_{\mathrm{p}}^m e^{-\|\mathbf{p}_i-\mathbf{p}_j\|_2^2} + \theta_{\mathrm{a}}^m e^{-\|\mathbf{a}_i-\mathbf{a}_j\|_2^2} + \theta_{\mathrm{n}}^m e^{-\|\mathbf{n}_i-\mathbf{n}_j\|_2^2} \tag{3.4}$$

Here, $\mathbf{p}_i$, $\mathbf{a}_i$ and $\mathbf{n}_i$ are respectively the 3D world coordinate position, RGB appearance, and surface normal vector of the reconstructed surface at voxel i, and $\theta_{\mathbf{p}}$, $\theta_{\mathbf{a}}$ and $\theta_{\mathbf{n}}$ are parameters obtained by cross-validation. Note that surface normals are calculated using the TSDF values (Newcombe *et al.*, 2011a). In general, we obtain high-quality normals (see Fig. 3.5), which helps in achieving very smooth output.

3.5.2 Efficient Volumetric Filtering-based Mean-Field Inference

One of the most popular approaches for multi-label CRF inference has been graph-cuts based α-expansion (Boykov *et al.*, 2001), which finds the maximum a posteriori (MAP) solution. However, graph-cut leads to slow inference and is not easily parallelisable. Given the form of the energy function defined above, we follow a filter-based variant of the mean-field optimization method, that has been shown to be very efficient for densely-connected CRFs in 2D image segmentation (Krähenbühl and Koltun, 2011). As we have already shown in §2.1.2, in the mean-field framework, we approximate the true distribution P by a family

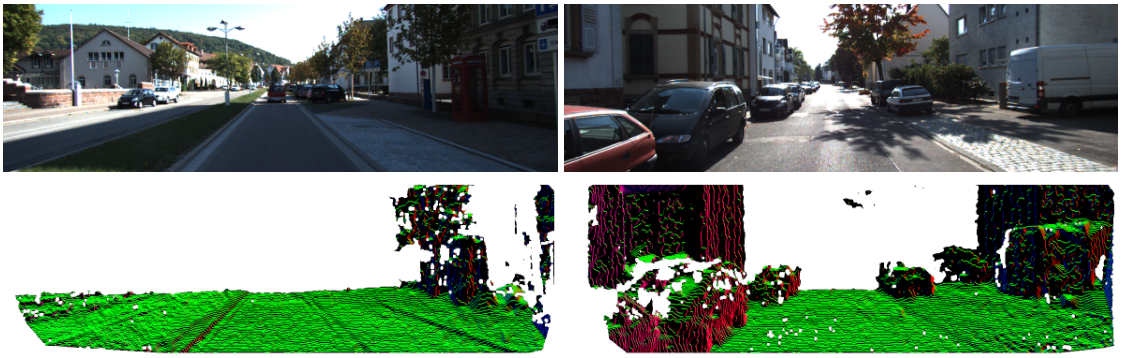


Figure 3.5: An example of the normals we generate from the TSDF surfaces. These provide a lot of information about surface orientation and curvature that we use in pairwise potentials.

of Q distributions that factorize as the product of all components' marginals (components are independent) $Q = \prod_i Q_i(x_i)$. The mean-field inference then attempts to minimize the KL-divergence $D_{\mathrm{KL}}(Q||P)$ between the tractable distribution Q and true distribution P.

Next, we discuss our online volumetric mean-field approach. Although this *online* mean-field approach has previously been applied in 2D (Medrano *et al.*, 2009), we believe this is the first time it has been applied in a 3D setting. The most time-consuming step in the mean-field inference is the pairwise update, whose complexity is $\mathcal{O}(N^2)$. Hence, we use the cross bilateral filtering approach of (Krähenbühl and Koltun, 2011) which reduces this complexity to $\mathcal{O}(N)$. This approach allows to efficiently approximate parallel updates in $\mathcal{O}(MNL)$ time for the Potts model. The algorithm is run for a fixed number of iterations, and the MPM solution extracted by choosing $X_i \in \mathrm{argmax}_l\, Q_i(x_i = l)$ from soft predictions at the final iteration. We use high-dimensional filtering on the 3D volumetric data, where the filtering is a simple extension of the 2D permutohedral lattice-based filtering shown in (Krähenbühl and Koltun, 2011) to 3D.

3.5.3 Online mean-field

Given unlimited computation, one might run multiple update iterations until convergence. However, in our online system, we assume that the next frame's updates to the volume (and thus to the energy function) are not too radical so we can make the assumption that the Q_i distributions can be temporally propagated from one frame to the next, rather than re-initialized (*e.g.* to uniform) at each frame (Valentin *et al.*, 2015). Thus, running even a *single iteration* of mean-field updates per frame effectively allows us to amortize an otherwise expensive inference operation over multiple frames and maintain real-time speeds.

As described above, the output of the classifier responses is used to update the unary

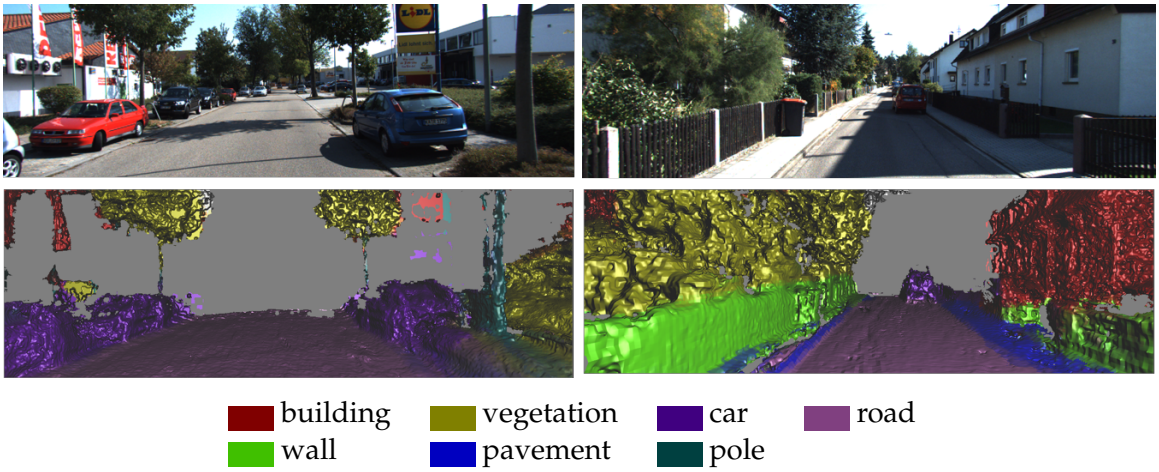


Figure 3.6: Our approach not only reconstructs and labels entire outdoor scenes that include roads, pavements and buildings, but also accurately recovers thin objects such as lamp posts and trees.

potentials, which will, over several frames, impact the final segmentation that results from the online mean-field inference. However, to speed up convergence, rather than simply propagating the Q_i^{t-1}s from the previous frame, we instead provide the next iteration of mean-field updates with a weighted combination of Q_i^{t-1} and the classifier prediction $P_u(X_i = l \mid \mathbf{D})$. We thus use

$$\bar{Q}_i^{t-1}(l) = \gamma Q_i^{t-1}(l) + (1-\gamma) P_u(X_i = l \mid \mathbf{D}) \tag{3.5}$$

in place of Q_i^{t-1}, where γ is a weighting parameter.

3.6 Experiments

We demonstrate the effectiveness of our approach for both 3D semantic segmentation and reconstruction. We evaluate our system on the KITTI dataset (Geiger *et al.*, 2012), which contains a variety of outdoor sequences, including a city, road and campus. All sequences were captured at a resolution of 1241×376 pixels using stereo cameras (with baseline 0.54m) mounted on the roof of a car. The car was also equipped with a Velodyne HDL-64E laser scanner (LIDAR). The KITTI dataset is very challenging since it contains many moving objects such as cars, pedestrians and bikes, and numerous changes in lighting conditions.

For both voxel labelling and 3D reconstruction, we show our results on static and dynamic scenes. This allows us to evaluate how well our approach handles motion. For static scenes, we used the dataset of Sengupta *et al.* (2013), which consists of 45 training and 25 test images labelled with the following classes: road, building, vehicle, pedestrian,

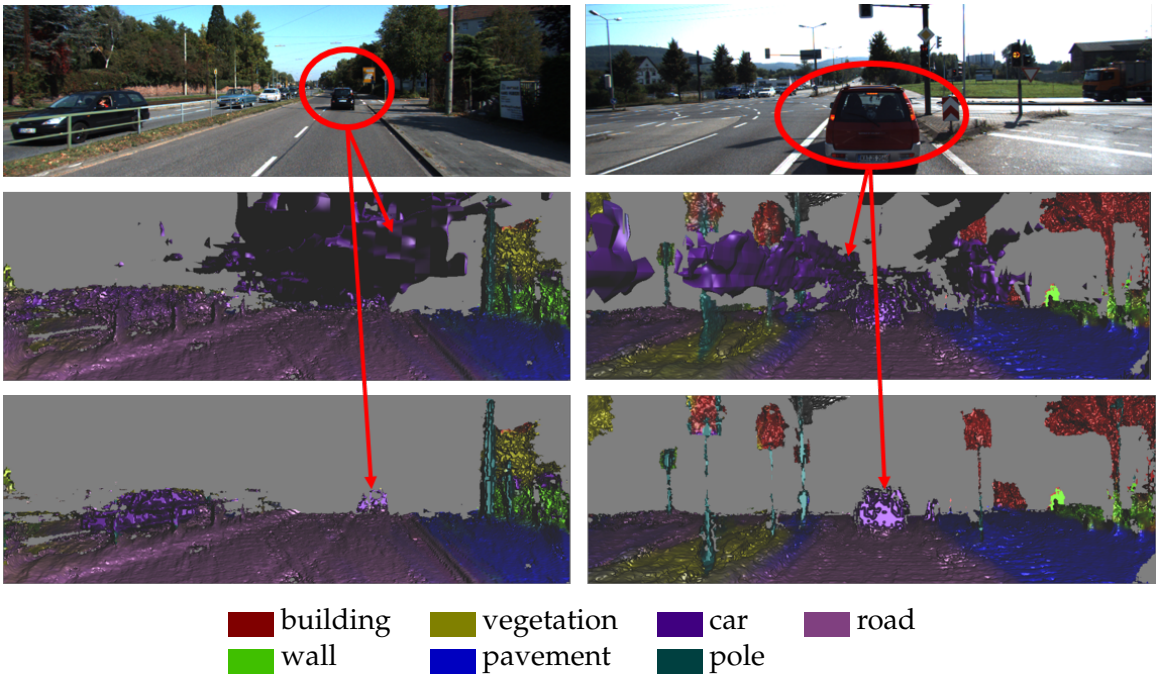


Figure 3.7: Our semantic fusion technique enables us to avoid corrupting a static scene with data from moving objects. First row: input image; second row: reconstructed scene without semantic fusion; third row: reconstructed scene with semantic fusion. Note the way in which semantic fusion helps suppress the trail of spurious voxels that moving objects would normally leave behind.

pavement, tree, sky, signage, post/pole and wall/fence. For dynamic scenes, we manually annotated sequences from the KITTI dataset that contained many moving objects. We compare the timings and accuracy achieved by our voxel-labelling approach against two baselines (Ladicky *et al.*, 2014; Sengupta *et al.*, 2013). To evaluate our 3D reconstruction, we compare with the depth data generated using (Geiger *et al.*, 2010)'s approach, using LIDAR data from the Velodyne scanner as ground truth. To perform qualitative and quantitative evaluation, we back-project the voxel labels and reconstructed surfaces onto the camera's image plane, ignoring those that are farther than 25 metres from the camera.

Figure 3.8: A high-quality mesh recovered from the long (1000 images) sequence 5 of the KITTI odometry dataset, superimposed over the corresponding Google Earth image. This shows the ability of our method to reconstruct and label large scenes.

3.6.1 Qualitative KITTI results.

First, we show some qualitative results for our semantic reconstruction approach. In Fig. 3.6 we highlight the ability of our approach not only to reconstruct and label entire outdoor scenes that include roads, pavements and buildings, but also to accurately recover thin objects such as lamp posts and trees. In Fig. 3.7 we show the advantages of our semantic fusion approach in handling moving objects (in this case, a car). Note in particular that with semantic fusion turned on, the static scene is far less corrupted by moving objects than it would be otherwise. Fig. 3.8 shows a high-quality mesh recovered from a long KITTI sequence (1000 images), superimposed over the corresponding Google Earth image. This shows the ability of our method to reconstruct and label large scenes. In Fig. 3.9 we show a close-up view of a dense semantic 3D model produced using our method, in which the arrows indicate the image locations and their corresponding positions in the 3D model, and colours indicate the object labels. This shows that even though our approach is an incremental one, we are able to achieve smooth surfaces for large-scale outdoor scenes.

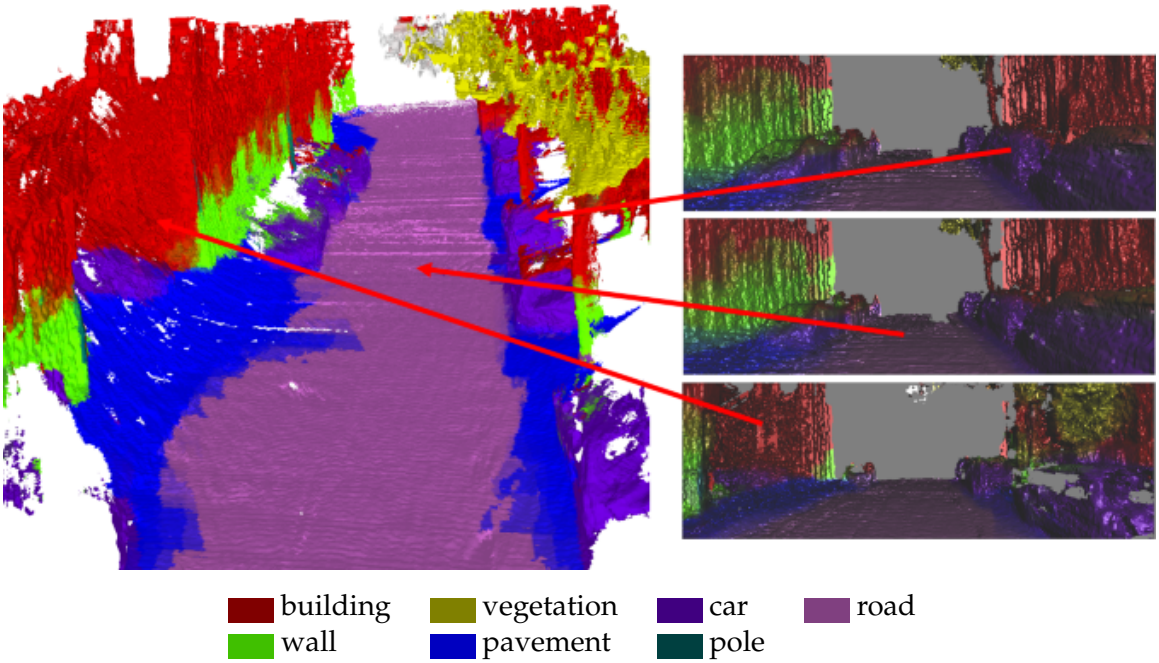


Figure 3.9: A close-up view of a semantic model produced using our method, in which the arrows indicate the image locations and their corresponding positions in the 3D model, and colours indicate the object labels. This shows that even though our approach is an incremental one, we are able to achieve smooth surfaces for outdoor scenes.

3.6.2 Quantitative KITTI Results

Semantic segmentation. Next, we quantitatively evaluate the speed and accuracy of our mean-field-based volumetric labelling approach. Mean-field updates take roughly 20ms. Although the timings change as a function *e.g.* of the number of visible voxels, in all tests we performed, we observed real-time performance. We assess the overall percentage of correctly-labelled voxels (global accuracy) and the intersection/union (I/U) score defined in terms of the true/false positives/negatives for a given class, *i.e.* TP/(TP+FP+FN).

Quantitative results for static scenes are shown in Tab. 3.2 (a). In comparison to the 2D approach of Ladicky *et al.* (2014), we achieve a 0.49% improvement in global accuracy and a 0.84% improvement in I/U score. We also significantly improve upon the 3D approach of Sengupta *et al.* (2013), achieving a 10.8% improvement in global accuracy and a 6.7% improvement in I/U. More importantly, our approach achieves encouraging improvements in global accuracy and I/U for thin objects such as poles.

In Tab. 3.2 (b), we evaluate the accuracy of our labellings on sequences containing many moving cars. We observe that our non-semantic fusion approach reduces accuracy by over 10% in comparison to (Ladicky *et al.*, 2014); however, our semantic fusion approach improves overall accuracy by 1.5%. For cars, we observe an improvement of 2.2% in global accuracy and 5.5% in I/U. Note that our semantic fusion approach significantly improves both the global accuracy and I/U of our method, in both cases by over 10%. The improvements for cars are even more significant, highlighting the importance of using semantic fusion for scenes containing moving objects.

Reconstruction. Next, we quantitatively evaluate the efficiency and accuracy of our reconstruction approach. Camera pose estimation takes roughly 20ms, stereo estimation takes around 40ms (on our 12 core systems) and fusion takes 14ms. In order to evaluate accuracy, we follow the approach of Sengupta *et al.* (2013), who measure the number of pixels whose distance (in terms of depth) from the ground truth (in our case the Velodyne data) after projection to the image plane is less than a fixed threshold.

Quantitative results for depth evaluation are summarised in Fig. 3.10 for both static and dynamic scenes. We observe that for static scenes, our non-semantic fusion approach itself achieves almost 90% and 95% accuracy when the thresholds are 1m and 4m respectively. We therefore achieve an improvement of almost 20% over the initial depth estimated using the stereo output from Geiger *et al.* (2010)'s approach. However, for sequences in which there are many moving objects, non-semantic fusion does not perform that well and leads

Table 3.2: Quantitative results for our segmantic segmentation approach on the KITTI dataset. We compare global accuracy and intersection/union on both (a) static and (b) moving scenes. For static scenes, we compare our approach without semantic fusion [Ours(1)] against the state-of-the-art approaches of Ladicky *et al.* (2014) and Sengupta *et al.* (2013). For moving scenes, we compare our approach with semantic fusion [Ours(2)] against [Ours(1)] and (Ladicky *et al.*, 2014).

(a) Static

	Global Accuracy			Intersection/Union		
Class	Ladicky *et al.* (2014)	Sengupta *et al.* (2013)	Ours(1)	Ladicky *et al.* (2014)	Sengupta *et al.* (2013)	Ours(1)
building	97.0	96.1	**97.2**	86.1	83.8	**88.3**
vegetation	93.4	86.9	**94.1**	82.8	74.3	**83.2**
car	93.9	88.5	**94.1**	78.0	63.5	**79.5**
road	98.3	97.8	**98.7**	94.3	**96.3**	94.7
wall	**48.5**	46.1	47.8	**47.5**	45.2	46.3
pavement	91.3	46.1	**91.8**	73.4	68.4	**73.8**
pole	49.3	38.2	**51.4**	39.5	28.9	**41.7**
Average	81.7	71.4	**82.2**	71.7	65.8	**72.5**

(b) Moving

	Global Accuracy			Intersection/Union		
Class	Ladicky *et al.* (2014)	Ours(1)	Ours(2)	Ladicky *et al.* (2014)	Ours(1)	Ours(2)
building	90.9	89.1	**93.1**	82.1	81.9	**82.7**
vegetation	89.2	66.9	**92.1**	77.6	64.3	**79.0**
car	92.1	78.5	**94.3**	72.0	56.4	**77.5**
road	**98.6**	87.8	97.7	91.3	86.3	**92.1**
wall	46.7	42.1	**48.1**	49.5	42.2	**50.3**
pavement	93.3	84.5	**94.8**	72.4	63.4	**75.8**
pole	46.2	36.7	**47.4**	34.1	24.6	**36.7**
Average	79.6	69.4	**81.1**	68.4	59.9	**70.6**

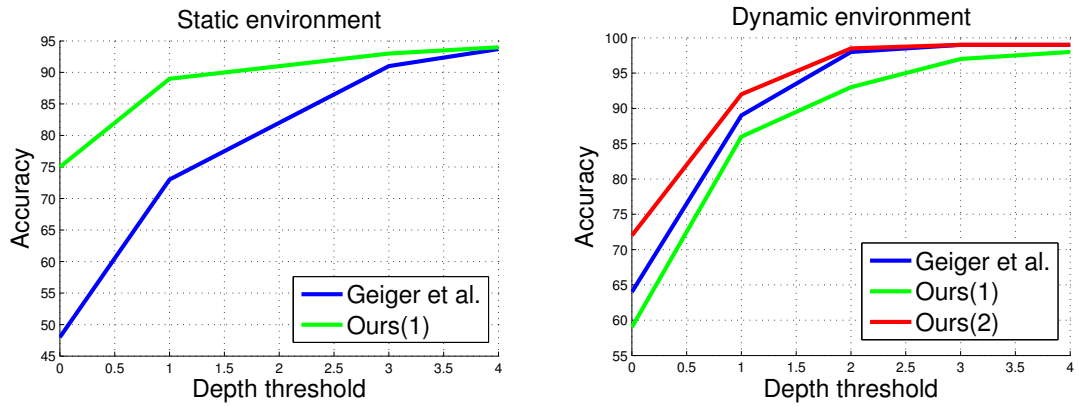


Figure 3.10: Quantitative results for depth evaluation for static (left) and moving (right) scenes.

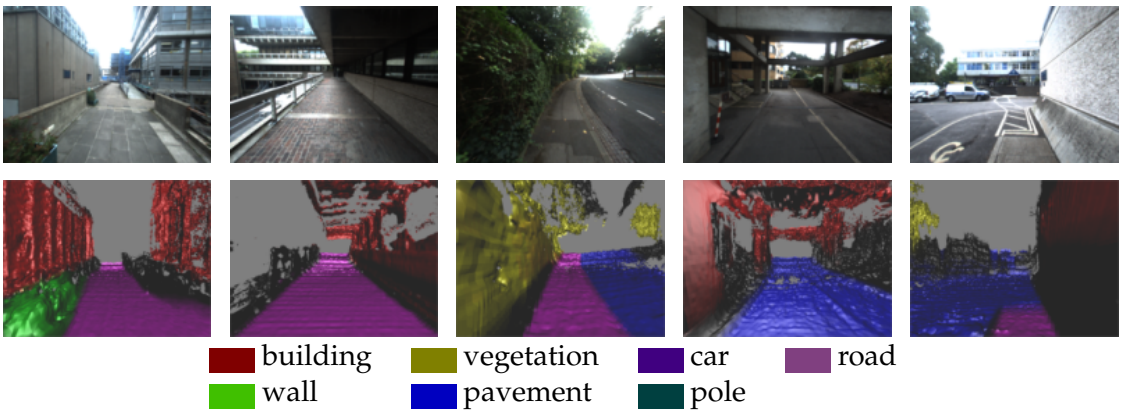


Figure 3.11: Final labelling surfaces for four reconstructed sequences (the last two columns belong to the same sequence).

to a decrease in accuracy of almost 5% compared to Geiger *et al.* 's method. By contrast, our semantic fusion approach achieves an almost 5% improvement in accuracy.

We would like to highlight that the real-time aspect of our semantic reconstruction pipeline does not include the feature evaluation time. However, features can be implemented on GPU to provide real-time performance, as shown in (Prisacariu and Reid, 2009) for a handcrafted variant or as modern convolutional neural networks.

3.6.3 Other Qualitative Results

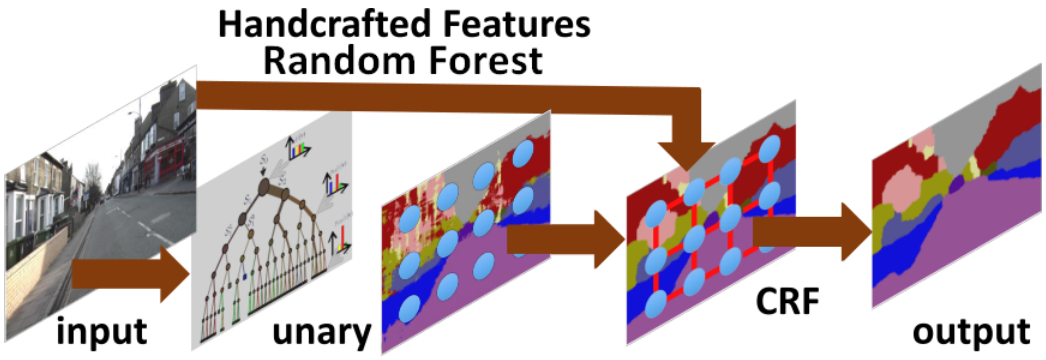
We show additional qualitative results on four new, challenging sequences that we captured using a head-mounted stereo camera. Fig. 3.11 shows the final smooth semantic reconstructions obtained by running our mean-field inference procedure. The images clearly indicate the sharp boundaries between different conflicting semantic classes. For example, observe the extremely accurate boundary between the pavement and the road in the sequence in the third column. More results are provided at `http://www.miksik.co.uk`

3.7 Conclusion

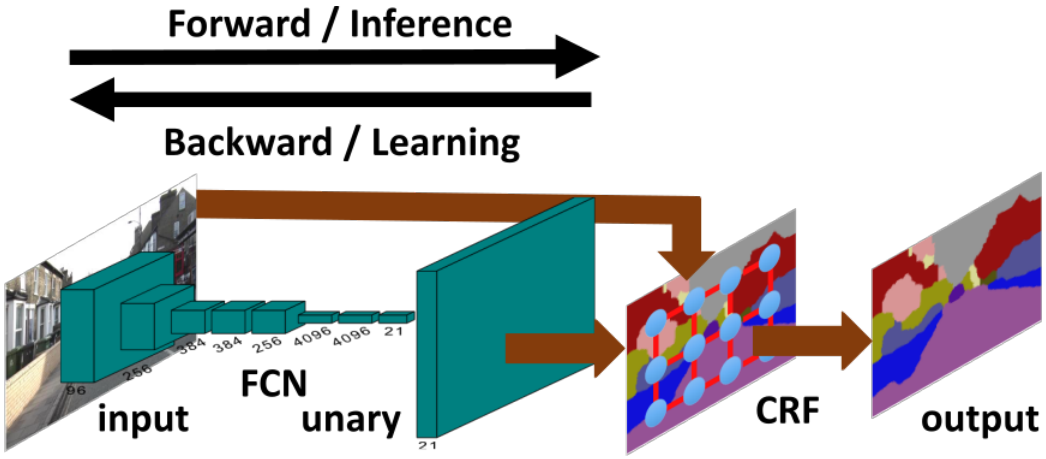
We have presented a robust and accurate approach for incremental dense large-scale semantic reconstruction of outdoor environments in real time from a stereo camera. At the core of our algorithm is a hash-based fusion approach for 3D reconstruction and a volumetric mean-field inference approach for object labelling. By performing reconstruction and recognition in tandem, we capture the synergy between the two tasks. By harnessing the processing power of modern GPUs, we can perform semantic reconstruction at real-time rates, even for large-scale environments. We have demonstrated our system's effectiveness for both high-quality dense reconstruction and scene labelling on the KITTI dataset.

Seeing it from perspective of 2017. Our system uses handcrafted features and textonboost to generate unary potentials (Fig. 3.12 (a)). As of 2017, this is quite outdated approach and significantly better results can be obtained with convolutional neural networks. The first variant replaces these handcrafted unary potentials by fully convolutional network (Shelhamer *et al.*, 2017) (Fig. 3.12 (b)), while the CRF-as-RNN paradigm (Zheng *et al.*, 2015) allows end-to-end learning with enforced structural constraints (Fig. 3.12 (c)). Similarly, dense depth prediction can be replaced by learnt CNN (Zbontar and LeCun, 2016). This has the advantage, that both networks can be combined into a single computational graph to predict semantic and depth labelling jointly (similarly to (Song *et al.*, 2017)) and hence exploit correlation between the two (*e.g.* road is typically flat, *etc.*).

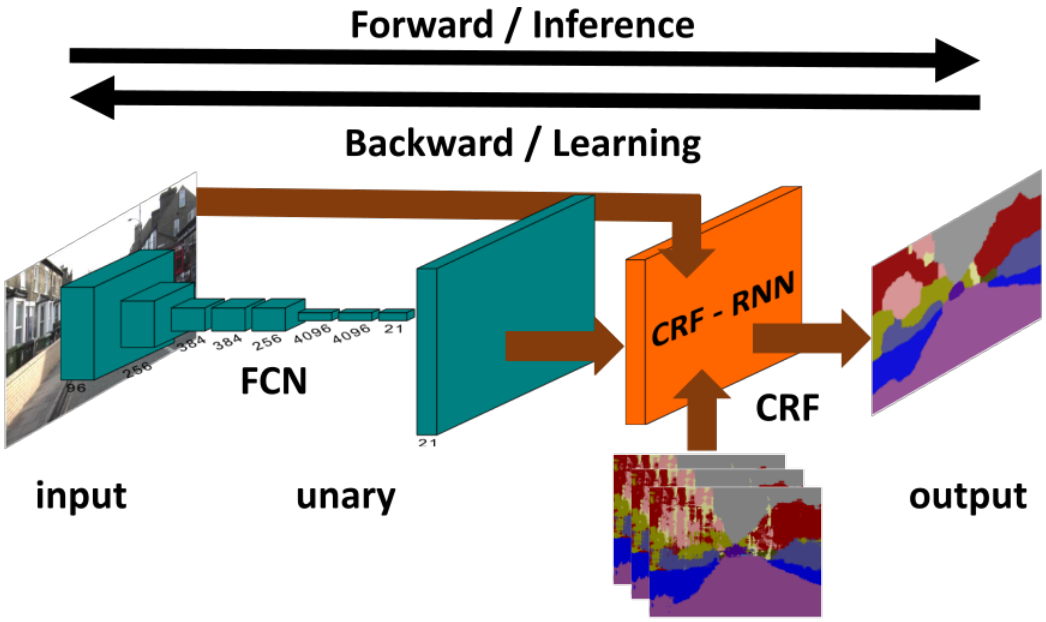
Our 3D reconstruction suffers from two major limitations. The first issue is that this approach is a fusion and not full SLAM with loop closures. As such it is prone to drift; avoiding this is still an active area of research (Dai *et al.*, 2017; Kähler *et al.*, 2016; Whelan *et al.*, 2016). The second issue is with dynamically moving objects; we fuse all data into a single, global reference frame. Hence we cannot model dynamically moving objects properly and we only avoid "ghost" artifacts. Instead, we should segment moving objects, estimate their relative poses with respect to the global reference frame and reconstruct them independently in their own volumes.



(a) Our system with handcrafted features for unary potentials.



(b) Unary features replaced by fully convolutional network (Shelhamer *et al.*, 2017).



(c) With CRF as RNN (Zheng *et al.*, 2015).

Figure 3.12: Replacing handcrafted unary potentials of our system by deep learning models trained in an end-to-end manner is a relatively straightforward step.

4 The Semantic Paintbrush: Interactive 3D Mapping and Recognition in Large Outdoor Spaces

We present an augmented reality system for large scale 3D reconstruction and recognition in outdoor scenes. Unlike existing prior work, which tries to reconstruct scenes using active *depth cameras, we use a purely passive stereo setup, allowing for outdoor use and extended sensing range. Our system not only produces a map of the 3D environment in real-time, it also allows the user to draw (or 'paint') with a laser pointer directly onto the reconstruction to* segment *the model into objects. Given these examples our system then learns to segment other parts of the 3D map during online acquisition. Unlike typical object recognition systems, ours therefore very much places the user 'in the loop' to segment particular objects of interest, rather than learning from predefined databases. The laser pointer additionally helps to 'clean up' the stereo reconstruction and final 3D map,* interactively.

The Semantic Paintbrush builds on top of the system we developed in the previous chapter and extends it to interactive scenarios. Using our system, within minutes, a user can capture a full 3D map, segment it into objects of interest, and refine parts of the model during capture. We provide full technical details of our system to aid replication, as well as quantitative evaluation of system components. We demonstrate the possibility of using our system for helping the visually impaired to navigate through outdoor environments. Beyond this use, our system can be used for playing large-scale augmented reality games, shared online to augment streetview data, and used for more detailed car and person navigation.

4.1 Introduction

Maps help us to navigate and discover the world. Companies such as Google and Microsoft use aerial and/or street-level imagery to produce virtual 3D maps on a *global scale*. These 3D maps form the basis of many navigation systems we use in our cars or mobile devices.

Whilst much progress has been made in 3D mapping, particularly with the advent of real-time depth cameras, most of these virtual maps are still at a *geometric* level, representing the 3D structure of the scene, as opposed to understanding or *recognizing* the higher level objects or scene structure. Furthermore, these maps are captured ahead of time, and often at a coarse level, instead of being captured *live* and reflecting the detailed nature of the scene.

In the previous chapter, we have partially addressed this issue and developed a system for *live* dense large-scale semantic 3D reconstruction from passive cameras. However, the resulting maps are still general-purpose, *i.e.* the system uses only a predefined and hence very limited set of labels. Moreover, even if we managed to train a model with thousands of labels, such "one-size-fits-all" approach would never be capable of *personalisation*, it would never manage to recognize *e.g. my favorite cup* class since such class would never be captured in any training dataset.

In this chapter, we present a new mapping system that is capable of creating large-scale semantic maps of outdoor scenes *interactively*. The word "interactive" is of particular importance, as this not only implies live capture of the map, but also a system that keeps the user "in the loop" to guide the mapping towards objects and elements of the map that are of particular interest. More specifically, we present a novel augmented reality (AR) system for large scale 3D reconstruction and recognition in outdoor scenes. Unlike prior work, which tries to reconstruct scenes using *active* depth cameras, we use a purely passive stereo setup, allowing for outdoor use and extended range sensing. This allows us to reconstruct large and/or distant structures, such as building facades, roads and cars.

Our system not only produces a map of the 3D environment in real-time, it also allows the user to draw (or "paint") with a laser pointer directly onto the reconstruction. The user simply points at an object with the laser pointer, performs a brush-like stroke, and then issues a voice command to interactively segment and label the 3D scene into different object classes. Unlike typical object recognition systems, which work in a "closed-world" scenario, with a fixed, pre-trained set of object classifiers, our system is fully interactive, allowing the user to add new classes on the fly, and even correct object labels.

The laser pointer is additionally triangulated by the stereo camera rig during capture,

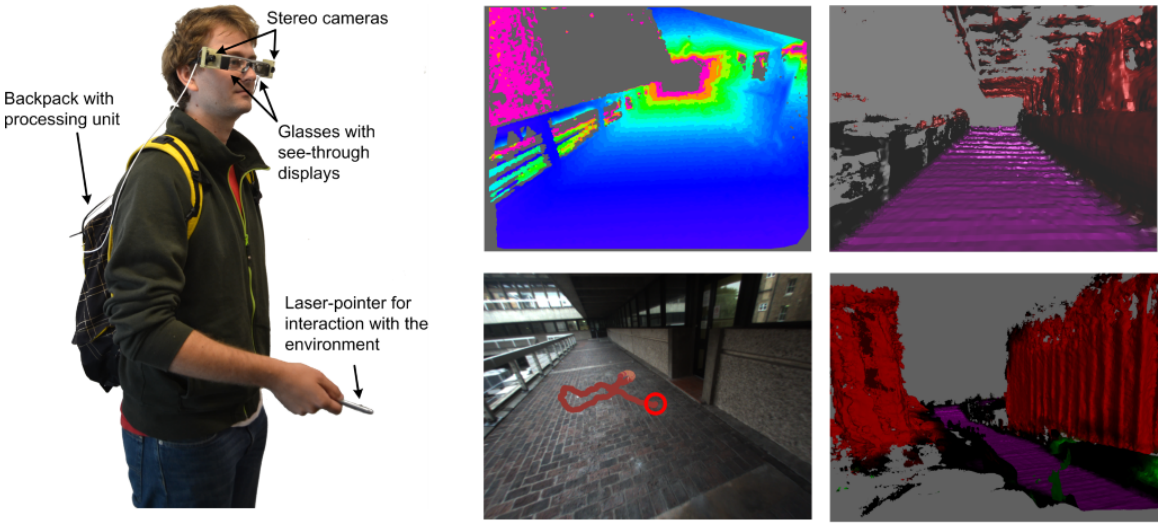


Figure 4.1: The Semantic Paintbrush comprises of an off-the-shelf pair of optical see-through glasses, with additional stereo RGB-Infrared cameras, and an additional handheld infrared/visible light laser. The passive stereo cameras are used for depth estimation. The user can see these reconstructions immediately using the heads-up display, and can use a laser pointer to draw onto the 3D world to semantically segment objects (once segmented these labels will propagate to new parts of the scene). The laser pointer can also be triangulated precisely in the stereo infrared images allowing for interactive "cleaning up" of the model. Final output is the dense semantic 3D map of the scene.

which provides a strong 3D prior to help *interactively* "clean up" the stereo reconstruction and final 3D map. Stereo algorithms typically break in textureless regions, causing major errors. Here, these errors can be quickly and interactively cleaned up by the user, in an online manner. To our knowledge, this is the first such system that allows the user to see the results of object and stereo estimation in real time and interactively correct them.

With our system, within minutes, a user can capture a full 3D map, segment it into objects of interest and refine parts of the model during capture, all by simply exploring the space and moving a handheld laser pointer device, metaphorically "painting" or "brushing" onto the world. We provide full technical details of our system to aid replication, as well as quantitative evaluation of system components.

We are particularly interested in applications that can exploit these large-scale semantic 3D maps. We demonstrate the possibility of using our system for helping the visually impaired navigate through spaces. Here, the semantic segmentation allows us to highlight objects of interest using the AR glasses. The metric and precise reconstruction can be used for navigation, and the laser pointer can be used to pinpoint objects within proximity. Beyond this use, these semantic maps can be used for playing large-scale AR games, shared online to augment streetview data, and used for more detailed car and person navigation.

The Semantic Paintbrush builds on top of the system for real-time dense large-scale 3D reconstruction of outdoor environments presented in the previous chapter. As such, it has the ability to reconstruct objects at greater distances and in direct sunlight, beyond the capabilities of active depth cameras such as the Kinect. Our contributions can therefore be summarized as follows:

- A novel augmented reality hardware system comprising of transparent LED glasses, attached RGB-Infrared stereo cameras, and a one-button laser pointer.
- The ability for users to semantically segment captured 3D maps into object regions using a simple laser pointer and "brushing" metaphor.
- A machine learning pipeline for learning from these object examples to automatically segment the captured 3D models, in real-time, at scale, and with noisier data than previous systems *e.g.* (Valentin *et al.*, 2015).
- Integration of accurate yet sparse measurements from a laser pointer to interactively improve the quality of the stereo depth estimation and reconstruction.
- A first prototype of our semantic mapping system for the visually impaired.

4.2 Related Work

We have discussed the state of the art methods for large-scale semantic 3D mapping in Section §3.2. However, most of these approaches are typically far from real-time and require offline processing. A more recent trend has explored real-time or online object recognition directly during 3D mapping (Salas-Moreno *et al.*, 2013), in particular to help compress 3D models further and aid in relocalization. SemanticPaint (Valentin *et al.*, 2015) takes this concept further by allowing users to label the scene during capture by *touching* surfaces and providing an online learning framework to infer class labels for unseen parts of the world. We build on that framework in this chapter, but change the algorithm to handle large scale outdoor mapping using far nosier stereo data. Furthermore, we fundamentally change the input modality. Using a laser pointer allows for the correction of failures in the estimated stereo depth, which are endemic when moving to outdoor scenes. This ability to create semantic segmentations of the map and correct both the geometry and labels is a critical part of our system.

Laser pointers have been used extensively for HCI scenarios, especially for interacting with large displays in intuitive ways (Wienss *et al.*, 2006; Qin *et al.*, 2010; Olsen and Nielsen,

(a) (b)

Figure 4.2: State-of-the-art technology for helping the visually impaired through AR glasses. This system enhances object boundaries obtained by thresholded depth maps captured by Kinect-like sensors. The depth maps are thresholded in each frame independently and as such are temporary inconsistent and cannot provide the user any semantic knowledge. The use of an active camera limits this system to indoor environments (Hicks *et al.*, 2014).

2001). Our approach uses the laser pointer as a means for interactively segmenting the scene into objects of interest. However, as a by-product the laser pointer can be precisely triangulated through the stereo pair. This allows us to also refine the 3D model, based on accurate 3D measurements taken from the laser pointer. This is in the spirit of systems such as (Habbecke and Kobbelt, 2008) and other scanning laser depth sensors. However, our method puts the user "in the loop" allowing the laser to refine parts that the user cares about or observes as noisy. In contrast to (Nguyen *et al.*, 2013), our method allows the user to label any unknown (indoor/outdoor) environment into objects and semantic parts.

We demonstrate our real-time large-scale semantic mapping system in the context of helping visually impaired users to navigate through outdoor spaces. Here, laser pointing devices have a long history as digital aids for the partially sighted (see (Iannacci *et al.*, 2011) for a review). We however take further inspiration from a relatively new trend of helping the visually impaired through augmented vision (Hicks *et al.*, 2014; Froissard *et al.*, 2014). The basic principle is based on capturing images using a regular camera or depth sensor, and enhancing features of the image such as edges (Froissard *et al.*, 2014) or objects of interest (Hicks *et al.*, 2014). These enhanced images are then displayed to the user on head-mounted AR-glasses, hence stimulating the residual vision of the user.

4.3 System Overview

Now we describe our system from a hardware, user interaction, and software perspective.

4.3.1 Hardware

The hardware for our system is shown in Fig. 4.1 and Fig. 4.3. It is composed of optical see-through AR glasses (EPSON MOVERIO BT-200) with a resolution of 960×540 and field of view of approx. 23° corresponding to a 40″ virtual screen at 2.5 metres. Attached to these glasses is a pair of Omnivision RGB-Infrared (RGB-I) cameras (OV4682 RGB IR) with a resolution of 2688x1520 pixels. These cameras are capable of imaging both visible and infrared (IR) spectra. The cameras are set apart with a baseline of 22cm, calibrated, and using a stereo algorithm (described in the next section), natural features in the RGB image of the left camera are *matched* with those in the right, to estimate the scene disparity. This allows a *dense* depth map to be computed per frame. Additionally, we use a standard red laser operating at visible and IR spectra between 680-730nm, with output far less than 5mW making its usage eye safe. This laser emits both red light for the user to see, but also IR light which can be sensed by the IR sensitive pixels of the stereo cameras. This laser point can be triangulated and used to localize the pointer with respect to the dense 3D reconstruction.

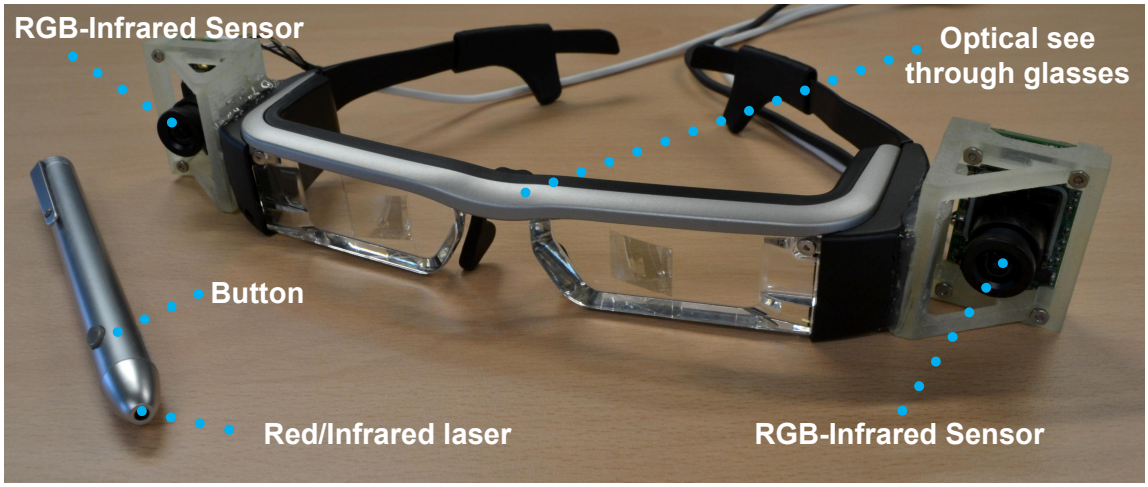


Figure 4.3: The main hardware components of our AR glasses. See text for details.

4.3.2 User Interface and Interaction

As the user wears the AR glasses, she is provided with immediate feedback, as the reconstruction is captured live. The reconstruction is based on a scalable variant of the

KinectFusion system (Nießner *et al.*, 2013). Our system not only produces a map of the 3D environment in real-time, it also allows the user to draw (or 'paint') with a laser pointer directly onto the reconstruction.

In Fig. 4.9 and the accompanying video at `http://www.miksik.co.uk`, we show how the laser pointer is used for interaction. In a basic scenario, a user wears the AR glasses and carries the laser pointer and backpack with processing unit (see Fig. 4.1). The user simply points at an object with the laser, performs a stroke, and then issues a voice command to interactively segment and label the 3D scene into different object classes. Unlike typical object recognition systems, our system therefore very much places the user 'in the loop' to segment particular objects of interest, rather than learning from predefined databases. The laser pointer is additionally triangulated by the stereo camera rig during capture, which provides a strong 3D prior to help 'clean up' the stereo reconstruction and final 3D map, *interactively*. Stereo algorithms typically break in textureless regions, causing major errors and outliers. Here, these errors can be quickly and interactively cleaned up by the user, in an online manner. The immediate feedback is visualized on the AR glasses. Our system also supports multi-user interactions, this scenario is thoroughly discussed in §4.7.

4.4 Software Pipeline

The entire pipeline consists of several steps (*cf.* Fig. 4.4). First, we capture a pair of frames from the RGB-I cameras, which we separate into a pair of color and IR images. Next, we estimate the depth and camera pose from the color images, and detect and track the laser dot in the IR images. Then, the system fuses the input data to the current 3D model and performs 3D inference to propagate the semantic information and improves the reconstruction by high quality depth from triangulated laser dots. The user interacts with the system at all stages, by moving with the wearable AR device and by laser pointer to label the objects and improve the depth. The output of the system is continuously visualized to the user with the optical see-through glasses.

The Semantic Paintbrush relies on semantic 3D reconstruction system developed in §3, hence we will move directly onto the more novel and interactive aspects.

4.4.1 The Laser Paintbrush

Fig. 4.5 outlines the process of laser pointer tracking. Since we use cameras with IR sensitive pixels and emit IR from the laser, we can readily localize the laser pointer in the IR input

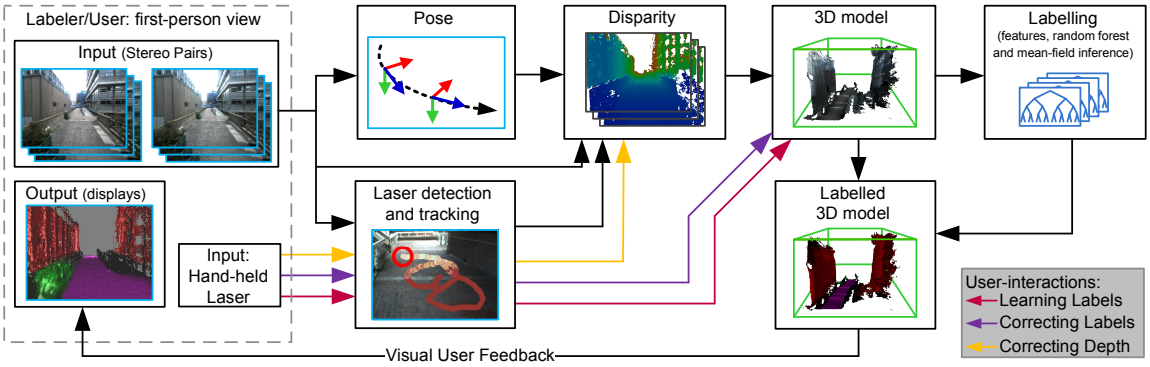


Figure 4.4: Overview of the Semantic Paintbrush system.

images. In most cases, the IR images will contain high-intensity pixels associated with the laser dot and some spurious noise. Due to the presence of noise, we cannot simply extract pixels with the highest intensity. Hence we "track" a local window around the laser dot.

A user first initializes the tracker by pointing the laser into a predefined rectangle in the center of the image. To avoid the spurious noise (often with higher-intensity values than the laser dot), we need to keep the tracked window as local as possible. However, we also need to handle rapid motion of the camera or the laser pointer (or both) resulting in a very large displacement in the image plane. To this end, the tracker uses a Kalman filter, which predicts a pose of a local window in frame $t + 1$. In this frame, we move the local window into the predicted position, and threshold the patch. The tracker automatically switches into the "re-detection" mode if i) the mean intensity of a patch is higher than some fraction α of the highest intensity, ii) the thresholded pixels are not connected, or iii) the number of thresholded pixels is much higher than the expected size of the laser dot at a given distance.

Next, the measured pose is used to "correct" the Kalman filter prediction. If the laser tracker is in the "re-detection mode", it attempts to re-initialize in a small area close to the last known position – this is important since the laser dot intensity is often decreased to the level of noise (or can even completely disappear) due to diffraction and other optical effects. Further robustness is ensured by epipolar constraints; we run two laser trackers (left and right camera) in parallel and if the predictions do not satisfy the epipolar constraint, the tracker switches to the re-detection mode.

The final step is a 3D triangulation of the tracked 2D points observed in left and right images. To this end we implemented simple and efficient linear triangulation method as described in (Hartley and Zisserman, 2003). We then use the 6DoF pose estimated from the visual odometry as a means to back-project this 3D point from camera reference frame into

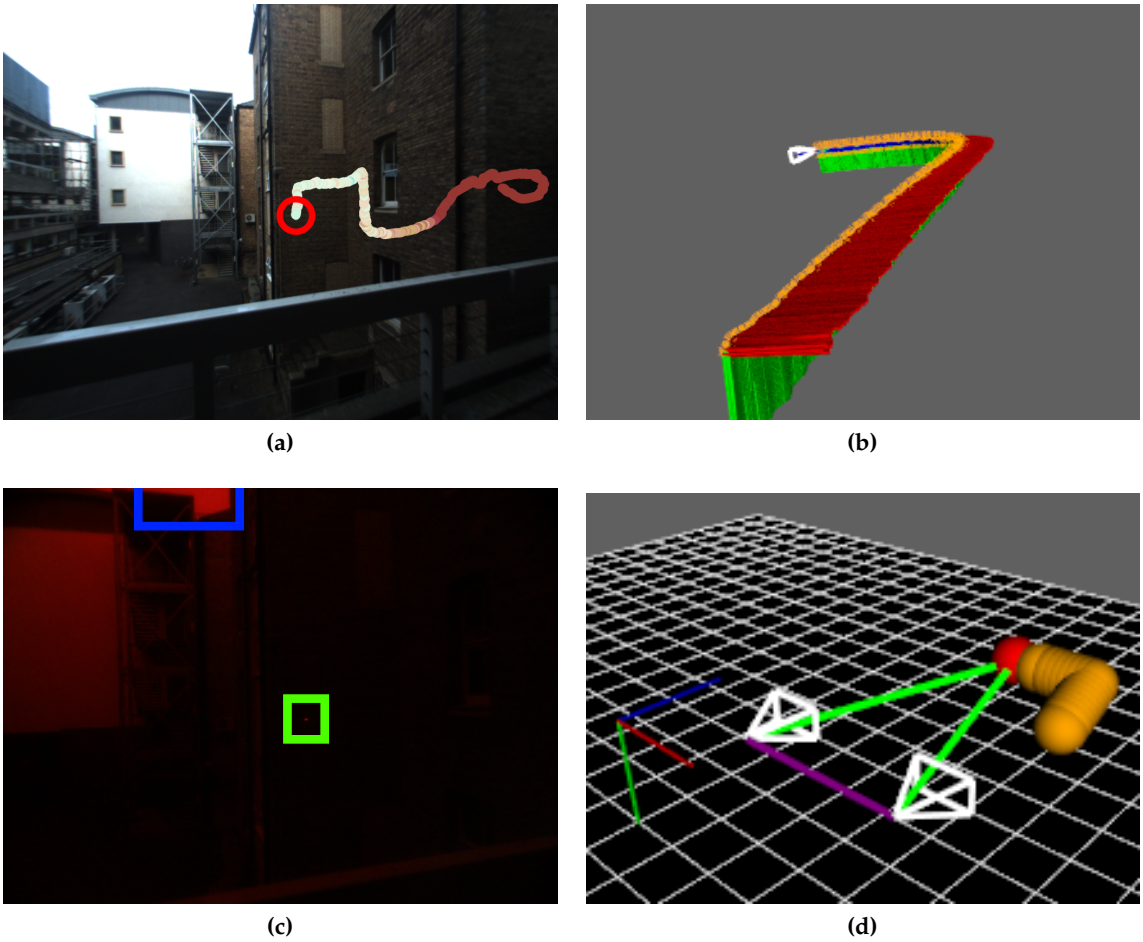


Figure 4.5: Laser pointer tracker: a) tracked and triangulated (mapped) points projected onto the RGB frame, b) part of visual odometry c) filtered images with local window (green) which is necessary since the image often contains brighter areas (blue), d) triangulation.

the global world world reference frame. This adds a desirable temporal consistency of the 3D points, allowing 3D point tracks to be built up over time to detect "brush" strokes and other gestures.

4.4.2 Interactive Improvement of Disparity

Whilst the laser pointer provides the main method for user input, it also carries another important benefit: the triangulated 3D point is very robustly matched and consequently a high-quality depth/disparity estimate is provided at a single sparse point. In general, dense depth/disparity estimation in large outdoor areas using passive stereo is a difficult problem. Although there has been a tremendous progress over the past decade, the typical output of a real-world sequence suffers from several issues – disparity estimates in homogeneous and/or over-exposed (saturated) areas are usually incorrect or completely missing. Moreover, most algorithms evaluate disparity independently per stereo-pair, *i.e.* disparity evaluated on a video sequence typically exhibits a "flickering" effect.

Our applications allow us to, at least partially, recover incorrect and/or missing disparity values by laser interactions. A user labels the scene surface by laser pointer. The laser tracker finds such dots, triangulates them and use them to obtain high-quality depth estimates. These can be used as prior in disparity estimation algorithm (Torr and Criminisi, 2004). To this end, we modified the approach of Geiger *et al.* (2011). A desirable side-effect is that the tracked points are stable over time, hence also the prior is stabilized and the estimated disparity within regions corrected by interactions is more temporarily consistent.

Fig. 4.6 shows the steps used in disparity correction using the sparse 3D laser point. The method of Geiger *et al.* (2011) robustly matches a set of sparse corner-like features (called support points) first and forms a Delaunay triangulation which serves as prior (a piece-wise linear function) in a generative model for stereo matching. This reduces the disparity search space to plausible regions and tends to disambiguate dense matching even without any global optimization method.

To improve disparity by interactivity, we inject the tracked laser points into a set of the support points before it computes triangulation. Feature matches in homogeneous areas are often incorrect, hence we form a convex hull around laser points and remove all support points from its interior. We prefer this conservative strategy since we can always add missing matches by interactivity; however, we need to make sure there are no incorrect support points for prior. We consider three situations for each triangle: 1) if all vertices of a triangle are laser points, we decrease the weight of feature matching term by β_{decr} (*cf.* (Geiger *et al.*, 2011), *Eq.* 8) and rely much more on the prior since we are very certain about the correct disparity value from the laser points. 2) If at least one vertex is a laser point, we decrease the weight of the feature matching term only by some fraction of β_{decr}. 3) Finally, if no vertex of a triangle corresponds to laser points, we maintain the current state.

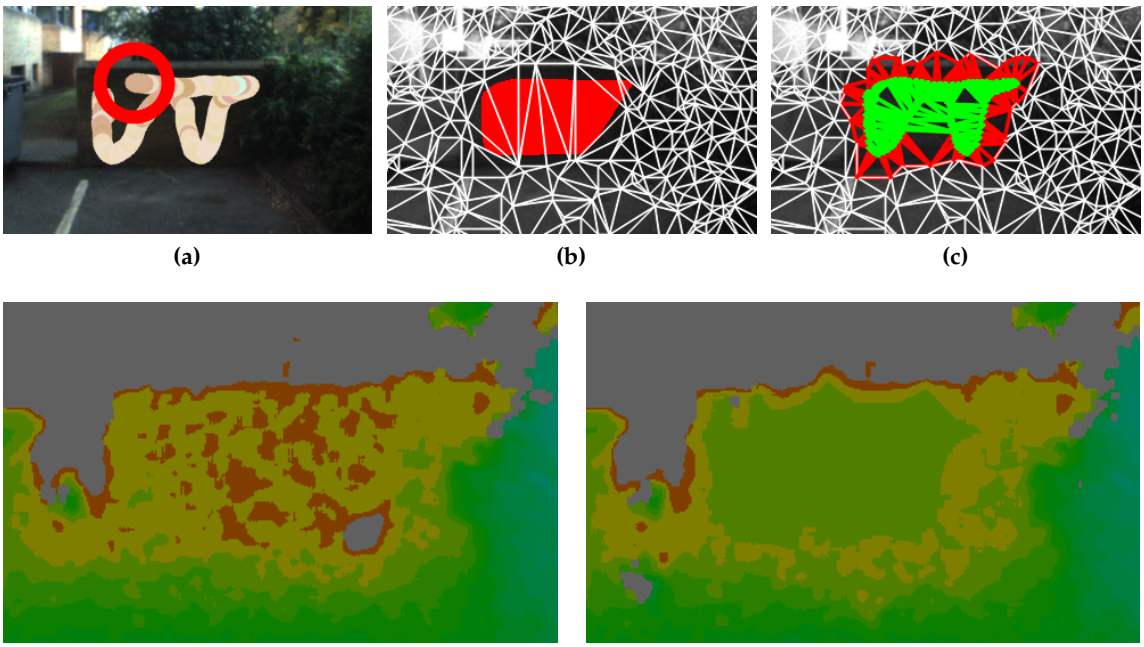


Figure 4.6: Disparity correction: a) tracked laser point, b) removed support points c) support points provided by interaction, d) disparity without interactivity, e) improved disparity.

Unfortunately, all real-world measurements contain some level of noise, which includes our laser tracker, triangulation, and pose estimation (projection of 3D point to a current frame). In order to handle noisy input data and make disparity estimates more robust in the case of planar surfaces, we find connected components of triangles, fit a plane (using RANSAC with least squares refinement on inliers) into all support points obtained by the laser tracker, and share the estimated plane as a more robust prior (this can be viewed as a regularization) by all triangles within a connected component. In the case of unreliable surface measurements, a user can easily add more support points in these regions to clean them up during refinement.

4.4.3 Interactive Learning of Semantic Labels

In addition to map refinement, the laser is used to mark objects of interest through simple gestures such as strokes. Our system is then able to learn a classifier for labeled parts of a scene, where the semantic labels l_i are provided through speech recognition. Note that the user need not precisely label every voxel belonging to the object and can instead roughly mark a small set of voxel and the algorithm will automatically propagate labels to the other voxels belonging to the same part of a scene. Laser interaction is more convenient in this

outdoor scenario than touch gestures, which are used in the SemanticPaint system (Valentin *et al.*, 2015). The laser allows interaction at a distance and does not corrupt the 3D reconstruction; moving hands in front of a camera causes serious issues if corresponding depth values are not segmented and masked out properly.

Similarly to §3, at the heart of our system is a densely-connected pairwise dynamic Conditional Random Field (CRF) defined on voxels. In such a model, each voxel i in the 3D reconstruction volume $\mathcal{V}$ of the scene is represented by a discrete random variable x_i that represents the semantic class $l \in \mathcal{L}$ (*e.g.* road, sidewalk, wall, ...) that the voxel belongs to. The unary potentials are evaluated by streaming decision forests that are extremely fast both to test and train online on voxel oriented patches (VOP) features. For the pairwise potentials, we employ the standard Potts model that enforces smoothness but preserves edges (we use RGB, surface normal vector and 3D world coordinate as features). These potentials take form of mixture of Gaussian kernels that allow efficient, filter based inference which is a message passing algorithm (*cf.* 2.1.2). Although the CRF is defined on continuously changing data, its energy landscape changes only gradually from one frame to the next. This allows us to amortize the optimization cost over multiple frames and a GPU implementation allows super real-time speeds (one update of the messages requires 6 ms).

4.5 Augmented Visualization

The last step of our pipeline involves rendering our synthetic scene on the (full-color) displays of the glasses. These displays are transparent, allowing our raycasted 3D model to be superimposed over the user's view of their physical environment. For interactive segmentation of the scene, superimposing the two in this way provides a natural way of interacting with the 3D model, providing users with a way to verify the accuracy of the interactive labeling of the scene in real time. Our rendering through the glasses shows various semantic classes using a number of easily-distinguished colors. Fig. 4.7 shows some visualization examples.

4.5.1 Interactive Reduction of Visual Clutter

In addition to its uses for semantic labeling and improving disparity, the laser pointer can also be used to select individual semantic classes for visualization. That is, the user points at a part of the scene that is labeled with a particular class and the system then highlights all parts of the scene that belong to that class, whilst graying out those parts that belong to other

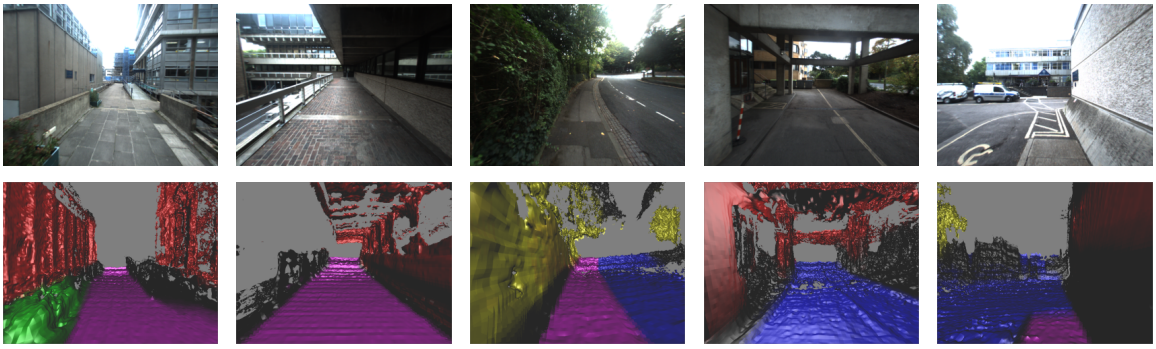


Figure 4.7: Final mean-field inference results for four sequences (the last two columns belong to the same sequence). Our streaming decision forest is able to learn per-voxel predictions about the object classes present in the scene. Each pixel is classified independently, and so the forest predictions can be somewhat noisy. The mean-field inference effectively smooths these predictions to produce a final labelling output to display to the user.

classes. This provides a useful way of reducing the visual clutter in a scene, *e.g.* it might be useful for a visually-impaired person trying to follow a footpath to be able to prominently highlight the footpath and grey out classes such as the road and the surrounding buildings. The user can either select a class that should be highlighted until the system is informed otherwise (which is useful for tasks such as following a footpath), or switch into interactive highlighting mode, in which case the class being highlighted changes in real time as the user moves the laser pointer around. This could also be augmented with audio feedback for visually impaired users to determine the type of objects in view.

4.5.2 Map Sharing

In order to allow optional multi-user interactions, we need to share information between users. For ease of exposition, we discuss a two user scenario. We assume that users A and B are close to each other so they observe almost the same part of the scene. In our scenario, only user A builds a single common map and provides raycasted visualizations to user B. At the beginning, we estimate a relative pose between the users and run visual odometry for each of them. Then, the user B sends only a 6DoF pose and receives a raycasted visualization from her own perspective so both users can interact. Though this approach adds computational load on the user A, this is not an issue in practice, since raycasting takes only 5 ms. Though our map sharing method is simple and efficient, the quality depends on precision of visual odometry. In order to prevent drifting, we re-estimate a relative pose between the users every 500 frames.

4.6 Experimental Results

4.6.1 Qualitative Results

Propagation of User Labels

The user indicates the surface of objects in the physical world using the laser pointer. Our system interprets such indications as a paint stroke, and voice input is used to associate an object class label to the corresponding voxels. Then, our mean-field inference engine (see related paper (Valentin *et al.*, 2015)) propagates these labels through the reconstructed scene very efficiently. Thanks to the pairwise potentials we use, the result is a spatially smooth segmentation that adheres to object boundaries. Examples of label propagation are shown in Fig. 4.8 and supplementary video at `http://www.miksik.co.uk`.

Semantic Labelling

Our system learns a streaming decision forest classifier in a background CPU thread given the labels provided by the user. At some point, the user selects 'test mode', and the forest starts classifying all voxels. Fig. 4.7 and supplementary video (`http://www.miksik.co.uk`) shows the final smooth results obtained by running our mean-field inference procedure on our decision forest predictions. We show results on four new challenging sequences captured using a head-mounted display (the last two columns belong to the same sequence). The images clearly indicate the sharp boundaries that we manage to achieve between different conflicting semantic classes. For example, observe the extremely accurate boundary between the pavement and the road in the sequence in the third column.

4.6.2 Quantitative Results

We evaluate the accuracy of our mean-field filtering of the forest predictions, based on a variety of test sequences captured from outdoor scenes. For each sequence, a series of keyframes were hand-labeled with object segmentations. Keyframes were selected to ensure full coverage of the scene. These ground-truth images are then projected and aggregated onto the underlying TSDF, and then back-projected to all the views of each sequence. We use the results to calculate global accuracies for each of our semantic classes (see Table 4.1).

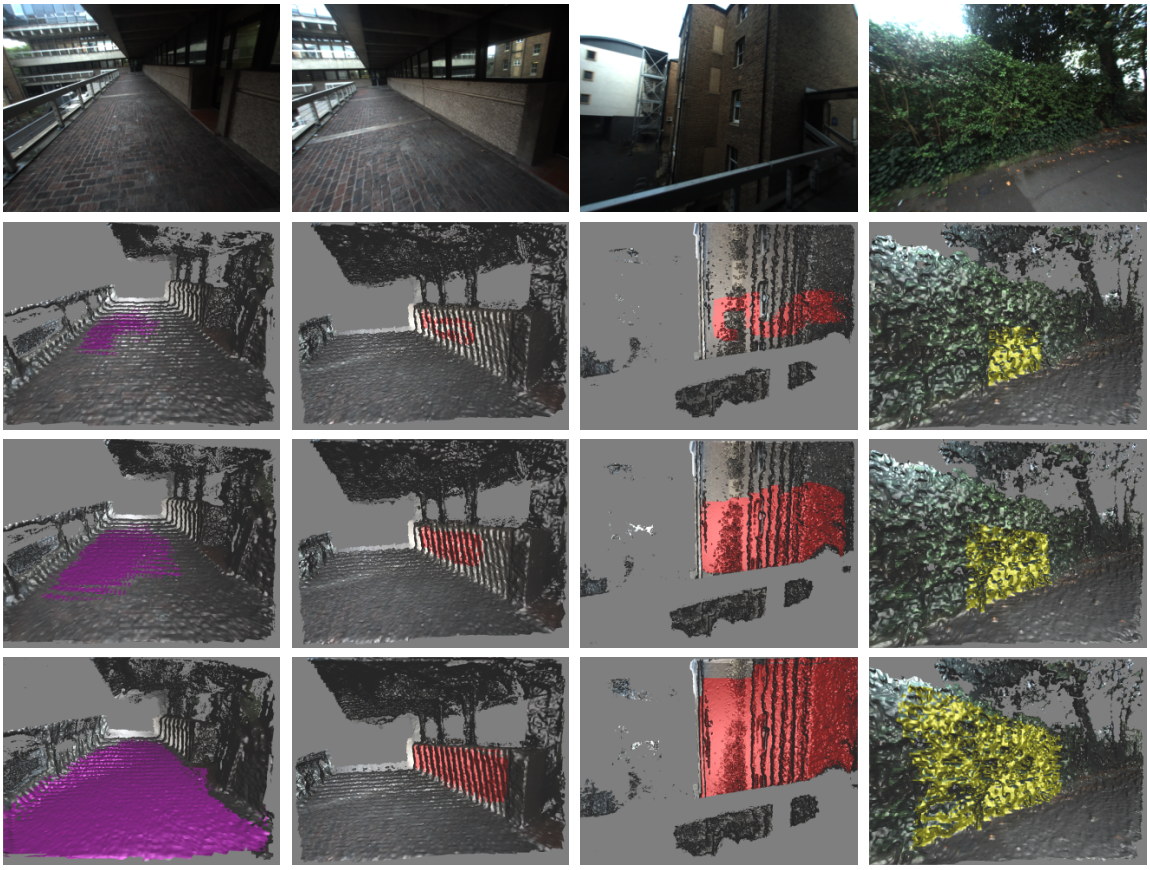


Figure 4.8: Label propagation. Our efficient inference engine smoothly propagates class labels from the voxels indicated by the user to the rest of the volume. Here we show examples from four sequences. The first row shows the raw input data; the second row shows the labelling after a couple of propagation steps; the third and fourth rows show the labelling at later stages of the propagation. The pairwise terms in our energy encourage a smooth segmentation that respects object boundaries.

4.6.3 Computational Efficiency

The inherently volumetric nature of our approach parallelizes well on modern GPUs. We have employed laptops with an Nvidia GeForce GTX 880M with 8 GB of GPU RAM, and quadcore Intel i7 processor with 24 GB of CPU RAM. We provide approximate system timings in Table 4.2. Although the timings change as a function of the number of visible voxels and resolution, in all tests we observed interactive frame rates. Numbers are provided for $4 \times 4 \times 4$ cm^3 voxel resolution, 1024×768 pixels and we do not fuse voxels beyond 20 m. The semantic segmentation pipeline runs on a GPU while the disparity estimation, laser tracker, disparity correction, visual odometry and forest learning run on the CPU. Note that the forest learning runs asynchronously in a background thread so it does not influence reconstruction and labeling. This thread continuously samples new labeled training data from the current view frustum and updates itself. This ensures an up-to-date forest is

Table 4.1: Global accuracies (true positives / total numbers) for each class we use. The first column shows results for labelling in the image domain; the second column shows results for labelling in 3D and then projecting those labels to 2D.

Class	Background	Building	Road	Pavement	Tree	Bin
2D Labelling	87.6	85.4	86.2	86.9	82.3	80.5
3D Labelling	88.5	89.3	88.9	89.2	89.3	84.0

Table 4.2: Approximate system timing. Despite small fluctuations we observed consistently good, interactive frame rates. Note, the reconstruction and labeling are independent of forest update step.

Component	**Time**
Disparity Estimation	80 ms
Laser Tracker	2 ms
Disparity Correction	3 ms
Visual Odometry	20 ms
Fusion	15 ms
Forest Update	170 ms
Forest Evaluation	5 ms
Mean-field inference	2-10 ms
Wifi latency	5-10 ms

available for classification whenever the user requests it. As shown in Table 4.2, the most time consuming step is disparity estimation, but this can be implemented on GPU as well. Note, the reconstruction and labeling are independent of forest update step.

For a two-user scenario, we established a peer-to-peer wireless network. Since we transfer only 6DoF pose and raycasted visualizations, the data transfer is fast enough. Latency is not a huge issue, since we accumulate all the interactions over multiple frames and transfer the data when the user is satisfied.

Finally, the size of the environment that our system is able to map is limited by 1) drift of the visual odometry, 2) battery life of a laptop under heavy load (1 hour) and 3) GPU and CPU RAM. Considering these limits, we were able to run our system in environments of up to 100-500 metres. A standard scene can be reconstructed and labeled at human walking speed and we use no post-processing.

4.7 Applications

So far we have described and evaluated our novel mapping system and uncovered its low-level interactive capabilities. Whilst the focus of our work is technical, we believe the system as a whole could have dramatic impact for HCI applications. We demonstrate the potential application areas in the next section.

4.8 Semantic Maps for the Visually Impaired

There are more than 285 million people in the world living with sight loss which has a significant impact on their daily lives. Over 85% of these individuals have some remaining vision (Mariotti, 2010). Recently, there has been an interest in developing smart glasses (Hicks *et al.*, 2014; Froissard *et al.*, 2014), to provide these people with additional information from the nearby environment through stimulation of the residual vision. The aim is to increase the information level regarding the close environment using depth edges. This rather simplistic, though effective method enables the user to more independently traverse and navigate areas by providing the user with richer information than residual vision could provide (*cf.* Fig. 4.2).

We believe our live semantic maps can be directly used to highlight user-specific objects learned through online teaching with a carer/helper/trainer. This will present the visually impaired user with even more information regarding the surrounding environment to understand it more clearly. Examples of these user-specific objects or regions could be stairs, road-crossings, bus-stops, doors, booths ...

The basic scenario is that a visually impaired and a helper, both wearing smart-glasses displaying individual views of a shared 3D reconstructed environment, label the user-specific objects or regions of interest through usage of a laser pointer handled by the helper. The user can learn to use the system within familiar environments highlighting only regions that the user finds useful with help from the carer. The objects are labeled and learned by the system online, hence immediate response can be provided to both users. Fig. 4.9 shows a demonstration of this scenario.

At a later stage, once the objects are labeled, the visually impaired user can return to the same scene, and view the semantic map using the heads-up display. The transparency of the displays is advantageous for a visually-impaired person using the glasses to navigate around a pre-labeled scene, since it allows her to enhance, rather than replace, her remaining vision with the spatial information provided by the 3D mapping. Furthermore, transparent displays allow other people to see the wearer's eyes, which is helpful for social interaction.

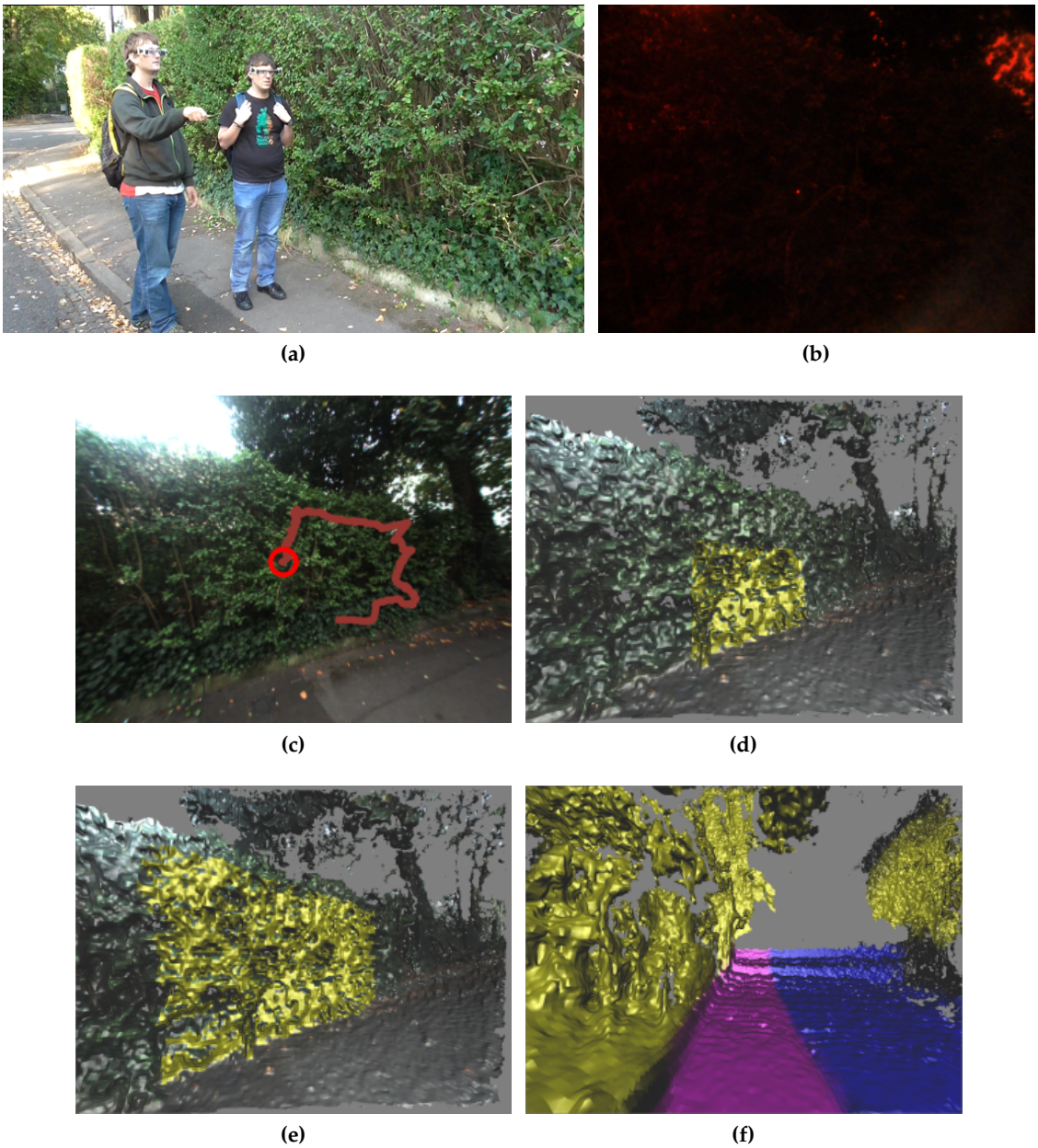


Figure 4.9: A potential application area for semantic 3D maps for aiding visually impaired people to navigate outdoor spaces. The basic scenario is that a visually impaired and a helper, both wearing smart-glasses displaying individual views of a shared 3D reconstructed environment, label the user-specific objects or regions of interest through usage of a laser pointer handled by the helper. As shown in (a and b) the helper is indicating the object "tree" using the laser pointer. Our system then starts to learn this "tree" model. We first track and detect the laser dots in the IR images (b). These points are detected and tracked over a sequence of frames (c). After interaction, the label propagates to segment the tree (d and e), and other instances are detected in the scene (f).

Once a scene has been labeled by a sighted user, it can be converted into a suitable form for assisting a visually-impaired person to understand the nature of her environment, and navigate safely around it. Existing techniques such as those of Hicks *et al.* (2014) have shown the usefulness of whole-image techniques, such as depth-to-brightness mapping, for helping visually-impaired individuals to avoid obstacles. The inclusion of semantic labeling has the potential to add an additional dimension to this kind of system by providing the wearer with more information about the objects around them and the boundaries between surfaces in their local environment (*e.g.* between a footpath and a road).

4.9 Other Applications for Semantic Maps

Personalized semantic maps with known object segmentations could also be used for a variety of other way-finding and navigation applications, either for robots or end users. For example, imagine self driving cars or quadcopters being able to follow particular paths and avoid obstacles. Additionally, users could interact with these robots, asking them to find particular instances of objects by semantically breaking down the world and using the laser pointer (*e.g.* "please go to *that* building"). Furthermore, if such a model was maintained and updated over time, finding points of interest could be as simple as uttering a few words (*e.g.* "where is the nearest bus stop").

These personalized maps could also be interactively captured, shared online and played back. For example, a user could give fine details for navigation to a friend, by actually capturing their path through the city, and then sharing it online, allowing for a detailed retracing of the steps, potentially with audio feedback. Another aspect is the ability for users to add semantic information to online maps. Here, by crowd-sourcing multiple personalized maps, a larger corpus of semantic maps could be generated. Users could use these semantic labels for searching, *e.g.* "find the nearest bus stop to my map location", or "please find the entrance to the building". This latter point is also very important, as it allows a level of detail not yet available in regular maps, allowing for a more fine-grained level of way finding and navigation. Finally, augmented reality mobile gaming could be a rich source of application. Imagine quickly scanning in and labeling an outdoor space, and then associating object classes with aspects of the game. For example, game characters could hid behind particular objects, or follow particular paths or enter buildings. Such augmented reality scenarios could be expanded for planning the renovation of buildings and cities, automating inventory, and town planning.

User Experience. The proposed system has been used by 15 users. All users felt comfortable with it. In particular, they liked the system performing a 3D reconstruction and labeling at interactive framerates and the laser pointer providing a natural means of interacting at a distance in outdoors environment as opposed to touching. The users also liked the see-through glasses allowing to see the real world with overlayed outputs providing an extra information about the environment.

Though the users provided a positive feedback in general, they also suggested a few modifications to make the system more comfortable, mostly on the hardware side. They suggested in particular to balance the center of gravity of the AR glasses better to prevent sliding off from the user's nose. Another recommendation was to change the position of wires in order to less restrict the motion of user's head. On software side, the users mostly complained about drift of the visual odometry.

Limitations. Despite very encouraging results, our system is not without limitations. As with all recognition algorithms, the segmentation results are not always voxel-perfect, as shown in the results and accompanying video. One possibility, however, is to allow the user to interactively make corrections to help reduce such errors. We believe additional modes of interaction such as voice priors (*e.g.* "walls are vertical"), as well as more intelligently sampling the training examples could further improve results. From a computational standpoint, our system is fairly GPU heavy, which limits us to laptop only uses currently. With the advent of mobile GPGPU there are likely ways of addressing this in future work.

Further, our system is currently state based; *i.e.* it requires the use of voice commands to switch between annotation, training, and test modes. We are planning an extension where both the learning and forest predictions are always turned on. This will require considerable care to avoid "drift" in the learned category models: the feedback loop would mean that small errors could quickly get amplified. Finally, algorithmic parameters such as the pairwise weights are currently set at compile time (these are cross-validated and common across datasets shown). Given a small training set (perhaps boot-strapping), more reliable settings could be automatically selected online.

Since our system uses an IR laser pointer with output far less than 5mW, the pointer will not work in direct sunlight, but the IR laser can be replaced *e.g.* by LIDAR-based pointer. In general, the system works well in an urban environment, but fails in areas where the visual odometry and/or disparity estimation fail (*e.g.* those containing highly reflective or specular surfaces, or textureless regions).

4.10 Conclusions

In this chapter, we have presented an interactive 3D mapping system that can semantically label large-scale unknown outdoor scenes. The system can take advantage of interactive input from the user in order to guide the mapping towards objects and elements of interest in the scene. Rather than using active depth cameras, we capture our input using a passive stereo approach, making it possible to reconstruct large or distant structures outdoors.

Our system comprises a pair of see-through glasses, two RGB-Infrared stereo cameras, and a one-button laser pointer. The laser pointer helps the user highlight objects of interest and, in combination with voice commands, can provide semantic labels for objects (even distant objects) in an online fashion. The laser pointer can also be used to provide accurate, sparse measurements to the system in order to improve the estimated stereo depth, and thereby improve the final reconstruction.

We believe our mapping system could be of particular use for the visually-impaired, who in many cases can benefit from a more accurate understanding of the nature of objects in their environment. Whilst we have focused our work on the technical details, we feel this could be a high impact area for future work. For example, our system's ability to differentiate footpaths from roads has the potential to be extremely helpful in providing visually-impaired people with a safer way to navigate independently outdoors.

Seeing it from perspective of 2017. The Semantic Paintbrush is a direct extension of the system for dense large-scale semantic 3D reconstruction proposed in the previous chapter, hence it inherits most of its limitations. However, the specific applications and interactive setting introduce few more demands.

Our camera pose estimation uses solely visual odometry. This significantly reduces possibility of re-using maps in the future. This deficiency can be addressed by using large-scale re-localization methods (Torii *et al.*, 2015; Kendall *et al.*, 2015) which allow us to re-use maps for regular daily activities such as commuting to work (Churchill and Newman, 2012). This is related with models we train; currently, we use a single model that should work well everywhere however this is somewhat unnecessary and it would be better to use models specific to particular locations.

In interactive settings, it is necessary to relax the assumption of static environments. It is necessary to be able to handle objects that are not just moving but also deformable (Orts-Escolano *et al.*, 2016; Dou *et al.*, 2016; Innmann *et al.*, 2016).

5 Incremental Dense Multi-modal 3D Scene Reconstruction

Acquiring reliable depth maps is an essential prerequisite for accurate and incremental 3D reconstruction used in a variety of robotics applications. Depth maps produced by affordable Kinect-like cameras have become a de-facto standard for indoor reconstruction and the driving force behind the success of many algorithms. However, Kinect-like cameras are less effective outdoors where one should rely on other sensors. Often, we use a combination of a stereo camera and Lidar. However, processing the acquired data in independent pipelines generally leads to sub-optimal performance since both sensors suffer from different drawbacks. In this chapter, we propose a probabilistic model that efficiently exploits complementarity between different depth-sensing modalities for incremental dense scene reconstruction. Our model uses a piecewise planarity prior assumption which is common in both indoor and outdoor scenes. The proposed model can be seen as an extension of the laser paintbrush from previous chapter to non-interactive scenarios. We demonstrate the effectiveness of our approach on the KITTI dataset, and provide qualitative and quantitative results showing high-quality dense reconstruction of a number of scenes.

5.1 Introduction

Aquiring reliable depth maps is an essential prerequisite for accurate and incremental 3D reconstruction used in a variety of robotics applications, including navigation (Urmson *et al.*, 2008; Vineet *et al.*, 2015), object recognition (Song and Xiao, 2014; Gupta *et al.*, 2014), wearable and/or assistive technology (Miksik *et al.*, 2015b), and grasping (Potapova *et al.*, 2014). Depth maps produced by affordable Kinect-like cameras have become a de-facto standard for indoor perception (Whelan *et al.*, 2013; Silberman *et al.*, 2012) and the driving force behind

the success of many algorithms. However, Kinect-like cameras are less effective outdoors where one should rely on other sensors. With the advent of an increasingly wide selection of sensing modalities (*e.g.* 2D/3D laser range finders, optical cameras, stereo/depth cameras, flash lidars, radars, *etc.*), it is now common to obtain multiple observations of a given scene; a typical example are sensors mounted on (un)manned vehicles. Using observations from different modalities is generally advantageous as they are complementary but at the same time challenging since there often is no one-to-one correspondence across modalities.

Let us consider, for instance, an optical camera and a lidar, as illustrated in Fig. 5.1. The camera has a limited dynamic range (Fig. 5.1, top) and many parts of the perceived scene can easilly get saturated (specular highlights, reflections, over-exposure, ...). Stereo matching algorithms generally fail in areas with lack of texture, resulting in large holes in the dense depth maps (Fig. 5.1, 3rd row). This reconstruction problem is ill-posed even for images without any illumination artifacts due to ambiguity in dense correspondence matching (textureless areas, repetitive patterns, ...) and performance is usually determined by a trade-off between accuracy and efficiency. Fast algorithms typically use only (non-regularized) per-pixel predictions with heuristic postprocessing reducing noise (Je and Park, 2013), while accurate but slow methods rely on (semi) global optimization enforcing smoothness and ordering constraints (Woodford *et al.*, 2009; Sinha *et al.*, 2014). Moreover, most algorithms operate on a per-frame basis, which reduces their efficiency and temporal consistency. In contrast, lidars (Fig. 5.1, 2nd row) are often able to sense in areas in which video information is not exploitable and provide more accurate/reliable measurements. However, lidars often have smaller field-of-view than cameras, depth readings are limited to a certain maximum range and are obtained at much slower temporal rate (except with the most expensive systems, which are not suitable for many applications).

Processing data from different modalities in independent pipelines and fusing only their outputs generally leads to sub-optimal performance. In this chapter, we propose a model that efficiently exploits complementarity between different depth-sensing modalities for incremental dense scene reconstruction. For ease of exposition, we demonstrate our method on measurements from a calibrated stereo camera and lidar, however the method is general and can accomodate other sensors as well (*e.g.* radar for obstacle detection, *etc.*). We directly integrate the lidar data into the stereo reconstruction algorithm to predict accurate depth maps and we show that superior results can be obtained even with relatively cheap, second-class sensors (Fig. 5.1, bottom).

At the core of our system is a pairwise conditional random field (CRF) that captures

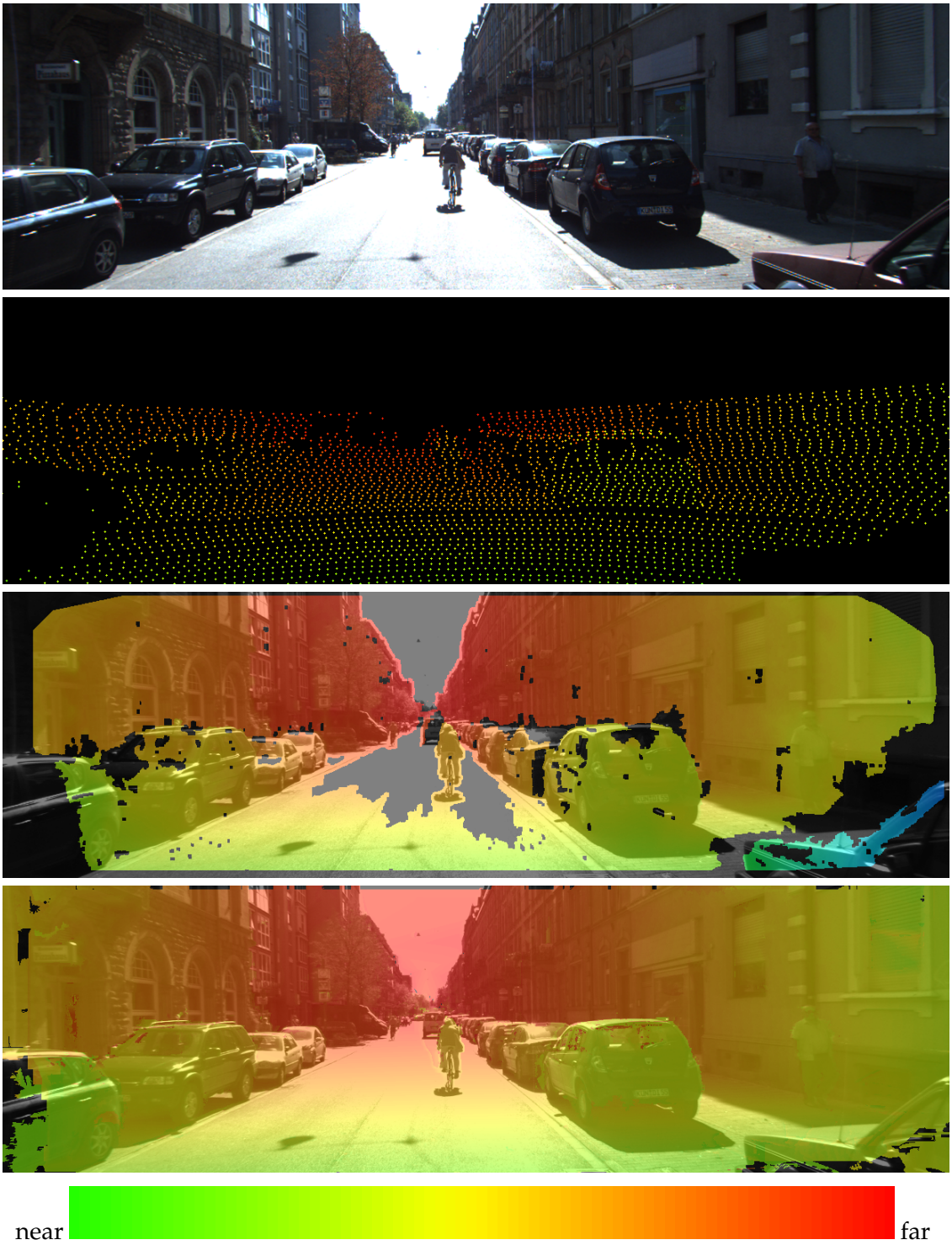


Figure 5.1: An image captured by a calibrated stereo camera with multiple reflections, specularities and over-exposed areas (top), 3D point cloud captured by a Velodyne HDL-64E laser scanner (2nd row), stereo reconstruction (Geiger *et al.*, 2010) (3rd row), and output from our system (bottom), as seen from a moving platform on-the-fly.

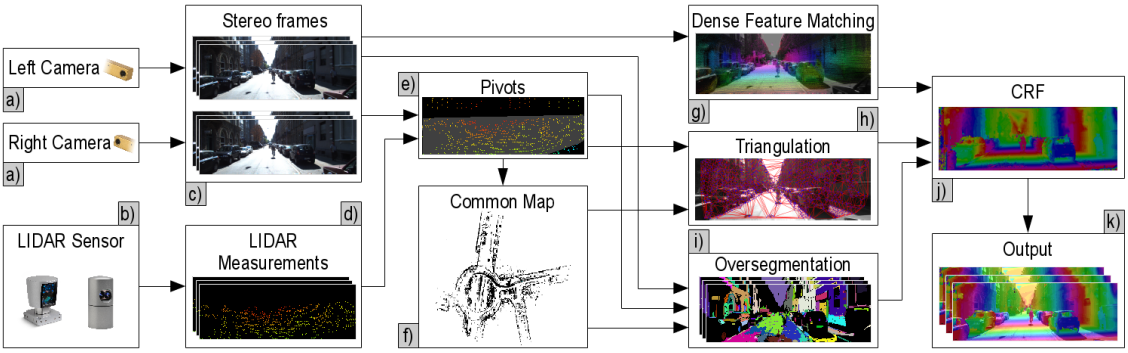


Figure 5.2: Overview of our system: (a) given a pair of calibrated cameras and (b) lidar, we (c) capture stereo images and (d) 3D point cloud, (e) generate an initial set of pivots and (f) project them on a common map. Given the pivots within the current frustum and stereo images, we evaluate (g) unary potential and piecewise planar term based on (h) the Delaunay triangulation of pivots and (i) oversegmentation over which we (j) define a pairwise CRF to (k) infer the final solution.

interactions between the pixels and efficiently combines information from the stereo camera and lidar (Fig. 5.2 (k)). To this end, we assume having a set of sparse but very accurate 3D points that provide partial prior knowledge about the scene. We call these points *pivots* (Fig. 5.2 (e)) and they correspond to lidar 3D measurements and 3D points generated by robustly matched and triangulated sparse 2D keypoints. To exploit this prior knowledge in our model, we significantly reduce the unary costs attached to these points, so that the optimal depth assignments are attracted towards pivots' depth, and pivots *guide* dense matching. Our unary potentials (Fig. 5.2 (g)) are based on dense matching of 2D features along the epipolar lines and a piecewise-planar prior defined by various groupings of pivots (Fig. 5.2 (h, i)). Such priors typically model only small scene fragments and/or do not respect object boundaries (Geiger *et al.*, 2010). Thus, we group pivots over a multiscale hierarchy of oversegmented regions that provide knowledge about potential object boundaries and model planarity over larger surfaces such as the whole road segment or a table top. Pivots also help to disambiguate dense matching by constraining the searched range which results in more confident unary predictions and their faster evaluation. Our pairwise terms propagate information into uncertain (*e.g.* saturated) areas and enforce smoothness among the neighbouring pixels (including the lidar data). Note that we do not introduce any hard constraints forcing variables at pivots' coordinates to take the estimated depth, hence, this leaves the chance to recover if the pivot is assigned an incorrect measurement.

Further, we project the pivots on a common map (Fig. 5.2 (f)) to maintain the temporal consistency and not to discard any measurements. Hence all measured data are available to the algorithm on request (and not just the latest sensor readings). To maintain the computational and memory complexity, we use a sparse hash-table-driven data structures that ignore unoccupied space and swap/stream map data between device and host memories as needed to fit the data into GPU memory and process only the data within a current frustum.

In order to infer the approximate maximum posterior marginal (MPM) solution efficiently, we use a mean-field inference technique that refines the marginals of a node with a bilateral filter (Krähenbühl and Koltun, 2011). This allows us to run inference in each frame (only a few mean-field updates are required), which is of utmost importance in most of the robotics settings where output is required at real-time or interactive rates. The system outputs a per-pixel probability distribution instead of a single label, which is desirable in robotics as it allows probabilistic interpretation in other subsystems. All parts of our system are trivially parallelizable, hence suitable for GPU implementation.

It should be noted, that our approach is not specific to this application, can be used with multiple sensors and/or other modalities, naturally accommodates other priors and can be extended to handle other tasks such as semantic segmentation.

5.2 Related work

Dense depth map estimation from stereo images is one of the most studied problems in computer vision (Scharstein and Szeliski, 2001). Fast methods usually treat each pixel independently, capture context only in a very small area and smoothness is often achieved heuristically through postprocessing (Je and Park, 2013). Algorithms relying on (semi-)global optimization capture the structure (Hirschmüller, 2008; Geiger *et al.*, 2010), encode higher order constraints (*e.g.* slanted planes) (Woodford *et al.*, 2009) and use segmentation (Bleyer *et al.*, 2011; Sinha *et al.*, 2014; Yamaguchi *et al.*, 2014) to inject knowledge about spatial extend of objects. However, these methods do not exploit complementarity and partial knowledge about the scene obtained from *different* modalities. Torr and Criminisi (2004) proposed pivoted dynamic programming, which attracts the optimal disparity path along a scanline towards the prior disparity at matched keypoints. All these methods process data on a per-frame basis, resulting in temporally inconsistent prediction. In the previous chapter, we have shown that pivots improve temporal consistency of stereo algorithms, however we assumed a user in the loop.

Other approaches focus on inpainting (Herrera *et al.*, 2013; Payen de La Garanderie and Breckon, 2014) of depth maps from active sensors, however, the captured depth maps have to be fairly dense. Diebel and Thrun (2005) proposed an MRF model for upsampling of laser measurements with enforced smoothness across areas with constant color. Though their method uses both the laser and color data, they do not use planarity prior as we do in the proposed method. Further, they do not consider motion while processing depth data. In experiments, we show that this is necessary to achieve high accuracy and efficiency. Dolson *et al.* (2010) proposed a filtering framework for dynamic scenes. Badino *et al.* (2011) showed how to integrate sparse lidar measurement directly into disparity estimation. Though this method also tries to solve a problem similar to ours, there are some key differences which are necessary for achieving high accuracy efficiently. First we propose to use a region based slanted-plane prior which is necessary to model planarity over large regions such as road, table-tops *etc.* Further we solve the energy minimization problem in a mean-field framework which can be run fully in parallel compared to the dynamic programming based method of Badino *et al.* (2011).

On application side, Munoz *et al.* (2012) proposed 2D-3D (camera-lidar) co-inference for semantic segmentation. Held *et al.* (2013) combined lidar and camera for object tracking and Premebida *et al.* (2014) for pedestrian detection. Recently, Arnab *et al.* (2015) used audio-visual cues for interactive semantic segmentation.

5.3 Dense Multi-modal Depth-Map Estimation

Our system exploits partial prior knowledge about the scene provided by sparse but accurate 3D measurements, called *pivots*. Hence, the first step is to project the lidar measurements into the camera coordinate system. Since lidars often have smaller field-of-view than cameras, we augment these points by robustly matched keypoints. Next, we use dense matching and slanted-plane prior to evaluate the potentials for the CRF model. The following subsections assume synchronized data and process them per-frame. We relax this assumption in §5.3.7.

5.3.1 Setting the stage

In our setup, we assume that all sensors are calibrated. In case of cameras, this comprises: 1) intrinsic camera calibration to compute the geometric parameters of each camera lens (focal length, principal point, radial and tangential distortion); 2) stereo calibration to compute

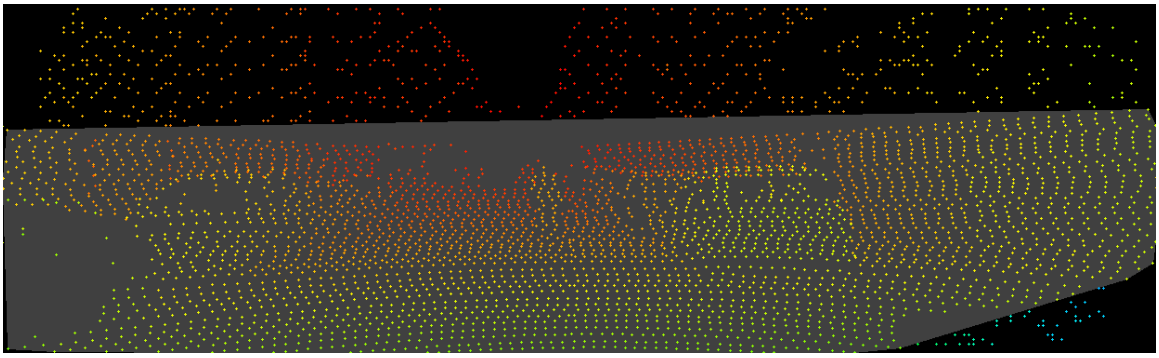


Figure 5.3: Pivots – gray area contains lidar measurements, outside this region we perform sparse feature matching.

the geometric relationship between the two cameras, expressed as a rotation matrix and a translation vector; 3) stereo rectification to correct the camera image planes such that they are scanline-aligned and disparity computation is simplified. Without loss of generality, the reference camera coordinate system has origin in the top-left corner of the left camera (consult (Hartley and Zisserman, 2003) for more details).

The laser scanner is registered with respect to the reference camera coordinate system. In this section, we also assume the cameras and laser scanner are synchronized and data are "untwisted" in case of spinning lidars. The optimization is carried out in the disparity image space with standard conversion to depth through disparity-to-depth mapping.

5.3.2 Pivots

In order to disambiguate dense matching, we first define a set $\mathcal{P}$ of accurate 3D points capturing partial prior knowledge about the scene, so called *pivots*. Each pivot $p = (x_p, y_p, d_p)$ is represented by coordinates $(x_p, y_p) \in \mathbb{N}^2$ and disparity $d_p \in \mathbb{N}$ defining the displacement of the corresponding matching point along the epipolar line in the right image. We assume that each pivot p is associated with its own uncertainty.

A natural choice for pivots is the set of all lidar measurements projected into the image plane. However, Lidars often have smaller field-of-view than cameras (Fig. 5.1 (b)) and do not return any measurement due to reflections or for objects located past the maximum range limit. Hence, we augment the set of Lidar pivots by a robustly matched keypoints. See Section §5.4.1 for implementation details.

5.3.3 Model

We define a random field over random variables $\mathcal{X} = \{X_1 \dots X_N\}$, conditioned on data $\mathcal{I} = \{\mathbf{I}^{(l)}, \mathbf{I}^{(r)}, \mathcal{P}\}$ consisting of a pair of 2D images $\mathbf{I}^{(l)}, \mathbf{I}^{(r)}$ and pivots $\mathcal{P}$. We assume that each discrete random variable X_i is associated with a pixel $i \in \mathcal{N} = \{1...N\}$ in the image of the reference camera (left) and takes a label $d \in \mathbb{N}$ from an ordered finite disparity label set $\mathcal{D} = \{d_1, \dots, d_D, d_{D+1}\}$. A dummy label d_{D+1} with some constant cost indicates invalid depth (outliers/occlusions). We formulate the problem of assigning disparity labels to the pixels as one of solving a densely-connected, pairwise Conditional Random Field (CRF)

$$\begin{aligned} P(\mathbf{x}|\mathcal{I}) &= \frac{1}{Z(\mathcal{I})} \exp(-E(\mathbf{x}|\mathcal{I})) \\ E(\mathbf{x}|\mathcal{I}) &= \sum_{i \in \mathcal{N}} \psi_u(X_i) + \sum_{i<j} \psi_p(X_i, X_j), \end{aligned} \tag{5.1}$$

in which $E(\mathbf{x}|\mathcal{I})$ is the energy associated with a configuration $\mathbf{x} = (X_1 \dots X_N)$, conditioned on the data $\mathcal{I}$, $Z(\mathcal{I}) = \sum_{\mathbf{x}'} \exp(-E(\mathbf{x}'|\mathcal{I}))$ is the (data-dependent) partition function and $\psi_u(\cdot)$ and $\psi_p(\cdot,\cdot)$ are the unary and pairwise potential functions, respectively, both implicitly conditioned on the data $\mathcal{I}$. This model is not constrained to our particular application and can be extended to, *e.g.* joint depth prediction and semantic or motion segmentation.

5.3.4 Unary Potential Function

Our unary potential function is inspired by guided dense stereo matching proposed by Torr and Criminisi (2004) and large-scale stereo estimation algorithm of Geiger *et al.* (2010). It consists of two terms, i) feature matching and ii) piecewise-planar term that we included directly into the unary potential function.

Let $\mathbf{f}_i \in \mathbb{R}^R$ be an image dependent feature vector (pixel intensity or patch descriptor) for pixel $i = (x_i, y_i) \in \mathbb{N}^2$ and superscripts $^{(l)}$,$^{(r)}$ denote left and right images, respectively.

Feature Matching

We express the contribution of the data term as a constrained Laplace distribution capturing cost for one dimensional dense feature matching along the epipolar line

$$\psi^d(\cdot|X_i = d, \mathbf{f}) \propto \begin{cases} \exp\left(-\beta \|\mathbf{f}_i^{(l)} - \mathbf{f}_{i-d}^{(r)}\|_1\right), & \forall d \in \bar{\mathcal{D}}_i \\ \infty, & \text{otherwise} \end{cases} \tag{5.2}$$

where $\mathbf{f}_i$ are features for pixels i and $\bar{\mathcal{D}}_i \subseteq \mathcal{D}$ (defined below) is a subset of disparities that reduces the searched range and implicitly enforces the epipolar constraint.

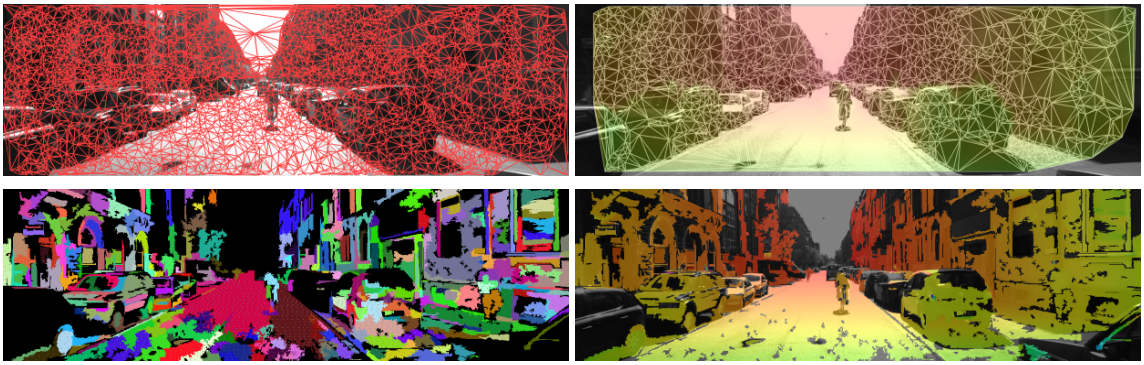


Figure 5.4: Piecewise planar prior defined on regions obtained with Delaunay triangulation (top) and multiscale over-segmentation, here we show the 3rd level (bottom).

Piecewise-planar term

Pivots $\mathcal{P}$ provide partial knowledge about the scene. Hence, we use them to define the prior proportional to a sampled Gaussian

$$\psi^p(\cdot|X_i = d, \mathcal{P}) \propto \begin{cases} \exp\left(-\frac{[d-\mu(\tau_t,i)]^2)}{2\sigma^2}\right), & \text{if } d \in \bar{\mathcal{D}}_i \\ \infty, & \text{otherwise} \end{cases} \tag{5.3}$$

where $\sigma \in \{\sigma_l, \sigma_k, \sigma_{lk}\}$ are hyper-parameters set by cross-validation determining our belief in plane τ defined by lidar measurements (σ_l), robustly matched keypoints (σ_k) or both (σ_{lk}). Here $\bar{\mathcal{D}}_i = \{|d - \mu(\cdot)| < 3\sigma \vee d \in N_P\}$ is a subset of disparity levels for which the equation is evaluated. We evaluate only disparities within 3σ from the mean to gain speed. The condition $d \in N_P$ enables the prior to locally extend its range to better handle disparity discontinuities in places where the linearity assumption might be violated (N_P is a set of all support point disparities in a small neighbourhood). We define $\mu(\cdot)$ to be a piecewise linear function

$$\mu(\tau_t, i) = a_t x_i + b_t y_i + c_t \tag{5.4}$$

interpolating subsets $\mathcal{T} = \{\tau_1, ..., \tau_T\}$ of pivots $\mathcal{P}$ partitioned in two different ways:

Delaunay triangulation. We partition the set of pivots $\mathcal{P}$ into a set of non-overlapping triangles $\mathcal{T}_D \subseteq \mathcal{T}$, *i.e.* $\cup_{t \in \mathcal{T}_D} \tau_t = \mathcal{P}$. Thus, this partitioning captures a coarse estimate of a 3D structure. For each triangle τ_t, we form a linear system of equations and solve for the plane parameters $\{a_t, b_t, c_t\}$ by SVD. The mode μ of the proposed distribution is a linear combination of the pivots in triangle τ_t.

Oversegmentation. The Delaunay triangulation partitions pivots into non-overlapping triangles, however, such triangles may cover multiple objects. Also, if the pivots are imprecise (which often happens with real-world measurements), such prior may result into *e.g.* non-coplanar neighbouring planes on a flat surface. We overcome both issues by partitioning pivots $\mathcal{P}$ into sets $\mathcal{T}_O \subseteq \mathcal{T}$ defined by object-aware segments – these are often sensitive to potential object boundaries and often contain many pivots, hence "regularize" priors defined by non-overlapping triangles.

A natural question is how to define grouping of image pixels. In contrast to object recognition, even if we had a method that could perfectly segment the objects from each other, it would not be enough for disparity estimation, since a single object often consists of many shapes/parts. Hence we use a multi-scale over-segmentation (details in sec. 5.4.1) to define such regions and RANSAC with least squares refinement to robustly fit a plane (*i.e.* estimate $\{a_t, b_t, c_t\}$) into a subset of pivots associated with each segment $\tau_t \in \mathcal{T}_O$.

Unary potential function. Combining feature matching term (*Eq.* 5.2) and piecewise-planar term (*Eq.* 5.3) together, taking the negative logarithm and introducing a "discount" function Ω for pivots yields

$$\psi_u(\cdot) = \Omega_i \left[\beta \|\mathbf{f}_i^{(l)} - \mathbf{f}_{i-d}^{(r)}\|_1 + \sum_{\tau_t \subseteq \mathbb{I}[i \in \mathcal{T}]} \frac{[d - \mu(\tau_t, i)]^2}{2\sigma^2} \right] \tag{5.5}$$

where $\mathbb{I}[\cdot]$ is an indicator function returning all subsets $\tau_t \subseteq \mathcal{T}$ that contain pixel (x_i, y_i), and discount function

$$\Omega_i = \begin{cases} \omega, & \text{if } (x_i, y_i, d) = p \in \mathcal{P} \\ 1, & \text{otherwise} \end{cases} \tag{5.6}$$

drastically reduces the cost of configurations assigning measured depth at pivots $p \in \mathcal{P}$ by some constant ω. Using different constants ω for pivots obtained by lidar (ω_l) and robust keypoint matching (ω_k) allows us to model our belief into precision of these measurements.

Note, that we do not introduce any hard constraints forcing variables at pivots' coordinates to take the measured disparity. Hence this leaves the chance for recovery if pivot has assigned incorrect disparity. Also, the piecewise planar term can be replaced by a set of functions with Minimum Description Length (MDL) prior to better model non-planar surfaces such as conics, *etc.*

5.3.5 Pairwise Potentials Function

The pairwise potential function $\psi_p(\cdot,\cdot)$ enforces consistency over pairs of random variables and thus generally leads to a smooth output. In our application, we use a weighted mixture of Gaussian kernels (with unit covariance matrix) that depend on appearance features

$$\psi_p(d,d') = \Delta(d,d') \sum_{m=1}^{M} w^{(m)} k^{(m)}(\bar{\mathbf{f}}_i^{(m)}, \bar{\mathbf{f}}_j^{(m)}) \tag{5.7}$$

where weights $w^{(m)}$ associated with m-th kernel are obtained by cross-validation, $\bar{\mathbf{f}}_i^{(m)}, \bar{\mathbf{f}}_j^{(m)}$ are 2D features extracted from image data $\mathbf{I}^{(l)}$ at the ith and j^{th} pixels (respectively) and $\Delta(d,d')$ is the compatibility function. We use a combination of the Gaussian kernel

$$k^{(1)}(\bar{\mathbf{f}}_i^{(1)}, \bar{\mathbf{f}}_j^{(1)}) = w^{(1)} \exp\left(-\frac{\|\mathbf{c}_i - \mathbf{c}_j\|_2^2}{2\theta_\gamma}\right) \tag{5.8}$$

removing small isolated areas and bilateral kernel

$$k^{(2)}(\bar{\mathbf{f}}_i^{(2)}, \bar{\mathbf{f}}_j^{(2)}) = w^{(2)} \exp\left(-\frac{\|\mathbf{c}_i - \mathbf{c}_j\|_2^2}{2\theta_\alpha} - \frac{\|\mathbf{I}_i^{(l)} - \mathbf{I}_j^{(l)}\|_2^2}{2\theta_\beta}\right) \tag{5.9}$$

enforcing neighbouring pixels with similar appearance to take the same label. Parameters θ controls the spatial extent of the kernel, $\mathbf{c}_i = (x_i, y_i)$ are pixel cordinates and $\mathbf{I}$ is color. This form of potential introduces a small fronto-parallel bias, which can be overcome by higher-order potentials. We decided not to use higher-orders, as it would make the inference slower; instead we directly included the "slanted" areas prior directly into our unary potentials. We use the standard truncated L_1 or L_2 compatibility functions

$$\Delta(d,d') = \min(\|d - d'\|_\Gamma, \alpha) \tag{5.10}$$

where $\|\cdot\|_\Gamma$ is the L_1 or L_2 norm, respectively, and α is the clipping parameter.

5.3.6 Efficient inference

Similarly to the previous chapter (§3.5.2), we again use efficient mean-field inference. Dense long-range interactions are in particular attractive for tasks with a small number of labels and a constant label over large areas (*e.g.* object segmentation). However, for disparity estimation, we often have a large state space and neighbouring pixels tend to take different labels (typically slanted areas). Hence we further exploit partial prior knowledge about the scene, and evaluate the pairwise updates only for labels within the range defined by prior (*i.e.* states with evaluated unary potential) plus some small slack λ_s (*e.g.* 5 disparity labels) allowing to handle imprecise pivots, *i.e.* $d' \in (\mathcal{D} \cup \lambda_s)$. The algorithm is inherently parallel,

runs for a fixed number of iterations, and the MPM solution is extracted by choosing $x_i \in \operatorname{argmax}_d Q_i(x_i = d)$ from soft predictions at the final iteration.

5.3.7 Temporal Sequences of Images

Often, robotic platforms perceive a gradually changing scene with multiple sensors operating at different rates (*e.g.* cameras at 25Hz, Lidar at 15Hz). So far, our system has required synchronized sensors and processed only the latest batch of data. Discarding all previous measurements results in temporally inconsistent predictions (even for static scenes due to noise) and need for all sensors to operate at rate of the slowest sensor.

In the previous chapter, we have shown how maintaining pivots over the temporal sequences stabilizes the predicted disparity. To this end, we replace per-frame keypoint matching by more robust temporal matching, *i.e.* the per-frame robustly matched features are propagated over time with a mutual exclusive check, and project both the Lidar readings and matched keypoints, into a common map. Consequently, all the measurements are available to the algorithm on a request and we do not discard any pivots. The only assumption is that the 6DoF pose is available. Our map is represented by a sparse hash-table-driven data structure that ignores unoccupied space. Further, we process only the data within a current frustum (Nießner *et al.*, 2013; Vineet *et al.*, 2015). This results in the more stable set of keypoints over the temporal sequence of images.

5.4 Experiments

5.4.1 Implementation details

Pivots from different modalities can be defined and modeled in numerous ways. Our implementation relies on a simple yet reasonable assumption that lidar measurements are generally more accurate than feature matching. Hence, we perform sparse feature matching only in areas that are not covered by lidar measurements (such areas are discovered by simple dilation of lidar measurements). Though a variety of fast feature detectors and descriptors has been proposed (Miksik and Mikolajczyk, 2012), we follow (Geiger *et al.*, 2010) who showed that matching points sampled on a regular grid using the L_1 distance between the descriptors consisting of concatenated horizontal and vertical Sobel responses is both, fast and stable. To impose no restrictions on the disparities, we allow a large disparity 1D search range along the epipolar line. Non-stable keypoints are eliminated by mutual exclusive check (Nister *et al.*, 2004b) and the best to the second best match ratio. We

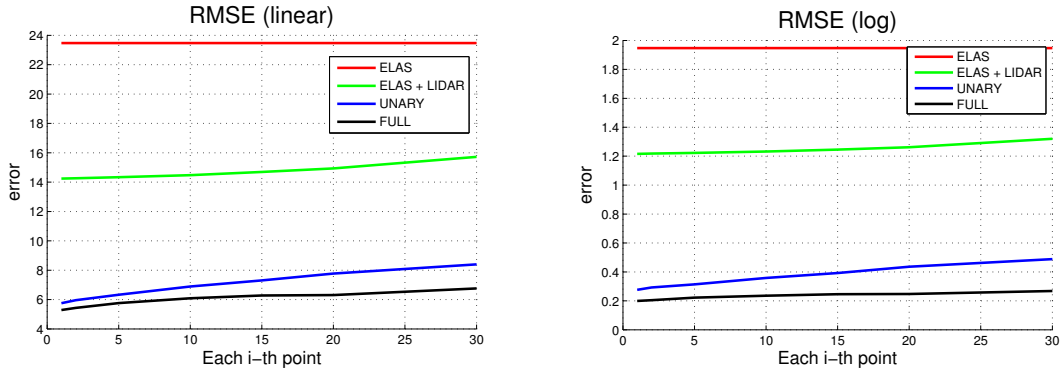


Figure 5.5: Quantitative results: RMSE linear (top), RMSE log (bottom). See text for details.

also remove all keypoints which exhibit disparity values dissimilar from all surrounding support points. For videos, we use the Fovis visual odometry library (Huang *et al.*, 2011) to estimate 6DoF pose and per-frame feature matching (for pivots) is replaced by features tracked by Fovis to increase temporal consistency and robustness.

In principle, our framework can be used with any super-pixel grouping algorithm (k-means, SLIC, ...). Our implementation uses multi-scale (4 levels) oversegmentation by Felzenszwalb and Huttenlocher (Felzenswalb and Huttenlocher, 2004) since it is fast and it is easy to control size of segments.

5.4.2 Dataset and baselines

We demonstrate the effectiveness of our approach on the KITTI dataset (Geiger *et al.*, 2012), which contains a variety of outdoor sequences (city, road, campus). All sequences were captured at a resolution of 1241×376 pixels using stereo cameras (with baseline 0.54m) mounted on the roof of a car. The cameras were calibrated and captured images rectified. The car was also equipped with a spinning Velodyne HDL-64E laser scanner (LIDAR). All sensors were synchronized, the dataset was captured at 10Hz and cameras triggered when lidar was rotated forward.

The KITTI dataset is very challenging. It contains numerous changes in lighting conditions resulting in textureless areas, repetitive patterns (road, facades, ...), *etc.* We report both, qualitative and quantitative results and show substantial improvement with respect to our baselines. The first baseline is the disparity matching algorithm (from passive stereo cameras) by Geiger *et al.* (2010) since part of our unary potentials follow this approach. Obviously, comparison with respect to the algorithm relying purely on data from cameras is not fair as this baseline use less data. Hence, the second baseline is a modified version that uses exactly the same set of support points as our approach.

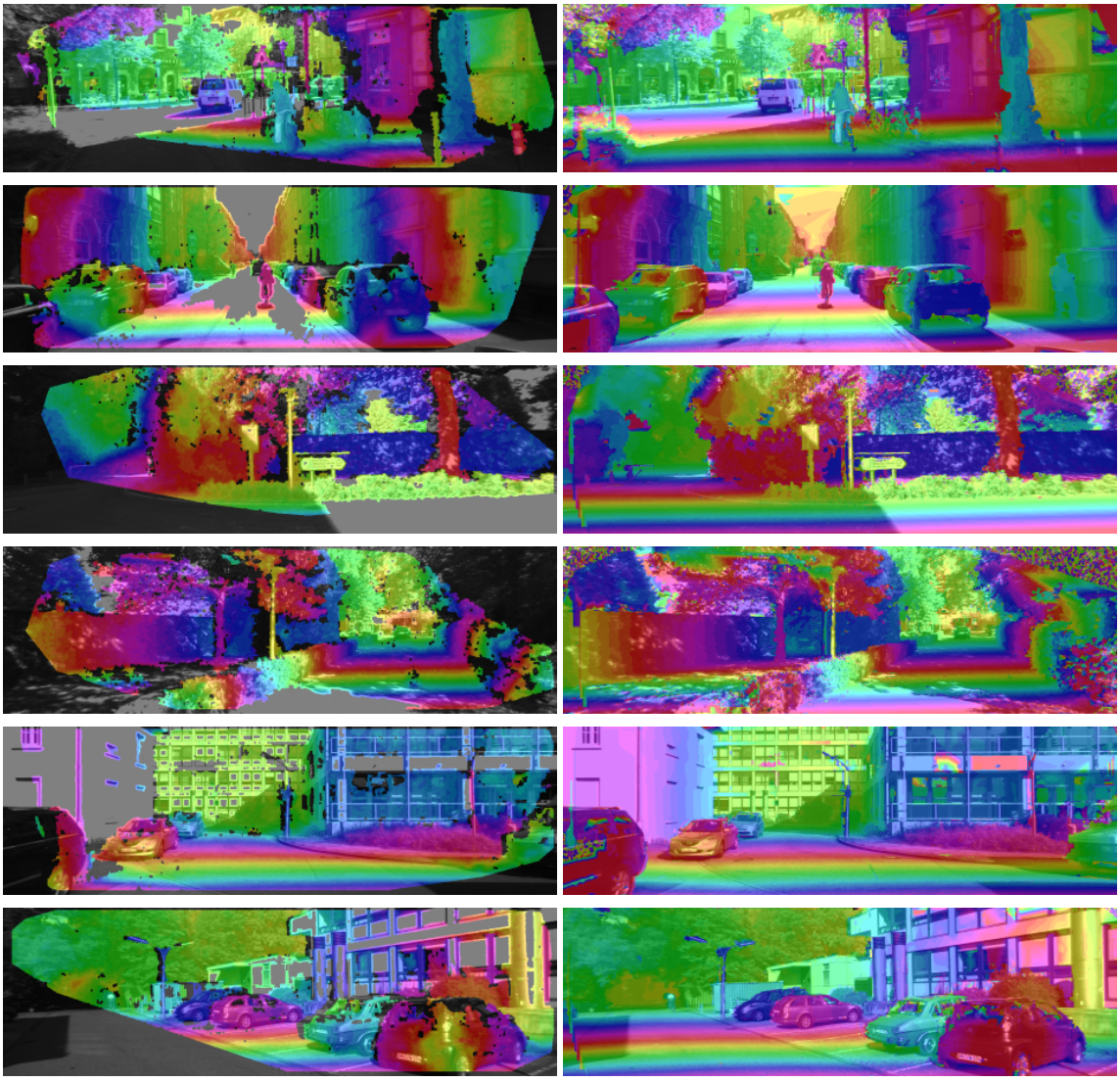


Figure 5.6: Qualitative results - left: Geiger *et al.* (2010), right: proposed. Cyclic colormap to enhance details.

5.4.3 Qualitative results

First, we show qualitative results for our algorithm. In Fig. 5.6, we highlight the ability of our approach not only to estimate disparity in saturated zones (*e.g.* filled holes in disparity images), but also to improve accuracy in areas with repetitive patterns (road surfaces under the cars, *etc.*) and also to accurately recover thinner objects such as walking pedestrians. Note in particular that with lidar data, and segment-based prior, the discontinuity in depth better follows the object boundaries.

5.4.4 Quantitative results

Next, we quantitatively evaluate the accuracy. We assess the overall performance by linear and logarithmic root mean square error (RMSE) that are standard metrics defined as $\text{RMSE}_{\text{linear}} = \sqrt{\frac{1}{N}\sum_{i\in N}\|d_i - d_i^*\|^2}$ and $\text{RMSE}_{\log} = \sqrt{\frac{1}{N}\sum_{i\in N}\|\log d_i - \log d_i^*\|^2}$, where d_i is the predicted disparity and d_i^* is the ground-truth. In spirit of disparity evaluation on the KITTI dataset, we use the lidar measurements as a ground-truth (as we do not have any other, more accurate and dense data). It is natural, that our approach performs well in these point measurements. However, our goal is to demonstrate that competitive performance can be achieved with worse sensors. Hence we reduce the number of lidar measurements that we use as pivots, *i.e.* we use each 2nd, 5th, 10th, *etc.* point and evaluate with respect to the unused points. Our approach significantly outperforms both baselines (elas (Geiger *et al.*, 2010), elas+lidar) and inference helps to get better results (unary vs. full), see Fig. 5.5 (x axis denotes how many points we preserve from lidar measurements, *e.g.* 10 means that we keep each 10th point). The error increases very slowly, which suggests that even with significantly worse sensors we are able to maintain the desired precision – 18000 lidar measurements can be decreased to only 900 points without significant drop in performance.

5.4.5 Limitations

Despite very encouraging results, our system is not without limitations. In particular, processing temporal sequences assumes the mapped pivots correspond to the static parts of a scene. Though we have not included it into our system, the pivots corresponding to moving objects can be marked by motion or semantic segmentation (Vineet *et al.*, 2015) (which can potentially be included into our energy function) and excluded from mapping. Also, the quality of estimated depth maps on temporal sequences depends on accuracy of estimated camera poses, however, this is not a limitation in practice as we anyway need accurate pose for 3D reconstruction.

For ease of exposition, we have not used any probabilistic model of lidar and/or camera taking sensor noise, resolution, *etc.* into account, however, both can be included into our energy function.

5.5 Conclusion

In this paper, we have proposed a probabilistic model that efficiently exploits complementarity between different depth-sensing modalities for online dense scene reconstruction.

Our model uses planarity prior which is common in both the indoor and outdoor scenes. We demonstrated the effectiveness of our approach on the KITTI dataset, and provide qualitative and quantitative results showing high-quality dense reconstruction and labeling of a number of scenes. More importantly, we show that we are able to get very high quality reconstruction using colour data and only a few hundreds of lidar points. We are planning to incorporate higher order terms to enforce slanted planarity priors as part of future work.

Seeing it from perspective of 2017. Multi-modal Auto-Encoders (conditioned on pivots) provide an end-to-end trainable alternative (Cadena *et al.*, 2016).

6 Coarse-to-fine Regularization for Dense Monocular 3D Reconstruction

So far, we have assumed a synchronized and calibrated stereo camera. Although a calibrated stereo camera has become a commodity sensor, widely available even in mobile phones nowadays, it still represents a major limitation in many situations. For instance, stereo cannot be used with legacy video footage recorded by a single camera. Similarly, using this setup for long range estimation, where stereo baselines are negligible is somewhat problematic. In this chapter, we relax the assumption of a calibrated stereo camera and focus on dense monocular reconstruction instead.

Simultaneous localization and mapping (SLAM) using the whole image data is an appealing framework to address shortcoming of sparse feature-based methods – in particular frequent failures in textureless environments. Hence, direct methods bypassing the need of feature extraction and matching became recently popular. Many of these methods operate by alternating between pose estimation and computing (semi-)dense depth maps, and are therefore not fully exploiting the advantages of joint optimization with respect to depth and pose. In this work, we propose a framework for monocular SLAM, and its local model in particular, which optimizes simultaneously over depth and pose. In addition to a planarity enforcing smoothness regularizer for the depth, we also constrain the complexity of depth map updates, which provides a natural way to avoid poor local minima and reduces unknowns in the optimization. Starting from a holistic objective we develop a method suitable for online and real-time monocular SLAM. We evaluate our method quantitatively in pose and depth on the TUM dataset, and qualitatively on our own video sequences.

6.1 Introduction

Simultaneous localization and mapping (SLAM) aims to produce trajectory estimations and a 3D reconstruction of the environment in real-time. In modern technology, its application ranges from autonomous driving, navigation and robotics to interactive learning, gaming and enhanced reality (Geiger *et al.*, 2012; Forster *et al.*, 2014; Engel *et al.*, 2014b; Schöps *et al.*, 2014; Miksik *et al.*, 2015b; Vineet *et al.*, 2015; Barfield, 2016). As we have already discussed in Section §2.3, SLAM typically comprises two key components: i) a local model, which generates fast initial odometry measurements (which often includes a local 3D reconstruction – *e.g.* a depth map – as byproduct), and ii) a global model, which performs loop closures and pose refinement *via* large scale sub-real-time bundle adjustment. In our work, we focus on the former, and propose a new strategy for local monocular odometry and depth map estimation.

Estimating the 3D position of tracked landmarks is a key ingredient in any SLAM system, since it directly allows for the poses to be computed w.r.t. a common coordinate frame. Historically, visual landmarks are induced by matched sparse keypoints, but there is a recent trend to utilize a dense (or semi-dense) set of points (leading to a dense or semi-dense reconstructions) (Stühmer *et al.*, 2010; Newcombe *et al.*, 2011b; Engel *et al.*, 2014a).

As we have seen in previous chapter, another trend is the inclusion of different sensing modalities for depth estimation. Often, methods exploit (a combination of) alternative sensors, such as infrared, lidar and stereo camera setups, which natively provide fairly accurate depth data (Newcombe *et al.*, 2011a; Salas-Moreno *et al.*, 2014; Yamaguchi *et al.*, 2014; Miksik *et al.*, 2015a). Such algorithms are quite advanced and are often employed even in consumer technology where hardware is controllable. Visual SLAM with only monocular camera streams is less common and still challenging in literature (Davison, 2003; Nister *et al.*, 2004a; Davison *et al.*, 2007; Klein and Murray, 2007; Newcombe *et al.*, 2011b; Wendel *et al.*, 2012; Pradeep *et al.*, 2013; Engel *et al.*, 2014a; Concha and Civera, 2015; Tarrio and Pedre, 2015). Nonetheless, the monocular setup is very suitable for (1) long range estimations, where stereo baselines are negligible, (2) light weight mobile and wearable devices aiming for a minimal amount of sensors to reduce weight and power consumption, and (3) legacy video footage recorded by a single camera.

Classical approaches for monocular visual SLAM are based on sparse keypoint tracking and mapping (Davison, 2003; Davison *et al.*, 2007; Klein and Murray, 2007), which produces a feature-based sparse depth hypothesis. A number of methods have since been proposed

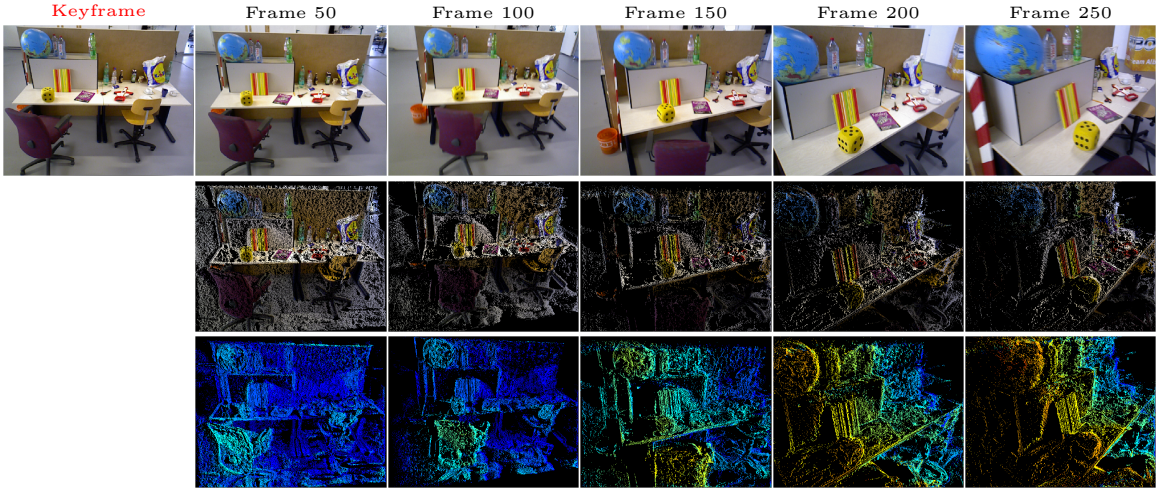


Figure 6.1: During keyframe-to-frame comparison a dense depth map is built. Image, point cloud and depth (top to bottom) are shown as they develop, for selected frames from a *single* keyframe. (While depth is dense at the keyframe, its projection may not be.)

which essentially alternate between tracking (and pose computation) and dense depth map estimation: Most prominently, (Newcombe *et al.*, 2011b) presents dense tracking and mapping (DTAM) which generates a dense depth map on a GPU. Similarly, (Wendel *et al.*, 2012; Pradeep *et al.*, 2013; Concha and Civera, 2015) provide dense depth maps, but like (Newcombe *et al.*, 2011b) also rely heavily on GPU acceleration for real-time performance. In contrast to these methods large-scale direct SLAM (LSD-SLAM) (Engel *et al.*, 2014a) focusses the computation budget on a semi-dense subset of pixels and has therefore attractive running-times, even when run on CPU or mobile devices. As a direct method, it computes the odometry measurements directly from image data without an intermediate representation such as feature tracks. Depth is then computed in a separate thread with small time delay. Note that all these methods employ an alternation strategy: odometry is computed with the depth map held fixed, and the depth map is updated with fixed pose estimates. In contrast, we propose joined estimation of depth and pose within a single optimization framework. Our method introduces only a minimal additional computational cost compared to that of the tracking thread of LSD-SLAM and is able to find structure and motion twice as fast as the whole LSD-SLAM.

6.1.1 Contributions

In this work, we present a local SLAM front-end which estimates pose and depth simultaneously in real-time (Fig. 6.1). We revisit traditional setups, and propose inverse depth estimation with a coarse-to-fine planar regularizer that gradually increases the complexity of the algorithm's depth perception. Note, that many systems for stereo vision or depth sensors incorporate local or global planar regularization (Salas-Moreno *et al.*, 2014; Yamaguchi *et al.*, 2014; Geiger *et al.*, 2010; Sinha *et al.*, 2014; Zhang *et al.*, 2015). Similarly, we employ global planar constraints into our monocular setup, and enforce local smoothness by representing each pixel as lying on a plane that is similar to its neighbours'. Furthermore, similarly to many algorithms in stereo (Geiger *et al.*, 2010; Miksik *et al.*, 2015a), we reduce depth complexity *via* discretization, in our case through planar splitting techniques which (in the spirit of graphical methods) create labels "on demand". In summary,

1. We formulate a global energy for planar regularized inverse depth that is optimized iteratively at each frame.

2. We revisit depth and pose optimization normally considered separately, and introduce a coarse-to-fine strategy that refines both truly simultaneously.

3. We establish our method as semi-dense, and find pose *and* depth twice as fast as LSD-SLAM, by adding minimal cost to LSD-SLAM's tracking thread.

4. We evaluate pose and depth quantitatively on the TUM dataset.

Closely related to our work is (Becker *et al.*, 2011), where depth and pose are optimized simultaneously given the optical flow of two consecutive images. This approach is based on image pairs. Our method considers video input and incrementally improves its belief. In (Concha *et al.*, 2015; Salas *et al.*, 2015) planarity is proposed in conjunction with scene priors, previously learned from data, and (Concha and Civera, 2015) presents a hole-filling strategy for semi-dense monocular SLAM. While these methods are real-time, they rely on keypoints at image corners or gradients, which are later enriched with a planar refinement. Importantly however, such methods fail in featureless environments. Finally, we emphasize DTAM (Newcombe *et al.*, 2011b) performs batch operations on a set of images taken from a narrow field of view, and henceforth introduces a fixed lag before depth is perceived by the system. As this is often unacceptable for robotics setups, our method updates depth incrementally after *each* frame.

6.2 Proposed Energy for Monocular Depth Estimation

We formulate our energy function for poses and depth w.r.t. the photometric error over time. Similarly to LSD-SLAM, we employ a keyframe-to-frame comparison to estimate camera displacement and each pixels' depth in the reference image. Let us denote the keyframe as I and its immediately succeeding images as $(I_t)_{t=1}^{T}$. The tuple of valid pixel locations on the keyframe's plane is represented by $\mathcal{X} = (\mathbf{x}_i)_{i=1}^{|\mathcal{X}|}$ in *normalized* homogeneous coordinates (*i.e.* $z_i = 1$), and their corresponding *inverse* depth values are expressed by $\mathcal{D} = (d_i)_{i=1}^{|\mathcal{X}|}$. Since we aim to model planar surfaces, we use an over-parametrization given by $\mathcal{S} = (\mathbf{s}_i^{\mathrm{T}})_{i=1}^{|\mathcal{X}|} \cong \mathbb{R}^{3|\mathcal{X}|}$, where $\mathbf{s}_i = (u_i, v_i, w_i)^{\mathrm{T}}$ are planes with disparity gradients u_i, v_i, and inverse depth at 0, w_i. Hence, the relation $d_i = \mathbf{s}_i^{\mathrm{T}}\mathbf{x}_i$ holds.

Tuple $\Xi = (\xi_t)_{t=1}^{T}$ denotes the changes in camera pose, where $\xi_t \in SE(3)$ is composed of rotation $\mathbf{R}_t \in SO(3) \subset \mathbb{R}^{3\times 3}$ and translation $\mathbf{t}_t \in \mathbb{R}^3$ between the keyframe I and frame I_t. In principle, the complete cost function should incorporate all available images associated with the current keyframe and optimize over the depth and all poses jointly,

$$\hat{E}_{Total}\left(\mathcal{S}, \Xi\right) = \sum_{t=1}^{T} E_{Match}^{(t)}(\mathcal{S}, \xi_t) + E_{Smooth}(\mathcal{S}). \tag{6.1}$$

Here $E_{Match}^{(t)}$ and E_{Smooth} are energy terms related to image-based matching costs and spatial smoothing assumptions, respectively. Before we describe these terms in more detail in subsequent sections, we modify $\hat{E}_{Total}$ to be more suitable for an incremental online approach. This is advisable since, the objective $\hat{E}_{Total}$ involves the complete history of all frames I_t mapped to the current keyframe I. Intuitively the optimization of the poses $(\xi_t)_{t=1}^{T-1}$ is no longer relevant at time T, as only the current pose ξ_T and $\mathcal{S}$ is required. Analytically, we introduce

$$E_{History}^{(T)}\left(\mathcal{S}\right) := \min_{(\xi_t)_{t=1}^{T-1}} \sum_{t=1}^{T-1} E_{Match}^{(t)}(\mathcal{S}, \xi_t) \tag{6.2}$$

where $(\xi_t)_{t=1}^{T-1}$ is the tuple of poses, minimized in previous frames. By splitting the first term in (6.1), the energy becomes

$$\hat{E}_{Total}\left(\mathcal{S}, \Xi\right) = E_{History}^{(T)}\left(\mathcal{S}\right) + E_{Match}^{(T)}(\mathcal{S}, \xi_T) + E_{Smooth}(\mathcal{S}). \tag{6.3}$$

Now we replace $E_{History}^{(T)}$ with its second order expansion

$$\left(\mathcal{S}^*, \xi_1^*, \ldots, \xi_{T-1}^*\right) = \operatorname*{argmin}_{\mathcal{S}, (\xi_t)_{t=1}^{T-1}} \sum_{t=1}^{T-1} E_{Match}^{(t)}(\mathcal{S}, \xi_t), \tag{6.4}$$

and thus we obtain an approximation of $E^{(T)}_{History}(\mathcal{S})$, denoted $E^{(T)}_{Temporal}(\mathcal{S})$:

$$\begin{aligned} E^{(T)}_{Temporal}(\mathcal{S}) &:= E^{(T)}_{History}(\mathcal{S}^*) + \left(\nabla_{\mathcal{S}} E^{(T)}_{History}(\mathcal{S}^*)\right)^{\mathrm{T}} (\mathcal{S} - \mathcal{S}^*) \\ &\quad + \frac{1}{2}(\mathcal{S} - \mathcal{S}^*)^{\mathrm{T}} \left(\nabla^2_{\mathcal{S}} E^{(T)}_{History}(\mathcal{S}^*)\right)(\mathcal{S} - \mathcal{S}^*) \\ &= E^{(T)}_{History}(\mathcal{S}^*) + \frac{1}{2}(\mathcal{S} - \mathcal{S}^*)^{\mathrm{T}} \left(\nabla^2_{\mathcal{S}} E^{(T)}_{History}(\mathcal{S}^*)\right)(\mathcal{S} - \mathcal{S}^*) \end{aligned} \tag{6.5}$$

As $\mathcal{S}^*$ is a local minimizer of $E^{(T)}_{History}$, $\nabla_{\mathcal{S}} E^{(T)}_{History}(\mathcal{S}^*) = 0$. Furthermore, as our choice of terms leads to a nonlinear least-squares formulation, $\nabla^2_{\mathcal{S}} E^{(T)}_{History}(\mathcal{S}^*)$ is computed using the Gauss-Newton approximation. Finally, since $E^{(T)}_{History}$ jointly optimizes the inverse depths (in terms of its over-parametrization $\mathcal{S}$) and (internally) the poses, but $E^{(T)}_{Temporal}$ is solely a function of $\mathcal{S}$, we employ the Schur complement to factor out the poses $(\xi_t)_{t=1}^{T-1}$. However, as the poses link the entire depth map, the Schur complement matrix will be dense. We obtain a tractable approximation by using its block-diagonal consisting of 3×3 blocks (corresponding to $\mathbf{s}_i = (u_i, v_i, w_i)^{\mathrm{T}}$).[1] The resulting objective at time T is therefore

$$E^{(T)}_{Total}(\mathcal{S}, \xi_T) = E^{(T)}_{Temporal}(\mathcal{S}) + E^{(T)}_{Match}(\mathcal{S}, \xi_T) + E_{Smooth}(\mathcal{S}). \tag{6.6}$$

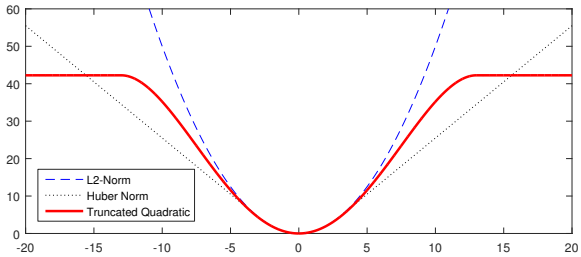
There is a clear connection between $E^{(T)}_{Total}$, extended Kalman filtering and maximum likelihood estimation. If $E^{(T)}_{History}$ is interpreted as log-likelihood, then $\left(\mathcal{S}^*, (\xi^*_t)_{t=1}^{T-1}\right)$ is an asymptotically normal maximum likelihood estimate with the Hessian as (approximate) inverse covariance (*i.e.* precision) matrix. The Schur complement factoring out the poses corresponds to marginalizing over the poses according to their uncertainty (in the energy-minimization perspective). $E^{(T)}_{Total}$ can be read as a probabilistic fusion of past and current observation, but this correspondence is limited, since we are searching for MAP estimates and not posteriors. In the following section we discuss the remaining terms in $E^{(T)}_{Total}$.

6.2.1 Photometric Energy

The matching cost $E^{(T)}_{Match}(\mathcal{S}, \xi_T)$ is derived from an appearance (*e.g.* brightness) consistency assumption commonly employed in literature, *e.g.* (Lucas and Kanade, 1981). Let us define the monocular warping function $W(\mathbf{x}_i, d_i, \xi_t)$ which maps point $\mathbf{x}_i$ in the keyframe to its representation $\mathbf{x}'_i$ in frame t by

$$\mathbf{x}'_i = W(\mathbf{x}_i, d_i, \xi_t) = \mathrm{hom}\left(\mathbf{R}_t^{\mathrm{T}}(\mathbf{x}_i - \mathbf{t}_t d_i)\right), \tag{6.7}$$

[1] The block-diagonal is an overconfident approximation of the precision. As compensation, we employ a forgetting factor $\lambda_{Temporal}$ in our implementation (see Sec. 6.3.2).



$$\|r\|_\tau = \begin{cases} \frac{1}{2}\|r\|^2 \left(1 - \frac{\|r\|^2}{2\tau^2}\right) & \text{if } \|r\|^2 \leq \tau^2 \\ \frac{\tau^2}{4} & \text{otherwise} \end{cases}$$

Figure 6.2: The smooth truncated quadratic compared to the squared L_2-norm and Huber cost (left), and the smooth truncated quadratic's mathematical representation (right).

under camera rotation $\mathbf{R}_t$ and translation $\mathbf{t}_t$, where $\mathrm{hom}(\cdot)$ normalizes the homogeneous coordinate. Now we express the matching energy as

$$E^{(T)}_{Match}(\mathcal{S}, \xi_T) = \sum_{\mathbf{x}_i \in \mathcal{X}} \|I(\mathbf{x}_i) - I_T\left(W(\mathbf{x}_i, d_i, \xi_T)\right)\|_{\tau_{Match}}, \tag{6.8}$$

where $I(\mathbf{x})$ and $I_T(\mathbf{x})$ are descriptors extracted around pixel $\mathbf{x}$ from keyframe and current frame respectively. We use image intensity values (*i.e.* a descriptor at pixel only), so that the disparity gradients do not need to be taken into account during warping. Robustness is achieved by employing a smooth truncated quadratic error (Li *et al.*, 2008) (visualized in Fig. 6.2) in the implementation of $\|\cdot\|_{\tau_{Match}}$.

6.2.2 Local Spatial Plane Regularizer

The smoothness constraint $E_{Smooth}(\mathcal{S})$ is based on a planar assumption often found in stereo setups (Yamaguchi *et al.*, 2014; Sinha *et al.*, 2014; Zhang *et al.*, 2015), which we adapt in this work to support monocular video data. Surface $\mathbf{s}_i$ induces a linear extrapolation of inverse depth *via* $\hat{d}_i(\mathbf{x}) = \mathbf{s}_i^{\mathrm{T}}\mathbf{x}$. Plugging this into the homographic transformation yields

$$W(\mathbf{x}, \hat{d}_i(\mathbf{x}), \xi_t) = \mathrm{hom}\left(\mathbf{R}_t^{\mathrm{T}}\left(\mathbf{x}_i - \mathbf{t}_t \mathbf{s}_i^{\mathrm{T}}\mathbf{x}_i\right)\right) = \mathrm{hom}\left(\mathbf{R}_t^{\mathrm{T}}\left(\mathbf{x}_i - \mathbf{t}_t \frac{\mathbf{n}_i^{\mathrm{T}}}{r_i}\mathbf{x}_i\right)\right), \tag{6.9}$$

where $\mathbf{n}_i$ is the plane normal and r_i is the point-plane distance to the camera center. Hence we can identify $\mathbf{s}_i \propto \mathbf{n}_i$ and therefore smoothing planes under the inverse depth parametrization also smoothes the alignment in 3D space (Fig. 6.3).

With λ_{Smooth} as balancing term, we define the spatial smoothness energy as

$$\begin{aligned} E_{Smooth}(\mathcal{S}) &= \lambda_{Smooth} \sum_{\mathbf{x}_i \in \mathcal{X}} \sum_{\mathbf{x}_j \in \mathcal{N}_i} \|\mathbf{s}_i^{\mathrm{T}}\mathbf{x}_i - \mathbf{s}_j^{\mathrm{T}}\mathbf{x}_i\|_{\tau_{Smooth}} \\ &= \lambda_{Smooth} \sum_{\mathbf{x}_i \in \mathcal{X}} \sum_{\mathbf{x}_j \in \mathcal{N}_i} \|d_i - (d_j + \mathbf{s}_j^{\mathrm{T}}(\mathbf{x}_i - \mathbf{x}_j))\|_{\tau_{Smooth}}, \end{aligned} \tag{6.10}$$

where $\mathcal{N}_i$ denotes the 4-neighborhood of $\mathbf{x}_i$. Thus, E_{Smooth} penalizes deviations between

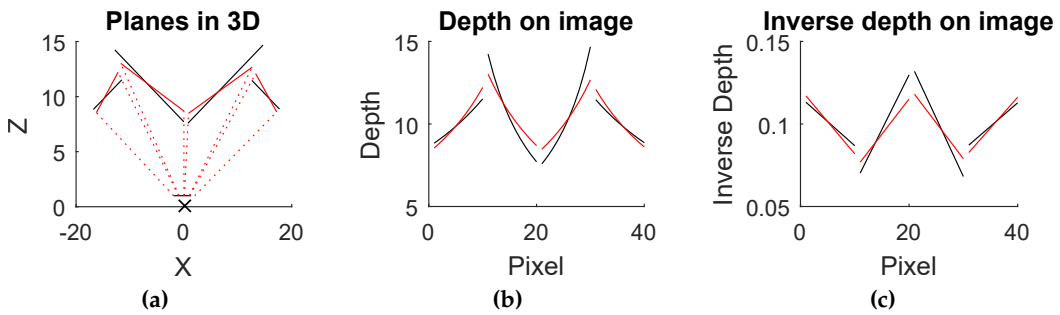


Figure 6.3: Planes in 3D space are aligned *via* smoothing in the inverse depth image (black represent original planes, red represents the smoothed versions).

linearly extrapolated depth at $\mathbf{x}_i$ and its actual depth. Although some methods try to introduce robustness by appearance-based edge detection, *e.g.* (Yang and Li, 2015), we again simply employ the smooth version of the truncated quadratic for $\|\cdot\|_{\tau_{Smooth}}$. Hence, our method is inherently robust without arbitrary color constraints. Unfortunately, (6.10) is not scale invariant, and scaling the baseline $\mathbf{t}_t$ scales the contribution of E_{Smooth}. This is a potential issue only for the first pair of frames (I, I_1), since subsequent frames have their scale determined by preceding frames. It is common usage to fix the initial scale by setting $\|\mathbf{t}_1\| = 1$, but this is a suboptimal choice, since the same 3D scene geometry is regularized differently depending on the initial baseline. A more sensible choice is to fix *e.g.* the average depth (or inverse depth) to make E_{Smooth} invariant w.r.t. baselines. For our reconstruction we constrain the average inverse depth to one.

6.3 Optimization Strategy

In this section we detail our optimization strategy for the energy in *Eq.* 6.6. We assume small changes between consecutive frames, as video data is used. Therefore we use a similar approach as in standard differential tracking and optical flow by locally linearizing the image intensities I_T in the matching term $E^{(T)}_{Match}$. The pseudocode of the proposed method is given in Algorithm 6.2. The underlying idea is to optimize the energy incrementally with increased complexity using the scale-space pyramid representation and our restricted depth map update which we detail below. The aim of doing this is two-fold: Firstly it substantially reduces the number of unknowns in the main objective and therefore makes the optimization much more efficient, and secondly it provides an additional level of regularization within the algorithm and combines naturally with a scale-space framework to avoid poor local minima. We discuss this constrained depth map update in the following, and then

Algorithm 6.2 Dense Incremental Planar Depth Estimation

Require: Keyframe I and images $(I_t)_{t=1}^{T}$.
Ensure: Final pose ξ and depth hypothesis $\mathcal{S}$.
1: $\mathbf{s}_i \leftarrow [0\ 0\ 1]^{\mathrm{T}}$ and $\Lambda_i \leftarrow \mathbf{0}$ for all $\mathbf{x}_i \in \mathcal{X}$.
2: compute resolution pyramid for the keyframe I.
3: $\xi \leftarrow (\mathbf{I} \in \mathbb{R}^{3\times 3}, [0\ 0\ 0]^{\mathrm{T}})$
4: **for** each frame I_t **do**
5: compute resolution pyramid for the frame I_t.
6: **for** each pyramid level **do**
7: optimize ξ *via* lie algebra $\mathfrak{se}(3)$ through Levenberg-Marquardt.
8: **repeat**
9: update ξ (and $\mathbf{s}_i \leftarrow \mathbf{s}_i + \mathbb{I}_c(\mathbf{x}_i)\Delta_c$ if applicable).
10: introduce new component Δ_c.
11: estimate $\mathbb{I}_c(\mathbf{x}_i)$ *via* eigenvector of $\sum_{\mathbf{x}_i \in \mathcal{X}} \nabla_{\mathbf{s}_i} \nabla_{\mathbf{s}_i}^{\mathrm{T}}$ (*Eq.* 6.13).
12: optimize ξ and Δ_c through Levenberg-Marquardt (*Eq.* 6.14).
13: **until** improvement below $\epsilon_{Complex}$ or maximum components reached
14: **end for**
15: update precision Λ_i and depth $\mathbf{s}_i^*$ for temporal constraint.
16: **end for**

introduce our optimization which exploits this update to allow for truly simultaneous pose and depth estimation. Finally we present a strategy for realtime performance on CPU.

6.3.1 Constrained Depth Map Updates

If we consider the current frame at time T and optimize E_{Total} (recall (6.6)) w.r.t. ξ_T and $\mathcal{S}$, then our algorithmic design choice is to restrict the update $\mathcal{S} - \mathcal{S}^*$ to have low complexity in the following sense:

$$\mathbf{s}_i = \mathbf{s}_i^* + \sum_{c=1}^{C} \mathbb{I}_c(\mathbf{x}_i)\Delta_c, \tag{6.11}$$

where $\mathbb{I}_c : \mathcal{X} \rightarrow \{+1, -1\}$ is an indicator function, splitting the set of pixels into positive or negative parts. This means that a depth update at each pixel $\mathbf{x}_i$ is constrained to take one of 2^C values. With increasing cardinality C, the complexity of the depth map increases.

The optimization is performed greedily by adding a single component Δ_c at a time. Notice, if ξ_T and $\mathcal{S}$ were to be optimized simultaneously, an equation with $6 + 3|\mathcal{X}|$ unknowns had to be solved inside a nonlinear least squares solver (*i.e.* 6 parameters for an element in the lie algebra $\mathfrak{se}(3)$ and 3 for the over-parameterized depth values at each pixel). By using the constrained shape for the updates and a greedy framework, we reduce the optimization to $6 + 3$ variables at a time (*i.e.* $\mathfrak{se}(3)$ and the 3 vector Δ_c), improving the execution cost and robustness significantly.

Our methodology can be seen in analogy to multi-resolution pyramids which spatially increase the quantization of the image plane, but in addition to spatial resolution we also incrementally increase the quantization level of inverse depths. Specifically, we exploit the representation of a pixel's plane $\mathbf{s}_i$ as summed components Δ_c, given in (6.11). These values correspond to the inverse depth resolution which increases when new components are introduced.

This coarse-to-fine depth estimation is inspired by the human vision (Westheimer, 1979), which perceives depth in relation to other areas in the scene, rather than absolute values. Specifically, we perform the introduction of new distance values in a relational setting, splitting the data points based on their desired depth value direction. The advantages of this approach are three-fold: (1) we introduce depth by enforcing a regularization across all pixels, (2) our splitting function separates the image data into multiple planes, which naturally encode the image hierarchically from coarse to fine, and (3) the incremental introduction of depth enables fast computation whilst optimizing transformation and depth simultaneously. Moreover, we emphasize while our approach is greedy, it is not final since corrections can be made through further splitting.

Our design choice to regularize the updates of $\mathcal{S}$ requires us to determine the binary function $\mathbb{I}_c : \mathcal{X} \rightarrow \{+1, -1\}$. Essentially, if Δ_c is given, $\mathbb{I}_c(\mathbf{x}_i)$ corresponds to the sign of the correlation $\Delta_c^{\mathrm{T}} \nabla_{\mathbf{s}_i} E_{Total}$ between the depth update direction Δ_c and the gradient of the objective with respect to $\mathbf{s}_i$. Since Δ_c is subject to subsequent optimization, we determine an initial estimate $\tilde{\Delta}_c$ as follows: given the current gradients $\nabla_{\mathbf{s}_i} E_{Total}$ (which we abbreviate to $\nabla_{\mathbf{s}_i}$), it is sensible to obtain $\tilde{\Delta}_c$ as principal direction of the set $\{\nabla_{\mathbf{s}_i}\}_{i=1}^{|\mathcal{X}|}$, due to the symmetric range in $\mathbb{I}_c$:

$$\tilde{\Delta}_c \leftarrow \underset{u:\|u\|=1}{\operatorname{argmax}} \left\{ u^{\mathrm{T}} \sum\nolimits_{\mathbf{x}_i \in \mathcal{X}} \nabla_{\mathbf{s}_i} \nabla_{\mathbf{s}_i}^{\mathrm{T}} u \right\}. \tag{6.12}$$

This can be obtained by eigenvalue or singular value decomposition of the 3×3 scatter matrix $\sum_{\mathbf{x}_i \in \mathcal{X}} \nabla_{\mathbf{s}_i} \nabla_{\mathbf{s}_i}^{\mathrm{T}}$. Finally, the indicator function is given by

$$\mathbb{I}_c(\mathbf{x}_i) = \begin{cases} 1 & \text{if } \tilde{\Delta}_c^{\mathrm{T}} \nabla_{\mathbf{s}_i} \geq 0 \\ -1 & \text{otherwise} \end{cases} = \operatorname{sgn}\left(\tilde{\Delta}_c^{\mathrm{T}} \nabla_{\mathbf{s}_i} \right). \tag{6.13}$$

6.3.2 Simultaneous Pose and Depth Estimation

Let us assume we have an initial estimate for ξ_T and $\mathcal{S}$ available (*e.g.* $\xi_T \leftarrow \xi_{T-1}$ and $\mathcal{S} \leftarrow \mathcal{S}^*$, which is equivalent to $C = 0$ in (6.11)). Since our objective is an instance of nonlinear

least-squares problems we utilize the Levenberg-Marquardt (LM) algorithm for robust and fast second order minimization. The robust kernels $\|\cdot\|_{\tau_{Match}}$ and $\|\cdot\|_{\tau_{Smooth}}$ are handled by an iteratively reweighed least square (IRLS) strategy. Potentially enlarging the convergence basin *via* a lifted representation of the robust kernel (Zach, 2014) is a topic for future work.

As outlined in §6.3.1 the complexity of depth map updates is increased greedily, which means that new components Δ_c are successively introduced. We start with $C = 0$ and iteratively increase C by adding new components. After introduction of a new component Δ_c (and having an estimate for $\mathbb{I}_c$), minimizing E_{Total} with respect to Δ_c and ξ_T amounts to solving (using LM)

$$\begin{aligned}\underset{\xi_T,\Delta_c}{\operatorname{argmin}}\Big\{ &\sum_{\mathbf{x}_i\in\mathcal{X}} \|I(\mathbf{x}_i) - I_T\left(W(\mathbf{x}_i, (\mathbf{s}_i + \mathbb{I}_c(\mathbf{x}_i)\Delta_c)^{\mathrm{T}}\mathbf{x}_i, \xi_T)\right)\|_{\tau_{Match}} \\ &+ \lambda_{Smooth} \sum_{\mathbf{x}_i\in\mathcal{X}} \sum_{\mathbf{x}_j\in\mathcal{N}_i} \|(\mathbf{s}_i + \mathbb{I}_c(\mathbf{x}_i)\Delta_c)^{\mathrm{T}}\mathbf{x}_i - (\mathbf{s}_j + \mathbb{I}_c(\mathbf{x}_j)\Delta_c)^{\mathrm{T}}\mathbf{x}_i\|_{\tau_{Smooth}} \\ &+ \sum_{\mathbf{x}_i\in\mathcal{X}} \|\mathbf{s}_i^* - (\mathbf{s}_i + \mathbb{I}_c(\mathbf{x}_i)\Delta_c)\|_{\Lambda_i}\Big\}\end{aligned} \tag{6.14}$$

followed by the update $\mathbf{s}_i \leftarrow \mathbf{s}_i + \mathbb{I}_c(\mathbf{x}_i)\Delta_c$. We emphasize, as Δ_c is shared between all pixels, this problem is unlikely to be rank deficient. Further components Δ_c are introduced as long as E_{Total} is reduced sufficiently (*i.e.* an improvement larger than $\epsilon_{Complex}$). Notice, while our algorithm iteratively introduces new components Δ_c, it optimizes pose and depth simultaneously. Analogous to the resolution-based scale-space pyramid, the indicator function acts as surrogate for increased resolution in depth.

For the first frame I_1 matched with the keyframe I we need to enforce that the average inverse depth is 1 (recall Section 6.2.2), which implies that

$$\sum_{\mathbf{x}_i} (\mathbf{s}_i + \mathbb{I}_c(\mathbf{x}_i)\Delta_c)^{\mathrm{T}}\mathbf{x}_i = \sum_{\mathbf{x}_i} \left(d_i + \mathbb{I}_c(\mathbf{x}_i)\Delta_c^{\mathrm{T}}\mathbf{x}_i\right) = 1 \tag{6.15}$$

must hold. If d_i already satisfies $\sum_{\mathbf{x}_i} d_i = 1$, then the above reduces to

$$\sum_{\mathbf{x}_i} \mathbb{I}_c(\mathbf{x}_i)\mathbf{x}_i^{\mathrm{T}}\Delta_c = 0. \tag{6.16}$$

We chose a projected gradient approach by projecting the gradient w.r.t. Δ_c to the feasible subspace defined by (6.16) inside the LM optimizer. Note that the planes are initialized to $\mathbf{s}_i = (0, 0, 1)^{\mathrm{T}}$ in the beginning of the algorithm, and by induction $\sum_{\mathbf{x}_i} \mathbf{s}_i^{\mathrm{T}}\mathbf{s}_i = \sum_{\mathbf{x}_i} d_i = 1$ is always satisfied for the first frame. In subsequent frames the constraint in (6.16) is not active.

Finally, to determine the precision matrices $\Lambda_i \in \mathbb{R}^{3\times 3}$ needed for $E^{(T+1)}_{Temporal}$, we employ the approximate Hessian *via* the Jacobian $\mathbf{J}_{Match}$ of $E^{(T)}_{Match}$:

$$\begin{pmatrix} \tilde{H}_{\mathcal{S},\mathcal{S}} & \tilde{H}^{\mathrm{T}}_{\mathcal{S},\xi_T} \\ \tilde{H}_{\mathcal{S},\xi_T} & \tilde{H}_{\xi_T,\xi_T} \end{pmatrix} := \mathbf{J}^{\mathrm{T}}_{Match}\mathbf{J}_{Match}, \tag{6.17}$$

and the 3×3-diagonal block of the Schur complement $\tilde{H}_{\mathcal{S},\mathcal{S}} - \tilde{H}^{\mathrm{T}}_{\mathcal{S},\xi_T}\tilde{H}^{-1}_{\xi_T,\xi_T}\tilde{H}_{\mathcal{S},\xi_T}$ (denoted Λ_{Match}). We employ a forgetting factor $\lambda_{Temporal}$ to reduce the overconfident precision matrix, and update $\Lambda_i \leftarrow \lambda_{Temporal}\Lambda_i + \Lambda_{Match}$. Recall that $\tilde{H}_{\xi_T,\xi_T} \in \mathbb{R}^{6\times 6}$ and $\tilde{H}_{\mathcal{S},\xi_T}$ are very sparse.

6.3.3 CPU Computation in Realtime

Thus far, we present our energy for each pixel in the input video stream. While this is generally useful for dense depth estimation, we may adopt our approach to semi-dense computation to reduce runtime. Similar to LSD-SLAM, we can represent the image by its significant gradient values. By only computing on these gradients, execution is significantly reduced. In fact, in comparison to LSD-SLAM, we only need one additional LM iteration per split to introduce depth on top of pose estimation. Finally, we can limit the number of introduced depth components per resolution level to achieve constant running time.

6.4 Results

We perform our experiments on 13 video sequences in total, using 6 TUM (Sturm *et al.*, 2012) image streams and 7 sequences recoded ourselves. The TUM dataset comprises a number of video sequences with groundtruth pose, as recorded by a Vicon system, and approximate depth through depth sensors (Sturm *et al.*, 2012). We select a subset of the handheld SLAM videos to measure system performance (*i.e.* fr1-desk, fr1-desk2, fr1-floor, fr1-room, fr2-xyz and fr3-office). As we are interested in the local aspect of SLAM (operating with single keyframe), we further divide these into smaller sequences. Notice, as we perform keyframe-to-frame comparison, the videos need to contain enough overlap with the reference image. Additionally, we record 7 videos, using a GoPro Hero 3 with a wide angle lens at 30 fps.

As a monocular approach, our method does not fix the scale. Hence, we employ a scale corrected error (SCE) for translation:

$$e(\mathbf{t}_t, \hat{\mathbf{t}}_t) = \left\| \mathbf{t}_t \frac{\|\hat{\mathbf{t}}_t\|}{\|\mathbf{t}_t\|} - \hat{\mathbf{t}} \right\|, \tag{6.18}$$

where $\mathbf{t}_t$ is the translational displacement of the pose ξ_t, and $\hat{\mathbf{t}}_t$ is the groundtruth with respect to the keyframe (or initial frame). An error in rotation is indirectly captured, as

Table 6.1: Median Scale Corrected Error (in mm) for the compared methods after the listed frame number for different TUM-Dataset sequences. (Note, different characteristics of camera motion in each video lead to different length of keyframe overlaps.)

		LSD-SLAM	LSD-Key	Disjoint	SIP	DIP
fr1-desk	frame 5	34	34	33	**25**	27
	frame 10	44	62	55	43	**30**
	frame 30	106	130	119	135	**46**
fr1-desk2	frame 5	68	68	53	23	**18**
	frame 10	103	115	87	**41**	44
	frame 20	207	-	162	163	**64**
fr1-floor	frame 5	30	30	36	**30**	34
	frame 10	**55**	58	76	58	60
	frame 15	85	88	111	**79**	86
fr1-room	frame 5	13	13	19	**10**	16
	frame 10	40	40	52	**39**	42
	frame 25	**9**	79	117	-	53
fr2-xyz	frame 10	15	15	10	**9**	**9**
	frame 30	54	68	28	**18**	23
	frame 100	121	88	**45**	**45**	47
fr3-office	frame 10	**29**	30	41	32	33
	frame 50	90	121	182	**53**	100
	frame 150	206	-	265	-	**123**

it effects the translation of future frames. We now introduce a scale invariant measure to evaluate the depth's completeness. Given true inverse depth at the keyframe $\hat{\mathcal{D}} = (\hat{d}_i)_{i=1}^{|\mathcal{X}|}$ we define the completeness as the proportion of depth values, satisfying a given accuracy ϵ:

$$c\left(\hat{\mathcal{D}}, \mathcal{D}\right) = \max_{\alpha} \sum_{i=1}^{|\mathcal{X}|} \frac{n_\alpha(\hat{d}_i, d_i)}{|\mathcal{X}|}, \text{where } n_\alpha(\hat{d}_i, d_i) = \begin{cases} 1 & \text{if } \|\frac{1}{\hat{d}_i} - \frac{\alpha}{d_i}\| < \epsilon \\ 0 & \text{otherwise} \end{cases}. \tag{6.19}$$

Parameter α represents scale and is found *via* grid search and refined through gradient descent. In our work, $\epsilon = 0.05$ which corresponds to ± 5cm.

6.4.1 Quantitative Evaluation on the TUM Dataset

We compare the proposed dense and semi-dense system (DIP and SIP respectively) to two versions of LSD-SLAM: (1) we carefully implement a LSD-SLAM version that only uses a single keyframe (LSD-Key), and (2) the original LSD-SLAM as provided by authors of (Engel *et al.*, 2014a), without loop closures or other constraints (LSD-SLAM). We further ensure that mapping is guaranteed to run after every tracking step in both LSD-SLAM systems. Finally, we include our method as disjoint optimization for pose and depth separately and sequentially. Table 8.1 shows the median SCE for different numbers of frames. The median is calculated over all snippets taken from the individual TUM sequences.

The sequences fr1-desk and fr1-desk2 show an office environment with high camera motion and little overlap towards keyframes. The trajectories are quickly lost when a single

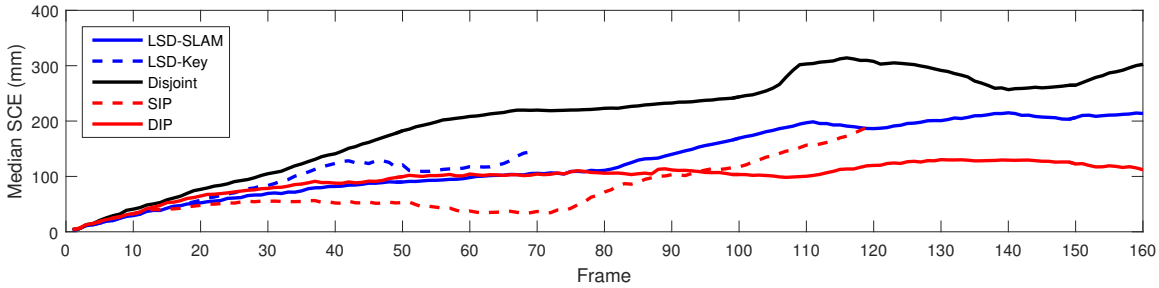


Figure 6.4: Median SCE for videos of fr3-office. LSD-SLAM and DIP track long-term, while SIP is more accurate early on. LSD-Key loses track quickly, and the disjoint optimization (Disjoint) is consistently worse.

keyframe is used. SIP performs best at early stages, while DIP is more suitable for longer tracking. The sequences fr1-floor and fr1-room also have little keyframe overlap, but with slower motion. LSD-SLAM performs competitively, as it benefits from keyframe generation.

Long-term tracks are evaluated with fr2-xyz and fr3-office. Fig. 6.4 plots the median SCE for each duration of fr3-office. We see that LSD-SLAM and DIP perform similarly early on, but DIP performs better at latter stages. Notice, as LSD-SLAM generates new reference images, the baseline is typically small. In contrast DIP benefits from larger baselines. LSD-Key loses track quickly, while SIP performs well in early stages. The trajectory and inverse depth maps for the very first 300 frames are shown in Fig. 6.6. Fig. 6.7 plots the depth completeness. Here, DIP and SIP reach a peak correctness with increasing baseline, after which they slightly degrades as points are outside the current view, and smoothing takes over their energies.

We remark, similar to many approaches based on gradient decent, our method converges to local minima. However our method relies on graduated optimization which aims to avoid getting trapped in bad minima by optimizing a smoother energy with gradually increased complexity (Mobahi and Fisher, 2015). In contrast to LSD-SLAM, we employ graduated optimization in depth perception as well as traditional scale-space image pyramids leading to superior results. The indicator function is a surrogate for the scale-space pyramid in depth. Finally, we note that the disjoint version is consistently worse in virtually all experiments. The difference is the impact of graduated optimization. For Disjoint, changes in perceived depth are not utilized for pose at the current frame. In contrast, joint optimization finds pose and depth at the same time, yielding improved performance.

In terms of runtime, LSD-SLAM and LSD-Key perform tracking and mapping at 14 fps, while SIP performs twice as fast at 30 fps on CPU. DIP is slower on CPU (2 fps), but its GPU implementation runs in realtime (30 fps).

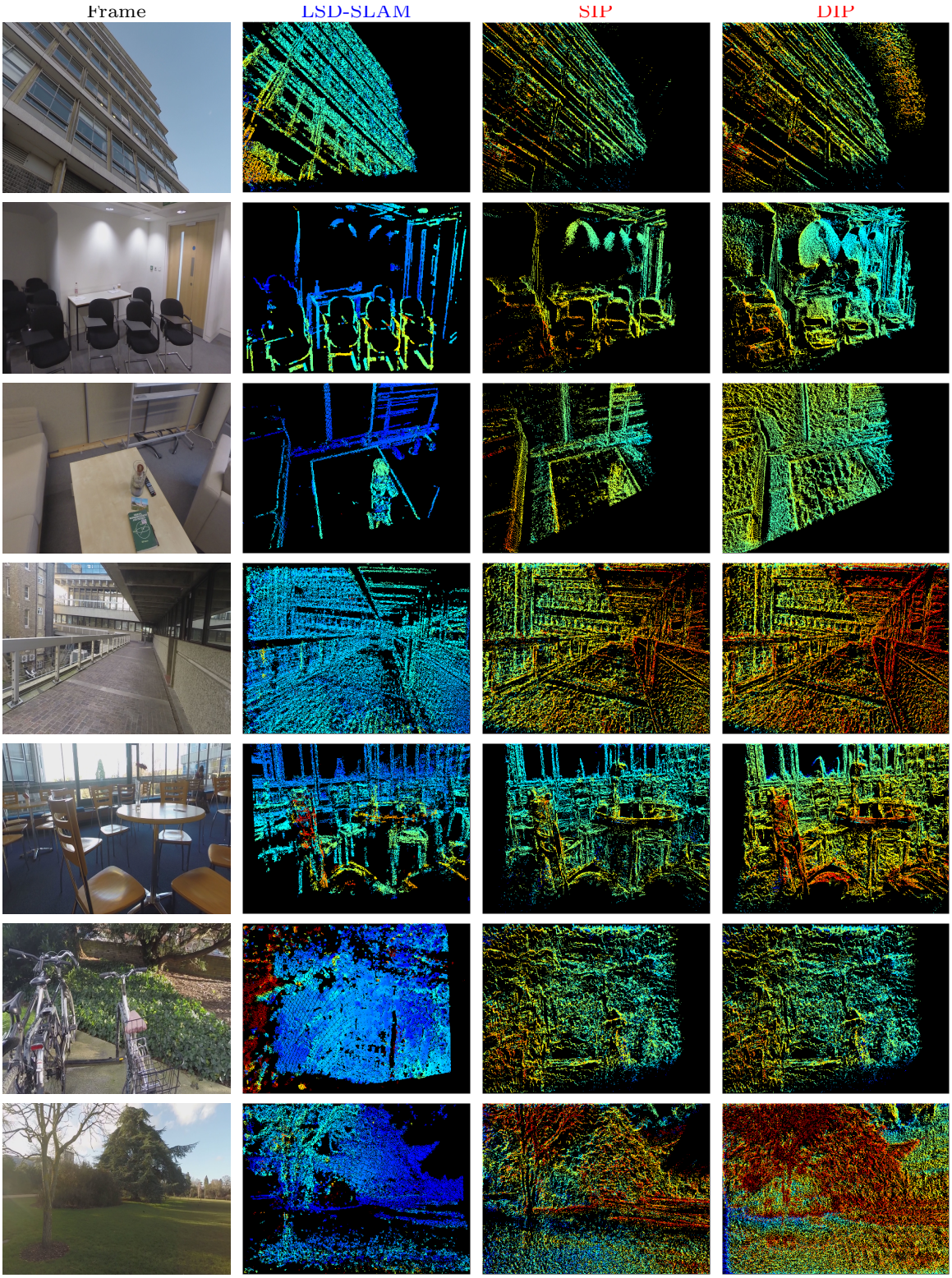


Figure 6.5: Inverse depth of LSD-SLAM, SIP and DIP for 7 qualitative video sequences (far is blue, near is red). In most scenes, the local planar surface assumption holds and our method performs well. In non-urban environments and where the initialization with frontal planar surfaces does not hold, our method fails (bottom row).

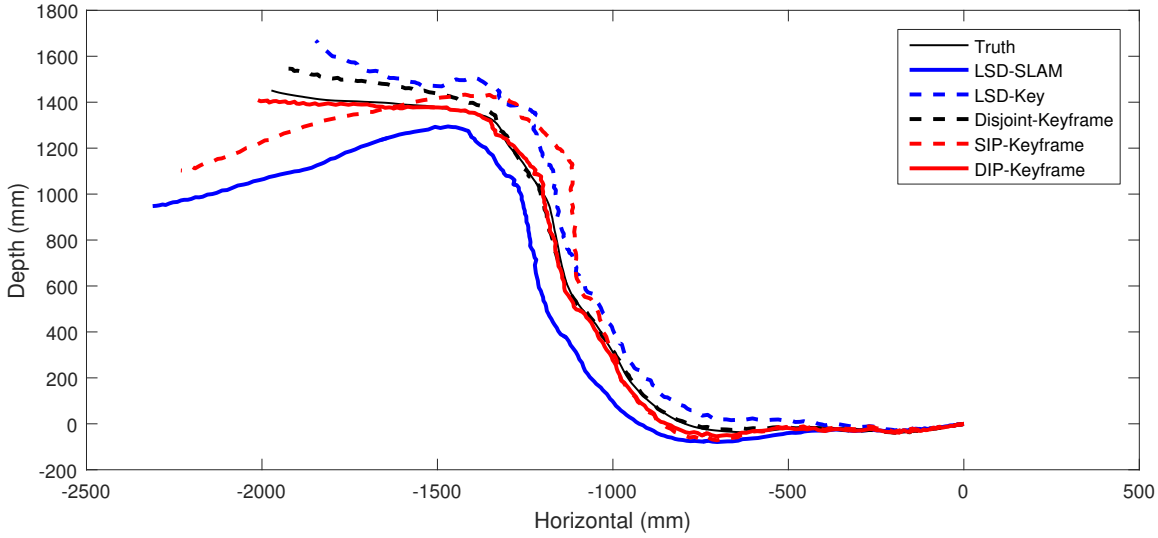


Figure 6.6: Trajectories (left) and inverse depth maps (right) of LSD-SLAM, SIP and DIP for the initial 300 images in fr3-office. LSD-SLAM is inaccurate due to scale drift. DIP uses a single keyframe and hence does not drift as significantly. For depth, SIP and DIP benefit from larger keyframe-to-frame baseline, resulting in qualitative better depth.

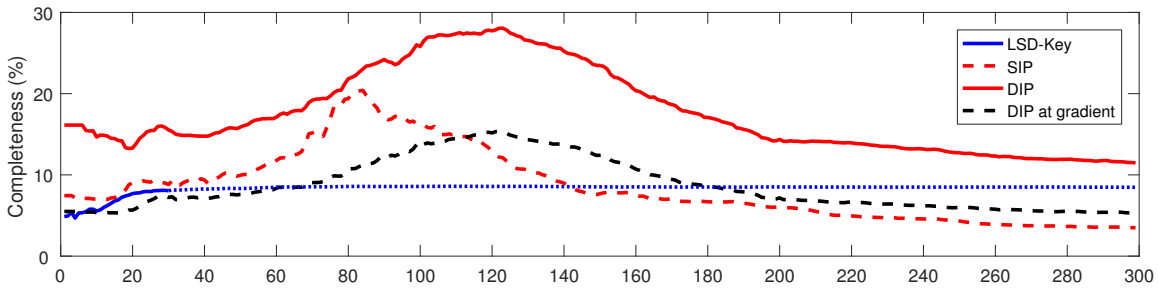


Figure 6.7: Depth completeness of LSD-Key, SIP and DIP for initial images in fr3-office. As LSD-Key and SIP only produces depth for high gradient pixels, the results of DIP at gradient only are also shown. Note, LSD-Key remains unchanged after poor tracking.

6.4.2 Qualitative Results

We conclude the experiments with sample frames of our 7 additional video sequences (Fig. 6.5). Generally, LSD-SLAM works well in the local neighborhood, while SIP and DIP perform more consistently on the global inverse depth hypothesis. The local planar surface assumption is reasonable in most environments, as was also witnessed by recent stereo systems, *e.g.* (Yamaguchi *et al.*, 2014; Sinha *et al.*, 2014; Zhang *et al.*, 2015). Nonetheless, in non-urban scenes, and in situations where the initial frontal plane assumption is significantly wrong (recall initialization of $\mathbf{s}_i = (0, 0, 1)^{\mathrm{T}}$), the results are less favorable as seen in the last row of Fig. 6.5.

6.5 Conclusion

We introduced a carefully derived coarse-to-fine planar regularization strategy that optimizes for both pose and depth simultaneously from monocular streams. Our framework is keyframe-based, and incrementally improves its depth hypothesis at each frame as new data arrives. As semi-dense approach, the proposed method runs in realtime on CPU, while realtime for the dense version can be achieved on GPU. In our evaluation, we improved upon the front-end of LSD-SLAM whilst increasing execution time by a factor of two.

Seeing it from perspective of 2017. The main limitation of this chapter is that we address only the local model and do not propagate information between multiple keyframes. Another drawback is that we use only grayscale features. Instead, we should use either handcrafted or learnt feature descriptors which are more invariant to illumination changes. High-dimensional feature spaces typically complicate the optimization, however, this can be overcome with supervised descent methods (Xiong and la Torre, 2014, 2015).

We could also use outputs provided by learnt regressors (CNN or Random Forest) as prior or coarse initialization (Eigen *et al.*, 2014). Scale drift represent an inherent issue for all monocular methods, however, it can be suppressed if we are able to recognize objects of known sizes within a scene (Frost *et al.*, 2016).

7 ROAM: a Rich Object Appearance Model with Application to Rotoscoping

Rotoscoping, the detailed delineation of scene elements through a video shot, is a painstaking task of tremendous importance in professional post-production pipelines. While pixel-wise segmentation techniques can help for this task, professional rotoscoping tools rely on parametric curves that offer the artists a much better interactive control on the definition, editing and manipulation of the segments of interest. Sticking to this prevalent rotoscoping paradigm, we propose a novel framework to capture and track the visual aspect of an arbitrary object in a scene, given a first closed outline of this object. This model combines a collection of local foreground/background appearance models spread along the outline, a global appearance model of the enclosed object and a set of distinctive foreground landmarks. The structure of this rich appearance model allows simple initialization, efficient iterative optimization with exact minimization at each step, and on-line adaptation in videos. We demonstrate qualitatively and quantitatively the merit of this framework through comparisons with tools based on either dynamic segmentation with a closed curve or pixel-wise labelling.

Figure 7.1: ROAM **for video object segmentation**. Designed to help *rotoscoping*, the proposed object appearance model allows the automatic delineation of a complex object in a shot, starting from an initial outline provided by the user.

7.1 Introduction

Modern high-end visual effects (vfx) and post-production rely on complex workflows whereby each shot undergoes a succession of artistic operations. Among those, rotoscoping is probably the most ubiquitous and demanding one (Bratt, 2011; Li *et al.*, 2016b). Rotoscoping amounts to outlining accurately one or several scene elements in each frame of a shot. This is a key operation for compositing (Wright, 2006) (insertion of a different background, whether natural or synthetic), where it serves as an input to subsequent operations such as matting and motion blur removal.[1] Rotoscoping is also a pre-requisite for other important operations, such as object colour grading, rig removal and new view synthesis, with large amounts of elements to be handled in the latter case.

Creating such binary masks is a painstaking task accomplished by trained artists. It can take up to several days of work for a complex shot of only a few seconds, using dedicated tools within video editing softwares like Silhouettefx, Adobe After Effect, Autodesk Flame or The Foundry Nuke. As discussed in (Li *et al.*, 2016b), professional roto artists use mostly tools based on *roto-curves*, *i.e.* parametric closed curves that can be easily defined, moved and edited throughout shots. By contrast, these artists hardly use brush-based tools, even if empowered by local appearance modelling, graph-based regularization and optic flow-based tracking as the After Effect's ROTOBRUSH.

Due to its massive prevalence in professional workflows, we address rotoscoping in its closed contour form, which we aim to facilitate. Roto-curves being interactively placed in selected keyframes, automation can be sought either at the key-frame level (reducing the number of user's inputs) or at the tracking level (reducing the number of required key-frames). Recently, Li *et al.* (2016b) proposed the ROTO++tool that helps on both fronts, thanks to elegant shape modelling.

In the present work, we explore a complementary route that focuses on automatic tracking from a given keyframe. In essence, we propose to equip the roto-curve with a rich, adaptive modelling of the appearance of the enclosed object. This model, coined ROAM for Rich Online Appearance Model, combines in a flexible way various appearance modelling ingredients: (i) Local foreground/background colour modelling, in the spirit of VIDEO SNAPCUT (Bai *et al.*, 2009) but attached here to the roto-curve; (ii) Fragment-based modelling to handle large displacements and deformations and (iii) Global appearance

[1]The use of blue or green screens on set can ease compositing but remains a contrived set-up. Even if accessible, such screens lead to chroma-keying and de-spilling operations that are not trivial and are not suited to all foreground elements, thus rotoscoping remains crucial.

modelling, which has proved very powerful in binary segmentation with graph cuts, *e.g.* in (Boykov and Jolly, 2001).

We would like to emphasize that our model is the first that combines local appearance models along the closed contour with global appearance model of the enclosed object using discrete Green theorem, and pictorial structure to capture locally rigid deformations, in a principled structured prediction framework. As demonstrated on recent benchmarks, ROAM outperforms state-of-art approaches when a single initial roto-curve is provided. It is in particular less prone to spurious changes of topology that lead to eventual losses than After Effect's ROTOBRUSH, and more robust than ROTO++ (Li *et al.*, 2016b) in the absence of additional user inputs. This robustness makes it appealing to facilitate rotoscoping, either as a standalone tool, or combined with existing curve-based tools such as ROTO++.

7.2 Related work and motivation

Rotoscoping is a form of interactive "video object"[2] segmentation. As such, the relevant literature is vast. For sake of brevity, we focus mostly our discussion on works that explicitly target rotoscoping or very similar scenarios.

7.2.1 Rotoscoping and curve-based approaches

Li *et al.* (2016b) recently released a very detailed study of professional rotoscoping workflow. They first establish that trained artists mostly use parametric curves such as Bezier splines to delineate objects of interest in key-frames, "track" them from one frame to the next, edit them at any stage of the pipeline and, last but not least, pass them in a compact and manipulable format to the next stage of the vfx pipeline, *e.g.* to the compo-artists. Professional rotoscoping tools such as Silhouettefx, Blender, Nuke or Flame are thus based on parametric curves, which can be either interpolated between key-frames or tracked with a homographic "planar tracker" when suitable. Sticking to this ubiquitous workflow, the authors propose ROTO++ to speed it up. Bezier roto-curves defined by the artist in the selected key-frames allow the real-time learning of a non-linear low-dimensional shape space based on a Gaussian process latent variable model. Shape tracking between key-frames, as well as subsequent edits, are then constrained within this smooth manifold (up to planar transforms), with substantial gains in work time. Our work is fully complementary

[2]Throughout, "video object", or simply "object", is a generic term to designate a scene element of interest and the associated image region in the video.

to ROTO++: while ROAM does not use a strong shape prior in its current form, it captures the dynamic appearance of the video object, something that ROTO++ does not address.

In their seminal rotoscoping work, Agarwala *et al.* (2004) proposed a complete interactive system to track and edit Bezier roto-curves. It relies on the popular active contour framework (Blake and Isard, 2000; Kass *et al.*, 1988): a curve, parametrized by control points, finely discretized and equipped with a second-order smoothness prior is encouraged to evolve smoothly and snap to strong image edges. Their energy-based approach also uses local optical flow along each side of the shape's border. In contrast to this work, our approach offers a richer local appearance modelling along the roto-shape as well as additional intra-object appearance modelling.

Similarly to (Agarwala *et al.*, 2004), Lu *et al.* (2016) recently introduced an interactive object segmentation system called "coherence parametric contours" (CPC), which combines planar tracking with active contours. Our system includes similar ingredients, with the difference that the planar tracker is subsumed by a fragment-based tracker and that the appearance of the object and of its close surrounding is also captured and modeled. We demonstrate the benefits of these additional features on the evaluation dataset introduced by Lu *et al.* (2016).

7.2.2 Masks and region-based approaches

Other notable approaches to interactive video segmentation address directly the problem of extracting binary masks, *i.e.* labelling pixels of non-keyframes as foreground or background. As discussed in (Li *et al.*, 2016b; Lu *et al.*, 2016), a region-based approach is less compatible with professional rotoscoping, yet provides powerful tools. Bai *et al.* (2009) introduced VIDEO SNAPCUT, which lies at the heart of After Effect's ROTOBRUSH. Interaction in VIDEO SNAPCUT is based on foreground/background brushes, following the popular scribble paradigm of Boykov and Jolly (2001). The mask available in a given frame is tracked to the next frame through the propagation of local windows that straddle its border. Each window is equipped with a local foreground/background colour model and a local shape template, both updated through time. After propagation along an object-centric optical flow, these windows provide suitable pixel-wise unaries that are fed to a classic graph-cut. This approach provides a powerful way to capture on-the-fly local colour models and combine them adaptively with some shape persistence. However, being based on graph-cut (pixel-wise labelling), ROTOBRUSH can be penalized by its extreme topology flexibility: as will be showed in the experiments, rapid movements of the object, for instance, can cause

large spurious deformations of the mask that can eventually lead to complete losses in the absence of user intervention. In ROAM, we take inspiration from the local colour modelling at the object's border and revisit it in a curve-based segmentation framework that allows tighter shape control and easier subsequent interaction.

More recently, Fan *et al.* (2015) introduced JUMPCUT, another mask-based approach where frame-to-frame propagation is replaced by mask transfer from the key-frame(s) to distant frames. This long-range transfer leverages dense patch correspondences computed over the inside and outside of the known mask, respectively. The transfered mask is subsequently refined using a standard level set segmentation (region encoded via a spatial map). A salient edge classifier is trained online to locate likely fragments of object's new silhouette and drive the level set accordingly. They reported impressive results with complex deformable objects going through rapid changes in scene foreground. However, similarly to ROTOBRUSH, this agility might also become a drawback in real rotoscoping scenarios, as is the lack of shape parametrization. Also, the underlying figure/ground assumption (the object is moving distinctly in front of a background) is not met in many cases, *e.g.* rotoscoping of a static scene element or of an object in a dynamic surrounding.

7.3 Introducing ROAM

Our model consists of a graphical model with the following components: (i) a closed curve that defines an object and a collection of local foreground/background[3] appearance models along it; (ii) a global appearance model of the enclosed object; and (iii) a set of distinctive object's landmarks. While the global appearance model captures image statistics as in graph-cut approaches (Boykov and Jolly, 2001; Rother *et al.*, 2004), it is the set of local fg/bg appearance models placed along the boundary that enables accurate object delineation. The object's distinctive landmarks are organized in a star-shaped model (Fig. 7.4, left) and help to prevent the contour from sliding along itself and to control the level of non-rigid deformations. The landmarks are also used to robustly estimate a rigid transformation between the frames to "pre-warp" the contour, which significantly speeds-up the inference. In addition, the control points of the roto-curve, as well as the local fg/bg models and the landmarks are maintained through time, which provides us with different types of temporal correspondences.

[3] "Foreground/background" terminology, "fg/bg" in short, merely refers here to inside and outside of the roto-curve; it does not imply that the object stands at the forefront of the 3D scene with a background behind it.

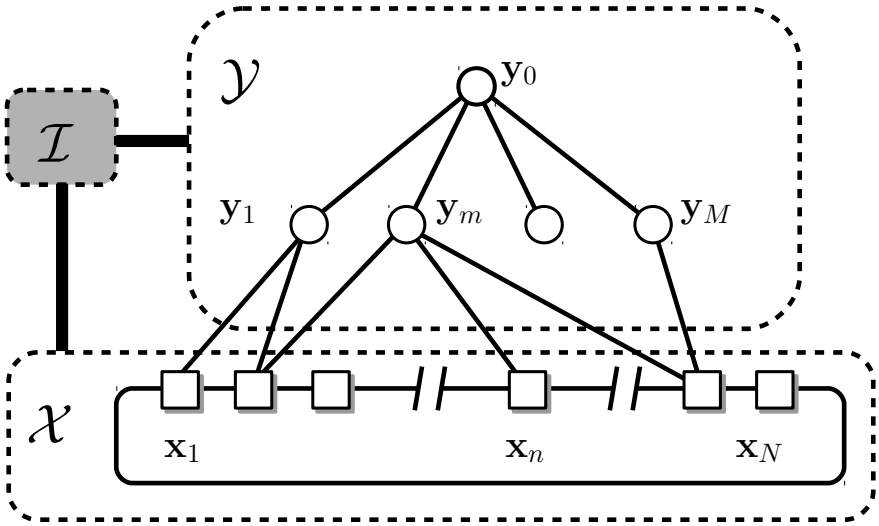


Figure 7.2: Graphical model of ROAM. In joint model defined by energy $E(\mathcal{X}, \mathcal{Y}; \mathcal{I})$ in (7.1), contour node variables (white squares) form a closed 1-st order chain conditioned on image data (grey box) and landmark variables (white circles), the latter variables forming a shallow tree conditioned on all others.

Given a colour image $\mathcal{I} = \{\mathbf{I_p}\}_{\mathbf{p}\in\mathcal{P}}$, a conditional graphical model (Fig. 7.2) is defined through the energy function

$$E(\mathcal{X}, \mathcal{Y}; \mathcal{I}) := E^C(\mathcal{X}; \mathcal{I}) + E^L(\mathcal{Y}; \mathcal{I}) + E^J(\mathcal{X}, \mathcal{Y}), \tag{7.1}$$

where E^C and E^L depend only on the roto-curve configuration $\mathcal{X}$ and the landmarks configuration $\mathcal{Y}$ respectively, and E^J links the two together (independently of the image). In the following, we describe these three energy terms in detail.

7.3.1 Curve-based modelling: E^C

While Bezier splines are a popular representation for rotoscoping (Agarwala *et al.*, 2004; Li *et al.*, 2016b), we simply consider polygonal shapes here: roto-curve $\mathcal{X}$ is a polyline with N vertices $\mathbf{x}_1 \dots \mathbf{x}_N \in \mathbb{Z}^2$ and N non-intersecting edges $\mathbf{e}_n = (\mathbf{x}_n, \mathbf{x}_{n+1})$, where $\mathbf{x}_{N+1}$ stands for $\mathbf{x}_1$, *i.e.* the curve is closed. Given an orientation convention (*e.g.* clockwise), the interior of this curve defines a connected subset $R(\mathcal{X}) \subset \mathcal{P}$ of the image pixel grid (Fig. 7.4, left), which will be denoted R in short when allowed by the context.

Energy E^C is composed of two types of edge potentials ψ_n^{loc} and ψ_n^{glob} that relate to local and global appearance respectively:

$$E^C(\mathcal{X}; \mathcal{I}) := \sum_{n=1}^{N} [\psi_n^{\text{loc}}(\mathbf{e}_n) + \psi_n^{\text{glob}}(\mathbf{e}_n)]. \tag{7.2}$$

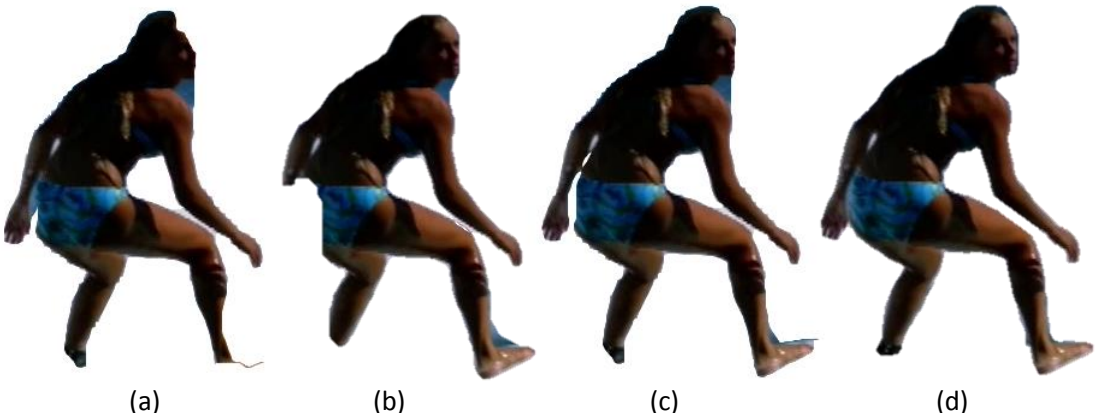


Figure 7.3: Assessing first part of the model. (a) Edge strength only; (b) Global colour model; (c) Edge strength combined with global colour model; (d) With full cost function E^C, including local colour modeling, on frame 13 from *surfer* sequence.

As with classic active contours (Kass *et al.*, 1988), the first type of potential will encapsulate both a simple ℓ_2-regularizer that penalizes stretching and acts as a curve prior (we are not using second-order smoothing in the current model), and a data term that encourages the shape to snap to strong edges. It will in addition capture colour likelihood of pixels on each side of each edge via local appearance models. The second set of potentials results from the transformation of object-wise colour statistics (discrete surface integral) into edge-based costs (discrete line integrals).

Note that, since we do not impose any constraint on the various potentials, the one specified below could be replaced by more sophisticated ones, *e.g.* using semantic edges (Dollar and Zitnick, 2013) instead of intensity gradients, or using statistics of convolutional features (Girshick *et al.*, 2014) rather than colour for local and global appearance modelling.

Local appearance model. Each edge $\mathbf{e}_n$ is equipped with a local appearance model $p_n = (p_n^{\mathrm{f}}, p_n^{\mathrm{b}})$ composed of a fg/bg colour distribution and of a rectangular support R_n, with the edge as medial axis and a fixed width in the perpendicular direction (Fig. 7.4, right). Denoting R_n^{in} and R_n^{out} the two equal-sized parts of R_n that are respectively inside and outside R, we construct a simple edge-based energy term (the smaller, the better) that rewards edge-configurations such that colours in R_n^{in} (resp. R_n^{out}) are well explained by model p_n^{f} (resp. p_n^{g}) and edge $\mathbf{e}_n$ is short and goes through high intensity gradients:

$$\psi_n^{\mathrm{loc}}(\mathbf{e}_n) := -\sum_{\mathbf{p}\in R_n^{\mathrm{in}}} \ln p_n^{\mathrm{f}}(\mathbf{I}_\mathbf{p}) - \sum_{\mathbf{p}\in R_n^{\mathrm{out}}} \ln p_n^{\mathrm{b}}(\mathbf{I}_\mathbf{p}) + \mu\|\mathbf{x}_{n+1} - \mathbf{x}_n\|^2 - \sum_{\mathbf{p}\in\mathbf{e}_n} \lambda\|\nabla\mathcal{I}(\mathbf{p}))\|^2, \quad (7.3)$$

with μ and λ two positive parameters.

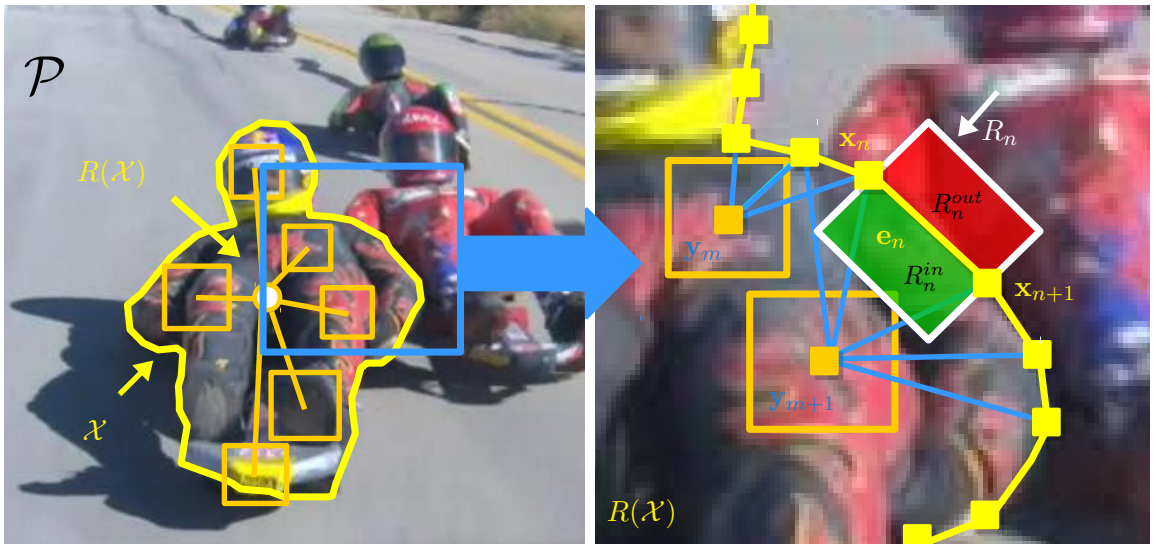


Figure 7.4: Structure and notations of proposed model. (Left) A simple closed curve $\mathcal{X}$ outlines the object region $R(\mathcal{X})$ in the image plane $\mathcal{P}$. Several landmarks, forming a star-shaped graphical model, are defined in this region. (Right) Each edge $\mathbf{e}_n$ of the closed polyline defines a region R_n that staddles $R(\mathcal{X})$; each node $\mathbf{x}_n$ of the polyline is possibly connected to one or several landmarks.

Global appearance model. A global appearance model captures image statistics over the object's interior. As such, it also helps pushing the roto-curve closer to the object's boundary, especially when local boundary terms are not able to explain foreground and background reliably. Defining $p_0 = (p_0^{\mathrm{f}}, p_0^{\mathrm{b}})$ the global fg/bg colour distribution, the bag-of-pixels assumption allows us to define the region energy term

$$\sum_{\mathbf{p}\in R} \ln \frac{p_0^{\mathrm{b}}(\mathbf{I}_\mathbf{p})}{p_0^{\mathrm{f}}(\mathbf{I}_\mathbf{p})}. \tag{7.4}$$

This discrete region integral can be turned into a discrete contour integral using one form of the discrete Green's theorem (Tang, 1982). Using horizontal line integrals for instance, we get

$$\sum_{\mathbf{p}\in R} \ln \frac{p_0^{\mathrm{b}}(\mathbf{I}_\mathbf{p})}{p_0^{\mathrm{f}}(\mathbf{I}_\mathbf{p})} = \sum_{n=1}^{N} \underbrace{\sum_{\mathbf{p}\in\mathbf{e}_n} \alpha_n(\mathbf{p}) Q(\mathbf{p})}_{:=\psi_n^{\mathrm{glob}}(\mathbf{e}_n)}, \tag{7.5}$$

where $Q(\mathbf{p}) = \sum_{\text{“}\mathbf{q}\leqslant\mathbf{p}\text{''}} \ln(p_0^{\mathrm{b}}(\mathbf{I}_\mathbf{p})/p_0^{\mathrm{f}}(\mathbf{I}_\mathbf{p}))$ is the discrete line integral over pixels to the left of $\mathbf{p}$ on the same row, and $\alpha_n(\mathbf{p}) \in \{-1, +1\}$ depends on the direction and orientation, relative to curve's interior, of the oriented edge $\mathbf{e}_n$. In (7.5), the second sum in r.h.s. is taken over the pixel chain resulting from the discretization of the line segment $[\mathbf{x}_n, \mathbf{x}_{n+1}]$ with the final vertex excluded to avoid double-counting.

7.3.2 Landmark-based modelling: E^L

Our model also makes use of a set $\mathcal{Y}$ of M distinctive landmarks $\mathbf{y}_1 \dots \mathbf{y}_M \in R(\mathcal{X})$ detected inside the object of interest. Similarly to pictorial structures (Felzenszwalb *et al.*, 2010), these landmarks form the leaves of a star-shaped graphical model[4] with a virtual root-node $\mathbf{y}_0$. This part of the model is defined by leaf potentials $\phi_m(\mathbf{y}_m)$ and leaf to root potentials $\varphi_m(\mathbf{y}_0, \mathbf{y}_m)$:

$$E^L(\mathcal{Y}; \mathcal{I}) := \sum_{m=1}^{M} \phi_m(\mathbf{y}_m) + \sum_{m=1}^{M} \varphi_m(\mathbf{y}_0, \mathbf{y}_m). \tag{7.6}$$

Each landmark is associated with a model, *e.g.* a template or a filter, that allows the computation of a matching cost at any location in the image. The leaf potential $\phi_m(\mathbf{y}_m)$ corresponds to the negative matching cost for m-th landmark. The pairwise potentials φ_m penalize the difference in ℓ_2-norm between the current configuration and the one, $\hat{\mathcal{Y}}$, estimated in previous frame:

$$\varphi_m(\mathbf{y}_0, \mathbf{y}_m) = \frac{1}{2}\|\mathbf{y}_m - \mathbf{y}_0 - \hat{\mathbf{y}}_m + \hat{\mathbf{y}}_0\|^2. \tag{7.7}$$

7.3.3 Curve-landmarks interaction: E^J

The joint energy $E^J(\mathcal{X}, \mathcal{Y})$ captures correlation between object's outline and object's landmarks. Based on proximity, shape vertices and landmarks can be associated. Let $n \sim m$ denote the pairing of vertex $\mathbf{x}_n$ with landmark $\mathbf{y}_m$. Energy term E^J decomposes over all such pairs as:

$$E^J(\mathcal{X}, \mathcal{Y}) = \sum_{n \sim m} \xi_{nm}(\mathbf{x}_n, \mathbf{y}_m). \tag{7.8}$$

For each pair $n \sim m$, the interaction potential is defined as:

$$\xi_{mn}(\mathbf{x}_n, \mathbf{y}_m) = \frac{1}{2}\|\mathbf{x}_n - \mathbf{y}_m - \boldsymbol{\mu}_{mn}\|^2, \tag{7.9}$$

where $\boldsymbol{\mu}_{mn}$ is the landmark-to-vertex shift vector in the first image.

7.4 Using ROAM

Sequential alternating inference. Using ROAM to outline the object of interest in a new image amounts to solving the discrete optimization problem:

$$\min_{\mathcal{X}, \mathcal{Y}} E(\mathcal{X}, \mathcal{Y}; \mathcal{I}), \tag{7.10}$$

[4]The star shape is used for its simplicity but could be replaced by another tree-shaped structure.

where E is defined by (7.1) and depends on previous curve/landmarks configuration $(\hat{\mathcal{X}}, \hat{\mathcal{Y}})$ through several of its components. Although this problem could be formulated as an integer linear program, we opt for a simpler alternating optimization with exact minimization at each step which converges within a few iterations.

In the first step, we fix the roto-curve $\mathcal{X}$ and find the best configuration of landmarks $\mathcal{Y}$ using dynamic programming. An essxact solution for such a problem can be obtained in two passes, solving

$$\min_{\mathbf{y}_0} \min_{\mathbf{y}_{1:M}} \sum_{m=1}^{M} \left(\phi_m(\mathbf{y}_m) + \varphi_m(\mathbf{y}_0, \mathbf{y}_m) + \sum_{n \sim m} \xi_{mn}(\mathbf{x}_n, \mathbf{y}_m) \right). \tag{7.11}$$

A default implementation leads to complexity $\mathcal{O}(MS^2)$, with S the size of individual landmark state-spaces, *i.e.* the number of possible pixel positions allowed for each. However, the quadratic form of the pairwise terms allows making it linear in the number of pixels, *i.e.* $\mathcal{O}(MS)$, by resorting to the generalized distance transform (Felzenszwalb *et al.*, 2010).

In the second step, we fix the landmarks $\mathcal{Y}$ and find the best configuration of contour $\mathcal{X}$. This is a classic first-order active contour problem. Allowing continuous values for node coordinates, a gradient descent can be conducted with all nodes being moved simultaneously at each iteration. We prefer discrete approach, whereby only integral positions are allowed and dynamic programming can be used (Amini *et al.*, 1990). In that formulation, exact global inference is theoretically possible, but with a prohibitive complexity of $\mathcal{O}(NP^3)$, where $P = \mathrm{card}(\mathcal{P})$ is the number of pixels in images. We follow the classic iterative approach that considers only D possible moves $\Delta\mathbf{x}$ for each node around its current position. For each of the D positions of first node $\mathbf{x}_1$, the Viterbi algorithm provides the best moves of all others in two passes and with complexity $\mathcal{O}(ND^2)$. Final complexity is thus $\mathcal{O}(ND^3)$ for each iteration of optimal update of previous contour, solving:

$$\min_{\Delta\mathbf{x}_1} \min_{\Delta\mathbf{x}_{2:N}} \sum_{n=1}^{N} \left(\psi_n^{\mathrm{loc}}(\mathbf{e}_n + \Delta\mathbf{e}_n) + \psi_n^{\mathrm{glob}}(\mathbf{e}_n + \Delta\mathbf{e}_n) + \sum_{m \sim n} \xi_{mn}(\mathbf{x}_n + \Delta\mathbf{e}_n, \mathbf{y}_m) \right). \tag{7.12}$$

Note that sacrifying optimality of each update, the complexity could even been reduced as much as $\mathcal{O}(ND)$ (Williams and Shah, 1992).

Given some initialization for $(\mathcal{X}, \mathcal{Y})$, we thus alternate between two *exact* block-wise inference procedures. This guarantees convergence toward a local minima of joint energy $E(\mathcal{X}, \mathcal{Y}; \mathcal{I})$. Also, the complexity of each iteration is linear in the number of vertices and landmarks, linear in the number of pixels, and cubic in the small number of allowed moves for a curve's vertex.

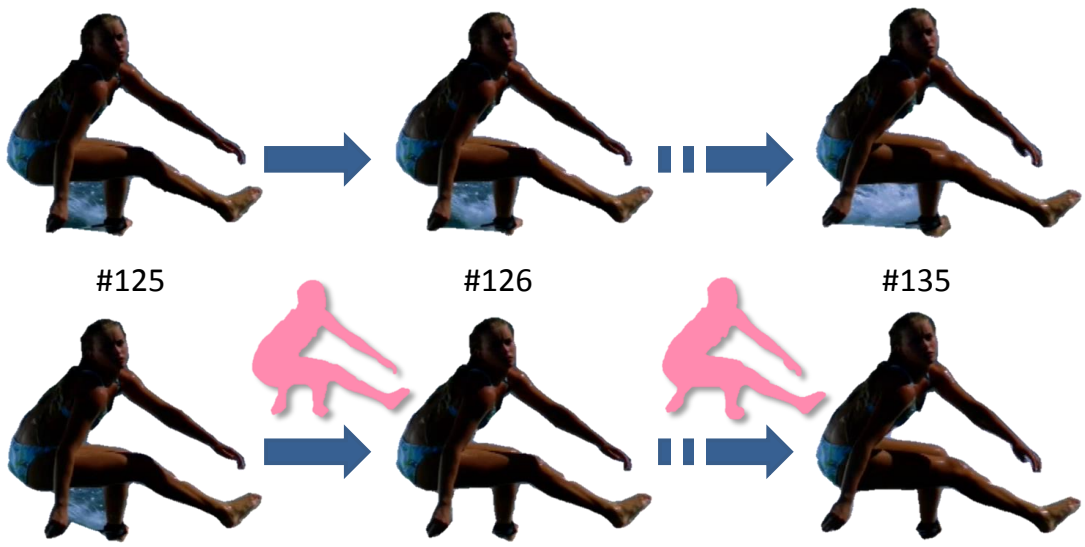


Figure 7.5: Using proposals based on graph-cut: Proposals (in pink) obtained through graph-cut minimization of an instrumental labeling energy using current colour models allows ROAM to monitor and accommodate drastic changes of object's outline (Bottom). Without this mechanism, parts of surrounding water get absorbed in surfer's region, between the leg and the moving arm (Top).

Online learning of appearance models. Local fg/bg colour models p_ns and global colour model p_0 are GMMs. Given the roto-curve in the initial frame, these GMMs are first learned over region pairs $(R_n^{\text{in}}, R_n^{\text{out}})$s and $(R, \mathcal{P} \setminus R)$ respectively and subsequently adapted through time using Stauffer and Grimson's classic technique (Stauffer and Grimson, 1999).

Selection and adaption of landmarks. A pool of distinctive landmarks is maintained at each instant. They can be any type of classic interest points. In order to handle texture-less objects, we use maximally stable extremal regions (MSERs) (Matas *et al.*, 2004). Each landmark is associated with a correlation filter whose response over a given image area can be computed very efficiently (Henriques *et al.*, 2015). At any time, landmarks whose filter response is too ambiguous are deemed insufficiently discriminative and removed from the current pool in the same way tracker loss is monitored in (Henriques *et al.*, 2015). The collection is re-populated through new detections. Note that correlation filters can be computed over arbitrary features and kernelized (Henriques *et al.*, 2015); for simplicity, we use just grayscale features without kernel function.

Allowing topology changes. Using a closed curve is crucial to comply with rotoscoping workflows and allows the definition of a rich appearance model. Also, it prevents abrupt

changes of topology. While this behavior is overall beneficial (See §7.5), segmenting a complete articulated 3D object as in Fig. 1 might turn difficult. Roto-artists naturally handle this by using multiple independent roto-curves, one per meaningful part of the object. As an alternative for less professional, more automatic scenarios, we propose to make ROAM benefit from the best of both worlds: standard graph-cut based segmentation (Boykov and Jolly, 2001), with its superior agility, is used to *propose* drastic changes to current curve, if relevant. Space-dependent unaries are derived in ad-hoc way from both global and local colour models and combined with classic contrast-dependent spatial regularization.[5] The exact minimizer of this instrumental cost function is obtained through graph-cut (or its dynamic variant for efficiency (Kohli and Torr, 2007)) and compared to the binary segmentation associated to the current shape $\mathcal{X}$. At places where the two differ significantly, a modification of current configuration (displacement, removal or addition of vertices) is proposed and accepted if it reduces the energy $E(\mathcal{X}, \mathcal{Y}; \mathcal{I})$.

[5]Note that this instrumental energy is too poor to compete on its own with the proposed model, but is a good enough proxy for the purpose of proposing possibly interesting new shapes at certain instants. It is also very different from the one in the final graph-cut of VIDEO SNAPCUT where unaries are based on the already computed soft segmentation to turn it into a hard segmentation. Also, graph-cut segmentation is the final output in VIDEO SNAPCUT, unless further interaction is used, while we only use it to explore alternative topologies under the control of our joint energy model.

7.5 Results

We report experimental comparisons that focus on the minimum input scenario: an initial object selection (curve or mask, depending on the tool) is provided to the system and automatic object segmentation is produced in the rest of the shot.[6] We do not consider additional user interactions.

Datasets. We evaluate our approach on the recent CPC rotoscoping dataset (Lu *et al.*, 2016). It contains 9 videos consisting of 60 to 128 frames which represents typical length of shots for rotoscoping. These sequences were annotated by professional artists using standard post-production tools. We also provide qualitative results on ROTO++ (Li *et al.*, 2016b) dataset for which the authors have not released the ground-truth yet, as well as from the VIDEO SNAPCUT dataset (Bai *et al.*, 2009).

We also use the DAVIS dataset (Perazzi *et al.*, 2016) which comprises 50 challenging sequences with a wide range of difficulties: large occlusions, long-range displacements, non-rigid deformations, camouflaging effects and complex multi-part objects. Let us note that this dataset is intended to benchmark pixel-based video segmentation methods, not rotoscoping tools based on roto-curves.

Evaluation Metrics. We use standard video segmentation evaluation metrics and report the average *accuracy*, *i.e.* the proportion of ground-truth pixels that are correctly identified, and the more demanding average *intersection-over-union* (IoU), *i.e.* the area of the intersection of ground-truth and extracted objects over the area of their union. We also report runtimes and evolution of IoU as sequences proceed.

Baselines. We compare with several state-of-the-art methods. Our main comparison is with recent approaches that rely on a closed-curve, *i.e.* CPC (Lu *et al.*, 2016) and ROTO++ (Li *et al.*, 2016b). We initialize all methods with the same object and measure their performance over the rest of each sequence. Since ROTO++ requires at least two key-frames to benefit from its online shape model, we report scores with letting the method access the ground-truth of the last frame as well. We also run it with the initial keyframe only, a configuration in which ROTO++ boils down to the Blender planar tracker.

In addition to that, we also compare with two approaches based on pixel-wise labelling: JUMPCUT (Fan *et al.*, 2015) and VIDEO SNAPCUT (Bai *et al.*, 2009) as implemented in After

[6] Video results are available at https://youtu.be/UvO7IacS9pQ

Table 7.1: Quantitative evaluation on CPC dataset (*: partial evaluation only, see text)

Method	Avg. Accuracy	Avg. IoU	Time (s) / frame min	max	avg
GCUT (ROTHER *et al.*, 2004) + KCF (HENRIQUES *et al.*, 2015)	.891	.572	0.394	0.875	0.455
AE ROTOBRUSH (Bai *et al.*, 2009)	.992	.895	—	—	—
ROTO++(1 keyframe) (Li *et al.*, 2016b)	.969	.642	—	—	0.108
ROTO++(2 keyframes) (Li *et al.*, 2016b)	.974	.691	—	—	0.156
CPC (Lu *et al.*, 2016)	.998*	.975*	—	—	—
NLCV (Faktor and Irani, 2014)	.896*	.194*	—	—	—
BSVS (Märki *et al.*, 2016)	.991	.872	—	—	—
OBJECTFLOW (Tsai *et al.*, 2016)	.968	.502	—	—	—
ROAM: Baseline Conf.	.993	.932	0.011	0.155	0.040
ROAM: Lean Conf.	**.995**	.938	0.092	0.377	0.102
ROAM: Medium Conf.	**.995**	.939	0.279	0.875	0.652
ROAM: Full Conf.	**.995**	**.951**	0.874	8.78	3.052

Effect ROTOBRUSH and three recent video-segmentation approaches (Faktor and Irani, 2014; Märki *et al.*, 2016; Tsai *et al.*, 2016). As a naive baseline, we use a combination of a bounding-box tracker (Henriques *et al.*, 2015) and GRABCUT (Rother *et al.*, 2004).

Ablation study. To evaluate the importance of each part of our model, we consider 4 different configurations:

- Baseline: negative gradient with ℓ_2-regularizer;
- Lean: baseline + local appearance model;
- Medium: lean + landmarks;
- Full: medium + automatic re-parametrization and global appearance model;

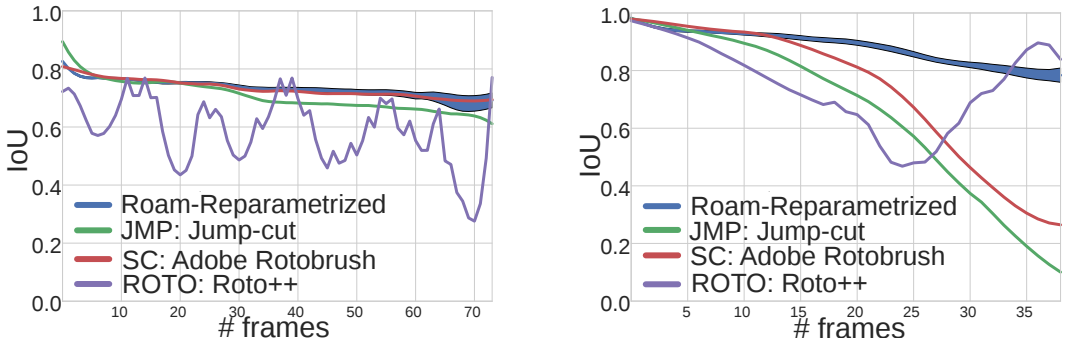
For all configurations, we used cross-validation (maximizing the mean IoU) on the training fold of the DAVIS dataset to set the parameters and kept them fixed for all experiments.

Quantitative results. The quantitative results for the CPC dataset are summarized in Tab. 7.1. While average accuracy is quite similar and saturated for all methods, all configurations of ROAM outperform all baselines. In terms of IoU, all versions of ROAM significantly outperform all others with the full configuration being the best. The reason why landmarks ("medium conf.") do not add much to ROAM is that the CPC dataset does not exhibit many large displacements.

Table 7.2: Quantitative comparisons on DAVIS dataset

Method	Average Accuracy			Average IoU			Time / frame (s)		
	Validation set	Training set	Full set	Validation set	Training set	Full set	min	max	avg
GRABCUT+ KCF Tracker	0.896	0.914	0.907	0.277	0.296	0.289	0.405	0.675	0.461
JUMPCUT (Fan *et al.*, 2015)	0.952	0.957	0.956	0.570	0.632	0.616	—	—	—
AE ROTOBRUSH (Bai *et al.*, 2009)	0.951	0.942	0.946	0.533	0.479	0.500	—	—	—
ROTO++ (single keyframe)(Li *et al.*, 2016b)	0.910	0.922	0.917	0.248	0.310	0.284	—	—	0.118
ROTO++ (two keyframes) (Li *et al.*, 2016b)	0.925	0.933	0.930	0.335	0.394	0.358	—	—	0.312
NLCV (Faktor and Irani, 2014)	0.948	0.963	0.957	0.551	0.701	0.641	—	—	—
BSVS (Märki *et al.*, 2016)	**0.966**	**0.974**	**0.971**	**0.683**	0.709	0.665	—	—	—
OBJECTFLOW (Tsai *et al.*, 2016)	—	—	—	0.600	**0.732**	**0.711**	—	—	—
ROAM: Baseline Conf.	0.930	0.937	0.932	0.358	0.385	0.377	0.017	0.113	0.049
ROAM: Lean Conf.	0.935	0.937	0.936	0.409	0.417	0.412	0.187	0.641	0.342
ROAM: Medium Conf.	0.942	0.952	0.948	0.532	0.591	0.564	0.302	1.785	0.746
ROAM: Full Conf.	0.952	0.956	0.953	0.583	0.624	0.615	0.546	7.952	3.058

Figure 7.6: Qualitative results on the DAVIS **dataset**: Comparisons on *blackswan* and *car-roundabout* sequences, between (from top to bottom for each sequence): JUMPCUT, ROTOBRUSH, ROTO++ and ROAM.



(a) Boat **(b)** Car-shadow

Figure 7.7: Evolution of IoU for different sequences of the DAVIS **dataset**. For our method, the blue shadow indicates influence of varying the label space size for each node (set of possible moves in dynamic programming inference).

The CPC method (Lu *et al.*, 2016) was evaluated only on the first ten frames of each sequence since their authors have released results only on these frames and have not yet released their code. Hence, the scores reported in Tab. 7.1 for CPC are based on partial videos, as opposed to the scores for all the other methods (including ours). When similarly restricted to the first 10 frames, ROAM performs on par with CPC for all the sequences except "drop" sequence. This sequence shows a water drop falling down – a transparent object, making color models (both local and global) useless if not harmful, and exhibiting a very smooth round shape. For this sequence, the CPC method (Lu *et al.*, 2016) performs better since it uses Bézier curves and relies solely on the strength of the image gradients.

Results for the DAVIS dataset are reported in Tab. 7.2. While our method is on par with JUMPCUT (pixel-wise labelling), we again significantly outperform ROTO++ by almost 25 IoU points (note that using ROTO++ with only two keyframes is not a typical scenario, however, this shows how complementary our approaches are). Although (Märki *et al.*, 2016) is better by 100 and (Tsai *et al.*, 2016) by 17 IoU points on DAVIS, our model outperforms (Märki *et al.*, 2016) by 80 and (Tsai *et al.*, 2016) by 450 points on the CPC. In other words, our approach should in the worst case be considered on par. However, we would like to stress that (Faktor and Irani, 2014; Märki *et al.*, 2016; Tsai *et al.*, 2016) are not our (main) competitors since all are based on pixel-wise labelling and as such **cannot** provide the same flexibility for rotoscoping as the closed contour counterpart (Li *et al.*, 2016b). Note, that we could not provide more quantitative comparisons since results/implementations of other methods were not available from the authors. In particular, comparison with the CPC method (Lu *et al.*, 2016) would be interesting since the DAVIS dataset (Perazzi *et al.*,

Figure 7.8: Qualitative results on four sequences from the ROTO++ dataset

2016) exhibits many large displacements and major topology changes.

Comparing the different configurations of ROAM – local appearance models add 3 points, landmarks 15 and global model with re-parametrization another 5 points – demonstrates the importance of all components of our framework. To examine behaviour of each method in detail, we report IoU score for each frame in Fig. 7.7, with the effect of varying the size of the label space in ROAM (from windows of 3×3 to 13×13 pixels) represented with a blue shadow. It can be seen that ROAM is more robust in time, not experiencing the sudden performance drops of others.

Importance of landmarks and warping. Using alternating optimization has one more benefit. We can use the predicted position of landmarks in the next frame to estimate the transformation between the two and "warp" the contour to the next frame. This allows us to reduce the number of D possible moves of nodes which i) significantly speeds-up the algorithm, ii) allows us to handle large displacement and iii) allows to better control non-rigid deformations.

We have experimented with three settings for warping of contour: a smoothed optical flow masked as in (Pérez-Rúa *et al.*, 2016), moving each node by averaging the motion of all landmarks connected to a given node and similarity transformation robustly estimated with RANSAC from positions of landmarks. Table 7.3 and Fig. 7.9 show the effect of using the last option.

Table 7.3: Different types of contour warping for handling long displacements on a subset of sequences of the DAVIS dataset.

Warping method	Average Accuracy	Average IoU
Optical flow	0.878	0.312
Node projection from landmark tracking	0.906	0.480
Robust rigid transformation from landmarks	**0.934**	**0.581**

Landmarks Frame 15 Frame 25

Figure 7.9: Benefit of landmarks-based modeling. Automatically detected landmarks (orange bounding boxes) are accurately tracked on the *plane* sequence. This further improves the control of the boundary (bottom), as compared to ROAM without landmarks (top).

Global colour models and reparametrization. We investigated the effects of adding reparametrization and global colour models to our framework. The numeric benefits of these elements can be seen in Tab. 7.2 and qualitative results on the *surfer* sequence from VIDEO SNAPCUT dataset are provided in Figs. 7.5 and 7.3. Observe that the local colour models are a powerful way to capture local appearance complexities of an object through a video sequence. However, self-occlusions and articulated motion can cause these models to fail (right arm crossing the right leg of the surfer). Our contour reparametrization allows the efficient handling of this situation. Furthermore, the beneficial effect of the global colour models can be observed in Fig. 7.3, where the right foot of the surfer is successfully tracked along the whole video.

Figure 7.10: More qualitative results on the DAVIS **dataset**: Comparisons on *scooter-gray* and *car-shadow* sequences between (from top to bottom): JUMPCUT, ROTOBRUSH, ROTO++ and ROAM.

Qualitative results. Result samples on several sequences from DAVIS dataset in Fig. 7.6 demonstrate the superior robustness of ROAM compared to other approaches when rotoscoping of the first image only is provided as input (and last image as well for ROTO++). Additional results obtained by ROAM on full videos from the DAVIS (Perazzi *et al.*, 2016), CPC (Lu *et al.*, 2016), VIDEO SNAPCUT (Bai *et al.*, 2009), and ROTO++ (Li *et al.*, 2016b) datasets are provided in Figs. 7.8, 7.10, 7.12 and supplementary video.

Timing breakdown. Table 7.4 provides detailed timing breakdown for our algorithm. These timings were obtained on an Intel Xeon 32@3.1GHz CPU machine with 8GB RAM and Nvidia GeForce Titan X GPU. Note that only part of the approach (evaluation of various potentials and dynamic programming) was run on the GPU. In particular, the re-parametrization steps could also be easily run on the graphics card, yielding real-time performance.

Table 7.4: Timing details (seconds / frame) for full configuration of ROAM.

Step	Min.	Max.	Avg.
DP Contour	0.018	0.113	0.084
DP Landmarks	0.003	0.072	0.052
Local models edge terms	0.342	0.671	0.581
Other terms	0.012	0.015	0.013
Reparametrization	0.032	7.403	2.226

Convergence. Fig. 7.11 demonstrates that the alternating optimization described in §7.4 converges quickly within a few iterations.

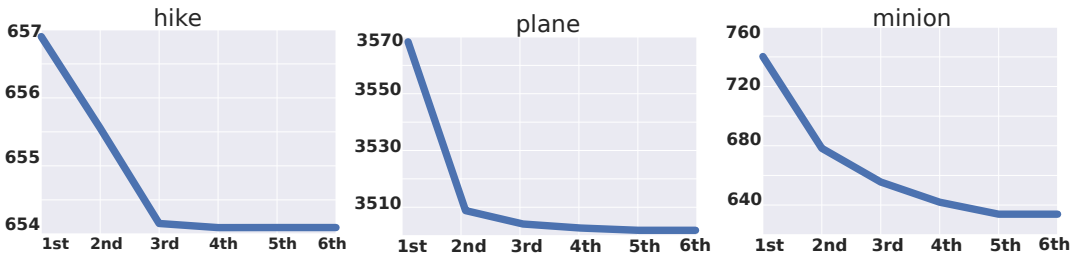


Figure 7.11: Energy vs. number of iterations on three sequences from the experimental datasets.

Figure 7.12: More qualitative results: Output of ROAM on very different sequences from DAVIS, CPC and VIDEO SNAPCUT datasets among others.

Parameters and reproducibility. The relative weights of the local and global appearance are 50 and 0.002 resp., $\lambda = 1.0$, $\mu = 0.75$, $P = 9^2$, $M = 20$ and other weights are set to 1. The full source code is available on our website. As can be seen from our ablation study, all terms contribute to the energy and none of them dominates. Confidence intervals in Fig. 7 suggest that our model is relatively parameter agnostic. Note that our fast alternating optimization with exactly solved blocks should allow us to learn parameters in the future.

7.6 Conclusion

We have introduced ROAM, a model to capture the appearance of the object defined by a closed curve. This model is well suited to conduct rotoscoping in video shots, a difficult task of considerable importance in modern production pipelines. We have demonstrated its merit on various competitive benchmarks. Beside its use within a full rotoscoping pipeline, ROAM could also be useful for various forms of object editing that require both accurate enough segmentation of arbitrary objects in videos and tracking through time of part correspondences, *e.g.* (Khan *et al.*, 2006; Rav-Acha *et al.*, 2008). Due to its flexibility, ROAM can be easily extended; in particular, with the recent ROTO++ and its powerful low-dimensional shape model.

Seeing it from perspective of 2017. The first drawback of our model is that it uses only simple RGB features. This deficiency can be mitigated by parameterizing the potential functions of appearance models by weights $\mathbf{w}$ which can be learnt in a structured output learning framework. An appealing property of this approach is that parameters $\mathbf{w}$ can be incorporated directly into CNN functions and learnt end-to-end (Jaderberg *et al.*, 2014). Online adaptation of local models can then be re-formulated as a feed-forward one-shot learner (Bertinetto *et al.*, 2016a).

Another limitation is that except for pre-warping of the contour, we do not use any information about motion (*e.g.* optical flow) at all. Last but not least, real rotoscoping application should support "keyframes" to *guide* or constrain contour deformations over longer sequences.

8 Playing Doom with SLAM-Augmented Deep Reinforcement Learning

In the previous chapters, we have developed an intermediate representation for decision making; specifically a dense large-scale semantic 3D map. However, we have followed the common evaluation metric, the Intersection over Union (IoU) score and thus far assumed that such representation is useful for decision making. In this chapter, we will address this deficiency and close the loop by showing how such a representation improves decision making of an agent.

A number of recent approaches to policy learning in 2D game domains have been successful going directly from raw input images to actions. However when employed in complex 3D environments, they typically suffer from challenges related to partial observability, combinatorial exploration spaces, path planning, and a scarcity of rewarding scenarios. Inspired from prior work in human cognition that indicates how humans employ a variety of semantic concepts and abstractions (object categories, localisation, etc.) to reason about the world, we build an agent-model that incorporates such abstractions into its policy-learning framework. We augment the raw image input to a Deep Q-Learning Network (DQN), by adding details of objects and structural elements encountered, along with the agent's localisation. The different components are automatically extracted and composed into a topological representation using on-the-fly object detection and 3D-scene reconstruction. We evaluate the efficacy of our approach in "Doom", a 3D first-person combat game that exhibits a number of challenges discussed, and show that our augmented framework consistently learns better, more effective policies.

8.1 Introduction

Recent approaches to policy learning in games (Mnih *et al.*, 2015, 2013) have shown great promise and success over a number of different scenarios. A particular feature of such approaches is the ability to take the visual game state directly as input and learn a mapping to actions such that the agent effectively explores the world and solves predetermined tasks. Their success has largely been made possible thanks to the ability of deep reinforcement learning (deepRL) networks, neural networks acting as function approximators within the reinforcement-learning framework. A particular variant, Deep Q-Learning Networks (DQN), has been widely used in a range of different settings with excellent results. It employs convolutional neural networks (CNN) as a building block to effectively extract features from the observed images, subsequently learning policies using these features.

For the majority of scenarios that have been tackled thus far, a common characteristic has been that the domain is 2-dimensional. Here, going directly from input image pixels to learned policy works well due to two important factors: i) a reasonable amount of the game's state is directly observable in the image, and ii) a combination of a lower-dimensional action space and smaller exploration requirements result in a smaller search space. The former ensures that the feature extraction always has sufficient information to influence policy learning, and the latter makes learning consistent features easier. Despite stellar success in the 2D domain, these models struggle in more complicated domains such as 3D games.

3D domains exhibit a multitude of challenges that cause the standard approaches discussed above to struggle. The introduction of an additional spatial dimension, first introduces notions of partial observability and occlusions, and secondly causes complications due to viewpoint variance. Not only is the agent viewing a relatively smaller portion (volume) of the environment, it also must reconcile observing a variety of other objects in different contexts under projective transformations. Furthermore, adding an extra dimension also *combinatorially* complicates matters in terms of exploration of the environment. This typically manifests itself in the form of *sparse feedback* in the learning process because the agent's inability to explore the environment directly penalizes its learning capacity. Moreover, complications in exploration directly affect any planning that may be required for tasks and actions. Finally, with larger search and state spaces comes the likelihood that any rewards that might help move learning along are also harder to come by.

Sutton *et al.* (1999) propose an extension of the RL framework that can potentially learn hierarchical policies. However this, and similar methods, have not been able to scale beyond

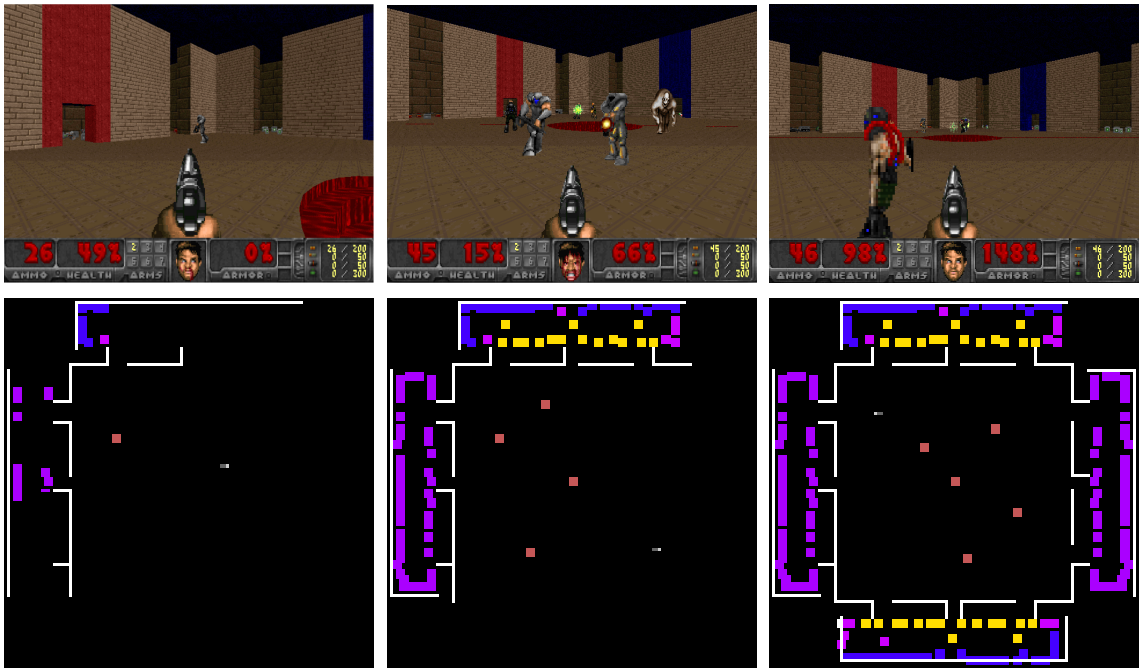


Figure 8.1: Motivation: As the agent explores the environment, the first-person-view (top) only sees a restricted portion of the scene, whereas in the semantic map (bottom), the effect of exploration is cumulative, indicating both *type* and *position*.

small gridworld domains (Barto and Mahadevan, 2003). Kulkarni *et al.* (2016) proposes to tackle environments with delayed rewards by coupling options learning and intrinsically driven exploration methods. Options are however notoriously hard to train, requiring a great deal of effort before intrinsically motivated agents can safely deal with generic hierarchical spatial domains.

Prior work in behavioural modelling and cognitive neuroscience suggests that humans employ particular, highly specialised mechanisms to construct representations of, and reason about, the world. These typically take the form of semantic concepts and abstractions such as object identity, categories, and localisation. Freedman and Miller (2008) review evidence from neurophysiology that explore the learning and representation of object categories. Burgess (2008) discusses evidence from neuroscience for the presence and combination of different viewpoints (*e.g.* egocentric) and the role of representing layouts (*e.g.* boundaries and landmarks) in the spatial cognition process. Moser *et al.* (2008) also discuss the presence of highly specialised representation regions in the brain that encode localisation and spatial reasoning. Denis and Loomis (2007) provide a review from behavioural psychology on the subject of spatial cognition and related topics.

In this chapter, we take inspiration from such work to propose a system that augments

the raw image input with explicitly constructed semantic and topological representation of the state (Fig. 8.1), in an attempt to learn policies more effectively in such complex 3D domains. To this end, we construct a novel model that incorporates an automatic, on-the-fly scene reconstruction component into a standard deep-reinforcement learning framework. Our work provides a streamlined system to immediately enhance current state-of-the-art learning algorithms in 3D spatial domains, additionally obtaining insight on the efficacy of spatially enhanced representations against those learned in a purely bottom-up manner.

8.2 Related work

We have provided an overview of the state of the art methods for large-scale semantic 3D mapping in Section §3.2. Hence, we discuss only deep reinforcement learning and object detection with which we replace segmentation in this chapter.

8.2.1 Deep Reinforcement Learning

Reinforcement Learning is a commonly employed set of techniques for learning agents that can execute generic and interactive decision making. Its mathematical framework is based on *Markov Decision Processes* (MDPs). An MPD is a tuple $(S, A, P, \mathcal{R}, \gamma)$, where S is the set of states, A is the set of actions the agent can take at each time step t, P is the transition probability of going from state s to s' using action a, $\mathcal{R}$ is the reward function defining the signal the agent receives after taking actions and changing states, and γ is a discount factor. The goal of Reinforcement Learning is to learn a policy $\pi : s \rightarrow a$ that maximises the expected discounted average reward over the agent run. A commonly used technique to learn such a policy is to learn the action-value function $Q^{\pi}(s, a)$ iteratively, so as to gradually approximate the expected reward in a model-free fashion.

They have, however, traditionally struggled with high-dimensional environments, due in large part to the curse of dimensionality. Deep Reinforcement Learning algorithms such as Deep-Q Networks extend model-free RL algorithms like Q-Learning to use Deep Neural Networks as function approximators, implicitly capturing hierarchies in the state representation that make the RL problem scale even to visual input states. Unfortunately, they still suffer from some of the problems that standard RL cannot deal with:

- Delayed reward requires non-stochastic exploration strategies (Kulkarni *et al.*, 2016).
- Learning to abstract policies hierarchically is currently an unsolved but key problem to make RL scale to tasks requiring long-term planning (Barto and Mahadevan, 2003).

- Partial observability in state requires models that can encode at least short-term memory (Hausknecht and Stone, 2015).

Some recent work has also explored ways to develop agents that can learn to play Doom. Lample and Chaplot (2016) take the approach of using a variant of DRQN (Hausknecht and Stone, 2015) together with some game features extracted directly from the game environment through its data structures. Our method is similar in spirit, but can be applied to any environment with a significant 3D navigation component, as our SLAM and object recognition pipeline is not intrinsically dependent to the VizDoom platform. Another interesting approach is the one presented by Dosovitskiy and Koltun (2017). This approach, featured as the winner in the VizDoom competition (Jakowski *et al.*, 2016), changed the supervision signal from a single scalar reward to a vector of measurements provided by the game engine. This is used to train a network that, given the visual input, the current measurements, and the goal, predicts future measurements. The action to perform is then chosen greedily according the predicted future measurements. This is orthogonal to our approach; both algorithms could benefit from the novelties introduced by the other, however we leave such an extension for future work.

8.2.2 Object detection

Early approaches to object detection include constellation models and pictorial structures (Fischler and Elschlager, 1973). The very first object detector capable of real-time detection rates was (Viola and Jones, 2004), who solved an inherent problem of sliding-window approaches by learning a sequential decision process that rapidly rejects locations which are unlikely to contain any objects. This concept has since then evolved into a distinct set of algorithms called *proposals*, whose only goal is to quickly localize potential objects (Hosang *et al.*, 2016). These locations are then fed into more complex classifiers to determine the class label (or assign a background). Deformable part models (Felzenszwalb *et al.*, 2010) are a prominent example of such, being able to represent highly variable object classes. Recently, it has been shown that deformable part models can be interpreted as a CNN (Girshick *et al.*, 2015), which led to replacement of handcrafted features by convolutional feature maps (Girshick, 2015). Finally the Faster-RCNN (Ren *et al.*, 2017) combines the region proposals and object detector into a single unified network trainable end-to-end with shared convolutional features which leads to very fast detection rates.

8.3 Semantic Mapping

In this chapter, we introduce an algorithm based on the Deep Q Network (DQN) that has been successfully applied to many Atari games (Mnih *et al.*, 2015). Inspired by prior work in human cognition that indicates how humans employ a variety of semantic concepts and abstractions (object categories, localization, *etc.*) to reason about the world, we build an agent-model that incorporates such abstractions into its policy-learning framework. We augment the first-person raw image input to a DQN by adding details about objects and structural elements encountered, along with the agents localization to cope with complex 3D environments. This is represented as a 2D map (top-down view) encoding three distinct sources of information: i) positions of static structures and obstacles such as walls, ii) position and orientation of the agent, and iii) positions and class labels of important objects such as health packs, weapons and enemies. Our representation is being updated over time as the agent explores the environment. This allows the agent to keep information about areas observed in the past and build an aggregated model of the 3D environment (Fig. 8.1). Such representation allows the agent to behave properly even with respect to the elements no longer present in the first-person view.

Semantic representation. As the agent explores the environment, we simultaneously estimate localization of the agent and obstacles (*e.g.* walls) in order to build the map of the surrounding 3D environment from the first-person-view at each frame. In parallel, we detect important objects in the scene such as weapons and ammunition. And since we want to minimize the dimensionality of the augmented representation to allow more efficient learning, we project all semantic information onto a single common 2D map of a fixed size. Essentially a "floor-plan" with the positions of objects and agents. This is achieved by encoding different entities by different gray-scale values, in the form of heat-maps (*cf.* bottom right of Fig. 8.1).

Our representation encodes position of walls and obstacles (white) extracted directly from the depth data provided by the VizDoom API. Information about agent's position and orientation on the 2D map is represented as a green directed arrow. We also want to provide the agent semantic information about a variety of objects present in the environment. For Doom, we encode the following five object categories: monsters (red), health packs (purple), high-grade weapons (violet), high-grade ammunition (blue), other weapons and other ammunition (yellow). Since these objects could either move or be picked up by another

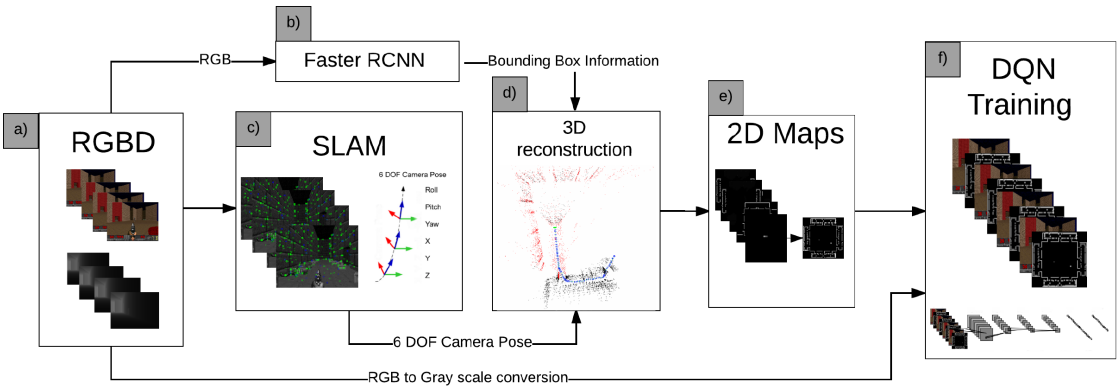


Figure 8.2: System overview: (a) Observing image and depth from VizDoom. Running Faster-RCNN (b) for object detection and SLAM (c) for pose estimation. Doing the 3D reconstruction (d) using the pose and bounding boxes. Semantic maps are built (e) from projection and the DQN is trained (f) using these new inputs.

player (*e.g.* deathmatch scenario), we project only objects visible in the current view onto the common map. This could be addressed by more advanced data association techniques such as (Wang *et al.*, 2007; Bibby, 2010), but this is beyond the scope of this chapter.

8.4 Recognition and Reconstruction

Fig. 8.2 depicts the architecture of our pipeline for automatic on-the-fly creation of semantic maps. As input, we use the image data provided by the VizDoom API, *i.e.* RGB video frames visualizing the 3D environment from agents (first person) perspective and a z-buffer providing depth information of the observed scene. In order to build a map of the 3D environment, we need to detect and remove all objects from the z-buffer since we want to i) provide explicit semantic information about various objects (monsters, weapons, *etc.*) and ii) avoid nuisance visual events such as weapon discharges in the depth buffer. We also need to know the current pose of the camera, so we run a camera-pose tracker in parallel with the object detector. Then, we project the observed scene on a common 3D map and provide its 2D visualization (top-down view) to the agent. Note, that the mapping system could work even without access to the z-buffer, *i.e.* using solely the RGB data (Eigen *et al.*, 2014). We now describe the components of our pipeline (object detection, camera pose estimation and map fusion) in greater detail.

8.4.1 Object detection

To detect the objects, we use the Faster-RCNN object detector (Ren *et al.*, 2017), which is a convolutional network that combines the attention mechanism (region proposals) and object detector into a single unified network, trainable end-to-end. The first module is a deep fully-convolutional network that simultaneously predicts object bounds and objectness scores at each position, and the second module is the Fast R-CNN detector (Girshick, 2015) that uses the proposed regions. Since both modules share the same features, it offers very fast detection rates.

As input, we use the RGB image resized to the standard resolution of 227×227 pixels. Next, the image is pushed through the network and a convolutional feature map is extracted. We use the model of Zeiler and Fergus (2014) (Zeiler and Fergus, 2014) to extract these feature maps. To generate region proposals, this feature map is processed in a sliding-window manner with two fully-connected layers predicting position of the region proposal and a binary class label indicating "objectness". For each region proposal, the corresponding (shared) feature maps are fed into 2 fully-connected layers with 2048 units that produce soft-max probabilities over K object classes (and background) and positions of bounding boxes of the detected objects. We trained this object detector on five classes corresponding to objects and monsters that are projected onto the common map.

8.4.2 Camera pose estimation

Despite using ground-truth depth maps provided by the z-buffer, ICP-like approaches (Besl and McKay, 1992) do not work well in game environments since such environments lack many geometrical features (they are typically represented as textured planar surfaces to allow fast rendering). Hence, we use the sparse feature-based ORB-SLAM2 for 6-DoF camera-pose estimation (Mur-Artal *et al.*, 2015) running on RGB images down-sampled to 320×240 pixels and a z-buffer.

First, we build an eight-level image pyramid with a scale factor $s_f = 1.2$. Then, we extract a set of sparse local features representing corner-like structures. For this, we use oriented multi-scale FAST detector (Rosten *et al.*, 2010) with an adaptively-chosen threshold to detect a sufficient number of features. The feature extraction step is biased by bucketing to ensure features are uniformly distributed across space and scale (at least 5 corners per cell). A constant-velocity motion model predicting the camera pose is used to constrain matching onto local search windows. The extracted features are associated with local

binary-patterns (256 bits ORB (Rublee *et al.*, 2011)) and matched using a mutual-consistency check. A robust estimate is performed in a classic way by RANSAC (Fischler and Bolles, 1981) with least-squares refinement on the inliers.

Robustness is further increased by keyframes that reduce drift when the camera viewpoint does not change significantly. If tracking is lost, the current frame is converted into a bag-of-words and queried against the database of keyframe candidates for global re-localization. The camera is re-localized using the PnP algorithm (Lepetit *et al.*, 2009) with RANSAC. Global consistency is achieved by loop-closing pose-graph optimization that distributes the error along the graph in a background thread (Kuemmerle *et al.*, 2011).

8.4.3 Mapping

Once we have the camera poses and a object-masked depth map, we can project the current frame on a common 3D map. At each frame k, we back-project all image pixels i into the current camera reference frame to obtain a vertex map $\mathbf{V_i^k}$

$$\mathbf{V}_i^k = d_i^k \mathbf{K^{-1}} \dot{\mathbf{u}}_\mathbf{i}. \tag{8.1}$$

Here, $\mathbf{K}^{-1}$ denotes the inverse of the camera calibration matrix (using parameters from the VizDoom configuration file), $\dot{\mathbf{u}}_i = [u_i, v_i, 1]^\top$ denote image pixels in homogeneous coordinates, and d_i^k is depth. We also want to maintain previously-visited areas in memory so we project the (homogenized) vertex map $\dot{\mathbf{V}}_i^k = [X_i, Y_i, Z_i, 1]^\top$ from camera to global reference frame as $\mathbf{V}_i^g = \mathtt{T}_{g,k}\dot{\mathbf{V}}_i^k$, where $\mathtt{T}_{g,k} = \{\mathtt{R}, \mathbf{t} | \mathtt{R} \in \mathbb{SO}_3, \mathbf{t} \in \mathbb{R}^3\}$ is a rigid body transformation mapping the camera coordinate frame at time k into the global frame g. Since the fixed volumetric 3D representation severely limits the reconstruction size that can be handled, we use the hash-based method of (Nießner *et al.*, 2013).

The resulting 2D map is generated by placing a virtual camera at the top-down view, ignoring all points above and below some height thresholds to remove areas that would otherwise occlude the map, such as ceilings and floors.

8.5 Implementation details

In this section, we describe details of the various components of our framework. Our framework is built on top of the ViZDoom (Kempka *et al.*, 2016) platform.

Recognition and Reconstruction. As described in Sec. 8.4.1, we use the Faster-RCNN detector and feed it with the RGB image given by the platform. We use a network pre-trained

on Imagenet (Russakovsky *et al.*, 2015) that we fine-tuned on a dataset consisting of 2000 training and 1000 validation examples extracted from the ViZDoom engine, performing 5-fold cross-validation. These images were manually annotated with ground-truth bounding boxes corresponding to 7 classes: monsters, health packs, high-grade weapons, high-grade ammunition, other weapons/ammunition, monsters' ammunition, and agent's ammunition. After fine-tuning, the model achieved an average precision of 93.21%. The reconstruction system presented in Sec. 8.4.2 uses the RGB-D images provided by the VizDoom platform.

Policy Learning. We use the DQN framework from (Mnih *et al.*, 2015) to perform policy learning with our augmented features. The only modification to the original algorithm is the CNN architecture that needs to be able to cope with the extended state. The first person view (FPV) images are resized to 84×84 pixel, converted to grayscale and normalized. The semantic 2D map is represented as a single channel image of the same resolution. The different object categories are encoded by different grayscale values. For the experiments that use both the FPV and the 2D map, we concatenate them along the channel dimension. The Q network is composed of 3 convolutional layers having respectively, 32, 64 and 64 output channels with filters of sizes 8×8, 4×4 and 3×3 and 4, 2 and 1 strides. The fully-connected layer has 512 units and is followed by an output SoftMax layer. All hidden layers are followed by rectified linear units (ReLU). Adding the 2D map associated to each FPV image changes input channels from 4 to 8 for the first convolutional layer, and thus increase the number of parameters from 77824 to 86016, a 10% increase. For training, we use the hyper-parameters from (Mnih *et al.*, 2013) and RMSProp for all experiments.

Action Space. The action space for this environment is an order of magnitude larger than the Atari environment. Indeed, "Doom" accepts any combination of 43 unique keystrokes as input. Following the observation that a human player uses only a small subset of these combinations to play the game, we recorded actions performed by humans and selected a representative subset. These actions can be divided into three groups: i) actions corresponding to a single keystroke allowing the agent to move and shoot, ii) combinations of two keystrokes corresponding to moving and shooting at the same time and iii) actions associated with switching weapons. We arbitrarily chose the top 13 actions performed by humans, categorising them into the 3 groups mentioned above. We did so primarily to constrain the action space to a reasonably tractable size, while still maintaining richness of actions that could be performed in the environment.

Reward Function. Our reward function is designed to capture the primary goal of the agent: to eliminate opponents. We represent this as Δ_k, an indicator variable for an opponent being eliminated since the last step. To encourage the agent to live longer, we also consider Δ_h, the health variation between the current step and the previous step. We explicitly structure the health reward to be zero-sum in order to remove any biases towards preserving health to the detriment of the primary goal. The reward $\mathcal{R}$ incorporating both these terms is written as: $\mathcal{R} = \Delta_h/100 + \Delta_k$ where $\Delta_h \in [-100; 100]$ and $\Delta_k \in \{0, 1\}$

Time Complexity. The complete framework has to be fast enough to allow playing at the game's native speed. To do so, we run the object detector in parallel with the camera-pose estimation. On average, the detector requires 60ms to process an image while camera-pose estimation and latency take 12ms and 10ms respectively. Semantic map construction takes 25ms, and DQN training requires 18ms to process a frame and perform one learning step. The complete pipeline is able to process, on average, 10 images per second. Given that inside the ViZDoom platform each step represents 4 frames of the game (as does the Atari emulator), our system plays at approximately 40 frames per second, which exceeds typical demands of gameplay. All experiments were run on a Intel Core i7-5930K machine with 32GB RAM and one NVidia Titan X GPU.

8.6 Experiments

In this section, we demonstrate the advantage of adding the semantic map presented in Sec. 8.3 to the standard first-person view while working inside the "Doom" environment. The quantitative results for all the experiments carried out are summarised in Tab. 8.1.

Platform. We use the ViZDoom (Kempka *et al.*, 2016) platform for all our experiments. It is built on top of the first person combat game "Doom", and allows easy synchronous control of the original game, where execution is user-controlled, getting the first-person-view from the engine at the current step, and stepping forward by sending it keystrokes. The environment where the player performs is specified as scenario.

In this paper, we focus on the *deathmatch* scenario, in which the map is a simple arena as can be seen in Fig. 8.1 and the goal is to eliminate as many opponents as possible before being eliminated. A proficient agent for this scenario would be the one that is efficient at eliminating enemies whilst being able to both collect more effective weapons and keep its own health as high as possible. This scenario was the basis of the CIG

Table 8.1: Best mean test rewards for the different frameworks run. Note that our pipeline performs strongly in comparison to both the baselines, and to the ablated versions considered. Also note that although the OSM is the best of the artificial systems considered, our pipeline, with the RSM is a lot closer to it than the others.

Settings	Rewards
Random Play	0.00
Noisy Oracle Semantic Maps (with player/objects locations)	2.94
Oracle Semantic Maps (map only / no first person view)	3.16
baseline	3.45
Noisy Oracle Semantic Maps (with objects)	3.53
Noisy Oracle Semantic Maps (with objects and walls)	3.92
Prior Dueling DQN	5.69
Reconstructed Semantic Maps (with localisation)	5.87
Oracle Semantic Maps (with localisation)	6.62
Reconstructed Semantic Maps	6.91
Oracle Semantic Maps	9.50
Human Player	45.00

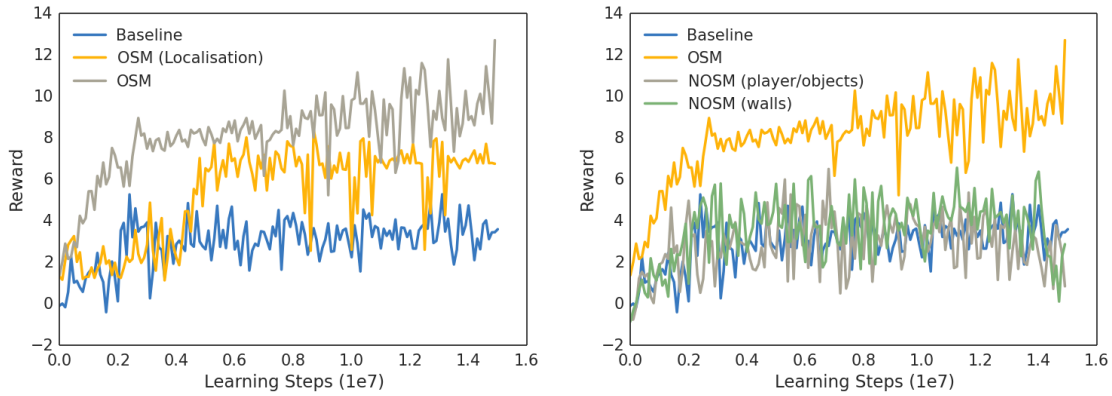
2016 competition (Jakowski *et al.*, 2016) where different autonomous agent competed in a deathmatch tournament.

Evaluation Metrics. We use two different scores to evaluate and compare different architectures. The main metric is the reward function as it allows observing the agent's behaviour with respect to the primary objective. The second reported metric is the number of steps the agent has lived. This is important as living increases the agent's chance to kill opponents and increase its reward in the longer term. All reported metrics are mean values over 100 test games.

8.6.1 Oracle Semantic Maps (OSM)

The first set of experiments allows us to evaluate the efficacy of our semantic representation. We first isolate potential errors introduced by the recognition and reconstruction pipeline by extracting ground-truth information about classes and positions of all objects that are used in the semantic map representation. In other words, this experiment presents the results we would get if we had perfect detection and reconstruction, and is used as an "oracle".

As the baseline, we use the standard DQN trained solely on the first person view images (referred to as baseline in the following). This baseline is compared to i) model trained with both, the first person view and the 2D map encoding ground-truth walls and player position (localisation OSM) ii) model trained with both, the first person view augmented by the complete 2D maps containing ground-truth walls and positions of player and objects.



(a) **(b)**

Figure 8.3: (top) Average reward. (bottom) Oracle Semantic Maps vs. Noisy Oracle Semantic Maps.

As can be seen in Fig. 8.3, the baseline is not able to learn as good policy as model with our semantic maps. Moreover, we see that the baseline model quickly reaches a plateau and does not improve afterwards. Adding a 2D map of the environment (*i.e.* without objects) allows the agent to learn a significantly better policy as the reward is almost doubled compared to the baseline. Adding the objects seen by the agent onto this map gives another significant improvement leading to reward of 10 compared to the $3-4$ achieved by the baseline. Moreover, we can see that the network provided with the complete 2D map (including objects) is able to learn faster than the models provided with fewer information. This result proves that providing higher level, complex representation of the surrounding of the agent allows it to learn faster and converge to a better policy.

8.6.2 Noisy Oracle Semantic Maps (NOSM)

Unfortunately, the detection and reconstruction pipelines are often imperfect in real world scenarios. Next, we study the impact of providing a very poor spatial representation to the agent. To do that, we add a significant amount of noise to the ground-truth data extracted from the game to see how the DQN framework reacts.

First, we consider the case where we add the same Gaussian noise to the agent's and all objects' positions, referenced as NOSM (player/objects), meaning that these elements are not properly positioned with respect to the static objects. Fig. 8.4(left) shows the results of adding that noise. The OSM map is shown on top and its noisy version is shown below. One thing to note here is that these maps have gray scale pixel values to define different abstractions and objects. This gray scaled format was used for training as discussed in the previous sections. Next, we add Gaussian noise to the positions of walls, referenced as

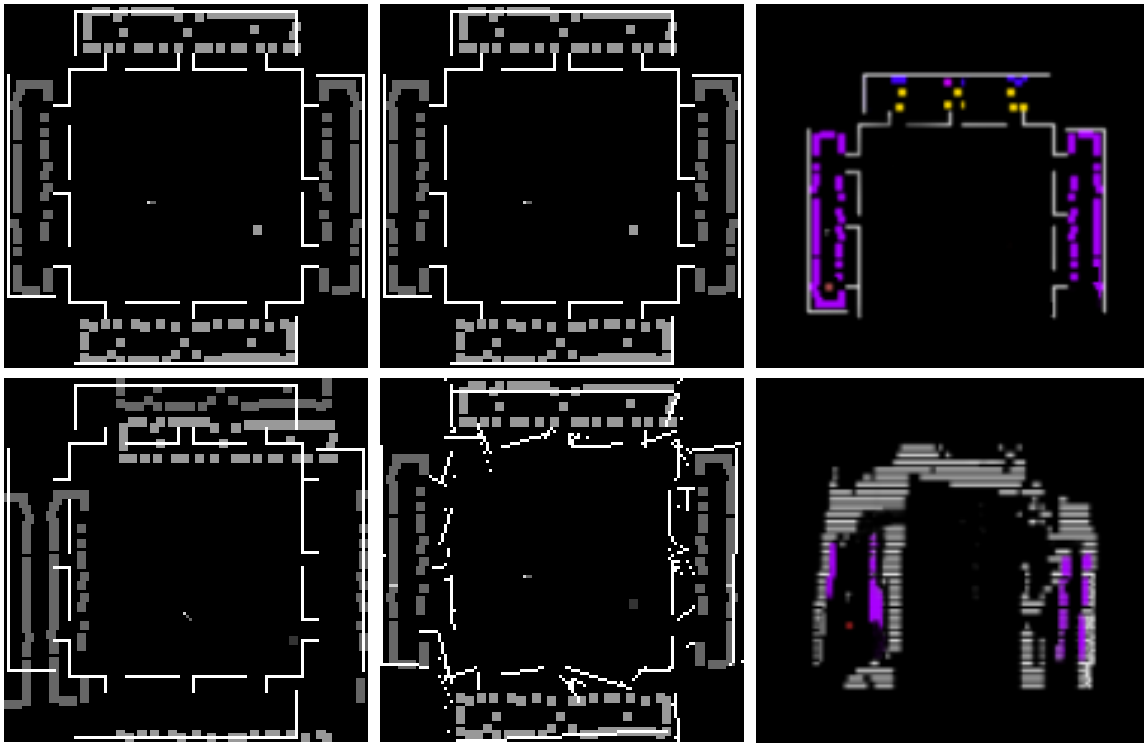


Figure 8.4: The top maps for each column are all taken from the oracle. The maps on the bottom are (left) Oracle map with noise on player and objects' positions. (middle) Oracle map with noise on the walls. (right) Semantic map reconstructed, independent of the oracle, by our pipeline.

NOSM (walls), meaning that some element that appear accessible in the 2D map cannot be reached in the real environment. Fig. 8.4(middle) shows the results of adding that noise.

As can be seen in Fig. 8.3(bottom), this very high amount of noise in the 2D maps prevent the DQN framework to learn a good policy. However, it is important to note that in the worst case, the noisy version matches the performances of the baseline as the network learns to ignore it.

8.6.3 Reconstructed Semantic Maps (RSM)

In §8.6.1, we have shown the efficacy of Q-learning with ground-truth version of our semantic maps. As a proof of concept, we now evaluate performance with the real maps generated on-the-fly by the approach described in §8.4 (RSM). This experiment allows us to evaluate the quality of the policy that can be learned when using the standard detection and mapping techniques without any extra engineering. In other words, we measure the drop in performance caused by imperfect object detection and SLAM in a real world scenario with respect to the oracle. The difference between the OSM and the RSM is seen in Fig. 8.4(right). Here, the semantic categories are coloured instead of greyscale levels for emphasis.

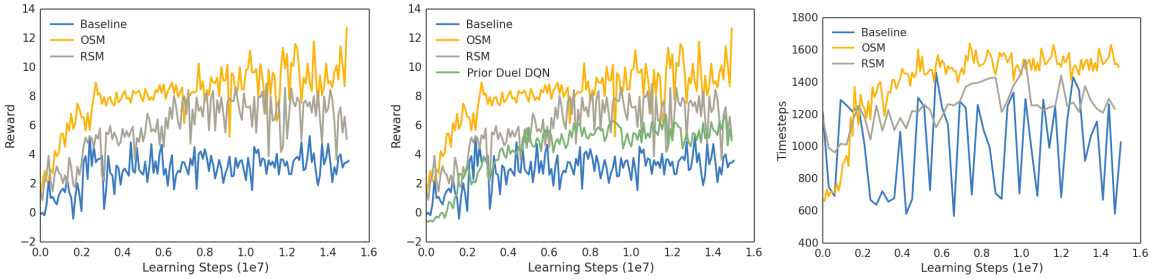


Figure 8.5: (left) OSM vs. RSM (middle) Our method vs. dual DQN with prioritized ER. (right) OSM vs. DQN on mean run-length

As seen in Fig. 8.5(left), the reconstructed map leads to significantly better results than the baseline. Even though it doesn't match the oracle, we clearly see that the RSM is much closer to the OSM than the baseline. The remaining gap can be further reduced with progress in the field.

8.6.4 Prioritized Duel DQN

Combination of the prioritized experience replay (Schaul *et al.*, 2016) and dueling network architecture (Wang *et al.*, 2016) has demonstrated superior results on 57 Atari games (2D environment) compared to the vanilla DQN approach that is the baseline considered above. In this experiment, we compare this successful model (referred as dDQN) with the basic DQN model augmented with our semantic maps.

Fig. 8.5(middle) shows that while the combination of PRL with dual DQN achieves better results than the DQN baseline, the model with our semantic maps, despite trained with the basic DQN, outperformed the PRL with dual DQN trained on first person views. It is also interesting to note that these two approaches are orthogonal and could be combined. We leave this study for the future work.

8.6.5 Mean Run Length

As can be seen in Fig. 8.5(right), the agent trained with semantic maps is able to typically live longer than the one trained only on the first-person view. This is a consequence of the fact that the OSM agent inherently attempts to build a representation of the environment it is in, which helps it adapt better from arbitrary initialisation points. The baseline however, does not have access to such capabilities, and hence performs incoherently in these situations. In keeping with the general characteristics of the results seen thus far, the RMS agent typically underperforms in relation to the ORM agent, but still significantly outperforms the baseline.

8.7 Discussion and Conclusion

We proposed to augment the standard DQN model with semantic maps; a representation that provides aggregated information about the 3D environment around the agent. We have demonstrated the efficacy of our approach with both oracle maps, and automatically reconstructed maps using object detection and SLAM, demonstrating the efficacy of our approach with standard computer-vision recognition and reconstruction pipeline and a standard off-the-shelf policy learner (DQN).

Our central thesis is exploring the benefits of semantic representations augmenting the directly-from-pixels learning approach typically employed. While we do not claim major contributions to policy-learning algorithms themselves, the effort nonetheless provides insight on the efficacy of such representations against those learned in a purely bottom-up manner. It also potentially serves as a benchmark for effectiveness of representations learned in a purely bottom-up manner. Moreover, our approach has the potential to extend and scale beyond the Doom environment by virtue of its applicability to any environment with a reasonable number of potential other entities and the extractability of 3D information.

Seeing it from perspective of 2017. Perhaps the main drawback is that we are unable to represent layered environments such as buildings ("stacked floors/levels"). A naive solution might be to provide the network also floors one level above and below to increase the information about the surrounding environment available to the network. Instead of using handcrafted pipeline for reconstruction, it may be better use a differentiable mapper (Gupta *et al.*, 2017).

9 Conclusions

9.1 Summary

In this thesis I have developed novel algorithms for (near) real-time dense 3D reconstruction and semantic segmentation of large-scale outdoor scenes from passive cameras. Motivated by "smart glasses" for partially sighted users, I have shown how such system can be integrated into an interactive augmented reality system which puts the user in the loop and allows her to physically interact with the world to learn personalized semantically segmented dense 3D models.

In the next part I have shown how sparse but very accurate 3D measurements can be incorporated directly into the dense depth estimation process and proposed a probabilistic model for incremental dense scene reconstruction. To relax the assumption of a calibrated stereo camera I have also addressed dense 3D reconstruction in its monocular form and shown how a local model for dense monocular 3D reconstruction can be improved by joint optimization over depth and pose.

The proposed video segmentation model has demonstrated how we can encode a single object instance as a closed curve, maintain correspondences across time and provide very accurate segmentation close to object boundaries. Although this model is motivated by rotoscoping, it can be used in a fully automatic setup for dense non-rigid 3D reconstruction of texture-less objects known as shape-from-silhouette.

Finally, instead of evaluating the performance in an isolated setup (IoU scores) which does not measure the impact on decision-making, I have shown how semantic 3D reconstruction can be incorporated into standard Deep Q-learning to improve decision-making of agents navigating complex 3D environments.

9.2 Future Directions and Open Questions

Throughout this thesis, I have been motivated by decision making of agents navigating complex 3D environments. At the end of each chapter, I have discussed how the major limitations of the proposed approach could be tackled with newer tools, typically a computational graph allowing end-to-end learning. In this section, I revisit the problems I have addressed in this thesis at a higher-level and discuss future directions, in particular in understanding of dynamic scenes.

First, I discuss recent progress in tracking of general objects and suggest that we should consider them as agents (§9.2.1). Hence we should not only recognize (follow) them but also understand their goals and intentions. A related line of research represents models of intuitive physics (§9.2.2) which should allow us to constrain the state space to physically plausible solutions. Next, I argue that we should not study moving objects (tracking) and stationary world (SLAM / SfM) in isolation since these tasks are mutually beneficial (§9.2.3). Section §9.2.4 discusses future directions for semantic scene understanding on more technical level and I conclude with discussion about topological SLAM based on a differential neural computer (DNC) model which could potentially lead to a new generation of SLAM models but represents a completely open ended question at the moment (§9.2.5).

9.2.1 Object Tracking as Question Answering

Progress in single object tracking has been significant during the past few years. It has primarily been fueled by i) the series of VOT Challenges (Kristan *et al.*, 2013-2017) and ii) holistic tracking-by-detection paradigm with models trained as ridge regressors (Henriques *et al.*, 2015; Bertinetto *et al.*, 2016b) (correlation filters). The VOT Challenges have ended the "wild west" of performance evaluation by enforcing standardized datasets and metrics and correlation filters enabled efficient online learning of holistic models for tracking-by-detection paradigm by by-passing sparse sampling heuristics of training examples.

The popularity of the VOT challenges, historical reasons (computational complexity, *etc.*) and recent success of tracking-by-detection paradigm have led many people to believe that the only difference between tracking and detection is in the VOT performance metrics. The VOT has for a long time forbidden use of the pre-trained class-specific models, and one might argue that the difference between the two would vanish by relaxing this constraint. In other words, the tracking problem has been degenerating to bounding box or object/motion segmentation mask prediction; with an ability to adapt the base model over time since we

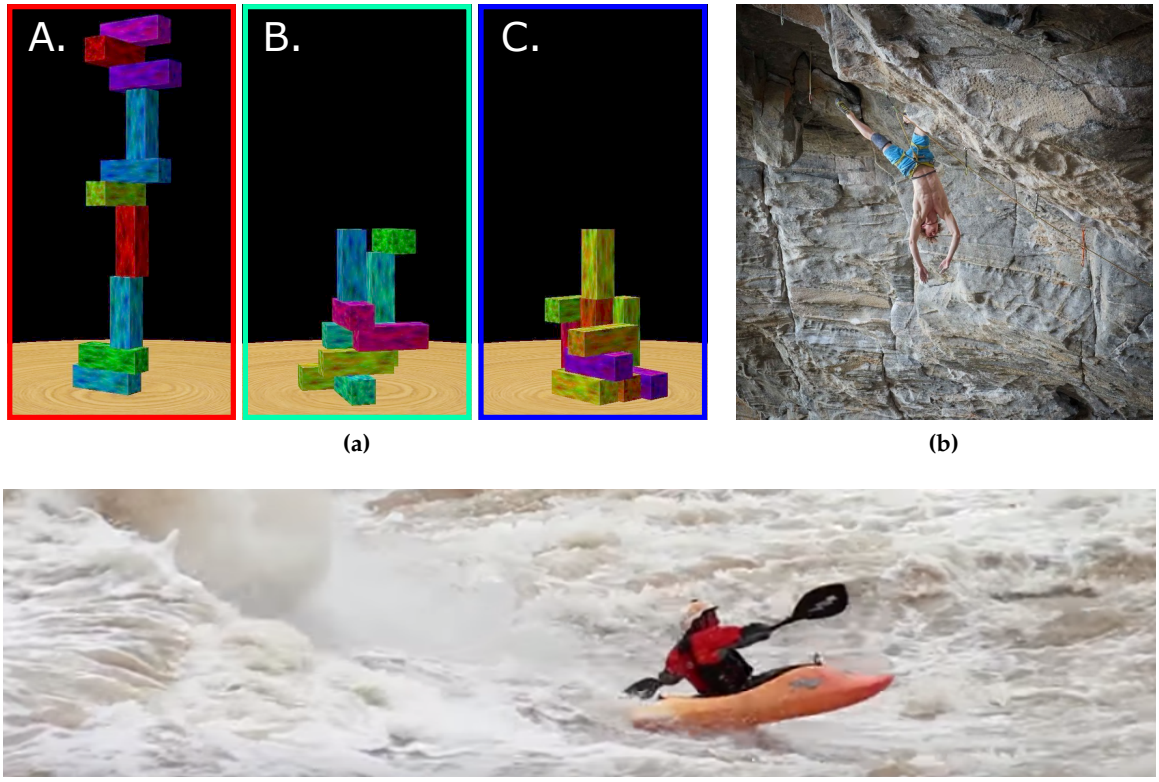
deal with causal image sequences. Ultimately, if our field believes we are able to solve object detection or segmentation, I would argue that there is no point of working on tracking defined in this way as it would be only a temporary problem that would completely vanish as a by-product of solved object detection / segmentation.

However, models that process videos per-frame independently would never be able to address the fundamental tracking task, which is (similarly to SfM/SLAM) *data association*. While we largely ignore its importance in single object scenarios (Kristan *et al.*, 2013-2017), and use it primarily only to improve per-frame detection scores in the multi-object scenario (Milan *et al.*, 2015-2017), it is a key concept for analyzing and understanding complex dynamic behavior. From this viewpoint, focusing on appearance representation (predicting bounding box, object or motion segmentation masks, *etc.*) definitively improves our ability to *follow* the object of interest but does not help us to *understand* it.

I would argue that our current view on tracking of general objects is too limited since we mostly model only the object appearance and (almost) completely ignore its behavior. Hence, we do not understand kinematic/dynamic constraints of objects, their intrinsic motivation (intentions and goals), and even whether we track a freely moving or stationary object whose motion is induced only by the camera itself. Consequently, we often see many completely illogical failures such as a car driving on a highway which within a single frame suddenly disappears into a lake, just because some appearance features failed in some way.

I believe that we should model all objects we could potentially track as agents. We should not only use appearance but also kinematic and dynamic models (stationary objects are fully determined by camera motion). We should categorize them into semantic classes (*e.g.* horses and cars behave very differently) and understand the surrounding environment. In fact, we should not only analyze actions that have already happened in the past (= previous frames) but we should also close the loop by understanding agents' intentions and forecasting their future actions. In other words, we should be able to understand and explain purpose, causes and effects.

This is not an easy task. In fact, it is not even clear how such task should be formulated at this stage, however, I believe that one potential way of escaping the current local minima in which we only keep improving modelling of object appearance would be to link tracking to *natural language* and *visual question answering*. For instance, we should be able to answer how many objects are within a scene, whether they are moving or not and if so whether an object is moving freely or following a crowd. We should be able to answer why an object can or cannot move in some particular area and forecast its potential goals and intentions.



(a) (b)

(c)

Figure 9.1: Modelling intuitive physics. (a) Three towers of varying height and stability (Hamrick *et al.*, 2011). (b) Despite this climber is in an unusual position, he is not falling down, just resting in a "no-hand" position during the world's first successful attempt on 9c route. This image would probably always violate our prior knowledge captured by data and we should resolve to explain *stability* and *support* instead. (c) Similarly, despite all objects are moving down the river, the kayaker is still at the same place. Despite the strong current from right to left, the kayaker is still at the same place. This would probably be always very difficult to explain without modeling hydrodynamics.

9.2.2 Modelling (Intuitive) Physics

We are interested in analyzing video sequences of the *real 3D world* (*i.e.* not simulated) in which all objects have to fulfil physics laws but our algorithms typically do not model physics at all. This is somewhat understandable since we do not want to over-handcraft our models. Moreover, modelling physics often require knowledge of object parameters that are difficult to estimate from images (*e.g.* object mass). However, our algorithms often fail in a completely illogical way; it is not uncommon that we track a car on a highway, and despite we know how it should be moving (*e.g.* constraints of Ackermann steering geometry), the algorithm predicts poses that would not be possible without omni-directional wheels.

I would argue, that our models should use at least *intuitive physics* (Wu *et al.*, 2015;

Li *et al.*, 2016a; Hamrick *et al.*, 2011) to constrain the state space to physically plausible solutions. Intuitive physics is also important for most manipulation tasks, in which we want to forecast or understand *what would happen if*. Last but not least, all our datasets follow a heavy-tailed distribution. Hence, the prior and context our model learns are typically unable to explain corner cases such as a kayak surfing the wave and not moving downstream or a climber resting in "no-hand" and not falling down (*cf.* Fig. 9.1) since our models are unable to understand the underlying physics (and potentially support planes). Augmenting our models with intuitive physics might be a way to overcome these issues.

9.2.3 SLAM with Dynamically Moving Objects

Structure-from-Motion and SLAM systems typically assume that the observed world is always stationary. Such an assumption may be acceptable for applications such as dense 3D reconstruction of historical buildings and museums (Xiao and Furukawa, 2014), however represents a major and limiting constraint for agents navigating complex dynamic 3D environments. One way of relaxing this constraint is to use motion segmentation to identify regions that should not be reconstructed within a static 3D map and simply track them (Kundu *et al.*, 2011). In other words, SLAM considers as positive only the stationary objects. The dynamic objects, which would normally degrade the performance, are ignored and tracked instead. While this solution is not perfect since it addresses both problems in isolation, it at least provides the agent some information about dynamically moving objects. More advanced solutions attempt to unify tracking and reconstruction tasks by including the dynamic objects directly within the SLAM framework. These tasks are mutually beneficial Wang *et al.* (2007); Bibby (2010); for instance, if we are able to understand camera motion, we are able to simplify tracking of static objects, because their "motion" is induced only by the camera itself. This is typically formulated as SLAM with generalized objects, which extends the state vector of each landmark by velocities. SLAM with generalized objects is similar to standard SLAM algorithms, but with additional structure enabling motion modelling of generalized objects (Fig. 9.2).

While mathematics of SLAM with generalized objects is relatively well understood and sparse variants have been successfully implemented Wang *et al.* (2007); Bibby (2010), its *dense* counterpart represents a challenging task. This is not only due to high computational complexity and inherent model selection problem (moving vs stationary). Perhaps, even more important and open question for dense SLAM with dynamic objects is map representation, in particular for non-rigid objects.

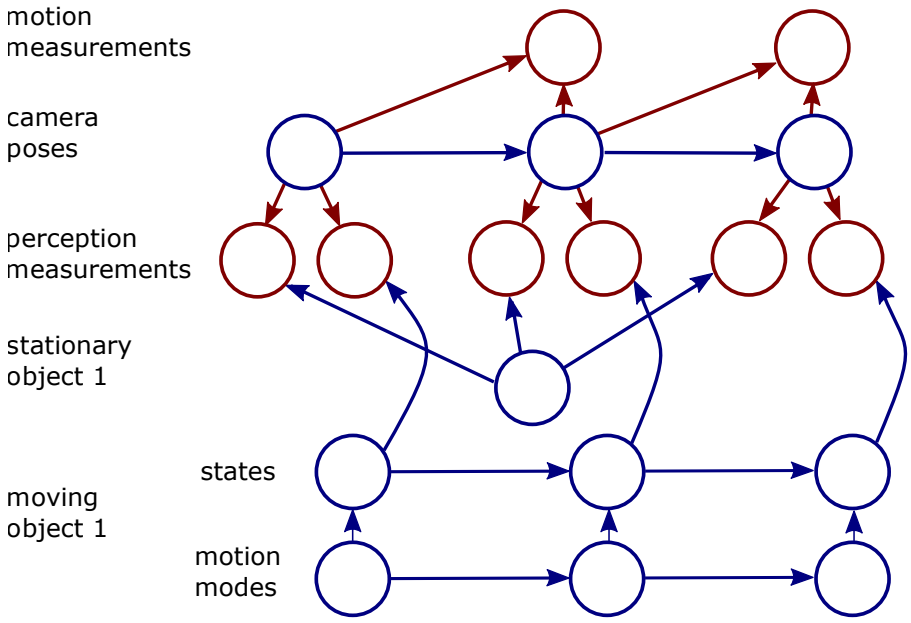


Figure 9.2: A Dynamic Bayesian Network of SLAM with generalized objects of duration three with one moving object and one stationary object (Wang *et al.*, 2007).

9.2.4 Future Directions for Semantic Scene Understanding

Keyframe-to-keyframe depth estimation. One of the most important contributions to sparse real-time SLAM was off-loading sparse bundle adjustment to a background thread. This allowed us to move from incrementally drifting visual odometry to constructing globally optimal sparse 3D reconstructions. In real-time dense 3D reconstruction we still usually perform only frame-to-keyframe matching and hence are limited to very narrow baselines (narrow baseline of a fixed camera rig or narrow dynamic baseline between frame and keyframe). Using the most recent camera frames is important to estimate camera pose in real-time, however, if we limit ourselves to such narrow baseline, the resulting 3D map would never be of the same quality as if we used *all* observed data. Hence, we should perform dense depth matching also *between* keyframes in a background thread to improve the quality of reconstructed 3D maps by using *dynamic* (and potentially very large) baselines and re-using most of the standard tools from off-line Structure-from-Motion.

Object instances. Most semantic segmentation approaches predict per-pixel maps with labels corresponding to semantic classes. While this is a great representation for *stuff* classes (sky, tree, road, *etc.*), it is not sufficient for objects (cars, pedestrians, signs, *etc.*) as it does not allow counting of object instances and does not provide any information about their locations. This makes many higher level reasoning tasks very complicated if not impossible.

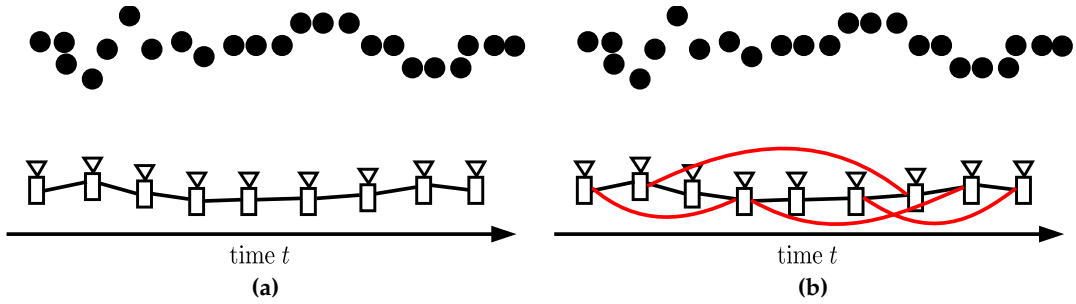


Figure 9.3: (a) Narrow baseline matching, (b) variable baseline matching.

Despite there have been some attempts and progress over the past few years to predict object *instances* explicitly, the best methods achieve only 32.0 average precision (AP) on CityScapes dataset (Cordts *et al.*, 2016) and hence this problem remains open.

Sparse long-range CRF / Inference on the level of objects. Throughout this thesis, I have often used the densely connected CRF. While this model is able to propagate information better than the standard 4/8-neighborhood CRFs and we often use the terms "fully connected" and "global optimization methods" (§2.1), it connects only variables in a relatively local neighborhood (Fig. 2.16, Fig. 9.4).

Imagine a large-scale model, for instance, between Oxford and London. Fast optimization is clearly beyond capabilities of any modern inference method. Moreover, such model contains many repetitive patterns (*e.g.* lamps, signs, *etc.*), however, there is no guarantee that particular instances of these objects are close enough to each other to be captured by pairwise potentials defined in local neighborhoods. Consequently, we do not use all information available to such a model.

In context of modern unary potentials based on convolutional neural networks which already capture large context, enforcing smoothness in local neighborhoods has been becoming less important (Zheng *et al.*, 2015). A much better model would use *sparse long-range* potentials on the level of objects. Sparse long-range potentials would allow us to reduce the number of variables in the model and hence at the same time define much richer and powerful interactions (Zhang *et al.*, 2016). This can be seen, as an extreme case of co-segmentation, in which objects reinforce each other. Another link can be seen to information propagation in *small-world networks* (Watts and Strogatz, 1998; Kleinberg, 2000).

Style and content decomposition / Deep intrinsic images. I have mostly used dense 3D reconstruction as an intermediate abstraction that glues together per-frame predictions

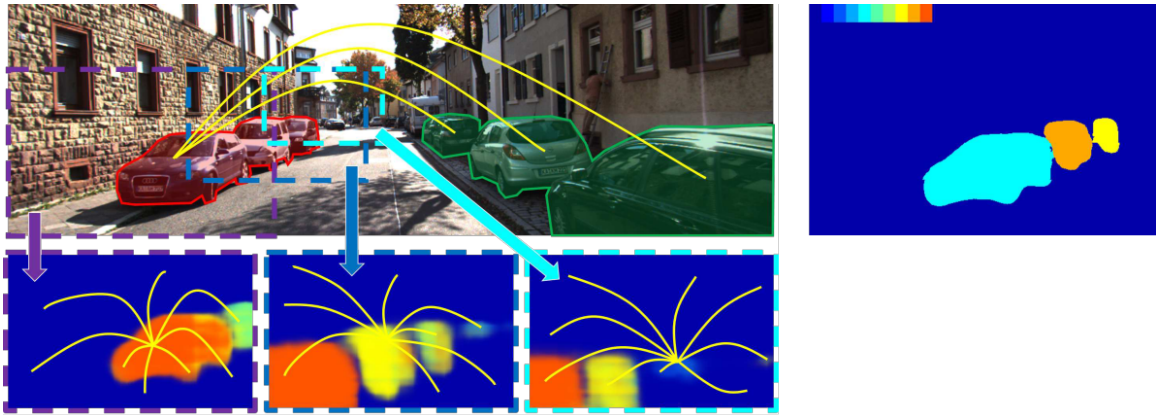


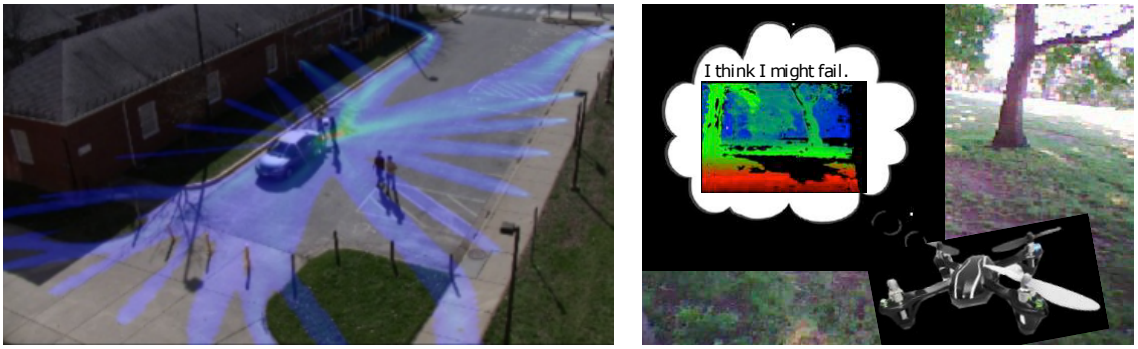
Figure 9.4: Sparse long-range potentials allow to connect different instances of the same semantic class even if they are not close to each other (left). Instance segmentation is better suited to represent objects (right) (Zhang *et al.*, 2016).

and at the same time provides relatively strong constraints. Although this representation is based on appealing mathematical concepts and offers straightforward interpretation, reconstructing dense 3D maps in the wild (YouTube videos, *etc.*) remains to be challenging.

In some sense, reconstructing dense 3D map means that we still approach computer vision as *measurement* and not as *perception*. Considering the fact that uncertainty of 3D reconstruction grows quadratically with depth (Gallup *et al.*, 2008), it might be sensible to abandon the idea of having a dense 3D map and rather focus on relative depth orderings. A natural proxy could be nowadays popular *style* and *content* decomposition. Such decomposition can be used to separate illumination, texture and style from content, which can be seen as a weak proxy for structure. One could push this idea further with learning deep disentangled intrinsic scene decomposition, *i.e.* learning a deep generative model with interpretable latent code.

Ill-posed GT labels (multi-to-multi labels). State-of-the-art models are very successful in learning many-to-one mappings. For instance, learning a (nonlinear) mapping transforming *all* observations of some object to a *single* class label have become a canonical supervised machine learning task and de-facto commodity. However, when we move to higher-level scene understanding, we find out that defining a *single* ground-truth label is often impossible. Consider, *e.g.* a personal robot anticipating human actions (activity forecasting (Kitani *et al.*, 2012; Lee *et al.*, 2017)); there is no single ground-truth label since this task is ill-posed by its nature.

Instead of inferring a single label, we should predict a set of top K answers. Despite this sounding like a very straightforward extension of standard models, the reality is



(a) (b)

Figure 9.5: (a) The problem of trajectory forecasting is inherently ill-posed and we need to predict K best diverse solutions (Kitani *et al.*, 2012). (b) Autonomous systems operating in complex environments have to be able to predict their own failures (Daftry *et al.*, 2016).

completely different. Even inferring the second best structured prediction is very difficult problem (Yanover and Weiss, 2003; Wainwright and Jordan, 2008). However, the real trouble is the fact that solution corresponding to the second best energy is typically completely useless - it often differs from the MAP prediction only by *a single pixel*. Similarly, we cannot naively evaluate *all* solutions and simply cluster them - the computational complexity is prohibitively large. Instead, we need to predict a small set of predictions, that are at the same time *relevant* yet *diverse* enough. An appealing framework for diverse predictions, which is however very difficult to generalize to structured prediction, is offer by Determinental Point Processes (Kulesza and Taskar, 2012; Gillenwater, 2014).

Introspective perception. Without any doubt, computer vision models have been improving performance on standard benchmarks so rapidly that the benchmarks are becoming quickly saturated and obsolete. This might lead us to a false impression that our models have after 50 years of research finally become mature and robust enough so that we can deploy them in any real-world application without much effort.

However, we should keep in mind that our models i) are typically evaluated in a *closed world* and ii) are unable to predict their own failures and self-diagnostics. This typically means that models are not aware of their own capabilities. For instance, if we train a model on dataset consisting of 20 PASCAL classes and deploy it in a real world with *open label set*, it typically would not be able to predict "I do NOT know" for previously unseen semantic classes but would simply assign the highest scoring label (using a dummy label with constant cost for outliers typically does not solve this problem). The second issue is, that our models are not able to "realize" that they *might have failed*. Lack of these capabilities

is of a relatively lower importance in applications such as photo-editing, however could lead to catastrophic failures in decision making process of autonomous systems operating for a long-term in complex dynamic environments (Daftry *et al.*, 2016; Grimmett *et al.*, 2015).

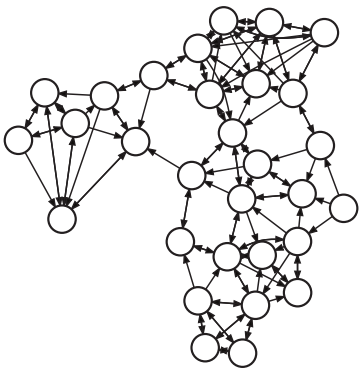
9.2.5 Models with Memory

The current generation of machine learning models excel at pattern recognition tasks such as classification, detection or semantic segmentation and quick reactive decision-making such as Atari Games. However, reactive decisions are not enough in many situations and we would like to reason using knowledge. While this deficiency can be somewhat mitigated by recurrent neural networks (RNNs, LSTMs, *etc.*) (Hochreiter and Schmidhuber, 1997), their reasoning abilities over complex and long sequences remain limited. To this end, our models need to be able to learn how to "store" even complex data structures in explicit *memory* which is decoupled from the decision-making process of the base model (*i.e.* would not modify it) and can be allocated on demand. One example might be multi-object tracking, which would be able to allocate explicit memory for each tracked object to allow online adaptation without modifying the whole model. Another application is 3D reconstruction. In fact, key-frame based SLAM is perhaps the only computer vision task which nowadays uses explicit memory (although it is not learnt), hence I will demonstrate this concept on a special case of *topological* SLAM (Werner *et al.*, 2009).

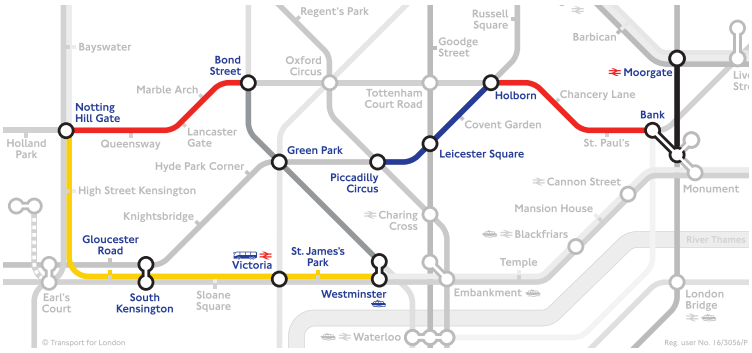
All visual odometry methods suffer from incremental pose updates since they accumulate errors associated every new observation which results in *drift*. This issue is inherent to all visual odometry methods and as such would never vanish. We typically address it by bundle adjustment, potentially with explicit *loop closure* detection. Most approaches, however, represent the map in a relatively naive way - usually with sparse metric point-cloud (or similar) which stores all parts of the environment within a single common 3D reference frame with the same density.

I would argue that this is not the best representation for most of the navigation tasks. For instance, if I were asked to describe the directions from my home to the office, I definitively would not use much of metric information. Such description, would rather be a sequence of a few *waypoints*, typically corresponding to *memorable places* (*e.g.* Radcliffe Camera, Broad Street, *etc.*) or locations at which an agent has to make some decision / action (*e.g.* turn left and continue 5 minutes). In that case, I would argue that *topological representation*, which uses graph nodes to model waypoints and edges to link them, would be better suited for most navigation tasks and at the same time much more scalable than classic

a Random graph



b London Underground



	Traversal	Shortest-path
Underground input: (OxfordCircus, TottenhamCtRd, Central) (TottenhamCtRd, OxfordCircus, Central) (BakerSt, Marylebone, Circle) (BakerSt, Marylebone, Bakerloo) (BakerSt, OxfordCircus, Bakerloo) (LeicesterSq, CharingCross, Northern) (TottenhamCtRd, LeicesterSq, Northern) (OxfordCircus, PiccadillyCircus, Bakerloo) (OxfordCircus, NottingHillGate, Central) (OxfordCircus, Euston, Victoria) 84 edges in total	**Traversal question:** (BondSt, _, Central), (_, _, Circle), (_, _, Circle), (_, _, Circle), (_, _, Circle), (_, _, Jubilee), (_, _, Jubilee), **Answer:** (BondSt, NottingHillGate, Central) (NottingHillGate, GloucesterRd, Circle) (Westminster, GreenPark, Jubilee) (GreenPark, BondSt, Jubilee)	**Shortest-path question:** (Moorgate, PiccadillyCircus, _) **Answer:** (Moorgate, Bank, Northern) (Bank, Holborn, Central) (Holborn, LeicesterSq, Piccadilly) (LeicesterSq, PiccadillyCircus, Piccadilly)

Figure 9.6: DNC was trained using randomly generated graphs (left). After training it was tested to see if it could navigate the London Underground by predicting shortest paths between stations (right) (Graves *et al.*, 2016).

metric maps. With topological representation, we only need to be able to navigate between these waypoints. This is a much simpler problem of path following which mostly boils down to obstacle avoidance and an ability to keep an agent moving in the right direction, however, the overall drift is not a problem anymore since it is "reset" when we reach the waypoint. In some sense, one might argue that the current visual odometry methods have already surpassed human-level abilities (we would hardly be able to draw a metric map of a large-scale environment), we just do not use them in the right way.

The key question is how to build such a graph and how to decide what data should be stored (*e.g.* nodes representing important locations could use dense 3D maps, while edges linking waypoints may use lighter sparse or semi-dense representation), in an automated way. One of the promising approaches is differential neural computer (Graves *et al.*, 2016).

9.3 Concluding Remarks

Most problems I have tackled in this thesis, have been formulated in the Conditional Random Field framework with per-pixel random variables. Although per-pixel CRFs have formed the basis for most low-level computer vision problems (optical flow, disparity estimation, segmentation, *etc.*) for a long time, they have been less and less efficient when used with unary potentials extracted by modern convolutional neural networks (CNNs) which capture local context well (Yu and Koltun, 2016). CNNs not only achieve state-of-the-art results but often also allow us to tackle multiple problems jointly in a principled way through multi-task learning (Kokkinos, 2017).

Nowadays, one can relatively easily train a state-of-the-art CNN for almost any low-level task. Typically, our main effort is to define a suitable (often per-pixel decomposable) loss function, set the hyper-parameters and architecture itself, gather sufficient amount of data and auto-differentiation handles the rest. While it may seem that creativity vanishes from our field, I would argue that this is actually great! We have finally reached the point when most of our low-level problems are mature enough for real-world applications. It just took 50 years of research, instead of a single summer as originally planned (Papert, 1966) but low-level computer vision has finally become a widely available commodity.

It is true that our field seems to be relatively flat at the moment, lacking diversity and generally not giving enough justice to "other-than-CNN" approaches. However, it is our choice to be less obsessed with well-defined but relatively low-level tasks and finally move on to higher-level tasks, to advanced scene understanding, approaching computer vision as "perception-not-measurement" and building complex "intelligent" machines.

The other question is whether deep learning will wash away the remaining parts of computer vision. Historically, we have always been borrowing ideas from other fields such as physics, optimization or machine learning. We have also always been moving on a spiral, forgetting and re-discovering the very same old ideas again and again. For someone who believes in structured prediction, end-to-end learning is a convenient tool which glues things together in a principled way. On the other hand, for people believing in "learning everything", structured loss functions and other handcrafted priors would become a convenient way to constrain the models to learn them more efficiently.

No matter what tools we will be (re-)using in the future, the geometry of a monocular camera *will always make our field unique!*

Bibliography

Abdel-Hakim AE, Farag AA (2006) CSIFT: A SIFT descriptor with color invariant characteristics. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Agarwal S, Mierle K, Others (2010) Ceres solver. http://ceres-solver.org

Agarwal S, Furukawa Y, Snavely N, Simon I, Curless B, Seitz SM, Szeliski R (2011) Building Rome in a day. Communications of the ACM 54:105–112

Agarwala A, Hertzmann A, Salesin DH, Seitz SM (2004) Keyframe-based tracking for rotoscoping and animation. ACM Transactions on Graphics (ACM TOG) 23(3):584–591, DOI 10.1145/1015706.1015764

Ajanthan T, Desmaison A, Bunel R, Salzmann M, Torr P, Kumar MP (2017) Efficient Linear Programming for Dense CRFs. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Amini AA, Weymouth TE, Jain RC (1990) Using dynamic programming for solving variational problems in vision. Transactions on Pattern Analysis and Machine Intelligence (T-PAMI) 12(9):855–867, DOI 10.1109/34.57681

Arnab A, Sapienza M, Golodetz S, Valentin J, Miksik O, Izadi S, Torr PHS (2015) Joint object-material category segmentation from audio-visual cues. In: British Machine Vision Conference (BMVC)

Arya S, Mount DM (1993) Approximate nearest neighbor queries in fixed dimensions. In: SODA '93: ACM-SIAM Symposium on Discrete algorithms

Arya S, Mount DM, Netanyahu NS, Silverman R, Wu AY (1998) An optimal algorithm for approximate nearest neighbor searching fixed dimensions. ACM-SIAM Symposium on Discrete algorithms 45(6):891–923, DOI 10.1145/293347.293348

Badino H, Huber D, Kanade T (2011) Integrating lidar into stereo for fast and improved disparity computation. In: International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT)

Bai X, Wang J, Simons D, Sapiro G (2009) Video snapcut: Robust video object cutout using localized classifiers. ACM Transactions on Graphics (ACM TOG) 28(3):70:1–70:11, DOI 10.1145/1531326.1531376

Barber D (2012) Bayesian Reasoning and Machine Learning. Cambridge University Press

Barfield W (2016) Fundamentals of Wearable Computers and Augmented Reality, Second Edition. CRC Press

Barron JT, Malik J (2012) Shape, albedo, and illumination from a single image of an unknown

object. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Barrow HG, Tenenbaum JM (1978) Recovering intrinsic scene characteristics from images. In: Hanson A, Riseman E (eds) Computer Vision Systems, Academic Press, pp 3–26

Barrow HG, Tenenbaum JM (1981) Interpreting line drawings as three-dimensional surfaces. Artificial Intelligence 17(1-3):75–116, DOI 10.1016/0004-3702(81)90021-7

Barto AG, Mahadevan S (2003) Recent advances in hierarchical reinforcement learning. Discrete Event Dynamic Systems

Bay H, Ess A, Tuytelaars T, Gool LV (2008) SURF: Speeded up robust features. Computer Vision and Image Understanding (CVIU) 110(3):346 – 359, DOI https://doi.org/10.1016/j.cviu.2007.09.014, Similarity Matching in Computer Vision and Multimedia

Becker F, Lenzen F, Kappes J, Schnörr C (2011) Variational Recursive Joint Estimation of Dense Scene Structure and Camera Motion from Monocular High Speed Traffic Sequences. In: International Conference on Computer Vision (ICCV)

Bertinetto L, Henriques JF, Valmadre J, Torr P, Vedaldi A (2016a) Learning feed-forward one-shot learners. In: Advances in Neural Information Processing Systems (NIPS)

Bertinetto L, Valmadre J, Golodetz S, Miksik O, Torr PHS (2016b) Staple: Complementary learners for real-time tracking. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Besl PJ, McKay ND (1992) A method for registration of 3-d shapes. Transactions on Pattern Analysis and Machine Intelligence (T-PAMI) 14(2):239–256, DOI 10.1109/34.121791

Bibby C (2010) Probabilistic methods for enhanced marine situational awareness. PhD thesis, University of Oxford

Bishop CM (2006) Pattern Recognition and Machine Learning. Springer

Blake A, Isard M (2000) Active contours. Springer-Verlag

Bleyer M, Rother C, Kohli P, Scharstein D, Sinha S (2011) Object stereo - joint stereo matching and object segmentation. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Borenstein J, Everett HR, Feng L (1996) Where am I? Sensors and Methods for Mobile Robot Positioning. Tech. rep., University of Michigan

Bouaziz S, Tagliasacchi A, Pauly M (2013) Sparse iterative closest point. In: Eurographics

Bouguet J (2000) Matlab camera calibration toolbox. http://www.vision.caltech.edu/bouguetj/calib_doc/, accessed: 2017-08-08

Boykov Y, Jolly MP (2001) Interactive graph cuts for optimal boundary and region segmentation of objects in nd images. In: International Conference on Computer Vision (ICCV)

Boykov Y, Veksler O, Zabih R (2001) Fast Approximate Energy Minimization via Graph Cuts. Transactions on Pattern Analysis and Machine Intelligence (T-PAMI) 23(11):1222–1239, DOI 10.1109/34.969114

Bratt B (2011) Rotoscoping: Techniques and tools for the Aspiring Artist. Taylor & Francis

Burgess N (2008) Spatial cognition and the brain. Annals of the New York Academy of Sciences 1124(1):77–97

Cadena C, Dick A, Reid I (2016) Multi-modal auto-encoders as joint estimators for robotics scene understanding. In: Robotics Science and Systems (RSS)

Calonder M, Lepetit V, Strecha C, Fua P (2010) BRIEF: Binary Robust Independent Elementary Features. In: European Conference on Computer Vision (ECCV)

Capturing Reality (2017) Capturing reality. https://www.capturingreality.com, accessed: 2017-08-08

Cashman TJ, Fitzgibbon AW (2013) What shape are dolphins? building 3D morphable models from 2D images. Transactions on Pattern Analysis and Machine Intelligence (T-PAMI) 35(1):232–244, DOI 10.1109/TPAMI.2012.68

Chen C, Seff A, Kornhauser A, Xiao J (2015a) Deepdriving: Learning affordance for direct perception in autonomous driving. In: International Conference on Computer Vision (ICCV)

Chen DM, Baatz G, Köser K, Tsai SS, Vedantham R, Pylvänäinen T, Roimela K, Chen X, Bach J, Pollefeys M, Girod B, Grzeszczuk R (2011) City-Scale Landmark Identification on Mobile Devices. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Chen J, Bautembach D, Izadi S (2013) Scalable real-time volumetric surface reconstruction. ACM Transactions on Graphics (ACM TOG) 32(4):113

Chen LC, Schwing AG, Yuille AL, Urtasun R (2015b) Learning Deep Structured Models. In: International Conference on Machine Learning (ICML)

Churchill W, Newman P (2012) Practice makes perfect? managing and leveraging visual experiences for lifelong navigation. In: International Conference on Robotics and Automation (ICRA)

Concha A, Civera J (2015) DPPTAM: Dense Piecewise Planar Tracking and Mapping from a Monocular Sequence . In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)

Concha A, Hussain W, Montano L, Civera J (2015) Incorporating Scene Priors to Dense Monocular Mapping. Autonomous Robots 39(3):279–292, DOI 10.1007/s10514-015-9465-9

Cordts M, Omran M, Ramos S, Rehfeld T, Enzweiler M, Benenson R, Franke U, Roth S, Schiele B (2016) The cityscapes dataset for semantic urban scene understanding. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Curless B, Levoy M (1996) A Volumetric Method for Building Complex Models from Range Images. In: SIGGRAPH, pp 303–312, DOI 10.1145/237170.237269

Daftry S, Zeng S, Bagnell JA, Hebert M (2016) Introspective perception: Learning to predict failures in vision systems. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)

Dahlkamp H, Kaehler A, Stavens D, Thrun S, Bradski G (2006) Self-supervised Monocular Road Detection in Desert Terrain. In: Robotics Science and Systems (RSS)

Dai A, Nießner M, Zollöfer M, Izadi S, Theobalt C (2017) Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface re-integration. ACM Transactions on Graphics (ACM TOG) 36(3):24:1–24:18, DOI 10.1145/3054739

Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: Interna-

tional Conference on Computer Vision and Pattern Recognition (CVPR)

Davison A (2003) Real-Time Simultaneous Localisation and Mapping with a Single Camera. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Davison AJ, Reid ID, Molton ND, Stasse O (2007) MonoSLAM: Real-Time Single Camera SLAM. Transactions on Pattern Analysis and Machine Intelligence (T-PAMI) 29(6):1052–1067, DOI 10.1109/TPAMI.2007.1049

Denis M, Loomis JM (2007) Perspectives on human spatial cognition: memory, navigation, and environmental learning. Psychological Research 71(3):235–239

Desmaison A, Bunel R, Kohli P, Torr P, Kumar MP (2016) Efficient continuous relaxations for dense crf. In: European Conference on Computer Vision (ECCV)

Diebel J, Thrun S (2005) An application of markov random fields to range sensing. In: Advances in Neural Information Processing Systems (NIPS)

Dollar P, Zitnick L (2013) Sketch tokens: A learned mid-level representation for contour and object detection. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Dolson J, Baek J, Plagemann C, Thrun S (2010) Upsampling range data in dynamic environments. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Dosovitskiy A, Koltun V (2017) Learning to act by predicting the future. In: International Conference on Learning Representations (ICLR)

Dou M, Taylor J, Fuchs H, Fitzgibbon AW, Izadi S (2015) 3d scanning deformable objects with a single RGBD sensor. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Dou M, Khamis S, Degtyarev Y, Davidson P, Fanello S, Kowdle A, Escolano SO, Rhemann C, Kim D, Taylor J, Kohli P, Tankovich V, Izadi S (2016) Fusion4d: Real-time performance capture of challenging scenes. In: SIGGRAPH, vol 35, pp 114:1–114:13, DOI 10.1145/2897824.2925969

Efros A (2017) Visual understanding without naming: Bypassing the language bottleneck. https://www.robots.ox.ac.uk/seminars/Extra/2015_07_13_AlyoshaEfros.pdf, accessed: 2017-08-08

Eigen D, Puhrsch C, Fergus R (2014) Depth map prediction from a single image using a multi-scale deep network. In: Advances in Neural Information Processing Systems (NIPS)

Engel J, Sturm J, Cremers D (2013) Semi-Dense Visual Odometry for a Monocular Camera. In: International Conference on Computer Vision (ICCV)

Engel J, Schöps T, Cremers D (2014a) LSD-SLAM: Large-scale direct monocular SLAM. In: European Conference on Computer Vision (ECCV)

Engel J, Sturm J, Cremers D (2014b) Scale-Aware Navigation of a Low-Cost Quadrocopter with a Monocular Camera. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)

Engel J, Stuckler J, Cremers D (2015) Large-scale direct slam with stereo cameras. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)

Everingham M, Van Gool L, Williams CKI, Winn J, Zisserman A (2010) The pascal visual object classes (voc) challenge. International Journal of Computer Vision (IJCV) 88(2):303–338, DOI 10.1007/s11263-009-0275-4

Faktor A, Irani M (2014) Video segmentation by non-local consensus voting. In: British Machine Vision Conference (BMVC)

Fan B, Wu F, Hu Z (2012) Rotationally invariant descriptors using intensity order pooling. Transactions on Pattern Analysis and Machine Intelligence (T-PAMI) 34(10):2031–2045, DOI 10.1109/TPAMI.2011.277

Fan Q, Zhong F, Lischinski D, Cohen-Or D, Chen B (2015) JumpCut: Non-successive mask transfer and interpolation for video cutout. ACM Transactions on Graphics (ACM TOG) 34(6):195:1–195:10, DOI 10.1145/2816795.2818105

Felzenswalb PF, Huttenlocher DP (2004) Efficient graph-based image segmentation. In: International Journal of Computer Vision (IJCV), vol 59, pp 167–181, DOI 10.1023/B:VISI.0000022288.19776.77

Felzenszwalb P, Zabih R (2011) Dynamic programming and graph algorithms in computer vision. Transactions on Pattern Analysis and Machine Intelligence (T-PAMI) 33(4):721–740, DOI 10.1109/TPAMI.2010.135

Felzenszwalb PF, Huttenlocher DP (2005) Pictorial structures for object recognition. International Journal of Computer Vision (IJCV) 61(1):55–79, DOI 10.1023/B:VISI.0000042934.15159.49

Felzenszwalb PF, Huttenlocher DP (2006) Efficient belief propagation for early vision. International Journal of Computer Vision (IJCV) 70(1):41–54, DOI 10.1007/s11263-006-7899-4

Felzenszwalb PF, Girshick RB, McAllester D, Ramanan D (2010) Object detection with discriminatively trained part-based models. Transactions on Pattern Analysis and Machine Intelligence (T-PAMI) 32(9):1627–1645, DOI 10.1109/TPAMI.2009.167

Fischler MA, Bolles RC (1981) Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Communications ACM 24(6):381–395, DOI 10.1145/358669.358692

Fischler MA, Elschlager RA (1973) The representation and matching of pictorial structures. IEEE Transactions on Computers C-22(1):67–92, DOI 10.1109/T-C.1973.223602

Floros G, Leibe B (2012) Joint 2D-3D Temporally Consistent Semantic Segmentation of Street Scenes. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Forster C, Pizzoli M, Scaramuzza D (2014) SVO: Fast Semi-Direct Monocular Visual Odometry. In: International Conference on Robotics and Automation (ICRA)

Fouhey DF, Delaitre V, Gupta A, Efros AA, Laptev I, Sivic J (2014) People watching: Human actions as a cue for single view geometry. International Journal of Computer Vision (IJCV) 110(3):259–274

Freedman DJ, Miller EK (2008) Neural mechanisms of visual categorization: insights from neurophysiology. Neuroscience & Biobehavioral Reviews 32(2):311–329

Froissard B, Konik H, Trémeau A, Dinet É (2014) Contribution of augmented reality solutions to assist visually impaired people in their mobility. In: Universal Access in

Human-Computer Interaction. Design for All and Accessibility Practice - 8th International Conference, UAHCI 2014, Held as Part of HCI International 2014, Heraklion, Crete, Greece, June 22-27, 2014, Proceedings, Part IV, Springer International Publishing, pp 182–191

Frost DP, Kahler O, Murray DW (2016) Object-aware bundle adjustment for correcting monocular scale drift. In: International Conference on Robotics and Automation (ICRA)

Furgale P, Maye J, Rehder J, Schneider T, Oth L (2015) Kalibr. https://github.com/ethz-asl/kalibr, accessed: 2017-08-08

Furukawa Y, Curless B, Seitz SM, Szeliski R (2009) Reconstructing Building Interiors from Images. In: International Conference on Computer Vision (ICCV)

Gallup D, Frahm JM, Mordohai P, Pollefeys M (2008) Variable baseline/resolution stereo. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Geiger A, Roser M, Urtasun R (2010) Efficient Large-Scale Stereo Matching. In: Asian Conference on Computer Vision (ACCV)

Geiger A, Ziegler J, Stiller C (2011) StereoScan: Dense 3d Reconstruction in Real-time. In: Intelligent Vehicles Symposium (IVS)

Geiger A, Lenz P, Urtasun R (2012) Are we ready for autonomous driving? The KITTI vision benchmark suite. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Gillenwater J (2014) Approximate inference for determinantal point processes. PhD thesis, University of Pennsylvania

Girshick R (2015) Fast R-CNN. In: International Conference on Computer Vision (ICCV)

Girshick R, Donahue J, Darrell T, Malik J (2014) Rich feature hierarchies for accurate object detection and semantic segmentation. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Girshick R, Iandola F, Darrell T, Malik J (2015) Deformable part models are convolutional neural networks. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Google (2014) ATAP Project Tango Google. URL http://www.google.com/atap/projecttango/

Grabner H, Matas J, Van Gool LJ, Cattin PC (2010) Tracking the invisible: Learning where the object might be. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Graves A, Wayne G, Reynolds M, Harley T, Danihelka I, Grabska-Barwinska A, Colmenarejo SG, Grefenstette E, Ramalho T, Agapiou J, Badia AP, Hermann KM, Zwols Y, Ostrovski G, Cain A, King H, Summerfield C, Blunsom P, Kavukcuoglu K, Hassabis D (2016) Hybrid computing using a neural network with dynamic external memory. Nature 538(7626):471–476

Grimmett H, Triebel R, Paul R, Posner I (2015) Introspective classification for robot perception. International Journal of Robotics Research (IJRR) 35(7):743–762, DOI 10.1177/0278364915587924

Gupta S, Girshick R, Arbeláez P, Malik J (2014) Learning rich features from RGB-D images

for object detection and segmentation. In: European Conference on Computer Vision (ECCV)

Gupta S, Davidson J, Levine S, Sukthankar R, Malik J (2017) Cognitive mapping and planning for visual navigation. arXiv preprint arXiv:170203920

Habbecke M, Kobbelt L (2008) LaserBrush: A Flexible Device for 3D Reconstruction of Indoor Scenes. In: ACM Symposium on Solid and Physical Modeling

Hammersley JM, Clifford PE (1971) Markov random fields on finite graphs and lattices. *Unpublished manuscript* URL http://www.statslab.cam.ac.uk/~grg/books/hammfest/hamm-cliff.pdf

Hamrick JB, Battaglia P, Tenenbaum JB (2011) Probabilistic internal physics models guide judgments about object dynamics. In: CogSci, cognitivesciencesociety.org

Häne C, Zach C, Cohen A, Angst R, Pollefeys M (2013) Joint 3D Scene Reconstruction and Class Segmentation. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Hartley R, Zisserman A (2003) Multiple view geometry in computer vision. Cambridge University Press

Hartley R, Kahl F, Torr PHS (2018) Discrete Optimization and Computer Vision, unpublished book

Hausknecht M, Stone P (2015) Deep recurrent q-learning for partially observable mdps. arXiv preprint arXiv:150706527

Held D, Levinson J, Thrun S (2013) Precision Tracking with Sparse 3D and Dense Color 2D Data. In: International Conference on Robotics and Automation (ICRA)

Henriques JF, Caseiro R, Martins P, Batista J (2015) High-speed tracking with kernelized correlation filters. Transactions on Pattern Analysis and Machine Intelligence (T-PAMI) 37(3):583–596

Hermans A, Floros G, Leibe B (2014) Dense 3D Semantic Mapping of Indoor Scenes from RGB-D Images. In: International Conference on Robotics and Automation (ICRA)

Herrera DC, Kannala J, Ladicky L, Heikkilä J (2013) Depth map inpainting under a second-order smoothness prior. In: Scandinavian Conference on Image Analysis (SCIA)

Hicks SL, Wilson I, Muhammed L, Worsfold J, Downes SM, Kennard C (2013) A Depth-Based Head-Mounted Visual Display to Aid Navigation in Partially Sighted Individuals. PLoS ONE DOI https://doi.org/10.1371/journal.pone.0067695

Hicks SL, Wilson I, van Rheede JJ, MacLaren RE, Downes SM, Kennard C (2014) Improved mobility with depth-based residual vision glasses. Investigative Ophthalmology & Visual Science 55

Hirschmüller H (2005) Accurate and Efficient Stereo Processing by Semi-Global Matching and Mutual Information. In: International Conference on Computer Vision and Pattern Recognition (CVPR), vol 2

Hirschmüller H (2008) Stereo processing by semiglobal matching and mutual information. Transactions on Pattern Analysis and Machine Intelligence (T-PAMI) 30(2):328–341, DOI 10.1109/TPAMI.2007.1166

Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural computation 9(8):1735–1780

Hosang J, Benenson R, Dollar P, Schiele B (2016) What makes for effective detection proposals? Transactions on Pattern Analysis and Machine Intelligence (T-PAMI) 38(4):814–830, DOI 10.1109/TPAMI.2015.2465908

Hu H, Munoz D, Bagnell JA, Hebert M (2013) Efficient 3-d scene analysis from streaming data. In: International Conference on Robotics and Automation (ICRA)

Huang AS, Bachrach A, Henry P, Krainin M, Maturana D, Fox D, Roy N (2011) Visual Odometry and Mapping for Autonomous Flight Using an RGB-D Camera. In: International Symposium of Robotics Research (ISRR)

Iannacci F, Turnquist E, Avrahami D, Patel SN (2011) The Haptic Laser: Multi-Sensation Tactile Feedback for At-a-Distance Physical Space Perception and Interaction. In: Human Factors in Computing Systems (CHI)

Innmann M, Zollhöfer M, Nießner M, Theobalt C, Stamminger M (2016) Volumedeform: Real-time volumetric non-rigid reconstruction. In: European Conference on Computer Vision (ECCV)

Izadi S, Kim D, Hilliges O, Molyneaux D, Newcombe RA, Kohli P, Shotton J, Hodges S, Freeman D, Davison A, Fitzgibbon A (2011) KinectFusion: Real-time 3D Reconstruction and Interaction Using a Moving Depth Camera. In: ACM Symposium on User Interface Software and Technology (UIST)

Jaderberg M, Simonyan K, Vedaldi A, Zisserman A (2014) Deep structured output learning for unconstrained text recognition. CoRR abs/1412.5903, URL http://arxiv.org/abs/1412.5903

Jaderberg M, Mnih V, Czarnecki WM, Schaul T, Leibo JZ, Silver D, Kavukcuoglu K (2016) Reinforcement learning with unsupervised auxiliary tasks. arXiv preprint arXiv:161105397

Jakowski W, Kempka M, Wydmuch M, Toczek J (2016) Vizdoom competition. URL http://vizdoom.cs.put.edu.pl/competition-cig-2016

Je C, Park HM (2013) Optimized hierarchical block matching for fast and accurate image registration. Signal Processing: Image Communication 28(7):779–791, DOI 10.1016/j.image.2013.04.002

Jegelka S, Bilmes J (2011) Submodularity beyond submodular energies: coupling edges in graph cuts. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Jégou H, Zisserman A (2014) Triangulation embedding and democratic aggregation for image search. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Jégou H, Perronnin F, Douze M, Sánchez J, Pérez P, Schmid C (2012) Aggregating local image descriptors into compact codes. Transactions on Pattern Analysis and Machine Intelligence (T-PAMI) 34(9):1704–1716, DOI 10.1109/TPAMI.2011.235

Jojic V, Gould S, Koller D (2010) Accelerated dual decomposition for MAP inference. In: International Conference on Machine Learning (ICML)

Kaess M, Johannsson H, Roberts R, Ila V, Leonard JJ, Dellaert F (2012) iSAM2: Incremental

smoothing and mapping using the Bayes tree. International Journal of Robotics Research (IJRR) 31(2), DOI 10.1177/0278364911430419

Kähler O, Prisacariu VA, Murray DW (2016) Real-time large-scale dense 3d reconstruction with loop closure. In: European Conference on Computer Vision (ECCV)

Kass M, Witkin A, Terzopoulos D (1988) Snakes: Active contour models. International Journal of Computer Vision (IJCV) 1(4):321–331, DOI 10.1007/BF00133570

Ke Y, Sukthankar R (2004) Pca-sift: A more distinctive representation for local image descriptors. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Kempka M, Wydmuch M, Runc G, Toczek J, Jaśkowski W (2016) Vizdoom: A doom-based ai research platform for visual reinforcement learning. arXiv preprint arXiv:160502097

Kendall A, Grimes M, Cipolla R (2015) Posenet: A convolutional network for real-time 6-dof camera relocalization. In: International Conference on Computer Vision (ICCV)

Khan EA, Reinhard E, Fleming RW, Bülthoff HH (2006) Image-based material editing. ACM Transactions on Graphics (ACM TOG) pp 654–663, DOI 10.1145/1179352.1141937

Kitani KM, Ziebart BD, Bagnell JAD, Hebert M (2012) Activity forecasting. In: European Conference on Computer Vision (ECCV)

Kitt B, Geiger A, Lategahn H (2010) Visual odometry based on stereo image sequences with ransac-based outlier rejection scheme. In: Intelligent Vehicles Symposium (IV)

Klein G, Murray DW (2007) Parallel tracking and mapping for small ar workspaces. In: International Symposium on Mixed and Augmented Reality (ISMAR)

Kleinberg JM (2000) Navigation in a small world. Nature 406(6798):845–845, DOI 10.1038/35022643

Kohli P, Torr PH (2007) Dynamic graph cuts for efficient inference in Markov random fields. Transactions on Pattern Analysis and Machine Intelligence (T-PAMI) 29(12):2079–2088, DOI 10.1109/TPAMI.2007.1128

Kohli P, Kumar MP, Torr PHS (2009) P3 & beyond: Move making algorithms for solving higher order functions. In: Transactions on Pattern Analysis and Machine Intelligence (T-PAMI), vol 31, pp 1645–1656, DOI 10.1109/TPAMI.2008.217

Kokkinos I (2017) Ubernet: Training a 'universal' convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. International Conference on Computer Vision and Pattern Recognition (CVPR)

Koller D, Friedman N (2009) Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning. The MIT Press

Kolmogorov V, Zabin R (2004) What energy functions can be minimized via graph cuts? Transactions on Pattern Analysis and Machine Intelligence (T-PAMI) 26(2):147–159, DOI 10.1109/TPAMI.2004.1262177

Komodakis N, Paragios N, Tziritas G (2011) MRF energy minimization and beyond via dual decomposition. Transactions on Pattern Analysis and Machine Intelligence (T-PAMI) 33(3):531–552, DOI 10.1109/TPAMI.2010.108

Koppula HS, Anand A, Joachims T, Saxena A (2011) Semantic Labeling of 3D Point Clouds

for Indoor Scenes. In: Advances in Neural Information Processing Systems (NIPS)

Krähenbühl P, Koltun V (2011) Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials. In: Advances in Neural Information Processing Systems (NIPS)

Kristan M, Leonardis A, Matas J, Felsberg M, Pflugfelder R, Čehovin L, Vojir T, Häger G, Lukežič A, Fernandez G (2013-2017) The visual object tracking challenge. http://www.votchallenge.net/

Kuemmerle R, Grisetti G, Strasdat H, Konolige K, Burgard W (2011) g2o: A general framework for graph optimization. In: International Conference on Robotics and Automation (ICRA)

Kulesza A, Taskar B (2012) Determinantal point processes for machine learning. arXiv preprint arXiv:12076083

Kulkarni TD, Narasimhan KR, Saeedi A, Tenenbaum JB (2016) Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. arXiv preprint arXiv:160406057

Kumar MP, Kolmogorov V, Torr PHS (2009) An analysis of convex relaxations for map estimation of discrete mrfs. Journal of Machine Learning Research (JMLR) 10:71–106

Kumar PM, Kohli P (2008) Map estimation algorithms in computer vision. http://www.robots.ox.ac.uk/~pawan/eccv08_tutorial/index.html, accessed: 2017-08-08

Kundu A, Krishna KM, Jawahar CV (2011) Realtime multibody visual slam with a smoothly moving monocular camera. International Conference on Computer Vision (ICCV)

Kundu A, Li Y, Dellaert F, Li F, Rehg JM (2014) Joint Semantic Segmentation and 3D Reconstruction from Monocular Video. In: European Conference on Computer Vision (ECCV)

Ladicky L (2012) Graphcut-based optimisation for computer vision. https://www.inf.ethz.ch/personal/ladickyl/graphcut.pdf, accessed: 2017-08-08

Ladický L, Sturgess P, Russell C, Sengupta S, Bastnlar Y, Clocksin W, , Torr PH (2012) Joint optimization for object class segmentation and dense stereo reconstruction. In: International Journal of Computer Vision (IJCV), vol 100, pp 122–133, DOI 10.1007/s11263-011-0489-0

Ladicky L, Russell C, Kohli P, Torr PH (2014) Associative hierarchical random fields. Transactions on Pattern Analysis and Machine Intelligence (T-PAMI) 36(6):1056–1077, DOI 10.1109/TPAMI.2013.165

Lafferty JD, McCallum A, Pereira FCN (2001) Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: International Conference on Machine Learning (ICML)

Lample G, Chaplot DS (2016) Playing fps games with deep reinforcement learning. arXiv preprint arXiv:160905521

LeCun Y, Boser B, Denker JS, Howard RE, Habbard W, Jackel LD, Henderson D (1990) Handwritten digit recognition with a back-propagation network. In: Advances in Neural Information Processing Systems (NIPS)

Lee N, Choi W, Vernaza P, Choy CB, Torr PHS, Chandraker M (2017) Desire: Distant future

prediction in dynamic scenes with interacting agents. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Lepetit V, Moreno-Noguer F, Fua P (2009) EPnP: An accurate O(n) solution to the PnP problem. International Journal of Computer Vision (IJCV) 81(2):155–166, DOI 10.1007/s11263-008-0152-6

Leutenegger S, Chli M, Siegwart R (2011) Brisk: Binary robust invariant scalable keypoints. In: International Conference on Computer Vision (ICCV)

Li H, Summer R, Pauly M (2008) Global Correspondence Optimization for Non-Rigid Registration of Depth Scans. Proceedings of the Symposium on Geometry Processing

Li W, Azimi S, Leonardis A, Fritz M (2016a) To fall or not to fall: A visual approach to physical stability prediction. CoRR abs/1604.00066

Li W, Viola F, Starck J, Brostow GJ, Campbell N (2016b) Roto++: Accelerating professional rotoscoping using shape manifolds. ACM Transactions on Graphics (ACM TOG) 35(4):62:1–62:15, DOI 10.1145/2897824.2925973

Liu A, Marschner S, Snavely N (2016) Caliber: Camera localization and calibration using rigidity constraints. International Journal of Computer Vision (IJCV) 118(1):1–21, DOI 10.1007/s11263-015-0866-1

Lowe DG (2004) Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision (IJCV) 60(2):91–110

Lu Y, Bai X, Shapiro L, Wang J (2016) Coherent parametric contours for interactive video object segmentation. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Lucas B, Kanade T (1981) An Iterative Image Registration Technique with an Application to Stereo Vision. In: International Joint Conference on Artificial Intelligence (IJCAI)

Mariotti SP (2010) Global Data on Visual Impairments 2010. Tech. rep., World Health Organization

Märki N, Perazzi F, Wang O, Sorkine-Hornung A (2016) Bilateral space video segmentation. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Matas J, Chum O, Urban M, Pajdla T (2002) Robust wide-baseline stereo from maximally stable extremal regions. British Machine Vision Conference (BMVC)

Matas J, Chum O, Urban M, Pajdla T (2004) Robust wide-baseline stereo from maximally stable extremal regions. Image and Vision Computing 22(10):761–767

Medrano C, Herrero J, Martnez J, Orrite C (2009) Mean field approach for tracking similar objects. In: Computer Vision and Image Understanding (CVIU)

Mikolajczyk K, Schmid C (2005) A performance evaluation of local descriptors. Transactions on Pattern Analysis and Machine Intelligence (T-PAMI) 27(10):1615–1630, DOI 10.1109/TPAMI.2005.188

Miksik O, Mikolajczyk K (2012) Evaluation of local detectors and descriptors for fast feature matching. In: International Conference on Pattern Recognition (ICPR)

Miksik O, Munoz D, Bagnell JA, Hebert M (2013) Efficient temporal consistency for streaming video scene analysis. In: International Conference on Robotics and Automation

(ICRA)

Miksik O, Amar Y, Vineet V, Pérez P, Torr P (2015a) Incremental Dense Multi-Modal 3D Scene Reconstruction. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)

Miksik O, Vineet V, Lidegaard M, Prasaath R, Nießner M, Golodetz S, Hicks SL, Perez P, Izadi S, Torr PHS (2015b) The semantic paintbrush: Interactive 3d mapping and recognition in large outdoor spaces. In: Human Factors in Computing Systems (CHI)

Milan A, Leal-Taixé L, Reid I, Roth S, Schindler K (2015-2017) MOT: A benchmark for multi-object tracking. arXiv:160300831 [cs] ArXiv: 1603.00831

Minka T (2005) Divergence measures and message passing. Tech. rep., Microsoft Research

Mnih V, Kavukcuoglu K, Silver D, Graves A, Antonoglou I, Wierstra D, Riedmiller M (2013) Playing atari with deep reinforcement learning. In: NIPS Deep Learning Workshop

Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, Bellemare MG, Graves A, Riedmiller M, Fidjeland AK, Ostrovski G, Petersen S, Beattie C, Sadik A, Antonoglou I, King H, Kumaran D, Wierstra D, Legg S, Hassabis D (2015) Human-level control through deep reinforcement learning. Nature 518(7540):529–533

Mobahi H, Fisher J (2015) On the Link between Gaussian Homotopy Continuation and Convex Envelopes. In: Energy Minimization Methods Computer Vision and Pattern Recognition (EMMCVPR)

Moser EI, Kropff E, Moser MB (2008) Place cells, grid cells, and the brain's spatial representation system. Annual Review of Neuroscience 31(1):69–89

Muja M, Lowe DG (2009) Fast approximate nearest neighbors with automatic algorithm configuration. In: In VISAPP ICCVTA

Munoz D, Bagnell JA, Vandapel N, Hebert M (2009) Contextual Classification with Functional Max-Margin Markov Networks. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Munoz D, Bagnell JA, Hebert M (2010) Stacked Hierarchical Labeling. In: European Conference on Computer Vision (ECCV)

Munoz D, Bagnell JA, Hebert M (2012) Co-inference machines for multi-modal scene analysis. In: European Conference on Computer Vision (ECCV)

Mur-Artal R, Montiel JMM, Tardós JD (2015) ORB-SLAM: a versatile and accurate monocular SLAM system. Transactions on Robotics (T-RO) 31(5):1147–1163, DOI 10.1109/TRO.2015.2463671

Newcombe RA, Izadi S, Hilliges O, Molyneaux D, Kim D, Davison AJ, Kohli P, Shotton J, Hodges S, Fitzgibbon A (2011a) Kinectfusion: Real-time dense surface mapping and tracking. In: International Symposium on Mixed and Augmented Reality (ISMAR)

Newcombe RA, Lovegrove SJ, Davison AJ (2011b) DTAM: Dense tracking and mapping in real-time. In: International Conference on Computer Vision (ICCV)

Newcombe RA, Fox D, Seitz SM (2015) Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Nguyen T, Grasset R, Schmalstieg D, Reitmayr G (2013) Interactive syntactic modeling with a single-point laser range finder and camera. In: International Symposium on Mixed and Augmented Reality (ISMAR)

NHTSA U (2008) National motor vehicle crash causation survey – report to US Congress. Tech. rep., U.S. Department of Transportation, DOT HS 811 059

Nießner M, Zollhöfer M, Izadi S, Stamminger M (2013) Real-time 3D Reconstruction at Scale using Voxel Hashing. ACM Transactions on Graphics (ACM TOG)

Nister D, Naroditsky O, Bergen J (2004a) Indoor Positioning Using Multi-Frequency RSS with Foot-Mounted INS. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Nister D, Naroditsky O, Bergen J (2004b) Visual odometry. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Nowozin S, Lampert CH (2011) Structured learning and prediction in computer vision. Foundations and Trends in Computer Graphics Vision 6:185–365, DOI 10.1561/0600000033

Nurutdinova I, Fitzgibbon AW (2015) Towards pointless structure from motion: 3d reconstruction and camera parameters from general 3d curves. In: International Conference on Computer Vision (ICCV)

Oliva A, Torralba A (2007) The role of context in object recognition. Trends in Cognitive Sciences 11(12):520–527

Olsen DR, Nielsen T (2001) Laser Pointer Interaction. In: Human Factors in Computing Systems (CHI)

Orts-Escolano S, Rhemann C, Fanello S, Chang W, Kowdle A, Degtyarev Y, Kim D, Davidson P, Khamis S, Dou M, Tankovich V, Loop C, Cai Q, Chou P, Mennicken S, Valentin J, Pradeep V, Wang S, Kang SB, Kohli P, Lutchyn Y, Keskin C, Izadi S (2016) Holoportation: Virtual 3d teleportation in real-time. In: ACM Symposium on User Interface Software and Technology (UIST)

Papert S (1966) The summer vision project. URL \unhbox\voidb@x\hbox{http://hdl.handle.net/1721.1/6125}

Payen de La Garanderie G, Breckon T (2014) Improved depth recovery in consumer depth cameras via disparity space fusion within cross-spectral stereo. In: British Machine Vision Conference (BMVC)

Perazzi F, Pont-Tuset J, McWilliams B, Gool LV, Gross M, Sorkine-Hornung A (2016) A benchmark dataset and evaluation methodology for video object segmentation. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Pérez-Rúa JM, Crivelli T, Pérez P (2016) Object-guided motion estimation. Computer Vision and Image Understanding (CVIU) 153:88–99, DOI https://doi.org/10.1016/j.cviu.2016.05.005

Perronnin F, Snchez J, Mensink T (2006) Fisher kernels on visual vocabularies for image categorization

Perronnin F, Snchez J, Mensink T (2010) Improving the fisher kernel for large-scale image classification. In: European Conference on Computer Vision (ECCV)

Pollefeys M, Koch R, Gool LJV (1999) A simple and efficient rectification method for general

motion. In: International Conference on Computer Vision (ICCV)

Potapova E, Varadarajan KM, Richtsfeld A, Zillich M, Vincze M (2014) Attention-driven object detection and segmentation of cluttered table scenes using 2.5d symmetry. In: International Conference on Robotics and Automation (ICRA)

Pradeep V, Rhemann C, Izadi S, Zach C, Bleyer M, Bathiche S (2013) Monofusion: Real-time 3D reconstruction of small scenes with a single web camera. In: International Symposium on Mixed and Augmented Reality (ISMAR)

Premebida C, Carreira J, Batista J, Nunes U (2014) Pedestrian detection combining rgb and dense lidar data. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)

Prince S (2012) Computer Vision: Models, Learning and Inference. Cambridge University Press

Prisacariu VA, Reid I (2009) fastHOG - a real-time GPU implementation of HOG. Tech. Rep. 2310/09, University of Oxford

Qin Y, Shi Y, Jiang H, Yu C (2010) Structured Laser Pointer: Enabling Wrist-Rolling Movements as a New Interactive Dimension. In: International Conference on Advanced Visual Interfaces

Rav-Acha A, Kohli P, Rother C, Fitzgibbon A (2008) Unwrap mosaics: A new representation for video editing. ACM Transactions on Graphics (ACM TOG) 27(3):17:1–17:11, DOI 10.1145/1360612.1360616

Ren S, He K, Girshick R, Sun J (2017) Faster R-CNN: Towards real-time object detection with region proposal networks. Transactions on Pattern Analysis and Machine Intelligence (T-PAMI) 39(6):1137–1149

Rosten E, Drummond T (2006) Machine learning for high-speed corner detection. In: European Conference on Computer Vision (ECCV)

Rosten E, Porter R, Drummond T (2010) Faster and better: A machine learning approach to corner detection. Transactions on Pattern Analysis and Machine Intelligence (T-PAMI) 32(1):105–119, DOI 10.1109/TPAMI.2008.275

Rother C, Kolmogorov V, Blake A (2004) Grabcut: Interactive foreground extraction using iterated graph cuts. ACM Transactions on Graphics (ACM TOG) 23(3):309–314, DOI 10.1145/1015706.1015720

Rublee E, Rabaud V, Konolige K, Bradski GR (2011) Orb: An efficient alternative to sift or surf. In: International Conference on Computer Vision (ICCV)

Rumelhart DE, Hinton GE, Williams RJ (1988) Learning representations by back-propagating errors. In: Anderson JA, Rosenfeld E (eds) Neurocomputing: Foundations of Research, MIT Press, Cambridge, MA, USA, pp 696–699

Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein M, Berg AC, Fei-Fei L (2015) ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV) 115(3):211–252, DOI 10.1007/s11263-015-0816-y

Salas M, Hussain W, Concha A, Montano L, Civera J, Montiel J (2015) Layout Aware Visual Tracking and Mapping. In: IEEE/RSJ International Conference on Intelligent Robots and

Systems (IROS)

Salas-Moreno R, Glocker B, Kelly P, Davison A (2014) Dense Planar SLAM. In: International Symposium on Mixed and Augmented Reality (ISMAR)

Salas-Moreno RF, Newcombe RA, Strasdat H, Kelly PHJ, Davison AJ (2013) SLAM++: SLAM at the Level of Objects. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Scharstein D, Szeliski R (2001) A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. International Journal of Computer Vision (IJCV) pp 131–140, DOI 10.1109/SMBV.2001.988771

Schaul T, Quan J, Antonoglou I, Silver D (2016) Prioritized experience replay. International Conference on Learning Representations (ICLR)

Schöps T, Engel J, Cremers D (2014) Semi-Dense Visual Odometry for AR on a Smartphone. In: International Symposium on Mixed and Augmented Reality (ISMAR)

Sengupta S, Sturgess P, Ladický L, Torr PHS (2012) Automatic Dense Visual Semantic Mapping from Street-Level Imagery. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)

Sengupta S, Greveson E, Shahrokni A, Torr PHS (2013) Urban 3D Semantic Modelling Using Stereo Vision. In: International Conference on Robotics and Automation (ICRA)

Shekhovtsov A (2014) Maximum persistency in energy minimization. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Shelhamer E, Long J, Darrell T (2017) Fully convolutional networks for semantic segmentation. Transactions on Pattern Analysis and Machine Intelligence (T-PAMI) 39(4):640–651, DOI 10.1109/TPAMI.2016.2572683

Shotton J, Johnson M, Cipolla R (2008) Semantic texton forests for image categorization and segmentation. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Silberman N, Hoiem D, Kohli P, Fergus R (2012) Indoor segmentation and support inference from rgbd images. In: European Conference on Computer Vision (ECCV)

Silver D, Huang A, Maddison CJ, Guez A, Sifre L, van den Driessche G, Schrittwieser J, Antonoglou I, Panneershelvam V, Lanctot M, Dieleman S, Grewe D, Nham J, Kalchbrenner N, Sutskever I, Lillicrap T, Leach M, Kavukcuoglu K, Graepel T, Hassabis D (2016) Mastering the game of Go with deep neural networks and tree search. Nature 529(7587):484–489, DOI 10.1038/nature16961

Sinha SN, Scharstein D, Szeliski R (2014) Efficient high-resolution stereo matching using local plane sweeps. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Sivic J, Zisserman A (2003) Video google: A text retrieval approach to object matching in videos. In: International Conference on Computer Vision (ICCV)

Snavely N, Seitz SM, Szeliski R (2006) Photo tourism: Exploring photo collections in 3d. In: SIGGRAPH, vol 25, pp 835–846, DOI 10.1145/1141911.1141964

Song S, Xiao J (2014) Sliding shapes for 3d object detection in depth images. In: European Conference on Computer Vision (ECCV)

Song S, Yu F, Zeng A, Chang AX, Savva M, Funkhouser T (2017) Semantic scene completion from a single depth image. International Conference on Computer Vision and Pattern Recognition (CVPR)

Stauffer C, Grimson E (1999) Adaptive background mixture models for real-time tracking. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Strasdat H (2012) Local accuracy and global consistency for efficient visual SLAM. PhD thesis, Imperial College London

Strasdat H, Montiel JMM, Davison AJ (2010) Real-time monocular SLAM: why filter? In: International Conference on Robotics and Automation (ICRA)

Stühmer J, Gumhold S, Cremers D (2010) Parallel Generalized Thresholding Scheme for Live Dense Geometry from a Handheld Camera. In: ECCV Workshops (1)

Stühmer J, Gumhold S, Cremers D (2010) Real-time dense geometry from a handheld camera. In: German Conference on Pattern Recognition (DAGM)

Sturm J, Engelhard N, Endres F, Burgard W, Cremers D (2012) A Benchmark for the Evaluation of RGB-D SLAM Systems. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)

Sutton RS, Precup D, Singh S (1999) Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. Artificial Intelligence 112(1-2):181–211, DOI 10.1016/S0004-3702(99)00052-1

Szeliski R, Zabih R, Scharstein D, Veksler O, Kolmogorov V, Agarwala A, Tappen M, Rother C (2008) A comparative study of energy minimization methods for markov random fields with smoothness-based priors. Transactions on Pattern Analysis and Machine Intelligence (T-PAMI) 30(6):1068–1080

Taneja A, Ballan L, Pollefeys M (2013) City-Scale Change Detection in Cadastral 3D Models using Images. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Tang GY (1982) A discrete version of Green's theorem. Transactions on Pattern Analysis and Machine Intelligence (T-PAMI) pp 242–249, DOI 10.1109/TPAMI.1982.4767241

Tarrio J, Pedre S (2015) Realtime edge-based visual odometry for a monocular camera. In: International Conference on Computer Vision (ICCV)

Taskar B, Guestrin C, Koller D (2003) Max-margin markov networks. In: Advances in Neural Information Processing Systems (NIPS)

Thrun S, Burgard W, Fox D (2005) Probabilistic Robotics. The MIT Press

Torii A, Arandjelović R, Sivic J, Okutomi M, Pajdla T (2015) 24/7 place recognition by view synthesis. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Torr PHS, Criminisi A (2004) Dense stereo using pivoted dynamic programming. In: Image and Vision Computing, vol 22, pp 795–806

Triggs B, McLauchlan PF, Hartley RI, Fitzgibbon AW (1999) Bundle adjustment - a modern synthesis. In: Workshop on Vision Algorithms

Tsai YH, Yang MH, Black MJ (2016) Video segmentation via object flow. In: International

Conference on Computer Vision and Pattern Recognition (CVPR)

Tsochantaridis I, Hofmann T, Joachims T, Altun Y (2004) Support vector machine learning for interdependent and structured output spaces. In: International Conference on Machine Learning (ICML)

Tuytelaars T, Mikolajczyk K (2008) Local Invariant Feature Detectors: A Survey. Now Publishers Inc.

Urmson C (2015) How a driverless car sees the road. https://www.ted.com/talks/chris_urmson_how_a_driverless_car_sees_the_road/discussion?share=1ea17737b

Urmson C, Anhalt J, Bae H, Bagnell JAD, Baker CR, Bittner RE, Brown T, Clark MN, Darms M, Demitrish D, Dolan JM, Duggins D, Ferguson D, Galatali T, Geyer CM, Gittleman M, Harbaugh S, Hebert M, Howard T, Kolski S, Likhachev M, Litkouhi B, Kelly A, McNaughton M, Miller N, Nickolaou J, Peterson K, Pilnick B, Rajkumar R, Rybski P, Sadekar V, Salesky B, Seo YW, Singh S, Snider JM, Struble JC, Stentz AT, Taylor M, Whittaker WRL, Wolkowicki Z, Zhang W, Ziglar J (2008) Autonomous driving in urban environments: Boss and the urban challenge. Journal of Field Robotics Special Issue on the 2007 DARPA Urban Challenge, Part I 25(8):425–466

Valentin J, Vineet V, Cheng MM, Kim D, Shotton J, Kohli P, Niessner M, Criminisi A, Izadi S, Torr P (2015) SemanticPaint: Interactive 3D Labeling and Learning at your Fingertips. ACM Transactions on Graphics (ACM TOG) 34(5):154:1–154:17, DOI 10.1145/2751556

Valentin JPC, Sengupta S, Warrell J, Shahrokni A, Torr PHS (2013) Mesh Based Semantic Modelling for Indoor and Outdoor Scenes. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Varma M, Zisserman A (2003) Texture classification: Are filter banks necessary? In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Veksler O (2005) Stereo correspondence by dynamic programming on a tree. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Vineet V, Miksik O, Lidegaard M, Nießner M, Golodetz S, Prisacariu VA, Kähler O, Murray DW, Izadi S, Perez P, Torr PHS (2015) Incremental dense semantic stereo fusion for large-scale semantic scene reconstruction. In: International Conference on Robotics and Automation (ICRA)

Viola P, Jones MJ (2004) Robust real-time face detection. International Journal of Computer Vision (IJCV) 57:137–154, DOI 10.1023/B:VISI.0000013087.49260.fb

Wainwright M, Jaakkola T, Willsky A (2002) Map estimation via agreement on (hyper)trees: Message-passing and linear programming approaches. IEEE Transactions on Information Theory 51(11):3697–3717, DOI 10.1109/TIT.2005.856938

Wainwright MJ, Jordan MI (2008) Graphical models, exponential families, and variational inference. Foundations and Trends in Machine Learning 1:1–305, DOI 10.1561/2200000001

Wang CC, Thorpe C, Thrun S, Hebert M, Durrant-Whyte H (2007) Simultaneous localization, mapping and moving object tracking. IJRR 26(9):889–916, DOI 10.1177/0278364907081229

Wang J, Kumar S, Chang SF (2010a) Semi-supervised hashing for scalable image retrieval. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Wang J, Yang J, Yu K, Lv F, Huang T, Gong Y (2010b) Locality-constrained linear coding for image classification. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Wang Z, Fan B, Wu F (2011) Local intensity order pattern for feature description. In: International Conference on Computer Vision (ICCV)

Wang Z, Schaul T, Hessel M, van Hasselt H, Lanctot M, de Freitas N (2016) Dueling Network Architectures for Deep Reinforcement Learning. arXiv preprint arXiv:151106581

Watts DJ, Strogatz SH (1998) Collective dynamics of 'small-world' networks. Nature 393(6684):440–442, DOI 10.1038/30918

Wendel A, Maurer M, Graber G, Pock T, Bischof H (2012) Dense Reconstruction On-the-fly. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Werner F, Maire F, Sitte J, Choset H, Tully S, Kantor G (2009) Topological SLAM using neighbourhood information of places. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)

Werner T (2007) A linear programming approach to max-sum problem: A review. Transactions on Pattern Analysis and Machine Intelligence (T-PAMI) 29(7):1165–1179, DOI 10.1109/TPAMI.2007.1036

Westheimer G (1979) Cooperative Neural Processes Involved in Stereoscopic Acuity. EBR 36:585–597, DOI 10.1007/BF00238525

Whelan T, Johannsson H, Kaess M, Leonard JJ, McDonald J (2013) Robust Real-Time Visual Odometry for Dense RGB-D Mapping. In: International Conference on Robotics and Automation (ICRA)

Whelan T, Salas-Moreno R, Glocker B, Davison A, Leutenegger S (2016) Elasticfusion: Real-time dense slam and light source estimation. International Journal of Robotics Research (IJRR) 35(14):1697–1716, DOI 10.1177/0278364916669237

Wienss C, Nikitin I, Goebbels G, Troche K, Göbel M, Nikitina L, Müller S (2006) Sceptre: An Infrared Laser Tracking System for Virtual Environments. In: ACM Symposium on Virtual Reality Software and Technology, pp 45–50, DOI 10.1145/1180495.1180506

Williams DJ, Shah M (1992) A fast algorithm for active contours and curvature estimation. Computer Vision and Image Understanding (CVIU) 55:14–26, DOI 10.1016/1049-9660(92)90003-L

Woodford OJ, Torr PHS, Reid ID, Fitzgibbon AW (2009) Global stereo reconstruction under second-order smoothness priors. Transactions on Pattern Analysis and Machine Intelligence (T-PAMI) 31:2115–2128, DOI 10.1109/TPAMI.2009.131

Wright S (2006) Digital compositing for film and video. Taylor & Francis

Wu J, Yildirim I, Lim JJ, Freeman B, Tenenbaum J (2015) Galileo: Perceiving physical object properties by integrating a physics engine with deep learning. In: Advances in Neural Information Processing Systems (NIPS)

Xiao J, Furukawa Y (2014) Reconstructing the world's museums. International Journal of Computer Vision (IJCV) 110(3):243–258

Xiong X, la Torre FD (2014) Supervised descent method for solving nonlinear least squares problems in computer vision. CoRR abs/1405.0601

Xiong X, la Torre FD (2015) Global supervised descent method. In: International Conference on Computer Vision and Pattern Recognition (CVPR)

Xiong X, Munoz D, Bagnell JA, Hebert M (2011) 3-D Scene Analysis via Sequenced Predictions over Points and Regions. In: International Conference on Robotics and Automation (ICRA)

Yamaguchi K, McAllester DA, Urtasun R (2014) Efficient joint segmentation, occlusion labeling, stereo and flow estimation. In: European Conference on Computer Vision (ECCV)

Yang J, Li H (2015) Dense, Accurate Optical Flow Estimation with Piecewise Parametric Model. In: European Conference on Computer Vision (ECCV)

Yanover C, Weiss Y (2003) Finding the m most probable configurations using loopy belief propagation. Advances in Neural Information Processing Systems (NIPS)

Yedidia JS, Freeman WT, Weiss Y (2003) Understanding belief propagation and its generalizations. In: Lakemeyer G, Nebel B (eds) Exploring Artificial Intelligence in the New Millennium, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp 239–269

Yu F, Koltun V (2016) Multi-scale context aggregation by dilated convolutions. In: International Conference on Learning Representations (ICLR)

Zach C (2014) Robust Bundle Adjustment Revisited. In: European Conference on Computer Vision (ECCV)

Zbontar J, LeCun Y (2016) Stereo matching by training a convolutional neural network to compare image patches. Journal of Machine Learning Research (JMLR) 17:2287–2318

Zeiler MD, Fergus R (2014) Visualizing and understanding convolutional networks. In: European Conference on Computer Vision (ECCV)

Zhang C, Li Z, Cheng Y, Cai R, Chao H, Rui Y (2015) MeshStereo: A Global Stereo Model with Mesh Alignment Regularization for View Interpolation. In: International Conference on Computer Vision (ICCV)

Zhang Z, Fidler S, Urtasun R (2016) Instance-level segmentation with deep densely connected mrfs. International Conference on Computer Vision and Pattern Recognition (CVPR)

Zheng S, Jayasumana S, Romera-Paredes B, Vineet V, Su Z, Du D, Huang C, Torr PHS (2015) Conditional random fields as recurrent neural networks. In: International Conference on Computer Vision (ICCV)