CORAL: Fast, Parallel Multi Model Fitting with Applications to Mapping, Perception and Scene Reconstruction and Understanding

Paul Amayo

Linacre College

University of Oxford

A thesis submitted for the degree of

Doctor of Philosophy

Trinity 2018

Acknowledgements

I would like to thank my supervisors, Professor Paul Newman, Dr Pedro Pinies and Dr Lina Maria Paz who have been a constant source of teaching, encouragement, correction and inspiration throughout my time in Oxford. From the patience they have shown me as I explored different blind alleys to the guidance that set me back on track I am eternally grateful. Being a part of MRG and now ORI has been greatly rewarding and the vision and effort that Paul put into it has been clear from the time I started. My thanks also go to my colleagues, the engineers and everyone that makes ORI the place it has been.

In addition thanks go to my examiners, Dr Jonathan Gammel and Prof Andrew Calway who's attention to details and thorough questions enabled me to greatly improve this work.

Most importantly I would like to thank my parents, Daddy and Mummy, and my brothers Ochieng and Joshua for the overwhelming love and support that they continue to give tirelessly. Without them this dream that I am fortunate to be a part of would have ended before it began. Thank you for inspiring me daily with everything that you do and allowing me to pursue this opportunity and equipping me with the tools to do it.

And finally to my friends and the many communities who have made this rainy little town a place I call home. From the ever-engaging and expanding Rhodes family, to the always warm faces at Linacre college, the camaraderie of the African society and all my friends from Nairobi and Cape Town who I always carry with me. You have my heartfelt thanks.

To God without whom all this would be impossible.

Abstract

Geometry plays an important role in our understanding of the world with its uses spanning multiple fields from astronomy and art to more contemporary fields such as computer vision and robotics. However, as geometry is seldom directly observed the fitting of geometric models becomes a crucial task in these fields. This is however not trivial: as data must be clustered according to the parameters of geometric models that must also be estimated, creating a chicken and egg problem. In addition, most data in practice originates from multiple models, whose numbers are also unknown, and contain noise and clutter which makes geometric multi-model fitting more challenging.

In this thesis we present a framework capable of swiftly and accurately fitting multiple geometric models to data contaminated with noise and clutter. Unlike previous greedy approaches this framework efficiently searches for a soft assignment of points to models by minimising a global energy that considers the joint classification of data points to geometric models. Additionally, the energy minimisation is performed through a **CO**nvex **R**elaxation **AL**gorithm (CORAL) that unlike other combinatorial approaches can be efficiently parallelised which leads to an energy minimisation that is significantly faster, as the resolution of data increases, while still performing as well or better than the state-of-the-art when evaluated.

While geometry and subsequently this CORAL formulation are not restricted to a single field, in this thesis we are particularly interested in how they can be used to improve various systems in contemporary robotics. We investigate this firstly through mapping wherein geometric models are incorporated to improve, in real-time, dense depth map estimation from cameras in regions where there is little texture. We see that when these improved depth maps are fused there is significantly more coverage (10%) as compared to the state-of-the-art over large-scales. Following this we demonstrate how in tandem with deep learning networks understanding of the road for autonomous vehicles can be improved. In this way, the human-effort needed to obtain training data for deep networks can be reduced while still retrieving qualitatively accurate results as seen in road markings and boundary classification. Finally we present a uniform formulation that encodes relationships between different geometric models. We show that by leveraging this formulation in sparse reconstructions of both indoor and outdoor environments with low-cost LiDAR platforms, we can obtain reconstructions that are as accurate, within the laser noise (0.03m), as professional high-fidelity surveys.

Contents

List of Figures **vi**

List of Abbreviations **xix**

1 Introduction **1**
- 1.1 Contributions . . . 3
- 1.2 Publications . . . 4
- 1.3 Thesis Structure . . . 5

2 Preliminaries **6**
- 2.1 Reference Frames . . . 7
  - 2.1.1 Special Euclidean Group . . . 7
  - 2.1.2 Convention . . . 8
- 2.2 Sensors . . . 9
  - 2.2.1 Cameras . . . 10
  - 2.2.2 LiDAR . . . 11
  - 2.2.3 Calibration . . . 13
- 2.3 Conclusions . . . 14

3 COnvex Relaxation ALgorithm (CORAL) **15**
- 3.1 Multi-Model Fitting . . . 17
  - 3.1.1 Greedy Multi-Model Fitting . . . 17
- 3.2 Energy Based Multi-Model Fitting . . . 25
  - 3.2.1 Energy Minimisation . . . 29
  - 3.2.2 Comparison of Discrete and Convex Energy Minimisation . . . 44
  - 3.2.3 Energy Minimisation On A Continuum of Labels . . . 50
  - 3.2.4 Implementation . . . 52
- 3.3 Conclusions . . . 56

4 CORAL Evaluation and Benchmarking **59**
- 4.1 Two-view Multi-Homography Estimation . . . 60
  - 4.1.1 Simulation Environment . . . 62
  - 4.1.2 AdelaideRmf . . . 67

4.2 Plane detection with RGBD images 68
4.3 Conclusions 71

5 Fast Global Labelling For Depth Map Improvement 72
5.1 Depth Map Estimation 74
5.2 Planar Priors For Depth Map Improvement 79
5.2.1 Planar Prior Generation 83
5.2.2 Planar Prior Labelling 88
5.3 Implementation 90
5.4 Results 92
5.5 Conclusions 96

6 Road Understanding For Autonomous Vehicles 97
6.1 Road Marking Classification 103
6.1.1 Road Marking Segmentation 103
6.1.2 Road Marking Geometric Primitive Fitting 104
6.1.3 Road Marking Geometric Primitive Clustering 107
6.1.4 Road marking tracking 109
6.2 Road Boundary Classification 112
6.2.1 Road Boundary Segmentation 112
6.2.2 Road Boundary Fitting 114
6.3 Implementation 117
6.4 Conclusion 118

7 A Unified Representation for Scene Reconstruction 119
7.1 Graph Based Optimisation For Architectural Constraints 122
7.2 Symmetries & Perturbations Model 124
7.2.1 Optimisation 127
7.3 Implementation 130
7.3.1 Constraint Generation 130
7.3.2 Data Association 132
7.4 Results 133
7.5 Conclusion 135

8 Conclusions 137
8.1 Future Work 140
8.2 Closing Remarks 142

9 CORAL Matlab Implementation 143

References 153

List of Figures

1.1 Illustration of some of the different applications of geometry through the years. From early, albeit flawed, models of the universe presented by Aristotle [1] to the more accurate laws offered by Kepler [2] (top row) geometry was important in the development of our understanding of the universe around us. It also found use in classical art and architecture with the golden ratio seen in the design of many buildings such as the Parthenon (middle right) as well as in famous paintings such as Leonardo Da Vinci's Vitruvius Man (middle row) and Katsushika Hokusai's The Great Wave (middle row) [3]. In recent times, it still underpins many technological fields such as computer vision and can be used to create sparse reconstructions of objects such as the Roman colosseum (bottom row) from crowd-sourced pictures [4]. 2

2.1 Example showing the $\mathbb{SE}(n)$ Transformation. A rigid-body object with the pose A(red triangle) is first rotated then translated through $\mathbf{T}_{B-A}$ to form object B(blue triangle). This is shown in $\mathbb{SE}(2)$ for illustrative purposes but is extendable to the $\mathbb{SE}(3)$ transforms used in this work. 8

2.2 Illustrative example showing the composition of poses. Given a dynamic rigid-body object with its position recorded at different times its current location, at time d, can be described relative to its original, at time a, through the transformation $T_{a-d} = T_{a-b} \oplus T_{b-c} \oplus T_{c-d}$. 9

2.3 Figure of the Pinhole Camera Model. This illustrates a simple model to convert 3D points $\mathbf{p}_{3D}$ to 2D points $\mathbf{p}_{2D}$ on the image plane. The camera lens focuses all light rays to the camera centre, $\mathbf{C}$, while the image's individual pixels (image plane) may be thought of as f metres (the focal length) in front of $\mathbf{C}$. The principal point, p, is located at the centre of the image plane. 10

2.4 Figure showing a robotic platform equipped with 2D LiDAR sensor in a push-broom orientation (left). Each scan shown in yellow represents a vertical snapshot of the environment which can be leveraged to create a sparse 3D reconstruction of the environment if the motion profile of the platform can be obtained (right). Additionally, if the platform contains cameras that are externally calibrated to the LiDAR sensor, the laser points can be coloured using images from the camera as shown in the right. 12

2.5 If the external calibration between a camera and LIDAR sensor is correct, points projected from the LIDAR sensor accurately overlay the 3D structure seen in images (left) taken from the Oxford RobotCar dataset [10]. A small error in this external calibration , 5° in RPY, leads the points out of alignment as can be seen in the brown wall in the right image. Inaccurate calibration leads to corruption if information is shared between the two sensors. 13

3.1 In this Figure a robust statistical method for line fitting is illustrated. In this method all the data points shown in blue (left column) are used to compute the parameters of a model, a line shown in red in this example (middle column), after which outliers to this model are removed. This then iterates between a parameter re-estimation using the previous inliers followed by outlier removal until no more outliers can be removed (right column). In the top right this results in a model consistent with the ground truth that is found in the presence of outliers but can be seen to break down in the bottom right as a "phantom" model is fitted. 19

3.2 Figure illustrating the application of sequential RANSAC for multi-model fitting. In this approach the RANSAC algorithm is first run to obtain a single model after which the inliers to this model are obtained and removed, after which RANSAC is run again. In the top row this results in the fitting of models that are consistent with the ground truth even in the presence of clutter however in many cases as illustrated in the bottom row errors in the initial model selection leads to a cascading of wrong, but geometrically valid, models. Data points in the right column are colour-coded according to their membership to the set of geometric models. 21

3.3 Figure illustrating the application of multi-RANSAC for multi-model fitting. In this approach a parallel rather than sequential approach is applied to the multi-model fitting which limits the errors introduced by a wrong initial model yielding better performance on the example in Figure 3.2. However, for this result the number of line models, in this case four, had to be provided to the algorithm beforehand limiting its use to situations where the number of geometric models is explicitly known. Data points in the right column are colour-coded according to their membership to the set of geometric models. . . . 22

3.4 Figure showing the results of different multi-model fitting algorithms on data that contains lines buried in noise. The mode-finding approaches such as mean-shift and RHA are unable to obtain the correct line models that correspond to the ground truth. Similar outcomes were observed with the RANSAC based approaches showing that clustering based approaches such as J-linkage offer better results in the presence of noise. Figure from [38]. 23

3.5 Figure showing a geometric line fitting example to illustrate the energy functions used in this work. From noisy data (top left) line models can be fitted and are displayed colour-coded. The first energy is associated to the distance from a point to the line or its geometric cost as shown in the top right image. The second energy regularises for smoothness over a defined neighbourhood and penalises points in the same neighbourhood that have different labels as seen in the bottom left. The third energy regularises for compactness and penalises redundant models (green line) as seen in the bottom right image. 27

3.6 Figure of a convex function. For a convex function the line segment between any two points lies above the graph as can be seen by the dotted line plotted and can then be minimised by gradient descent techniques. 30

3.7 Figure illustrating the choice between binary assignment for the indicator function $\{0, 1\}$ or its relaxation to a soft assignment $\phi(\mathbf{u}) \in [0, 1]$ proposed in this thesis. Observation of the smoothness energy from the binary assignment (left) reveals a discrete solution set shown by the blue points. As a result line segments between values in the solution set(the dotted) line contain values that are outside the solution (red diamond) and according to Equation 3.10 is thus non-convex. In contrast, observation of the same energy from the relaxed assignment (right) shows that a line segment (dotted line) between two possible solutions contains values that are still within the solution set making this function convex. Though this relaxation makes the smoothness energy convex, this energy happens to be non-smooth. However, techniques from continuous optimisation allow us to transform this to an alternate form over which standard gradient based techniques can be used for its minimisation. 31

3.8 Figure showing a geometric line fitting example to illustrate the ambiguities arising from a strict binary or *hard* labelling. Given data points from three lines that intersect with each other(left), a binary labelling can reveal multiple solutions as intersecting points may be correctly assigned to different models as is seen in the middle and right images. 31

3.9 Figure illustrating a graph cut example used in combinatorial optimisation techniques. In (a) the data points to be labelled and the binary choice nodes s and t are shown before edges that represent costs from an energy functional are added in (b) and (c). The fully constructed graph is shown in (d) with the graph cut (e) that provides the final labelling shown in (f). 34

3.10 Figure showing how the Legendre transformation encapsulates the information of a function $\mathbf{E}(\mathbf{r})$ (left) through its derivatives $\mathbf{q}$ into a dual $\mathbf{E}^*(\mathbf{q})$ (right). Given a tangent line(blue) at one point to the function as shown, through geometry its slope (q) multiplied by its adjacent side (r^*) is equal to its opposite side $qr^* = \mathbf{E}(r^*) + \mathbf{E}^*(q)$ which can be extended to give the transform $\mathbf{E}^*(\mathbf{q}) =< \mathbf{q}, \mathbf{r} > -\mathbf{E}(\mathbf{r})$ and the corresponding dual function (right). 36

3.11 Figure illustrating how the LF transform can transform a convex but non-smooth *primal* variable $|\nabla\phi_l(\mathbf{u})|$(left) to its convex, smooth *dual* $\Psi_l(\mathbf{u})$ (right). It can be seen that for $(\nabla\phi_l(\mathbf{u}) < 0)$ and $(\nabla\phi_l(\mathbf{u}) > 0)$ the primal is smooth and can be differentiated revealing slopes of $\Psi_l(\mathbf{u}) = -1$ and $\Psi_l(\mathbf{u}) = 1$ respectively. At $\nabla\phi_l(\mathbf{u}) = 0$ the primal cannot be differentiated and its gradient must be estimated, this is through a range of acceptable slopes $\Psi_l(\mathbf{u}) \in [-1, 1]$. Transforming the non-smooth *primal* to a smooth *dual* formulation allows for the use of standard gradient based optimisation techniques on this problem. 39

3.12 Figure illustrating the effect of the label order on the results of the geometric multi-model fitting by PEARL and CORAL. As α-expansion splits the energy minimisation into a series of binary problems it is sensitive to the label order as can be seen in the left column where points at the intersection of models have their label changed according to the label order, CORAL by performing the optimisation jointly is robust to this as seen in the right column. . . 45

3.13 Energy evolution of the combinatorial and convex optimisation algorithms for geometrical multi-model fitting in a simulation experiment with different initialisations that are then averaged. It can be seen that both of the approaches successfully minimise the energy given, with the combinatorial approach converging faster (left). However, when the final energy is examined closer, it can be seen that the convex approach returns a lower energy. This is as α-expansion is not robust to changes in initialisations which results in a higher energy and consequently slightly worse labelling performance than CORAL. 46

3.14 Comparison of the run-time per iteration of α-expansion and that of the primal-dual optimisation (PD) in an experiment with four models. The times are averaged over 100 runs on a machine with a 12 core Intel i7-5930K 3.50GHz CPU and a Nvidia GeForce GTX Titan Black 6048MB GPU. On this architecture, as the number of features increases so does the difference between the run times per iteration of the two methods. With the PD running at speeds two orders of magnitude (100x) faster than the α-expansion. 47

3.15 Convergence analysis of the primal dual algorithm in Algorithm 3. The optimisation is considered converged when the primal(blue)-dual(green) gap goes to zero. 48

3.16 Intermediate energy minimisation result showcasing the metrication errors when the Manhattan norm is used in discrete algorithms such as α-expansion as compared to CORAL in a plane-fitting experiment from dense RGBD data. In discrete algorithms the Manhattan norm preserves edges aligned with the axes leading to a stair-case effect that is observed when dealing with dense data, whereas CORAL due to its continuous formulation is still able to return a smooth boundary. 49
3.17 Illustrative example of energy minimisation for geometric multi-model fitting. From noisy data and a set of model proposals from random sampling, energy minimisations algorithms are able to converge on the correct underlying models through an iterative labelling and re-estimation process even in the presence of noise. 51
3.18 Figure illustrating the construction of the $\nabla_{\mathcal{N}}$ for i.i.d data. In this N_n nearest neighbours, the indexes of a point I_{p_i} and its neighbours are then added to $\nabla_{\mathcal{N}}$ as shown in Equation 3.47. 53
3.19 Flow-chart illustrating the use of CORAL for geometric multi-model fitting. From an initial set of N_p data points $\mathbf{u}$ supporting unknown numbers of geometric models, L models are proposed using RANSAC or minimal sample sets. From these model parameters, an initial data cost term can be retrieved. Similarly a gradient $\nabla_{\mathcal{N}}$ can be established through N_n nearest neighbours. This is sufficient to compute the energy in Equation 3.8 which is then minimised through Algorithm 4 to obtain an intermediate result after which the models are updated. Algorithm 5 suggests an iterative scheme of energy minimisation and model parameter estimation which continues until energy convergence when $(\mathbf{E}_{PD} - \mathbf{E}_{Re-est}) < T_{Re-est}$ giving the final result at the bottom right. 57

4.1 Figure showing a sample of the geometric models that can be obtained through the CORAL formulation. 61
4.2 The simulated environment created to compare the performance of multi-model fitting algorithms for multi-homography fitting. This environment consisting of three mutually orthogonal planes mirrors indoor and outdoor scenes of the end of a corridor or corner of a building respectively. Observation of the scene by a camera from different views reveals the pixel correspondences needed for homography fitting. 63
4.3 Misclassification error against sensor noise σ_{pixel} firstly in the absence of outliers (left) and then with a steady increase in outlier percentage for a fixed value of σ_{pixel}. The errors are obtained for RANSAC, PEARL, CORAL and T-Linkage. The ME is evaluated and averaged over ten different two-view pairs of the simulated environment. . . 64

4.4 The triangulated positions of noisy pixels together with the ground truth planes (red, green and blue patches) are shown under different values of σ_{pixel}. With the membership to the different homographies indicated by the different colours. RANSAC, PEARL and CORAL results are shown in the first, second and third row respectively. . . 65
4.5 Figure illustrating the effect of the regularisation parameters λ and β on the ME averaged over ten different experiments in the simulated environment. Low values of these parameters show no regularisation while as these values get too large they over-power the geometric errors resulting in a higher ME as an over-smooth and compact solution will be found. Optimal values lie in the valley between these two, with the size of it showing that there is a range of these parameters that still give good performance. 66
4.6 Time taken for the energy minimisation algorithms as the number of features increased in the multi-homography fitting simulation experiment. For the CORAL implementation a Nvidia GeForce GT 750M 2048MB GPU was used and for PEARL a 2.5 GHz Intel Core i7. 66
4.7 Sample of images from the AdelaideRmf Dataset. With the membership to the different homographies indicated by the different colours. Crosses are used to signify points that are mislabelled while the uniform outlier label $\emptyset$ is shown in black through all figures. 68
4.8 Plane segmentation on a sample of RGBD images from the NYU-Depth dataset. with the Ground Truth, RANSAC, PEARL and CORAL results presented for each image. Pixel membership to a plane model is shown by superposing on a pixel a colour-coded point that assigns it to a specific plane model. 69

5.1 Example of a depth image (left) and its corresponding 3D reconstruction (right) from back-projecting the individual pixels using their depth measurements. The depth of the individual pixels is found by matching correspondences in multiple images followed by a subsequent optimisation. If a stereo-pair of images is used the depth obtained is also metrically valid. 74
5.2 A qualitative perspective of this chapter. A dense depth map projection (top) of a planar wall surface created by a state-of-the-art Total Generalised Variation algorithm is displayed. As expected, the algorithm results in a noisy output on the largely textureless wall and road. CORAL discovers planar regions and from there invokes a planar prior to restricted areas. This results in an improved depth map projection (bottom), showing a marked improvement in the depth estimation along the wall and road. 75

5.3 An illustrative example showing the effects of the Total Variational (TV) and Total Generalised Variational (TGV) regularisation on reconstructing a noisy signal (left). TV (middle) is able to preserve the high variation changes and reconstruct piecewise constant surfaces however in regions of constant gradient, a staircase effect is observed. The TGV term (right) also deals with the high variation changes while dealing with affine surfaces, unlike in TV piecewise constant segments are approximated by linear segments. 76

5.4 Comparison of three depth map regularisers: TV, TGV, and TGV-Tensor. Using the reference image (top), the TV regulariser (left) favours fronto-parallel surfaces, therefore it creates a sharp discontinuity for the shadow on the road (red rectangle) and attaches the rubbish bin to the rear of the car (red circle). TGV (centre) improves upon this by allowing planes at any orientation, but it still cannot identify boundaries between objects: the rubbish bin is again estimated as part of the car. Finally, the TGV-Tensor (right) regulariser both allows planes at any orientation and is more successful at differentiating objects by taking into account the normal of the colour image's gradient. Figure from [73]. 78

5.5 An example scene showcasing depth map estimation of an urban scene. From the reference image (top) it can be seen that the variational technique struggles with the large, plain and planar surfaces with the walls (bottom left, bottom right) being estimated noisily while an erroneous estimate is made for the similarly plain road (bottom middle). 79

5.6 Overview of the proposed vision only depth map improvement pipeline. From image sequences (mono or stereo) a two-view homography segmentation in conjunction with visual odometry is used to create planar priors. These priors drive a fast global labelling that assigns planar regions to their corresponding prior and non-planar regions to a depth map obtained through the Total Generalised Variational (TGV) energy minimisation. From this labelling a more complete improved depth map is estimated. The fusion of consecutive improved depth maps are further fused to create a dense reconstruction. . . . 82

5.7 Figure showing multiple homography initialisation using affine transformations and affinity propagation clustering. With membership to a specific homography coded by colour it can be seen that this method results in an over-segmentation with a prevalence of redundant models. 85

5.8 Sample of the results from multi-homography fitting using CORAL with membership to different homographies encoded by colour-coded. CORAL can be seen to capture the large, plain and planar surfaces through the homography models in a variety of scenes and plane orientations. The planar priors can then be obtained through homography decomposition. 87
5.9 Semantic Segmentation of the pixels of an image (left) into its respective classes (right). Buildings and roads over which planar surfaces are expected to be found are labelled in red and pink respectively. 89
5.10 Sample results showing planar prior pixel labelling through an energy minimisation approach. This leverages planes obtained through CORAL and assigned pixels to them through a planar prior labelling. From planar priors obtained using CORAL pixels are assigned to these priors using a global energy function. The depth of these pixels can then be estimated solely using the plane hypotheses. 91
5.11 An illustration of the coarse–to-fine approach used to improve the speed of convergence in the multi-labelling problem. Groups of pixels are merged to create a coarser image in subsequent levels over which the per-pixel labelling is performed. 92
5.12 By projecting Velodyne laser data onto an image, CORAL can be used to create ground-truth plane annotations(top left), these can then be compared to the plane priors obtained through the homography decomposition (top right). A comparison of the parameters (middle row) shows a good relation between the ground truth plane parameters and those obtained by homography decomposition as shown by the small angle and distance errors in the boxplots. Additionally, as the homography decomposition provides the motion of the camera, this was compared to the ground-truth odometry (bottom row). With results showing that the decomposed homographies maintain a good estimation of the motion of the platform as seen in the boxplots. . . 94
5.13 Sample results of dense reconstructions from the fusion of depth maps in the two evaluated sections are shown in this figure. The left column of images corresponds to the reconstructions produced using the TGV depth maps. It can be seen that the noise on the depth maps results in some gaps in the reconstruction. By improving these depth maps with planar priors in our approach the results can be improved on, significantly, as shown in the right column. Resulting in a more accurate map of the world. 95

6.1 Example of the three layers deep fully convolutional U-net architecture [110]. In this architecture neurons in the first layer are only connected to pixels in their receptive fields, giving the networks the ability to concentrate on lower level features in the first layer, then aggregate them into higher level features in the next hidden layers. This makes them particularly suited for visual tasks. Additionally there is a pooling stage, that subsamples images to reduce the computational load and the number of parameters [93]. 101

6.2 Figure showing a LiDAR point cloud of a road surface and the corresponding camera image of the same scene. It can be seen that the high-reflectance pixels correspond to road markings which can be related to the image pixels using a Conditional Random Field. . 102

6.3 Semantic classification of road markings in an urban environment. From an image taken by the front-facing camera of an autonomous vehicle (top left), pixels of potential road markings can be identified by a trained deep neural network (top right) under various lighting conditions. A two-step CORAL optimisation retrieves the road marking classes. The first optimisation step reveals geometric primitives (lines) which encode road marking segments (bottom left), before a further optimisation step classifies these primitives into semantically meaningful road markings as seen in the bottom right image. 103

6.4 Road marking detection under different lighting conditions (overcast, night, rainy, and sunny). Despite the large changes in the prevailing conditions the trained deep segmentation network [112] is able to accurately identify the pixels belonging to the road markings from the monocular image. 105

6.5 A comparison of the results from road marking geometric primitive extraction under different prevailing conditions using the Hough Transform and CORAL approaches. It can be seen that our proposed energy optimisation approach is able to accurately extract linear segments in a diversity of scenes while the Hough Transform suffers in the presence of noise and in more complex scenes result in an over-parametrisation of the scene. CORAL is thus able to accurately reveal the underlying primitives for a large number of road markings present in the urban scene. 106

6.6 Figure showing manually annotated road markings after inverse perspective mapping. These show a junction (left), lane markings (second left), zig-zag lines (second right) and a road intersection (right) scene. In this birds eye view of the scene the effects of the camera perspective have been removed, road markings thus observed retain their configurations even as the autonomous vehicles traverse urban scenes. 108

6.7 Sample of results from the semantic classification of road marking pixels under different weather and lighting conditions (overcast/rain/night). Our proposed CORAL approach is able to detect multiple road marking segments in images from detected road marking pixels. These segments are aggregated through another energy minimisation into their semantic classes. Thereby we reveal the underlying meaning of the road markings in complex urban environments providing important cues for autonomous vehicles. These include indicating for upcoming road situations, which could require specific behaviour. For instance, the zig-zag markers (*purple*) indicate an upcoming pedestrian crossing and the give-way dashes (*cyan*) indicate a junction. 110

6.8 Given a fully observed road marking (left) our proposed approach is able to correctly detect the underlying semantic class of the road markings on a image. However when this becomes partially observed (middle) the interaction of the underlying geometric primitives with others in the scene can cause mis-classification, in this case the road junction was labelled as a zig-zag line. By introducing road marking tracking (right) even under partial observation the correct underlying can be arrived at, making the classifications more robust. 111

6.9 Given an input image (left), an adequately trained deep segmentation network is able to correctly identify the *visible* road marking pixels. However it produces a blurry output through occlusions (red rectangle) as it is unable to infer the underlying geometry that persists through the occlusions which would bias it to correctly detect the *occluded* road boundaries. 113

6.10 Illustration of the line objects used to parametrise the *occluded* road boundaries. In a particular cell, four anchor line classes are provided each at a different angle to the centre point of the cell. *Occluded* road boundaries are to be assigned to one of these four classes after which offsets to both the angle ($\omega^k_{i,j,gt}$), and distance ($\beta^k_{i,j,gt}$) of the lines to the centre-point computed for finer localisation. 114

6.11 Examples of parameterisation of *occluded* road boundary labels. In the left column pixel-wise labels are shown followed by the division of pixel-wise masks into a grid of squares. The final *occluded* road boundary masks are drawn based on parameterised labels in the right column. 115

6.12 Sample outputs from the combined networks under different levels of road boundary occlusion. These are able to seamlessly deal with and predict the position of the road boundaries in scenarios of low to no occlusion (left), some occlusion (middle) and full occlusion (right). The blue pixels are the visible road boundaries while the yellow pixels represent the occluded boundaries. 115

6.13 Comparison of results of geometric road boundary fitting from the output of the combined networks using Sequential RANSAC and CORAL. While CORAL is able to obtain the minimum set of cubic curves that represent the road boundaries, without any prior information of the number of models, in a diversity of viewed scenes the performance of RANSAC deteriorates in the presence of noise and in more complex scenes returning inaccurate road boundary models. 116

7.1 A large scale reconstruction obtained as solution of the graph optimisation, proposed in this chapter, over a 2km trajectory of an urban environment (top). By combining two different sensor modalities, a push broom laser and a stereo camera with a known cross-calibration, we can create low-cost reconstructions of the environment, these are however subject to "drift" that impacts their accuracy. In this chapter we offer a unified framework to handle multiple architectural constraints driven by the different geometric features discovered in the environment (e.g. points, planes, etc), these constraints can then be employed to improve the motion estimation resulting in accurate reconstructions over large-scales (bottom). 121

7.2 Description of the proposed geometric entities and constraints: a pose T^x, plane T^π, point T^p and line T^l with their respective representation within the proposed Lie algebra framework. The attached reference frame as well as the graphical representation of the entities are described together with the construction of $\exp(\boldsymbol{\Delta}^e)$, the exponential mapping of the differential to preserve the symmetry of each of the individual entities. 126

7.3 Sample of results for plane-fitting through CORAL on small sections of 3D LiDAR point-clouds projected onto camera images. It can be seen that CORAL can in a variety of cases, including indoor and outdoors, fit planes accurately to the LiDAR points which in this image are colour-coded according to their assigned models. 131
7.4 Figure showing the results of 4 surveys taken and optimised with this framework. The original surveys created using VO are shown in column 1, while a comparison of the surveys after optimisation with architectural constraints together with their subsequent histograms is shown in columns 2 and 3 respectively. Rows 1 and 2 present results from 2 independent indoor surveys of the Acland building in Oxford where the ground truth dense map is available courtesy of a professional survey and used for the comparisons. In row 3 two different independent outdoor maps are presented (grey and blue) of the Jericho triangle in Oxford. Since ground truth is not available the comparison is done between the 2 point clouds optimised independently. 134
7.5 This figure shows the configuration of planes from the survey of the Acland building. The structure of this environment can be described by the automatically generated orthogonal relationships (in purple) between the three sets of planes (in green) forming the corridors. . 135

List of Abbreviations

CORAL	Convex Relation Algorithm
TV	Total Variation
TGV	Total Generalised Variation
VO	Visual Odometry
1D	One dimensional
3D	Three dimensional
SLAM	Simultaneous Localisation and Mapping
RANSAC	Random Sample and Consensus
MSS	Minimal Sample Set
CS	Consensus Set
RHT	Randomised Hough Transform
HT	Hough Transform
RPA	Residual Preference Analysis
RCMSA	Random Cluster Model Simulated Annealing
ILP_RansaCov	Integer Linear Programming Ransac Cover
UFL	Uncapacitated Facility Location
MRF	Markov Random Field
LF	Legendre-Fenchel
PD	Primal Dual
DLT	Direct Linear Transform
ME	Misclassification Error
PPR	Piecewise Planar Reconstructions
SURF	Speeded Up Robust Features
SVD	Singular Value Decomposition
TSDF	Truncated Signed Distance Function

EKF . . . . . Extended Kalman Filter

CNN . . . . . Convolutional Neural Network

PEARL . . . . Propose Expand and Re-Estimate Labels

GPGPU . . . . General Purpose Graphical Processing Unit

GPU . . . . . Graphical Processing Unit

CPU . . . . . Central Processing Unit

CUDA . . . . . Compute Unified Device Architecture

SP . . . . . Symmetry and Perturbation

Just because something works;
Does not mean it can't be improved.

Shuri-Wakanda

1 Introduction

Contents

1.1 Contributions **3**
1.2 Publications **4**
1.3 Thesis Structure **5**

GEOMETRY is concerned with the shape, size and position of figures and their properties in space. It has been of great importance throughout recorded history finding applications from early navigation and astronomy through to classical physics, art and architecture and currently to more contemporary technology backed fields such as computer vision, robotics and artificial intelligence as illustrated in Figure 1.1. It thus follows that obtaining geometric models, as many of them cannot be directly observed, is a fundamental problem spanning multiple fields.

However, geometric model fitting is a chicken-and-egg problem: observed data needs to be assigned to models whose parameters are also unknown and to be estimated. Moreover, the number of models present in data is *a priori* unknown and the observed data is also often contaminated with noise and clutter further complicating the model fitting problem.

In this thesis we opt to formulate geometric multi-model fitting as a global energy based multi-labelling problem. This we show to be better suited for multi-model

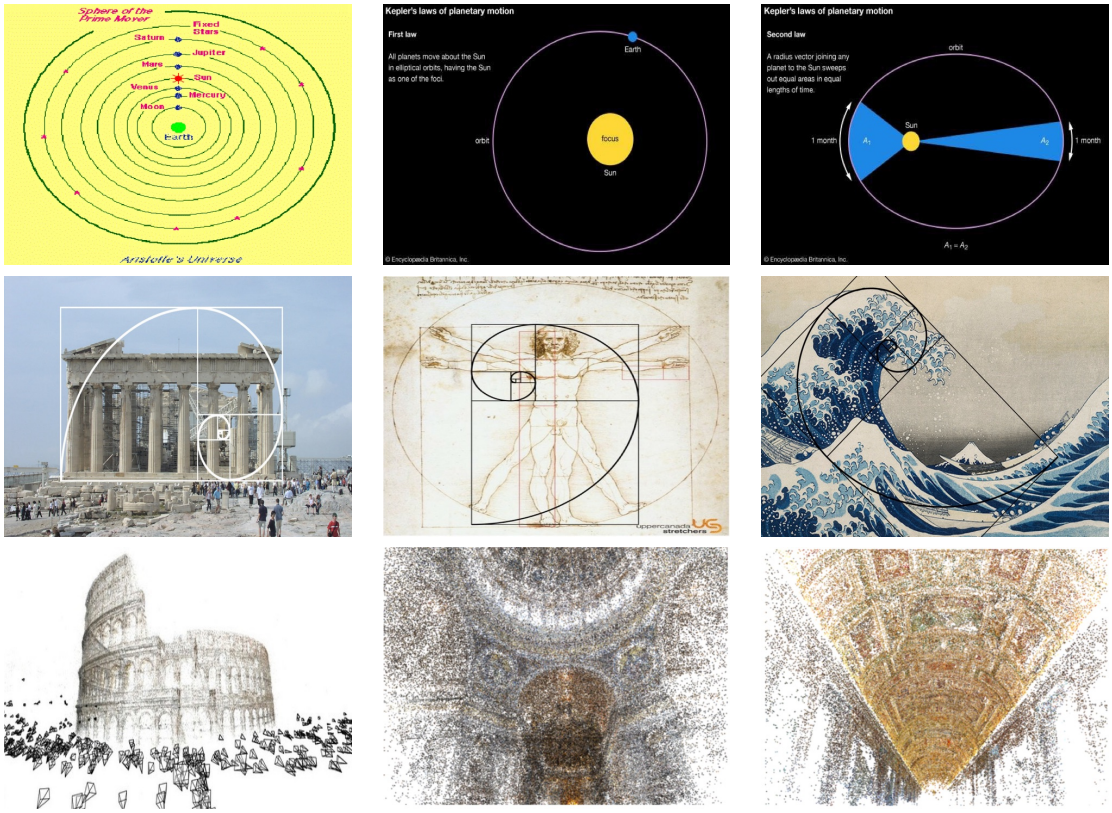


Figure 1.1: Illustration of some of the different applications of geometry through the years. From early, albeit flawed, models of the universe presented by Aristotle [1] to the more accurate laws offered by Kepler [2] (top row) geometry was important in the development of our understanding of the universe around us. It also found use in classical art and architecture with the golden ratio seen in the design of many buildings such as the Parthenon (middle right) as well as in famous paintings such as Leonardo Da Vinci's Vitruvius Man (middle row) and Katsushika Hokusai's The Great Wave (middle row) [3]. In recent times, it still underpins many technological fields such as computer vision and can be used to create sparse reconstructions of objects such as the Roman colosseum (bottom row) from crowd-sourced pictures [4].

fitting when compared to greedy heuristic approaches that dominate the literature yet suffer as noise and clutter increase. To enable this we introduce global objective functions that include not only the geometric errors, of the data to the proposed models, but also regularisation over spatial smoothness and model compactness. We then present for the first time a continuous optimisation approach for this global energy optimisation through a **CO**nvex **R**elaxation **AL**gorithm (CORAL) that creates a fast, robust and general geometric multi-model fitting algorithm.

While applications for geometric models exist across multiple fields we are particularly motivated by the case of mobile robotic platforms where the trade-off

between fast and accurate multi-model detection is especially crucial for safety. In this thesis we thus restrict the experiments and evaluations of multi-model fitting approaches to these platforms and tease out multiple applications relevant to contemporary robotics such as mapping, perception and scene understanding.

1.1 Contributions

This thesis makes the following contributions.

- A fast, parallel framework for the fitting of multiple geometric models with a **CO**nvex **R**elaxation **AL**gorithm **CORAL**. This uses a global energy approach with the energy related to the quality of the solution, trading off a data fidelity term which considers the geometric cost of a feature to a model with a regularisation term that encourages spatial smoothness between features in proximity with each other and a compactness term that eliminates model redundancy.

- A sliding-window real-time pipeline for improving the depth estimation of monocular or stereo images that capture large, plain and planar surfaces. Using the CORAL framework, we detect these underlying planes and through a subsequent energy approach obtain new depth map estimates.

- A hybrid approach to perception that combines state-of-the-art deep learning techniques with classical geometric multi-model fitting through CORAL. This is in a two-step approach that utilises the deep networks for the detection of salient road marking and boundary pixels from an image before CORAL extracts the underlying geometric model.

- A unified framework for capturing the variety of geometric models and their relationships that can be discovered in urban sparse reconstructions of laser surveys through a graph based framework. Leveraging the idea that most of the workspace can be described by the differential configuration of geometric

models, optimisation of which leads to more accurate reconstruction of the scene.

The theory, algorithms and software presented here are also used extensively within the Oxford Robotics Institute, from pipe detection to multiple motion estimation systems [5].

1.2 Publications

The major sections of this work have been published:

- **P. Amayo**, P. Pinies, L. M Paz, P. Newman. "Geometric Multi-Model Fitting Through a Convex Relaxation Algorithm" in Computer Vision and Pattern Recognition (CVPR), Salt Lake City, USA June 2018. (Chapters 3 and 4)

- **P. Amayo**, P. Pinies, L. M Paz, P. Newman. "Fast Global Labelling For depth Map improvement Via Architectural Priors" in International Conference on Robotics and Automation (ICRA), Brisbane, Australia, May 2018. (Chapter 5)

- **P. Amayo**, T. Bruls, P. Newman. "Semantic Classification of Road Markings From Geometric Primitives" in International Transport and Safety Conference (ITSC), Maui, USA, November 2018. (Chapter 6)

- T. Suleymanov, **P. Amayo**, P. Newman. "Inferring Road Boundaries Through and Despite Traffic" in International Transport and Safety Conference (ITSC), Maui, USA, November 2018. (Chapter 6)

- **P. Amayo**, P. Pinies, L. M Paz, P. Newman. "A Unified Representation for Application of Architectural Constraints in Large-Scale Mapping" in International Conference on Robotics and Automation (ICRA), Stockholm Sweden, May 2016. (Chapter 7)

1.3 Thesis Structure

In Chapter 2, the sensors and preliminary tools to supplement this thesis are presented. Chapter 3 presents our proposed formulation, situating it in the history of geometric multi model fitting methods before differentiating it from other energy based optimisation algorithms. We then proceed to benchmark its performance in various applications in Chapter 4 where the marked improvement in speed and accuracy given by CORAL is showcased.

In Chapter 5 we show how this method can be put in place to aid the extraction of depth maps of urban environments, improving on the outputs on state of the art Total Variation (TV) and Total Generalised Variation (TGV) methods. Following this we turn our attention to the perception problem in Chapter 6, focusing on the live detection of road markings and road boundaries from primitive geometric models. In Chapter 7, we show how configurations of geometric models can be used to enhance scenes traversed by robots, revealing high-accuracy maps of the environment.

A complex system that works is invariably found to have evolved from a simple system that works.

John Gaule

2 Preliminaries

Contents

2.1 Reference Frames 7
2.1.1 Special Euclidean Group 7
2.1.2 Convention 8
2.2 Sensors 9
2.2.1 Cameras 10
2.2.2 LiDAR 11
2.2.3 Calibration 13
2.3 Conclusions 14

While this thesis presents a general formulation for the fitting of multiple geometric models that is agnostic to any particular application, the evaluations presented in the subsequent chapters are concerned with robotic platforms. Using these we explore the improvements that could be offered by geometric models while they traverse their respective environments. In this chapter we present a few of the preliminary tools that underpin mobile robotic platforms. We begin with an explanation of reference frames and the coordinate transformations between them, which are used to define the position and orientation of a robotic platform and its associated sensors in time. An explanation of the sensors used in this thesis is then offered together with the importance of calibration between the different sensors when using data from different sensors.

2.1 Reference Frames

In this thesis we evaluate our geometric multi-model fitting algorithm using mobile robotic platforms together with their accompanying sensors as they explore their environments. Reference frames provide us with the ability to represent the position of the robotic platform or sensor at any point of time. In this work reference frames are mathematically defined through the Special Euclidean Group $\mathbb{SE}(n)$ described in the following section.

2.1.1 Special Euclidean Group

The $\mathbb{SE}$(n) special Euclidean group represents the set of all possible rigid-body rotations and translations available in **n**-dimensional space (typically n = 2, 3) [6]. $\mathbb{SE}$(n) is composed of a $n \times n$ rotation matrix $\mathbf{R}$ and a **n**-dimension translation vector.

For example, a point $\mathbf{p}$ is transformed as below:

$$\hat{\mathbf{p}} = \mathbf{R}\mathbf{p} + \mathbf{t} \tag{2.1}$$

where the point is first rotated about the origin's reference frame through $\mathbf{R}$ before being translated by $\mathbf{t}$. These two operations are often combined into a single matrix, $\mathbf{T}$,

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \tag{2.2}$$

where $\mathbf{T}$ may be applied to a $\mathbf{n}+1$-dimensional homogeneous vector,

$$\hat{\mathbf{p}} = \mathbf{T} \begin{bmatrix} \mathbf{p} \\ 1 \end{bmatrix} \tag{2.3}$$

As $\mathbf{R}$ is orthonormal, the inverse of this transformation can also be calculated

$$\mathbf{T}^{-1} = \begin{bmatrix} \mathbf{R^T} & -\mathbf{R^T}\mathbf{t} \\ 0 & 1 \end{bmatrix} \tag{2.4}$$

In $\mathbb{SE}(2)$, illustrated in Figure 2.1, $\mathbf{T}$ can be decomposed to:

$$\mathbf{T} = \begin{bmatrix} cos(\theta) & -sin(\theta) & t_x \\ sin(\theta) & cos(\theta) & t_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.5}$$

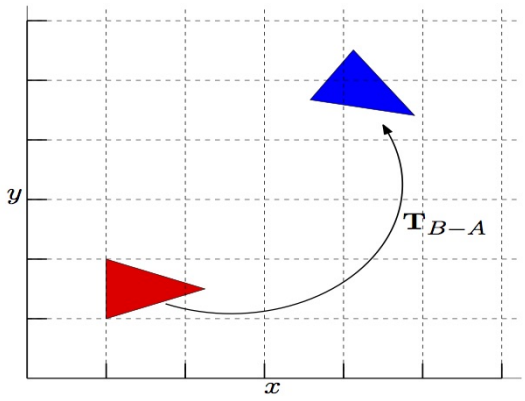


Figure 2.1: Example showing the $\mathbb{SE}(n)$ Transformation. A rigid-body object with the pose A(red triangle) is first rotated then translated through $\mathbf{T}_{B-A}$ to form object B(blue triangle). This is shown in $\mathbb{SE}(2)$ for illustrative purposes but is extendable to the $\mathbb{SE}(3)$ transforms used in this work.

where θ is the rotation angle about the origin and $\mathbf{t} = (t_x, t_y)^T$ is the translation vector. As can be seen from Figure 2.1 the transformed triangle is both the same shape and size as the original, this is as $\mathbb{SE}(n)$ transformations do not warp or scale objects.

2.1.2 Convention

As the robotic platform traverses the world, it observes points through its associated sensors which must be described relative to a reference frame. In this section we describe the two reference frames that we will use through this thesis. These are the **robot** and **camera** reference frames.

If a camera observes a point $\mathbf{p}_{\text{camera}} = (x_c, y_c, z_c, 1)^T$ this point is described relative to the **camera** frame. This point can be converted to the **robot** reference frame

$$\mathbf{p}_{\text{robot}} = \mathbf{T}_{\text{robot-camera}}\mathbf{p}_{\text{camera}} \tag{2.6}$$

and vice versa, if a point is observed from the **robot** frame it can be transformed to the **camera** frame

$$\mathbf{p}_{\text{camera}} = \mathbf{T}^{-1}_{robot-camera}\mathbf{p}_{\text{robot}} = \mathbf{T}_{\text{camera-robot}}\mathbf{p}_{\text{robot}} \tag{2.7}$$

Where each reference frame describes the same point but from differing origins and orientation, with the transform $T_{\text{robot-camera}}$ used to switch between the frames.

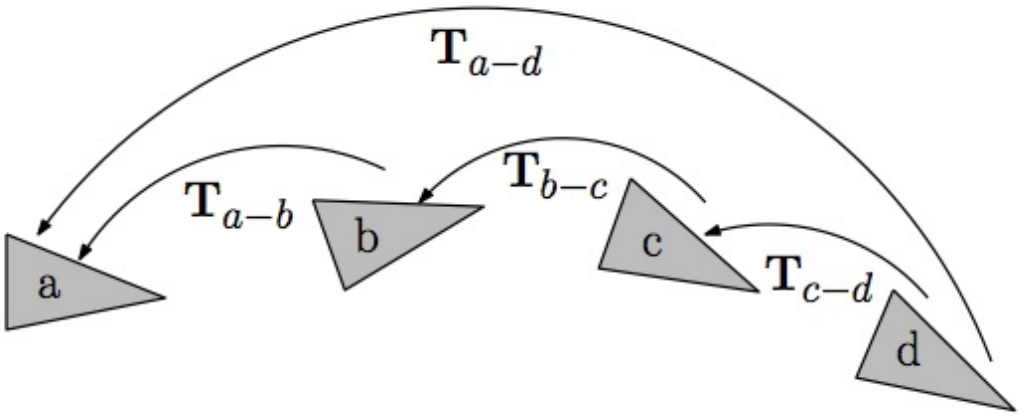


Figure 2.2: Illustrative example showing the composition of poses. Given a dynamic rigid-body object with its position recorded at different times its current location, at time d, can be described relative to its original, at time a, through the transformation $T_{a-d} = T_{a-b} \oplus T_{b-c} \oplus T_{c-d}$.

Transforms in this thesis are written in the standard form as below

$$T_{\text{DestinationFrame-SourceFrame}} \tag{2.8}$$

The sensors and robot platform are however not static themselves, to describe the transformations between different reference frames as the robot traverses the world as seen in Figure 2.2 the transform composition operator which is a matrix multiplication, $\oplus$ is introduced:

$$T_{a-d} = T_{a-b} \oplus T_{b-c} \oplus T_{c-d} \tag{2.9}$$

The inverse of this transform is,

$$T_{d-a} = (T_{a-b} \oplus T_{b-c} \oplus T_{c-d})^{-1} = (T_{a-d})^{-1} \tag{2.10}$$

With these we can describe the position of sensors and their observed points as the robot traverses the world from any desired reference frame.

2.2 Sensors

Through their associated sensors, robotic platforms are able to observe their surroundings. In this section we present the two main sensor modalities that

are mounted on mobile robotic platforms. Section 2.2.1 produces an overview of camera sensors, this is followed by Section 2.2.2 which presents the LiDAR sensors. Section 2.2.3 then showcases the importance of accurate calibration between different sensors in platforms with multiple sensors.

2.2.1 Cameras

For the cameras used in this work, the pinhole camera model shown in Figure 2.3 provides a simple geometric description of how cameras project 3D points to the 2D image plane and vice-versa [7].

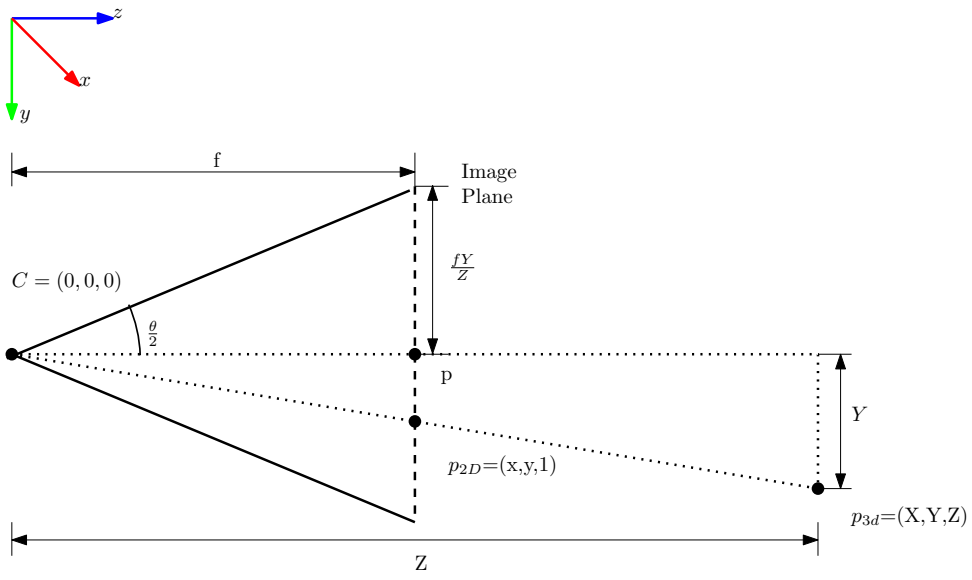


Figure 2.3: Figure of the Pinhole Camera Model. This illustrates a simple model to convert 3D points $\mathbf{p}_{3D}$ to 2D points $\mathbf{p}_{2D}$ on the image plane. The camera lens focuses all light rays to the camera centre, $\mathbf{C}$, while the image's individual pixels (image plane) may be thought of as f metres (the focal length) in front of $\mathbf{C}$. The principal point, p, is located at the centre of the image plane.

In this model the camera focal point $\mathbf{C}$ is the point at which all light is focused through the camera lens and the image plane denotes the virtual surface upon which the image rests. Light reflected off of an object thus travels to the camera centre through the image plane, with the pixel it passes through storing the

"colour" of the light photon. In this way, a 3D point $\mathbf{p}_{3D} = (X, Y, Z)$ is mapped to the image plane as below

$$(X, Y, Z)_{3D} \rightarrow (\frac{f_x X}{Z}, \frac{f_y Y}{Z})_{2D} \tag{2.11}$$

or in its matrix form

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \begin{bmatrix} f_x X \\ f_y Y \\ Z \end{bmatrix} = \begin{bmatrix} f_x & 0 & 0 \\ 0 & f_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{2.12}$$

Where different focal lengths, f_x and f_y, are used if the pixels are rectangular. We additionally account for the offset of the camera centre from the image origin through the principal point p as below.

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \begin{bmatrix} f_x X + Z p_x \\ f_y Y + Z p_y \\ Z \end{bmatrix} = \begin{bmatrix} f_x & 0 & p_x \\ 0 & f_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{2.13}$$

This transformation can be summarised into the widely-used camera calibration matrix $\mathbf{K}$.

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & p_x \\ 0 & f_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.14}$$

With this **world** 3D coordinates can be converted to their **image** 3D pixel equivalents:

$$\mathbf{p_{2d}} = \pi(\mathbf{p_{3d}}) = \mathbf{K p_{3d}} \tag{2.15}$$

and if the depth, d, of the pixels is known **image** coordinates can be converted back to **world** coordinates

$$\mathbf{p_{3d}} = \pi^{-1}(\mathbf{p_{2d}}, \mathbf{d}) = \mathbf{d K^{-1}} \begin{bmatrix} p_{2d} \\ 1 \end{bmatrix} \tag{2.16}$$

2.2.2 LiDAR

The next sensor we use is LiDAR, a time-of-flight sensor that actively projects light into the environment and calculates the distance d to the nearest surface via,

$$d = \frac{ct}{2} \tag{2.17}$$

where c is speed of light and t is the time from projecting the light pulse to when its return was detected.

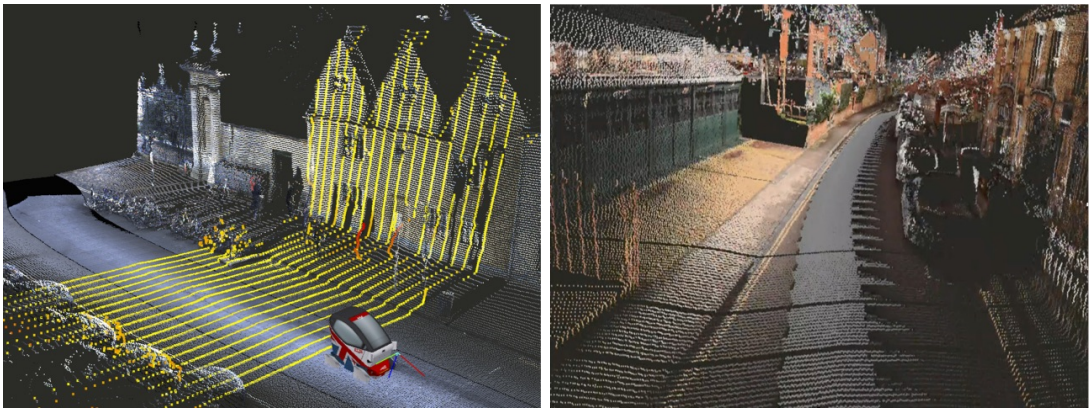


Figure 2.4: Figure showing a robotic platform equipped with 2D LiDAR sensor in a push-broom orientation (left). Each scan shown in yellow represents a vertical snapshot of the environment which can be leveraged to create a sparse 3D reconstruction of the environment if the motion profile of the platform can be obtained (right). Additionally, if the platform contains cameras that are externally calibrated to the LiDAR sensor, the laser points can be coloured using images from the camera as shown in the right.

In this work we utilise both 2D and 3D LiDAR sensors. While the 3D sensors obtain a full reconstruction, within observable range, of the environment, 2D LiDAR sensors only sweep a single laser beam at different angle increments along a plane. However, if these sensors are mounted in a push-broom orientation as shown in Figure 2.4 sparse 3D reconstructions of the environment can be created by integrating the laser scans with the motion of the laser as follows.

1. Compute the platform's trajectory via odometry.

2. Interpolate the $\mathbb{SE}(3)$ pose of each laser point from the LiDAR sensor.

Inevitably most odometry systems suffer from drift over time which leads to inaccuracy on the subsequent reconstructions thus they give a less accurate picture of the environment than that given by 3D LiDAR scanners but perform this at a much lower cost. For the platforms we interrogate in this thesis, Visual Odometry (VO) from an associated camera is used to obtain the motion estimates. Integrating these motion estimates with the LiDAR scans requires accurate external and temporal calibration which we present in the next section.

(a) Correct Calibration **(b)** 5° RPY error

Figure 2.5: If the external calibration between a camera and LIDAR sensor is correct, points projected from the LIDAR sensor accurately overlay the 3D structure seen in images (left) taken from the Oxford RobotCar dataset [10]. A small error in this external calibration , 5° in RPY, leads the points out of alignment as can be seen in the brown wall in the right image. Inaccurate calibration leads to corruption if information is shared between the two sensors.

2.2.3 Calibration

When using data from multiple sensors, as is required to create the sparse reconstructions from 2D LiDAR sensors shown in Figure 2.4, the relative $\mathbb{SE}(3)$ pose between the different sensors, or extrinsic calibration, must be known beforehand. As robotic platforms might employ a different number of sensors the calibration process is simplified in this work by defining them against the **robot** reference frame.

For the calibration process, coarse physical measurements are first taken before more sophisticated methods that use overlapping measurements are used to refine it [8, 9]. In the case of the calibration of cameras this can be done by placing objects whose location is known in the overlapping field of view of the cameras after which the unknown relative transform can be computed. The calibration between LiDAR sensors and cameras can be performed by seeking an overlap between the reflectance values in the 3D LiDAR point cloud and the camera images.

Figure 2.5 shows the effects of poor calibration. In the well-calibrated image, laser points projected into the image overlap properly with the scene's underlying 3D urban structure. However, in the poorly-calibrated image, the projected laser points do not align correctly in which case information from the camera to the laser

cannot be transferred without corruption or vice-versa. Accurate calibration thus enables the full use of all the sensors available on a robotic platform.

2.3 Conclusions

This chapter introduces some of the components underpinning robotic platforms that traverse the world. These form the main target for the evaluation of the multi-model fitting approach that we present in this thesis. The reference frames and transformations described in this chapter enable us to denote the precise location of a particular sensor or platform in time. The sensors, which the robot uses to observe the world are then described before underlining the importance of accurate calibration between sensors if information is shared between them.

With these preliminaries detailed, we proceed to present the theory for our proposed multi-model fitting algorithm in the following chapter. Firstly situating ourselves within the literature before describing the differences between our proposed formulation and the current state-of-the-art in Chapter 3 before benchmarking in Chapter 4. We then apply this formulation to different problems in contemporary robotics as detailed in Chapters 5, 6 and 7.

...in fact, the great watershed in optimisation isn't between linearity and nonlinearity, but convexity and nonconvexity.

R. Tyrrell Rockafellar, in SIAM Review, 1993

3

COnvex **R**elaxation **AL**gorithm (CORAL)

Abstract

Fitting of multiple geometric models to data is a chicken and egg problem: data must be clustered to models whose unknown parameters must also be estimated. Additionally in most cases the number of models is also not known *a priori* and the data is contaminated with noise and clutter.

In this chapter we present the theory for a novel method to fit and segment multi-structural data via a COnvex Relaxation ALgorithm (CORAL). Unlike greedy methods - which maximise the number of inliers - this approach efficiently searches for a soft assignment of points to models by minimising the energy of the joint classification. A classification that not only considers the geometric errors induced by the geometric models but also regularises for both spatial smoothness and compactness. This is similar to state-of-the-art energy minimisation techniques however, while these techniques take a combinatorial sequential approach our formulation can be efficiently parallelised. This results in an energy minimisation that is up to two orders of magnitudes faster, as the resolution of data increases, without compromising on accuracy.

Contents

3.1 Multi-Model Fitting 17
3.1.1 Greedy Multi-Model Fitting 17
3.2 Energy Based Multi-Model Fitting 25
3.2.1 Energy Minimisation 29
3.2.2 Comparison of Discrete and Convex Energy Minimisation 44
3.2.3 Energy Minimisation On A Continuum of Labels 50
3.2.4 Implementation 52
3.3 Conclusions 56

THIS thesis is concerned with the fitting of multiple geometric models to data consisting of clutter and points that constitute an initially unknown set of geometric models. Geometric models have long been of interest among multiple fields and play a key role in how the world is understood. They have found use in different applications from robotics which incorporates geometric models in its navigation processes as seen in the Simultaneous Localisation and Mapping (SLAM) literature [11–18] to computer vision which has geometric models at the core of several important processes such as camera calibration [19–22], ego-motion estimation [23] and 3D reconstructions [24–26]. This makes geometric models fundamental to a multitude of ways in which we view and interact with the world and obtaining them in an accurate and timely way an important task within these fields.

This chapter serves to introduce and show the evolution of the various geometric multi-model fitting algorithms that have been seen in the literature. This culminates in the description of the main contribution of this thesis, a novel geometric multi-model algorithm which is able to robustly obtain geometric primitives in the presence of clutter and noise.

Section 3.1 presents an overview of the most common geometric multi-model fitting algorithms and their benefits and shortcomings. Section 3.2 then describes the global energy formulation that this work follows going in detail through the options available for the minimisation of this global energy before teasing out the differences between our proposed formulation and other energy based approaches. Conclusions are then drawn in Section 3.3.

3.1 Multi-Model Fitting

In this section we perform a review of existing geometric multi-model fitting algorithms and their benefits and shortcomings before arriving at the energy based formulation in Section 3.2 selected in this thesis.

3.1.1 Greedy Multi-Model Fitting

The general goal of geometric model fitting is to cluster data points $\mathbf{u} \in \Omega$ according to their proximity to geometric models of a known type, where $\Omega \subset \mathbb{R}^m$ represents a continuous domain. This aims to fit the data to one of L models $\{1, 2, \cdots, L\}$. The parameters of these models must also be estimated from the membership of these points, creating a chicken and egg problem. Throughout this thesis membership to a model with parameters ρ_l is indicated by the function ϕ_l that depends on a given threshold T, where $l \in 1, 2, \cdot, L$.

This gives the clustering step as in Equation 3.1.

$$\phi_l(\mathbf{u}) = \begin{cases} 1 & \text{if } \rho_l(\mathbf{u}) < T \\ 0 & \text{otherwise} \end{cases} \tag{3.1}$$

wherein $\rho_l(\mathbf{u})$ is an overloaded notation that computes a distance between data points $\mathbf{u}$ and a model with parameters ρ_l. This is followed by the parameter estimation step in Equation 3.2

$$\hat{\rho}_l = \arg\min_{\rho_l} \sum \rho_l(\mathbf{u})\phi_l(\mathbf{u}) \tag{3.2}$$

The majority of contemporary algorithms treat the two steps outlined above, membership classification and parameter estimation, as isolated subproblems and aim to greedily maximise the membership of each model. This approach was popularised by the **RAN**dom **S**ample **A**nd **C**onsensus algorithm (RANSAC) [27] which found success in many applications, giving a robust solution when a single model is needed while dealing with a large numbers of outliers.

Prior to this, the use of robust statistical methods that considered all data points were the norm [28–30]. The governing assumption of these techniques is

that there are always enough "good values" i.e. inliers to the sought geometric model in the data to "smooth out" the outliers [27]. An assumption that falls apart as the number of outliers increase beyond a certain threshold. In many cases however gross outliers can still be identified from the biased model parameters and by removing these outliers a refined model estimation can be retrieved. Methods for model estimation using robust statistics therefore iterate through the following two steps until no more outliers are found.

- Refine the model parameters from the data.

$$\hat{\rho_1} = \arg\min_{\rho_1} \sum \rho_1(\mathbf{u})\phi_1(\mathbf{u})$$

- Identify and remove data points that are outliers to these parameters.

$$\phi_1(\mathbf{u}) = 0 \text{ if } \hat{\rho_1}(\mathbf{u}) > T$$

This heuristic method however fails in many cases returning models that are not consistent with the ground truth even at low outlier percentages as demonstrated in Figure 3.1.

Key to the approach of RANSAC, was that it ran opposite to the "smoothing" approach of robust statistics which start the model estimation using the maximal set of data and then aim to minimise the set of data. RANSAC conversely begins with a minimal set to obtain model parameters and then aims to maximise the set of data using a two-step hypothesis and test approach. In the first step a model is proposed by randomly sampling data points into a Minimal Sample Set (MSS) whose parameters are then computed, this model is then tested on the data to compute the data points that agree with this hypothesised model. This set of models forms the Consensus Set (CS) for the current model hypothesis. If the Consensus Set is large enough, based on the estimate of error on the data, this model is selected and refined through its support. If not a new model is hypothesised and the process continued until a predetermined number of trials is run after which the model with the largest consensus set is chosen. Algorithm 1 presents the full RANSAC algorithm.

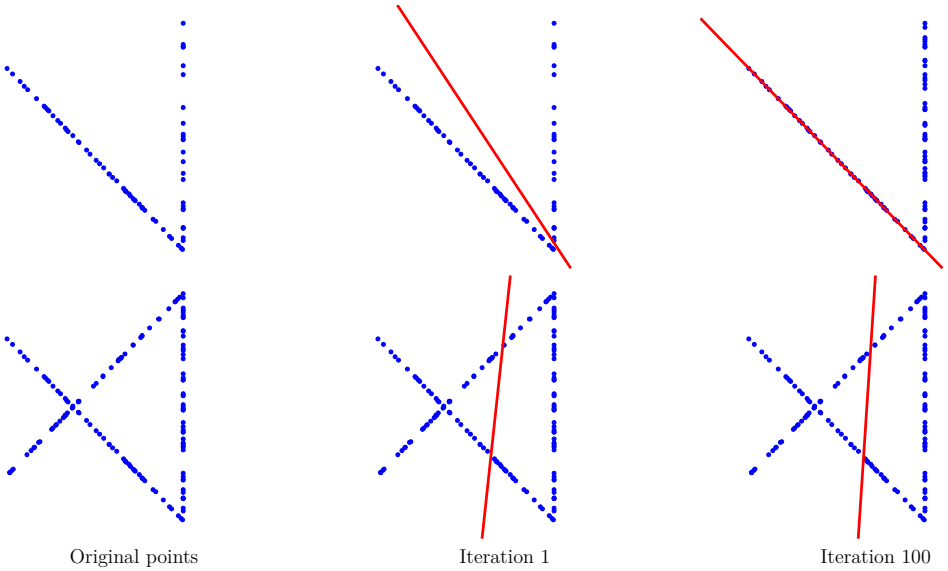


Figure 3.1: In this Figure a robust statistical method for line fitting is illustrated. In this method all the data points shown in blue (left column) are used to compute the parameters of a model, a line shown in red in this example (middle column), after which outliers to this model are removed. This then iterates between a parameter re-estimation using the previous inliers followed by outlier removal until no more outliers can be removed (right column). In the top right this results in a model consistent with the ground truth that is found in the presence of outliers but can be seen to break down in the bottom right as a "phantom" model is fitted.

As can be seen in Algorithm 1 the number of trials run is dependent on the probability P of selecting a good subset of points n that would correspond to a correct model. The inlier ratio w, which is the number of inliers as compared to the total number of points, gives the probability that the selected points are within the error threshold of the model. Additionally P captures the probability that at least one of the random selections is an error free set. From these two the number of trials N_t can be obtained.

$$\begin{aligned} (1 - w^n)^{N_t} &= (1 - P) \\ N_t &= \frac{log(1 - P)}{log(1 - w^n)} \end{aligned} \tag{3.3}$$

.

With this, a principled approach to fitting a single model with some guaranteed probability is obtained. This however does not immediately transfer to the multiple model case. As pointed out in [31] the probabilities of rejecting and accepting the

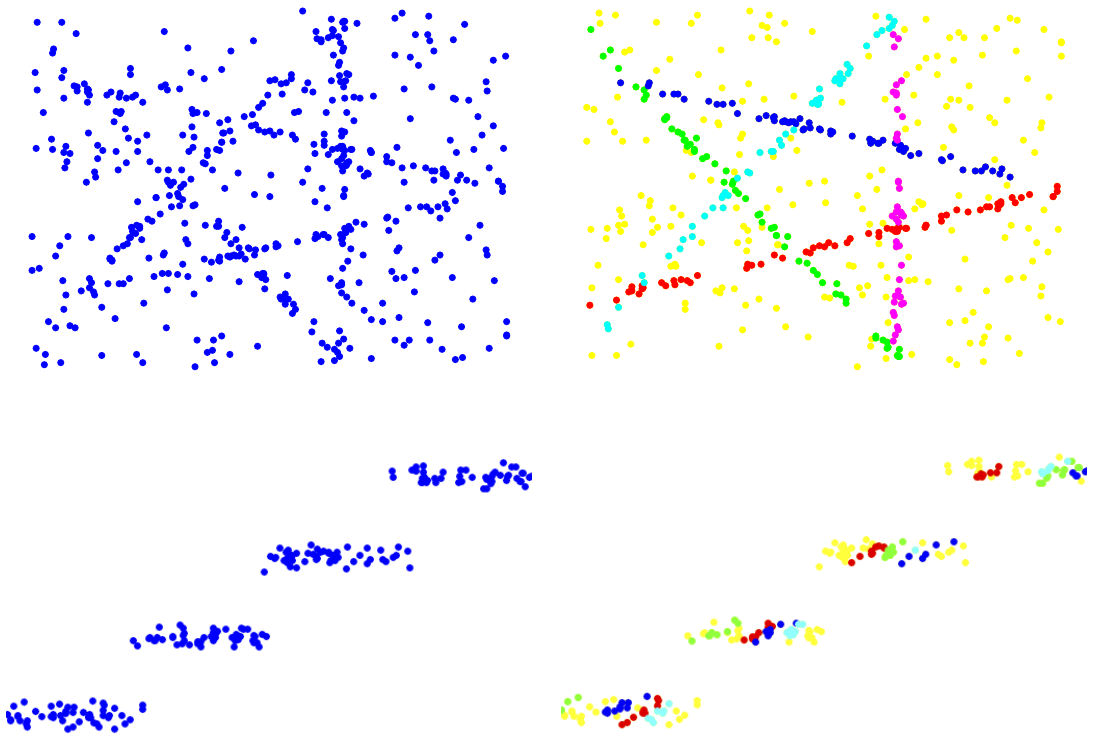
Algorithm 1: Single Model RANSAC Algorithm

Initialisation;
P = probability, n = Minimal Set Size, N_t = Number of trials;
$w = \frac{n}{num\_points}$;
$N_t = \frac{log(1-P)}{log(1-w^n)}$;
Largest set size = 0;
k=0;
while $k < N_t$ **do**
  Hypothesis Step;
  $\hat{\mathbf{u}} = MSS(\mathbf{u})$;
  $\rho_1 = \arg\min_{\rho_1} \rho_1(\hat{\mathbf{u}})$;
  {Test Step};
  $\phi_1(\mathbf{u}) = CS(\rho_1, \mathbf{u})$;
  Current set size = $\sum \phi_1$;
  if *Current set size* > *Largest set size* **then**
    Largest set size = Current set size;
    $N_t = \frac{log(1-P)}{log(1-w^n)}$;
    $\hat{\rho_1} = \arg\min_{\rho_1} \sum \rho_1(\mathbf{u})\phi_1(\mathbf{u})$;
  end
  k=k+1;
end

correct and incorrect hypotheses employed in RANSAC can only be considered to define the cost of a decision in a single model case and thus RANSAC cannot be trivially adapted to multiple model problems.

Various methods [31–34] have however proposed various generalisations for RANSAC with applications in multi-model fitting. In [31, 32] RANSAC is run sequentially, a single model is obtained using the classical RANSAC method shown in Algorithm 1 after which the model's inliers are removed and iteration continues for the next model. Zuliani et al. [33] argue that sequential RANSAC formulations are however sub-optimal and any inaccurate inlier estimation in the initial models will lead to a cascading of errors in subsequent models as can be seen in Figure 3.2.

A parallel rather than sequential formulation of RANSAC such as multi-RANSAC [33] is thus more suited for multi-model fitting, as the result in Figure 3.3 demonstrates. This approach is similar to data clustering techniques however crucially it requires prior knowledge of the number of models. A parameter that is

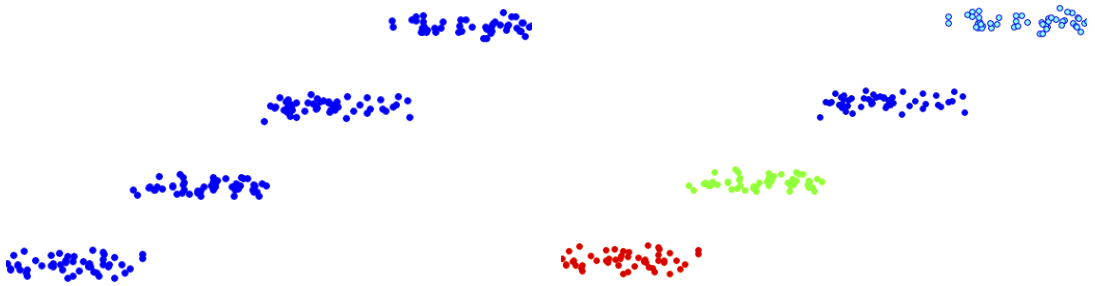


Original points Sequential RANSAC Result

Figure 3.2: Figure illustrating the application of sequential RANSAC for multi-model fitting. In this approach the RANSAC algorithm is first run to obtain a single model after which the inliers to this model are obtained and removed, after which RANSAC is run again. In the top row this results in the fitting of models that are consistent with the ground truth even in the presence of clutter however in many cases as illustrated in the bottom row errors in the initial model selection leads to a cascading of wrong, but geometrically valid, models. Data points in the right column are colour-coded according to their membership to the set of geometric models.

generally unavailable in most of the applications we are interested in thus limiting its use in practical applications.

In a similar manner to multi-RANSAC but without requiring any previous knowledge of the number of models, the Randomised Hough Transform (RHT) [35] is able to fit an *a priori* unknown number of models from data. This is through a hypothesis and voting technique over the model-parameter space, whereby a histogram is built through randomly selected minimal sample sets of the data points whose parameters are recorded. The multiple models are revealed as peaks in the histogram. As with all Hough transform methods the selected models are



Original points Multi RANSAC Result

Figure 3.3: Figure illustrating the application of multi-RANSAC for multi-model fitting. In this approach a parallel rather than sequential approach is applied to the multi-model fitting which limits the errors introduced by a wrong initial model yielding better performance on the example in Figure 3.2. However, for this result the number of line models, in this case four, had to be provided to the algorithm beforehand limiting its use to situations where the number of geometric models is explicitly known. Data points in the right column are colour-coded according to their membership to the set of geometric models.

sensitive to the choice and parametrisation of the parameters, with low accuracy and computational efficiency experienced with these techniques.

This was part of a wider trend in the literature of shifting from voting techniques to finding modes in the parameters space through a mapping, such as the Hough Transform, followed by application of a mode finding algorithm such as mean-shift [36]. Techniques such as Randomised Hough Transform [35] however were not intrinsically robust and suffered in the presence of gross outliers but could easily deal with multiple models which RANSAC and its variants could not. Zhang et al. [37] proposed a new method that attempted to overcome the shortcomings of these two dominant schools of thought in multi-model fitting. This was through an analysis of the residuals of data points obtained from RANSAC hypotheses. This signalled a move from studying only the distribution of residuals per hypothesis, which RANSAC and its variants did, to a more elaborate understanding of the distribution of residuals for each data point. It follows that the residuals have peaks corresponding to their true models in a similar fashion to the RHT. In practice

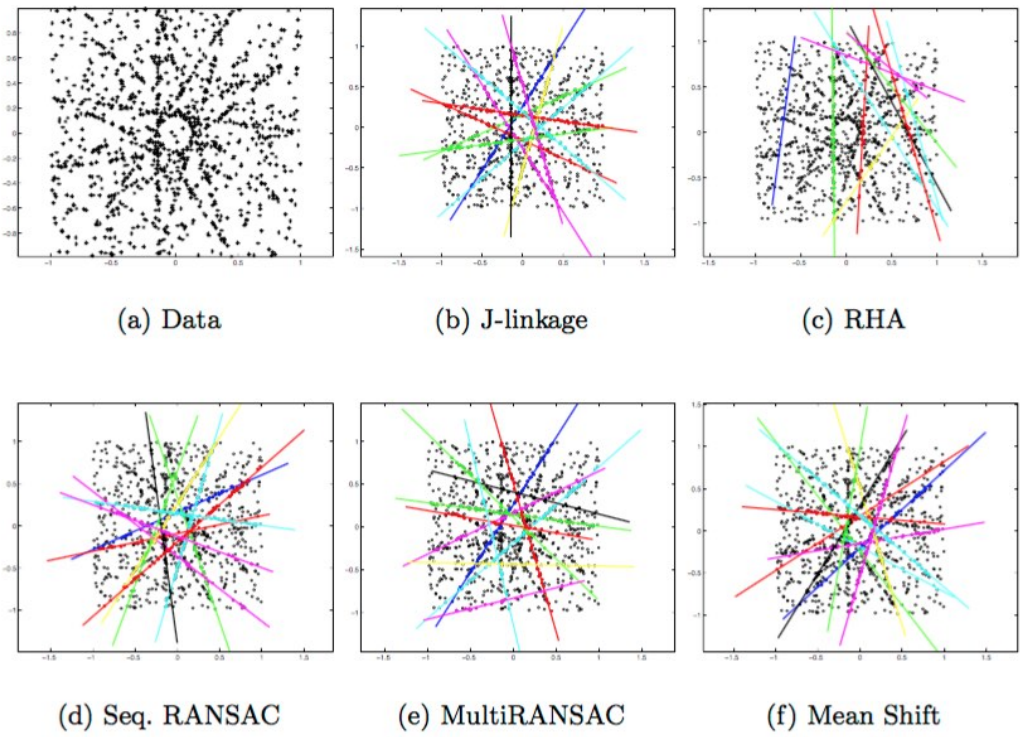


Figure 3.4: Figure showing the results of different multi-model fitting algorithms on data that contains lines buried in noise. The mode-finding approaches such as mean-shift and RHA are unable to obtain the correct line models that correspond to the ground truth. Similar outcomes were observed with the RANSAC based approaches showing that clustering based approaches such as J-linkage offer better results in the presence of noise. Figure from [38].

these modes proved to be hard to find particularly when points were distant to the hypothesised model, limiting application to low-noise conditions.

This was strengthened in J-Linkage with Toldo and Fusiello [38] using the same formulation while overcoming the difficulty in mode estimation through the transformation to the *conceptual representation* space rather than the parameter space relied on in [37]. In this space each point is represented by a characteristic function of the set of models M preferred by that point. This is generated by the calculating the distance of the point to the hypothesised models, if the distance is lower than ϵ then a value of 1 is added to a vector with the length of the models, otherwise a 0 is added to the vector. This vector is referred to as the preference space of a data point (PS). Points belonging to the same model will have similar

conceptual representations and cluster in this space $\{0,1\}^M$. The clustering of this represents the latest innovation of this method, with the Jaccard distance used to measure overlap between sets. It can be seen from the results presented in Figure 3.4 that J-linkage was able to return better performance than RANSAC and its variants on simulated data.

What can be seen from this approach is that although a maximum inlier approach is not explicitly given, the algorithm still greedily increases the size of its clusters which results in the same outcome. This remains true of the other variants of greedy clustering that have been developed in the literature, such as T-linkage [39], Residual Preference Analysis (RPA) [40], Random Cluster Model Simulated Annealing (RCMSA) [41] and Integer Linear Programming Ransac Cover (ILP-RansaCov) [42].

Isack and Boykov [43] however argue that this formulation is fundamentally flawed, as the overall classification of data is ignored by greedily maximising the number of inliers either implicitly (clustering) or explicitly (RANSAC) while seeking multiple models. In greedy formulations, situations of high noise and clutter could lead to "phantom" models being selected, as a random model originating from the clutter could potentially have more inliers than the ground-truth models sought.

In contrast, global energy-based approaches [43–45] present a more general optimisation framework that jointly fits all models present in multi-structural data. In practice, global energy methods aim to find an optimal fitting solution by accounting for the model error in a data fidelity term for a given metric. In addition, the set of solutions can be constrained by encouraging spatial smoothness in a regularisation term. A more complete formulation also considers the number of models as a parameter to optimise for [43] through an additional regularisation term that encourages compact solutions. With these methods shown to consistently outperform greedy techniques, this thesis adopts and adapt this formulation in our contribution.

In the following section we present in more detail the theoretical background for global energy formulations used in multi-model fitting problems for multi-model fitting before describing the approach adopted within this thesis.

3.2 Energy Based Multi-Model Fitting

While energy-based approaches have been applied widely in computer vision problems such as image segmentation, depth-map estimation and dense reconstructions [46–49] it was only until the work of Isack and Boykov [43] that a general approach to apply them to multi-model fitting was presented. Prior to this, energy approaches were used in specific problems such as motion segmentation [50] and affine model extraction [46, 48, 49].

Thus the work presented by Isack and Boykov [43], culminating in the **P**ropose **E**xpand **A**nd **R**e-estimate **L**abels (PEARL) algorithm can be considered to be the first general formulation of energy based geometric multi-model fitting applied to a multitude of problems without degradation of performance.

To develop this formulation, we first begin with an energy-based interpretation of the objective function found in many of the greedy model-fitting algorithms into a labelling problem using the indicator function $\phi_l(\mathbf{u})$ defined in Equation 3.1. In the case of a single model, the role of model fitting is to find the parameters ρ of the model with the largest number of inliers or Consensus Set (CS). This can be represented as the maximisation of the energy in Equation 3.4.

$$\mathbf{E}(\phi) = \sum_{\mathbf{u}} \phi_1(\mathbf{u}) \tag{3.4}$$

With this formulation only the inliers of a single model can be maximised over the model parameters. In this thesis we are however interested in the case where data supports multiple models. If say the data supports L models we now have a minimisation of the following energy:

$$\mathbf{E}(\phi) = \sum_{l=1}^{L} \sum_{\mathbf{u}} \rho_l(\mathbf{u}, \phi_l(\mathbf{u})) \tag{3.5}$$

wherein $\rho_l(\mathbf{u}, \phi_l(\mathbf{u}))$ is an overloaded function that returns the distance between data points $\mathbf{u}$ and the model with parameters ρ_l if the indicator function $\phi_l(\mathbf{u})$ at that point has assigned it to model l. This can be expressed mathematically as $\rho_l(\mathbf{u}, \phi_l(\mathbf{u})) = \rho_l(\mathbf{u})\phi_l(\mathbf{u})$.

This energy thus corresponds to the geometric errors of points to their assigned models. It must be noted that a naive minimisation of this energy in Equation 3.5 leads to a trivial solution whereby all the data points are assigned a model that perfectly fits them, minimising the geometric errors but overfitting the data. If the number of models is however prescribed, the minimisation of this energy is equivalent to the standard K-means algorithm. The limitations of this algorithm are unfortunately inherited when applied to multi-model fitting, such as its sensitivity to initialisation and bias towards equal distribution of points to models as was shown by Delong et al. [45].

In light of this and with the knowledge that the number of models present in data is seldom known, Li [51] proposed that the energy should also penalise the number of models $||L||$, minimisation of which would lead to a compact solution. This is commonly known as the *uncapacitated facility location* UFL problem.

$$\mathbf{E}(\phi) = \sum_{l=1}^{L} \sum_{u} \rho_l(\mathbf{u}, \phi_l(\mathbf{u})) + \beta \|L\| \tag{3.6}$$

Additionally Isack and Boykov [43] proposed another term to be added to the energy formulation that encouraged spatial smoothness. In other words, there exists a strong prior that points located in close proximity to each other would belong to the same model. This form of regularisation was common in Markov Random Field (MRF) based segmentation problems but had not been justified when data was independent and identically distributed (i.i.d.) as is the case in most of the geometric model fitting problems. However, Isack and Boykov [43] argue that the data does in fact come from regular objects thus allowing a notion of spatial regularisation to be brought up, with results that show an increase in performance justifying this approach. Additionally, local sampling in RANSAC and J-linkage has been shown to increase the performance of these algorithms, supporting the case for some spatial consideration when performing geometric multi-model fitting. This gives the global energy as in Equation 3.7 whose minimisation is known as the *metric labelling* problem.

$$\mathbf{E}(\phi) = \sum_{l=1}^{L} \sum_{u} \rho_l(\mathbf{u}, \phi_l(\mathbf{u})) + \beta \|L\| + \lambda \sum_{(p,q) \in N} w_{pq} \delta(\phi_l(u_p) \neq \phi_l(u_q)) \tag{3.7}$$

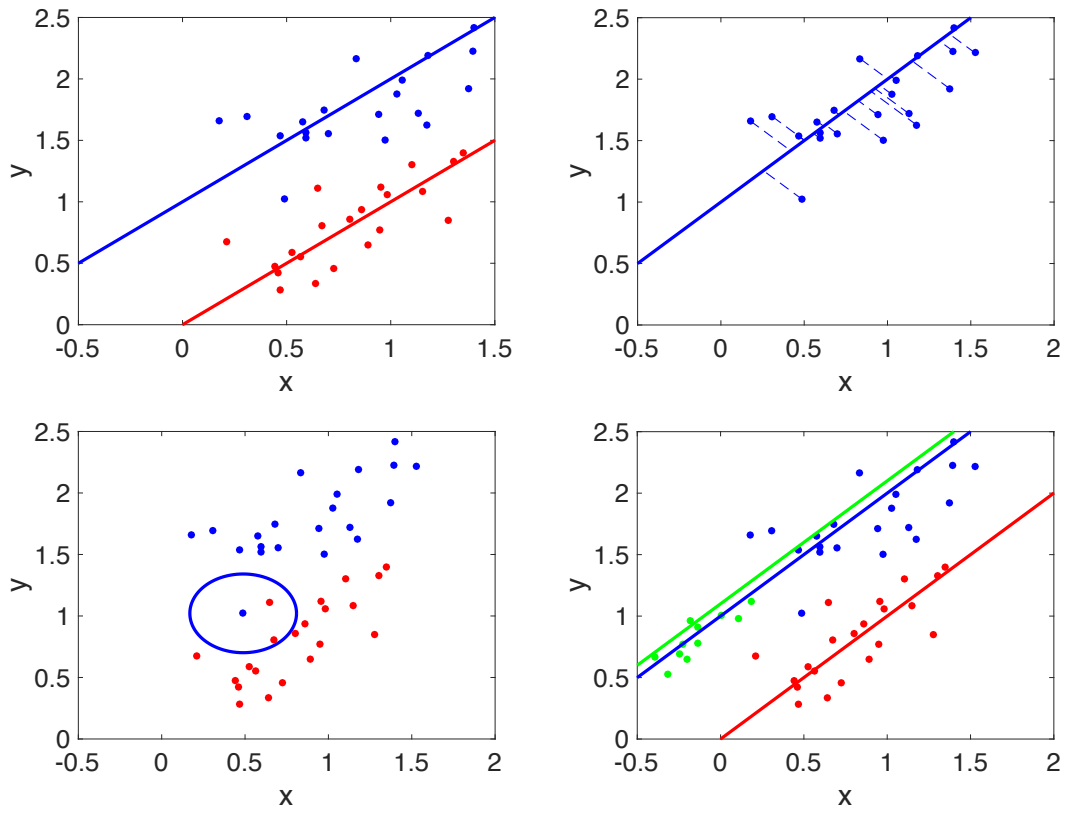


Figure 3.5: Figure showing a geometric line fitting example to illustrate the energy functions used in this work. From noisy data (top left) line models can be fitted and are displayed colour-coded. The first energy is associated to the distance from a point to the line or its geometric cost as shown in the top right image. The second energy regularises for smoothness over a defined neighbourhood and penalises points in the same neighbourhood that have different labels as seen in the bottom left. The third energy regularises for compactness and penalises redundant models (green line) as seen in the bottom right image.

Where N is some neighbourhood (e.g. edges on some near-neighbour graph). A penalty, weighted through w_{pq} is added when neighbouring data points such as u_p and u_q do not share the same model. These constituent energies are illustrated in Figure 3.5 through a toy line-fitting example.

With the energies defined, geometric multi-model fitting can be framed as an optimised labelling problem in which the quality of the solution is linked to an energy functional. With a view to the continuous energy minimisation optimisation approach that we opt for in this thesis that is presented in the subsequent sections,

we restate the energy functional given in Equation 3.7 in a continuous way as below:

$$\mathbf{E}(\phi) = \underbrace{\sum_{l=1}^{L} \int_{\Omega} \rho_l(\mathbf{u}, \phi_l(\mathbf{u}))d\Omega}_{\text{Geometric Error Energy}} + \lambda \underbrace{\sum_{l=1}^{L} \int_{\Omega} \omega_{\mathcal{N}} R(\nabla_{\mathcal{N}} \phi_l(\mathbf{u}))d\Omega}_{\text{Smoothness Energy}} + \underbrace{\beta \|L\|}_{\text{Compactness Energy}} \tag{3.8}$$

The first term in Equation 3.8 is a data fidelity term containing the geometric errors, defined over the data points $\mathbf{u} \in \Omega$, where $\Omega \subset \mathbb{R}^m$ represents a continuous domain. This can be seen as the geometric cost of a data point supporting a particular model with parameters ρ_l.

In practice, some data points might not be explained by a geometric model. For this case, a special label $\emptyset$ is added, representing the outlier model. In this way, a constant cost γ is assigned to points that cannot be explained by any geometric model. The model cost for the outlier model is simply given by $\rho_\emptyset(\mathbf{u}, \phi_\emptyset(\mathbf{u})) = \gamma$.

The second term in Equation 3.8 takes into account locality by promoting a homogeneous assignment of labels to neighbouring points. The $\nabla_{\mathcal{N}}$ operator calculates the gradient of the indicator function over the neighbourhood $\mathcal{N}$ of a point. Thus revealing points that are in the same neighbourhood but do not share the same model. $R(\cdot)$ is a functional that penalises lack of smoothness, this can be evaluated over different norms $|\cdot|_p$. For instance, the non-smooth $|\cdot|_1$ [1] norm. The parameter λ controls the trade-off between the smoothness/locality cost and the model cost with the weights $\omega_{\mathcal{N}}$ allowing for finer control of the dependencies between neighbouring points. For example, in cases where the k nearest neighbours for a given point are actually at a far distance, the effect of the smoothness term can be reduced by decreasing the value of ω according to the separation. Finally, the Compactness Energy term in Equation 3.8 penalises the number of models L by adding a constant cost β per model that encourages a more compact solution.

To reveal a compact set of geometric models that considers not only the geometric errors but the spatial neighbourhood around data points the energy functional presented in Equation 3.8 must be minimised. In the following section we present a brief overview of standard optimisation techniques and show why the proposed

[1] In the 2D case, $|\nabla_{\mathcal{N}} \phi_l(\mathbf{u})|_1 = |\nabla_x \phi_l(\mathbf{u})| + |\nabla_y \phi_l(\mathbf{u})|$

formulation cannot be minimised by these before presenting two different approaches to the minimisation based on the choice of the form of the indicator function.

3.2.1 Energy Minimisation

Before jumping into the details of the specific approaches for the energy minimisation of energy in Equation 3.8, let us consider a general objective function $\mathbf{E}(\mathbf{r})$

$$\mathbf{E}(\mathbf{r}) = \mathbf{E}_{data}(\mathbf{r}, \mathbf{u}) + \mathbf{E}_{regularisation}(\mathbf{r}) \tag{3.9}$$

where $\mathbf{E}(\mathbf{r})$ is the function to be minimised, $\mathbf{r}$ is the proposed solution, and $\mathbf{u}$ is the set of noisy input data. The two terms, regularisation and data, in general seek a solution that meets some desired priors (*regularisation*) without varying too far from the original noisy observations (*data*). Apart from trivial cases, there are generally no closed-form solutions to solve the energy-minimisation in a single step.

If $\mathbf{E}(\mathbf{r})$ is convex and smooth a global minimum solution exists. According to [52] a function $\mathbf{E}(\mathbf{r})$ is convex if its domain $\mathbf{r}$ is itself convex. Where $\mathbf{r}$ is a convex if and only if the line segment between any two points in $\mathbf{r}$ lies in $\mathbf{r}$; that is, if for any $x, y \in \mathbf{r}$ and $\alpha \in [0, 1]$, we have

$$\alpha x + (1 - \alpha)y \in \mathbf{r} \tag{3.10}$$

Additionally for any $x, y \in \mathbf{r}$ and $\alpha \in [0, 1]$, $\mathbf{E}(\mathbf{r})$ is convex if:

$$\mathbf{E}(\alpha x + (1 - \alpha)y) \leq \alpha \mathbf{E}(x) + (1 - \alpha)\mathbf{E}(y) \tag{3.11}$$

A simple illustration of a convex function is given in Figure 3.6 where it can be seen that for any $x, y \in \mathbf{r}$ the line segment between them (dotted line) lies in $\mathbf{r}$.

For convex functions, gradient descent techniques can be formulated to find their global minimum solution. Gradient descent works by taking small "steps" along the energy function, where each step moves closer to the global minimum. In the case of convex problems, there are no local minima (just a single global minimum), and the system will converge upon the same final solution, no matter the initial value of $\mathbf{r}_k$. Gradient descent is a continual loop executing [52],

$$\mathbf{r}_k = \mathbf{r}_{k-1} - \alpha \nabla \mathbf{E}|_{\mathbf{r}_k} \tag{3.12}$$

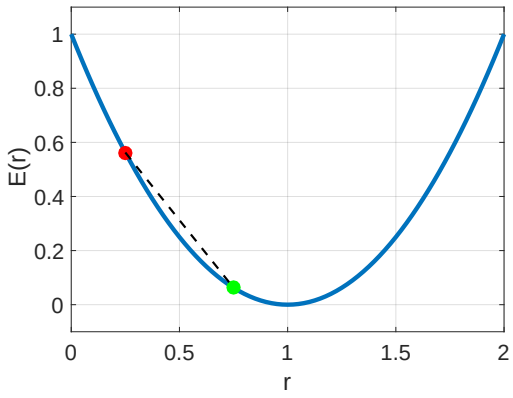


Figure 3.6: Figure of a convex function. For a convex function the line segment between any two points lies above the graph as can be seen by the dotted line plotted and can then be minimised by gradient descent techniques.

where $\nabla \mathbf{E}|_{\mathbf{r}_k}$ is the Jacobian of the energy function and α, is the step size. If α is sufficiently small, the system is guaranteed to converge upon the optimal solution for smooth convex energy functions.

Transitioning back to our specific energy function, the particular optimisation strategy depends on the form of the indicator function $\phi_l(\mathbf{u})$. Taking Equation 3.1, $\phi_l(\mathbf{u})$ is a binary or *hard* assignment of membership of data points to models. This binary assignment makes $\mathbf{E}(\phi)$ in Equation 3.8 non-convex which can be illustrated through the following example.

Let us consider the simple case where two points are to be assigned to two geometric models. If a binary assignment is used, it can be seen that this generates four different unique solutions, with the energy from the smoothness term of Equation 3.8 shown in Figure 3.7. From this Figure it can be seen that the set of solutions for this function is discrete and disjoint. A line segment drawn between two possible solutions contains values that are not part of the solution set, which as was shown in Equation 3.10 makes this function non-convex.

Additionally for the geometric multi-model fitting problem the hard assignment may be able to assign a particular data point to multiple models as can be seen in Equation 3.8. As a data point can only be assigned to a single model in a particular solution, these ambiguities leads to multiple solutions.

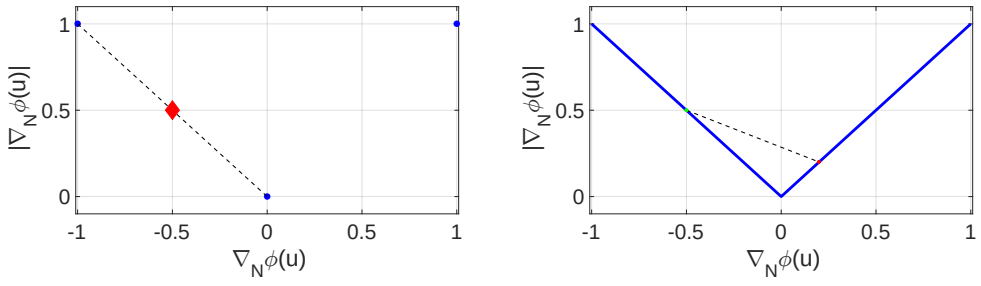


Figure 3.7: Figure illustrating the choice between binary assignment for the indicator function $\{0, 1\}$ or its relaxation to a soft assignment $\phi(\mathbf{u}) \in [0, 1]$ proposed in this thesis. Observation of the smoothness energy from the binary assignment (left) reveals a discrete solution set shown by the blue points. As a result line segments between values in the solution set(the dotted) line contain values that are outside the solution (red diamond) and according to Equation 3.10 is thus non-convex. In contrast, observation of the same energy from the relaxed assignment (right) shows that a line segment (dotted line) between two possible solutions contains values that are still within the solution set making this function convex. Though this relaxation makes the smoothness energy convex, this energy happens to be non-smooth. However, techniques from continuous optimisation allow us to transform this to an alternate form over which standard gradient based techniques can be used for its minimisation.

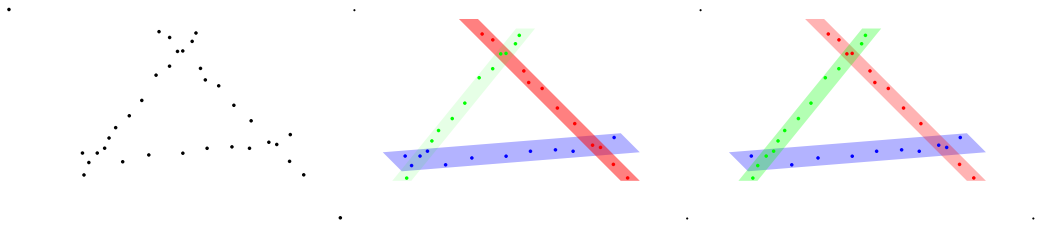


Figure 3.8: Figure showing a geometric line fitting example to illustrate the ambiguities arising from a strict binary or *hard* labelling. Given data points from three lines that intersect with each other(left), a binary labelling can reveal multiple solutions as intersecting points may be correctly assigned to different models as is seen in the middle and right images.

This in fact corresponds to discrete multi-labelling problems where it can be shown [53] that with three labels or more the labelling problems are **NP**-Hard and only approximate solutions can be found i.e. they do not converge to a global minimum. All multi-model fitting problems that we deal with in this thesis have at least three labels because of the presence of an outlier model hence suffer in the same way. Moreover, the minimisation of the spatial and compactness energies of Equation 3.8 are also **NP**-Hard. However, several combinatorial optimisation

techniques exist that provide approximate solutions for the energy minimisation the most notable of these being α-expansion [53] that was used in PEARL.

For this thesis, we relax the indicator function by allowing $\phi_l(\mathbf{u})$ to take values in the interval $\phi_l(\mathbf{u}) \in [0, 1]$ instead of the binary set $\{0, 1\}$. While this relaxation is not the tightest, it produces good results in practice [54]. This transforms the objective function from Equation 3.8 to a convex problem forming the basis for our proposed **CO**nvex **R**elaxation **AL**gorithm (CORAL). Any line segment drawn between two possible solutions contains values that are part of the convex set, which as stated in Equation 3.10 makes this function convex as can be seen in Figure 3.7.

While this relaxation convexifies the original problem in Equation 3.8, the presence of the $|\cdot|_1$ norm in this energy functional means the problem is still non-smooth as the $|\cdot|_1$ norm is non-differentiable at its origin. This similarly excludes the use of simple gradient descent for the energy minimisation. However, techniques commonly used for convex optimisation can be used for the energy minimisation.

In the following section, the two optimisation strategies for the energy minimisation, combinatorial and convex optimisation respectively, are explored in greater detail after which a comparison of the two is carried out.

Combinatorial Optimisation For Energy Minimisation

The goal of the optimisation is to find the labelling $\phi(\mathbf{u})$ that minimises the energy in Equation 3.8. As was stated in the previous section, if the indicator function represents a *hard* assignment there can exist a multitude of local minima. A labelling $\phi(\mathbf{u})$ describes a local minimum if:

$$\mathbf{E}(\phi) \leq \mathbf{E}(\phi') \quad \text{for all} \quad \phi' \quad \text{close to} \quad \phi \tag{3.13}$$

At a local minimum, the energy cannot be decreased by changing the label of a single data point. This change is referred to as a *standard* move. Techniques such as iterated conditional modes [55] and annealing [56] that use *standard* moves tend to produce sub-optimal results as they get stuck in local minima that can be far from the global minimum. Boykov at al. [53] instead propose an algorithm,

α-expansion, that considers large moves whereby multiple data points can change their labelling simultaneously. The local minima generated through these large moves are thus closer, to a factor of two [53], to the global minimum providing more accurate solutions to the energy minimisation problem.

α-expansion is based on the min/max-flow algorithms used in combinatorial optimisation or graph-cuts as they are referred to. It operates by giving a binary choice to all of the data points in a labelling $\mathbf{f}$, the choice is either to stick to their current labels or to change their labels to α allowing α to "expand" its support. The labelling $\mathbf{f} = \{f_1, f_2, \cdots, f_n\}$ stores the indices of the models that the data points are assigned to based on the indicator function $\phi(\mathbf{u})$. Since this assignment is always unique, the same point cannot be assigned to multiple models, it can be defined as:

$$f_n = l \quad \text{if} \quad \phi_l(\mathbf{u}_n) = 1 \tag{3.14}$$

Exploring this "expansion" in more detail, given the current labelling $\mathbf{f}'$, let $\mathbf{f}^\alpha$ be a feasible α-expansion with respect to this labelling. A set of binary indicator variables $\mathbf{x} = \{x_1, \cdots, x_n\}$ can be used to encode whether a particular label chooses to stay with its current label, or changes its label to α in $\mathbf{f}^\alpha$:

$$\begin{aligned} x_p = 0 &\iff f_p^\alpha = f_p' \\ x_p = 1 &\iff f_p^\alpha = \alpha \end{aligned} \tag{3.15}$$

Let $\mathbf{E}^\alpha(\mathbf{x})$ be the energy corresponding to encoding 3.15 relative to $\mathbf{f}'$. The α-expansion algorithm computes an optimum $\mathbf{x}^*$, and thereby $\phi^\alpha(\mathbf{u})$, in other words an α-expansion of $\mathbf{f}'$ by a single graph cut. Figure 3.9 illustrates the construction of a simple graph with two nodes s and t in which the graph cut corresponds to the binary choice of either keeping or switching a data point's label to α. The edges of the graph encode the costs from the energy functional. Points that are connected to node s have their labels changed to α while those connected to t keep their labels with a cut on this graph revealing this optimum labelling.

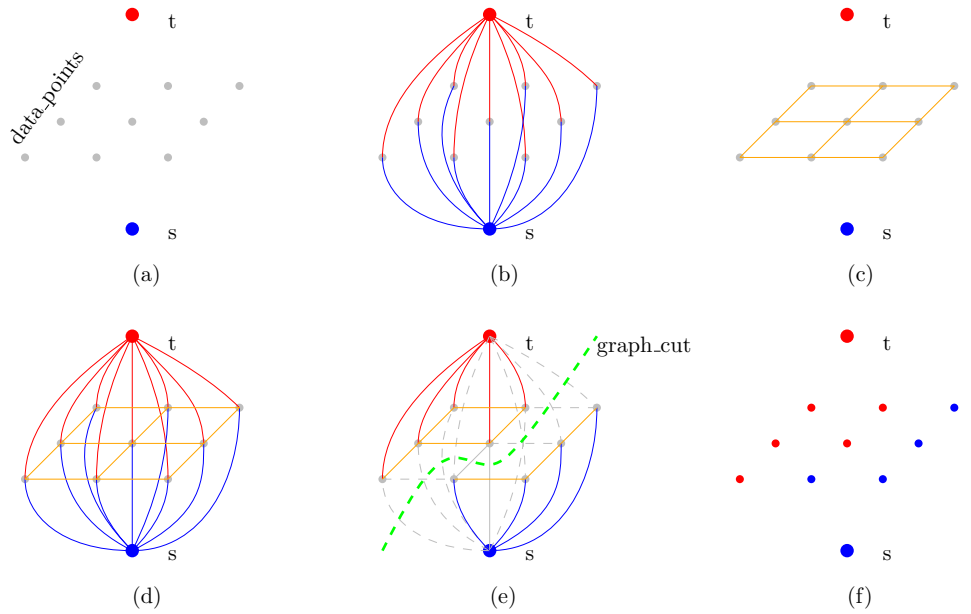


Figure 3.9: Figure illustrating a graph cut example used in combinatorial optimisation techniques. In (a) the data points to be labelled and the binary choice nodes s and t are shown before edges that represent costs from an energy functional are added in (b) and (c). The fully constructed graph is shown in (d) with the graph cut (e) that provides the final labelling shown in (f).

For example, suppose energy $\mathbf{E}(\mathbf{f})$ is such that the optimal expansion w.r.t. labelling $\mathbf{f}'$ is $\mathbf{f}^\alpha$:

$$\mathbf{f}' = \boxed{\beta}\boxed{\alpha}\boxed{\gamma}\boxed{\gamma}\boxed{\beta}\boxed{\beta} \implies \boxed{\alpha}\boxed{\alpha}\boxed{\alpha}\boxed{\gamma}\boxed{\beta}\boxed{\beta} = \mathbf{f}^\alpha$$
$$\boxed{1}\boxed{\underline{1}}\boxed{1}\boxed{0}\boxed{0}\boxed{0} = x^* \qquad (3.16)$$

where $\underline{1}$ means x_2 is fixed to 1. Here only f_1 and f_3 changed to label α while the rest preferred to keep their labels. The α-expansion algorithm iterates the above binary step until finally $\mathbf{E}^\alpha(x') = \mathbf{E}^\alpha(x^*)$ for all $\alpha \in L$ as shown in Algorithm 2.

While additional nodes are needed to completely encode the smoothness and label terms of the energy functional Equation 3.8, this is out of the scope of this thesis, and we refer the reader to [45] for further implementation details. However it remains that using α-expansion the energy minimisation of Equation 3.8 can be broken into a set of binary labelling problems that can be solved by graph

Algorithm 2: Alpha Expansion

$\phi'(\mathbf{u})$:= arbitrary labelling;
repeat for $\alpha \in L$ **do**
  $\phi^{\alpha}(\mathbf{u}) := \arg\min_{\phi} \mathbf{E}(\phi)$ where $\phi^{\alpha}(\mathbf{u})$ is an α-expansion of $\phi'(\mathbf{u})$;
  if $\mathbf{E}(\phi^{\alpha}) \leq \mathbf{E}(\phi')$ **then**
    $\phi(\mathbf{u})' := \phi(\mathbf{u})^{\alpha}$;
  end
end
until converged;

cuts, creating an efficient optimisation strategy for energy-based multi-model fitting objective functions using combinatorial optimisation techniques.

Convex Optimisation for Energy Minimisation

By relaxing the indicator function from a *hard* to a *soft* assignment $\phi_l(\mathbf{u}) \in [0, 1]$, the energy functional in Equation 3.8 becomes convex. This is still non-smooth due to the presence of the $|\cdot|_1$ norm which inhibits the use of standard gradient descent techniques. Techniques commonly used in convex optimisation can however be used to convert this functional to an alternate form over which standard techniques can then be used, in this thesis we utilise the Legendre-Fenchel (LF) Transformation [52, 57, 58] to perform the conversion which we detail in the following section.

The LF transform is a method, similar to the Laplace and Fourier methods, that provides an alternate view of the information in a function $\mathbf{E}(\mathbf{r})$ via its derivatives $\mathbf{q} = \frac{d\mathbf{E}(r)}{dr}$. To obtain this transformation, we first consider the equation of a line:

$$\mathbf{y} = m\mathbf{r} + b \tag{3.17}$$

If this line is tangent to $\mathbf{E}(\mathbf{r})$ at r^* Equation 3.17 can be restated as:

$$\begin{aligned} \mathbf{E}(r^*) &= qr^* + b \\ -b &= qr^* - \mathbf{E}(r^*) \end{aligned} \tag{3.18}$$

where b, the y-intercept, represents the value of the tangent when $r = 0$ and $q = \frac{d\mathbf{E}(r^*)}{dr}$. If $\mathbf{E}(\mathbf{r})$ is a smooth and convex function, there is a unique correspondence between the function $\mathbf{E}(\mathbf{r})$ and its derivatives $\mathbf{q}$ thus each value of the y-intercept

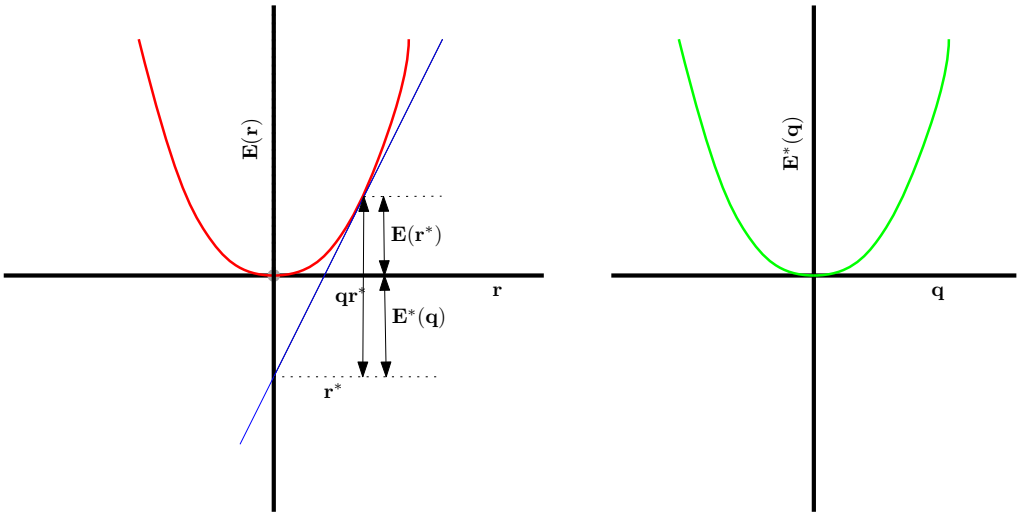


Figure 3.10: Figure showing how the Legendre transformation encapsulates the information of a function $\mathbf{E}(\mathbf{r})$ (left) through its derivatives $\mathbf{q}$ into a dual $\mathbf{E}^*(\mathbf{q})$ (right). Given a tangent line(blue) at one point to the function as shown, through geometry its slope (q) multiplied by its adjacent side (r^*) is equal to its opposite side $qr^* = \mathbf{E}(r^*)+\mathbf{E}^*(q)$ which can be extended to give the transform $\mathbf{E}^*(\mathbf{q}) =< \mathbf{q}, \mathbf{r} > -\mathbf{E}(\mathbf{r})$ and the corresponding dual function (right).

is unique. In this way the information of a function is encapsulated through its derivatives.

By extending Equation 3.18 to all the lines tangent to $\mathbf{E}(\mathbf{r})$ we arrive at the Legendre transformation as can be seen in the illustrative example in Figure 3.10:

$$\mathbf{E}^*(\mathbf{q}) =< \mathbf{q}, \mathbf{r} > -\mathbf{E}(\mathbf{r}) \tag{3.19}$$

Where $< \cdot, \cdot >$ is the inner product operator and $\mathbf{E}^*(\mathbf{q})$ is referred to as the *dual* and offers a parametrised representation of the primal $\mathbf{E}(\mathbf{r})$. We can see that this transformation is bijective $(\mathbf{r},\ \mathbf{E}(\mathbf{r})) \iff (\mathbf{q},\ \mathbf{E}^*(\mathbf{q}))$ i.e. it can be reversed to obtain the original function $\mathbf{E}(\mathbf{r})$ which then allows for the function to be approached simultaneously from two viewpoints: a *primal* $(\mathbf{E}(\mathbf{r}))$ and *dual* function $(\mathbf{E}^*(\mathbf{q}))$.

$$\begin{aligned}\mathbf{E}^{**}(\mathbf{r}) &=< \mathbf{r}, \mathbf{q} > -\mathbf{E}^*(\mathbf{q}) \\ &=< \mathbf{r}, \mathbf{q} > -(< \mathbf{q}, \mathbf{r} > -\mathbf{E}(\mathbf{r})) \\ &= \mathbf{E}(\mathbf{r})\end{aligned} \tag{3.20}$$

While this transformation in itself is useful it is restricted to convex and smooth functions whereby a unique correspondence between the function and its derivatives are maintained. An extension of this transformation, the Legendre-Fenchel (LF) transformation or convex conjugate, was thus proposed to extend this to non-smooth and non-convex functions:

$$\mathbf{E}^*(\mathbf{q}) = \arg\sup_{r\in\mathbb{R}^n}\{< \mathbf{q}, \mathbf{r} > -\mathbf{E}(\mathbf{r})\} \tag{3.21}$$

This in essence finds the point $\mathbf{r}$ on $\mathbf{E}(\mathbf{r})$ that maximises the y-intercept of a line that passes through the point $(\mathbf{r}, \mathbf{E}(\mathbf{r}))$ which is geometrically equivalent to computing the tangent line at $\mathbf{r}$ [58]. In this way points on the function $\mathbf{E}(\mathbf{r})$ can be transformed into slopes on the function $\mathbf{E}^*(\mathbf{q})$, and vice versa.

Of particular importance to the work in this thesis, is that the LF transform of the *dual* is always convex and smooth even when the original *primal* was not. This allows for the minimisation of non-smooth functions such as those seen in Equation 3.8. Additionally this formulation allows for the optimisation of objective functions from two different viewpoints: either from a *primal* problem standpoint $\mathbf{r}$ or the *dual* problem standpoint $\mathbf{q}$.

For instance, given an objective function such as that in Equation 3.9 the minimisation can be formulated as below:

$$\arg\min_{r\in\mathbb{R}^n}\{\mathbf{E}(\mathbf{r})\} = \arg\min_{r\in\mathbb{R}^n}\{\mathbf{E}(\mathbf{u},\mathbf{r})_{data} + \mathbf{E}(\mathbf{r})_{reg}\} \tag{3.22}$$

Using the LF transform we have that

$$\mathbf{E}(\mathbf{r})_{reg} = \arg\sup_{q\in\mathbb{R}^n}\{< \mathbf{r}, \mathbf{q} > -\mathbf{E}^*(\mathbf{q})_{reg}\} \tag{3.23}$$

which can be replaced in equation 3.22 to give

$$\arg\min_{r\in\mathbb{R}^n}\mathbf{E}(\mathbf{r}) = \arg\min_{r\in\mathbb{R}^n}\arg\sup_{q\in\mathbb{R}^n}\{< \mathbf{r}, \mathbf{q} > -\mathbf{E}^*(\mathbf{q})_{reg} + \mathbf{E}(\mathbf{u},\mathbf{r})_{data}\} \tag{3.24}$$

$$= \arg\min_{r\in\mathbb{R}^n}\arg\max_{q\in\mathbb{R}^n}\{< \mathbf{r}, \mathbf{q} > -\mathbf{E}^*(\mathbf{q})_{reg} + \mathbf{E}(\mathbf{u},\mathbf{r})_{data}\} \tag{3.25}$$

Under the assumptions in convex analysis the order of the min and max can be switched giving:

$$\begin{aligned}\arg\min_{r\in\mathbb{R}^n}\mathbf{E}(\mathbf{r}) &= \arg\max_{q\in\mathbb{R}^n}\arg\min_{r\in\mathbb{R}^n}\{<\mathbf{r},\mathbf{q}> -\mathbf{E}^*(\mathbf{q})_{reg}+\mathbf{E}(\mathbf{u},\mathbf{r})_{data}\}\\ &= \arg\max_{q\in\mathbb{R}^n}\{\arg\min_{r\in\mathbb{R}^n}\{<\mathbf{r},\mathbf{q}> +\mathbf{E}(\mathbf{u},\mathbf{r})_{data}\}-\mathbf{E}^*(\mathbf{q})_{reg}\}\end{aligned} \tag{3.26}$$

We also have that

$$\begin{aligned}\arg\max_{q\in\mathbb{R}^n}\{<\mathbf{r},\mathbf{q}> -\mathbf{E}(\mathbf{u},\mathbf{r})_{data}\} &= \mathbf{E}^*(\mathbf{u},\mathbf{q})_{data}\\ \arg\min_{q\in\mathbb{R}^n}\{-<\mathbf{r},\mathbf{q}> +\mathbf{E}(\mathbf{u},\mathbf{r})_{data}\} &= -\mathbf{E}^*(\mathbf{u},\mathbf{q})_{data}\\ \arg\min_{q\in\mathbb{R}^n}\{-<\mathbf{r},-\mathbf{q}> +\mathbf{E}(\mathbf{u},\mathbf{r})_{data}\} &= -\mathbf{E}^*(\mathbf{u},-\mathbf{q})_{data}\\ \arg\min_{q\in\mathbb{R}^n}\{<\mathbf{r},\mathbf{q}> +\mathbf{E}(\mathbf{u},\mathbf{r})_{data}\} &= -\mathbf{E}^*(\mathbf{u},-\mathbf{q})_{data}\end{aligned} \tag{3.27}$$

and replacing this in Equation 3.28

$$\arg\min_{r\in\mathbb{R}^n}\{\mathbf{E}(\mathbf{u},\mathbf{r})_{data}+\mathbf{E}(\mathbf{r})_{reg}\} = \arg\max_{q\in\mathbb{R}^n}\{-\mathbf{E}^*(\mathbf{u},-\mathbf{q})_{data}-\mathbf{E}^*(\mathbf{q})_{reg}\} \tag{3.28}$$

From which it can be seen that minimising the *primal* problem is equivalent, for convex functions, to maximising the *dual* for convex problems with both of these converging to the same solution [58]. This is critical for the minimisation of the non-smooth energy functions that we are interested in in this thesis.

Returning to our original energy functions in Equation 3.8 the non-smooth energy term can be rewritten below as:

$$\mathbf{E}_{smoothness} = \int_\Omega |\nabla_{\mathcal{N}}\phi_l(\mathbf{u})|_1 d\Omega \tag{3.29}$$

It must be noted that this is the well-established Total Variational (TV) norm that has been commonly used in various image segmentation, depth estimation and image smoothing problems though to the authors' knowledge is the first time this is being applied in the context of geometric multi-model fitting. The *primal* $\nabla\phi_l(\mathbf{u})$ can thus be restated in terms of its *dual* function $\Psi_l(\mathbf{u})$ using the LF transform in Equation 3.21 as

$$|\nabla\phi_l(\mathbf{u})|_1 = \arg\sup_{\Psi_l(\mathbf{u})}\{<\Psi_l(\mathbf{u}),\nabla\phi_l(\mathbf{u})> -\mathbf{E}^*(\Psi_l(\mathbf{u}))\} \tag{3.30}$$

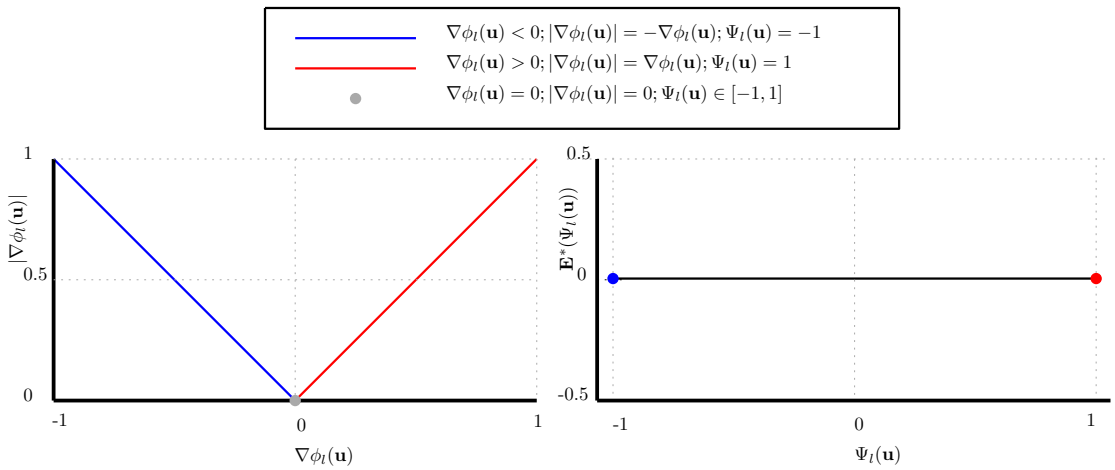


Figure 3.11: Figure illustrating how the LF transform can transform a convex but non-smooth *primal* variable $|\nabla\phi_l(\mathbf{u})|$(left) to its convex, smooth *dual* $\Psi_l(\mathbf{u})$ (right). It can be seen that for $(\nabla\phi_l(\mathbf{u}) < 0)$ and $(\nabla\phi_l(\mathbf{u}) > 0)$ the primal is smooth and can be differentiated revealing slopes of $\Psi_l(\mathbf{u}) = -1$ and $\Psi_l(\mathbf{u}) = 1$ respectively. At $\nabla\phi_l(\mathbf{u}) = 0$ the primal cannot be differentiated and its gradient must be estimated, this is through a range of acceptable slopes $\Psi_l(\mathbf{u}) \in [-1, 1]$. Transforming the non-smooth *primal* to a smooth *dual* formulation allows for the use of standard gradient based optimisation techniques on this problem.

with $\mathbf{E}^*(\Psi_l(\mathbf{u}))$ the convex conjugate of $\Psi_l(\mathbf{u})$. Which using the reversibility property of the LF transform can be rewritten as:

$$\mathbf{E}^*(\Psi_l(\mathbf{u})) = \underset{\Psi_l(\mathbf{u})}{\arg\sup}\{< \nabla\phi_l(\mathbf{u}), \Psi_l(\mathbf{u}) > -|\nabla\phi_l(\mathbf{u})|_p)\} \tag{3.31}$$

As the LF transform maps points on the *primal* to slopes on the *dual* and vice versa, for the smoothness energy in 3.29, its slope $\Psi_l(\mathbf{u})$ is constant for $\nabla\phi_l(\mathbf{u}) < 0; \Psi_l(\mathbf{u}) = -1$ and $\nabla\phi_l(\mathbf{u}) > 0; \Psi_l(\mathbf{u}) = 1$ as can be seen in the simple illustrative example in Figure 3.11. However, this energy is non-differentiable at point $\nabla\phi_l(\mathbf{u}) = 0$ and the slope must be estimated. At this point the slope could take a range of values $\Psi_l(\mathbf{u}) \in [-1, 1]$ with any slopes outside the range invalid.

Following this example 3.31 can be restated on a per-element basis as:

$$\begin{aligned}\mathbf{E}^*(\Psi_l(\mathbf{u})) &= \arg\sup_{\Psi_l(\mathbf{u})}\left(\sum_l \begin{cases}\nabla\phi_l(u)\Psi_l(u) - \nabla\phi_l(u) & \text{if } \nabla\phi_l(u) \geq 0\\ \nabla\phi_l(u)\Psi_l(u) + \nabla\phi_l(u) & \text{if } \nabla\phi_l(u) < 0\end{cases}\right)\\ &= \arg\sup_{\Psi_l(\mathbf{u})}\left(\sum_l \begin{cases}\nabla\phi_l(u)(\Psi_l(u) - 1) & \text{if } \nabla\phi_l(u) \geq 0\\ \nabla\phi_l(u)(\Psi_l(u) + 1) & \text{if } \nabla\phi_l(u) < 0\end{cases}\right)\end{aligned} \tag{3.32}$$

With the restriction of $\Psi_l(\mathbf{u}) \in [-1, 1]$ it can be seen that the two conditions in 3.32 are always less than or equal to zero. Therefore, the supremum is always at $\nabla\phi_l(\mathbf{u}) = 0$. The original supremum problem in Equation 3.31 may now be rewritten as:

$$\begin{aligned}\mathbf{E}^*(\Psi_l(\mathbf{u})) &= \underset{|\Psi_l(\mathbf{u})|\leq 1\forall\ l}{\arg\max}\ \{< \nabla\phi_l(\mathbf{u}), \Psi_l(\mathbf{u}) > -|\nabla\phi_l(\mathbf{u})|_1)\}\\ &= 0 \quad \text{if } |\Psi_l(\mathbf{u})| \leq 1\ \forall\ l\end{aligned} \tag{3.33}$$

Allowing a re-write of the smoothness energy 3.29 as

$$\int_\Omega |\nabla\phi_l(\mathbf{u})|_1 d\Omega \ = \ \max_{\Psi_l(\mathbf{u})} \int_\Omega < \nabla\phi_l(\mathbf{u}), \Psi_l(\mathbf{u}) > d\Omega \tag{3.34}$$

$$s.t.\ |\Psi_l(\mathbf{u})|_1 \leq 1 \tag{3.35}$$

Briefly neglecting the label costs, the energy in Equation 3.8 can be formulated as:

$$\begin{aligned}\mathbf{E}(\phi) = &\max_{\Psi_l(\mathbf{u})} \sum_{l=1}^{L} \int_\Omega \rho_l(\mathbf{u}, \phi_l(\mathbf{u})) + \lambda\omega_n < \nabla\phi_l(\mathbf{u}), \Psi_l(\mathbf{u}) > d\Omega\\ &s.t. |\Psi_l(\mathbf{u})|_1 \leq 1\\ &s.t. \sum_{l=1}^{L} \phi_l(\mathbf{u}) = 1\end{aligned} \tag{3.36}$$

A feature of the LF transform is that minimising of the *primal* is equivalent to maximising the *dual*. Therefore the minimisation of energy 3.36 is a saddle-point problem. The optimal values of $\phi(\cdot)$ and $\Psi(\cdot)$ are obtained from the first-order primal-dual optimisation shown in Algorithm 3. Which shows the steps of the primal-dual optimisation that minimises the data and smoothness energy terms. Where τ, α are the step size parameters of the algorithm, with θ controlling the relaxation. The values of these are determined using the diagonal preconditioning scheme [59].

Algorithm 3: Primal-Dual Optimisation

Initialisation;
$\tau, \alpha > 0, \quad \theta \in [0, 1]$;
$\phi^0 = \bar{\phi}^0 = \Psi^0 = 0$;
while $k < N$ **do**
  Dual Step;
  $\Psi^{k+1} = \pi_\Psi(\Psi^k + \tau \nabla \bar{\phi}^k)$;
  Primal Step;
  $\phi^{k+1} = \pi_\phi(\phi^k - \alpha(\rho_l(\mathbf{u}, \phi^k) + \lambda \nabla^T \Psi^{k+1})$;
  Relaxation step;
  $\bar{\phi}^{k+1} = \phi^{k+1} + \theta(\phi^{k+1} - \phi^k)$
end

To fulfil the first constraint in Equation 3.36, the gradient ascent step of the dual variable gets projected onto the feasible set $\Psi_l \in [-1, 1]$ with the projection, $\pi_\Psi(\cdot)$, defined by:

$$\pi_\Psi(\Psi) = \frac{\Psi}{\max(1, |\Psi|_1)} \tag{3.37}$$

Similarly, the function $\pi_\phi(\cdot)$ projects the gradient descent step of the primal variable onto the simplex $\phi_l(\mathbf{u}) \in [0, 1] \quad | \quad \sum_{l=1}^{L} \phi_l(\mathbf{u}) = 1$ fulfilling the second constraint in Equation 3.36.

At this stage the compactness energy has not been directly optimised for, however, when spatially connected regions are considered it can be seen that minimising of the smoothness energy enforces a single model over the region. In this way redundant models can be removed as they have no support in the data leading to a more compact solution. As this does not hold for disconnected regions, extra steps need to be taken to optimise the compactness energy. One approach is to perform the compactness optimisation after the primal dual optimisation, through an extra explicit step that merges separated models with similar parameters. In the presence of noise, merging two models results in an increase in the geometric error energy. If this increase is however less than β, the global energy would still decrease and a more compact solution can thus be obtained.

Work from the combinatorial optimisation space shows that however the optimisation of the three energies simultaneously provides better outcomes as seen in

[45]. The introduction of compactness costs into a convex optimisation paradigm was first explored by [60], whereby a compactness cost prior was introduced into a Total Variational (TV) Segmentation problem. In this work an auxiliary function $\gamma_l \in [0,1]$ was introduced that indicates if a label appears in the minimised energy result. $\gamma_l = 1$ if the label appears and $\gamma_l = 0$ otherwise. Therefore the compactness energy from Equation 3.8 can be reformulated as:

$$\beta\|L\| = \beta \sum_{l}^{L} \gamma_l \tag{3.38}$$

The presence of a label is already encoded in the labelling function ϕ_l. It follows that this auxiliary variable is related to the indicator function as follows;

$$\max_{\mathbf{u}\in\Omega} \phi_l(\mathbf{u}) = |\phi_l(\mathbf{u})|_\infty = \gamma_l \tag{3.39}$$

Therefore the infinity norm of the indicator function encodes the label cost prior, and the global energy in Equation 3.8 can be rewritten as:

$$\begin{aligned}\mathbf{E}(\phi) = &\underbrace{\sum_{l=1}^{L} \int_\Omega \rho_l(\mathbf{u}, \phi_l(\mathbf{u}))d\Omega}_{\text{Geometric Error Energy}} + \underbrace{\beta|\phi_l(\mathbf{u})|_\infty}_{\text{Compactness Energy}} \\ &+\lambda \underbrace{\sum_{l=1}^{L} \int_\Omega \omega_{\mathcal{N}} < \nabla\phi_l(\mathbf{u}), \Psi_l(\mathbf{u}) > d\Omega}_{\text{Smoothness Energy}}\end{aligned} \tag{3.40}$$

$$s.t.\ |\Psi_l(\mathbf{u})|_{p*} \leq 1$$

$$s.t. \sum_{l=1}^{L} \phi_l(\mathbf{u}) = 1$$

This norm as in the case of the smoothness energy is not smooth, applying the LF transform reveals its *dual* $\Lambda_l(\mathbf{u})$ where:

$$\max_{\mathbf{u}\in\Omega} \phi_l(\mathbf{u}) = \arg\sup_{\Lambda_l(\mathbf{u})} \{< \Lambda_l(\mathbf{u}), \phi_l(\mathbf{u}) > -\mathbf{E}^*(\Lambda_l(\mathbf{u}))\} \tag{3.41}$$

with $\mathbf{E}^*(\Lambda_l(\mathbf{u}))$ the convex conjugate of $\Lambda_l(\mathbf{u})$. Which when using the reversibility property of the LF transform can be rewritten as :

$$\mathbf{E}^*(\Lambda_l(\mathbf{u})) = \arg\sup_{\phi_l(\mathbf{u})} \{< \Lambda_l(\mathbf{u}), \phi_l(\mathbf{u}) > - \max_{\mathbf{u}\in\Omega} \phi_l(\mathbf{u})\} \tag{3.42}$$

and in a per-element basis as:

$$\mathbf{E}^*(\Lambda_l(\mathbf{u}) = \arg\sup_{\phi_l(\mathbf{u})} \sum_{l} (\Lambda_l(u)\phi_l(u) - \max_{\mathbf{u}\in\Omega} \phi_l(u)) \tag{3.43}$$

Algorithm 4: Primal-Dual Optimisation With Label Prior

Initialisation;
$\tau, \alpha, \nu > 0, \quad \theta \in [0,1]$;
$\phi^0 = \bar{\phi}^0 = \Psi^0 = \Lambda^0 = 0$;
while $k < N$ **do**
  Dual Step;
  $\Psi^{k+1} = \pi_\Psi(\Psi^k + \tau\lambda\nabla\bar{\phi}^k)$;
  $\Lambda^{k+1} = \pi_\Lambda(\Lambda^k + \nu\beta\bar{\phi}^k)$;
  Primal Step;
  $\phi^{k+1} = \pi_\phi(\phi^k - \alpha(\rho_l(\mathbf{u}, \phi^k) + \lambda\nabla^T\Psi^{k+1} + \beta\Lambda^{k+1}(\mathbf{u}))$;
  Relaxation step;
  $\bar{\phi}^{k+1} = \phi^{k+1} + \theta(\phi^{k+1} - \phi^k)$
end

Given that $\phi_l(\mathbf{u})$ is restricted to $[0,1]$ it follows that $\max_{\mathbf{u}\in\Omega}\phi_l(\mathbf{u}) \in [0,1]$ and correspondingly $\Lambda_l(\mathbf{u}) \in [0,1]$. Therefore it can be clearly seen that Equation 3.43 is always less than or equal to zero. Thus we can restate the LF transform of the compactness energy as:

$$\begin{aligned} &|\phi_l(\mathbf{u})|_\infty = \max_{\Lambda_l(\mathbf{u})} < \Lambda_l(\mathbf{u}), \phi_l(\mathbf{u}) > \\ &s.t. \int_\Omega \Lambda_l(\mathbf{u}) = 1 \quad \Lambda_l(\mathbf{u}) \geq 0, \ \forall \mathbf{u} \in \Omega \end{aligned} \tag{3.44}$$

From this Equation 3.40 can be rewritten as:

$$\begin{aligned} \mathbf{E}(\phi) = &\max_{\Psi_l(\mathbf{u}),\Lambda_l(\mathbf{u})} \sum_{l=1}^{L} \int_\Omega \Big(\rho_l(\mathbf{u}, \phi_l(\mathbf{u})) + \beta < \Lambda_l(\mathbf{u}), \phi_l(\mathbf{u}) > \Big) d\Omega \\ &+ \max_{\Psi_l(\mathbf{u}),\Lambda_l(\mathbf{u})} \sum_{l=1}^{L} \int_\Omega \lambda\omega_n < \nabla\phi_l(\mathbf{u}), \Psi_l(\mathbf{u}) > d\Omega \\ &s.t. |\Psi_l(\mathbf{u})|_1 \leq 1 \\ &s.t. \sum_{l=1}^{L} \phi_l(\mathbf{u}) = 1 \\ &s.t. \int_\Omega \Lambda_l(\mathbf{u}) = 1 \quad \Lambda_l(\mathbf{u}) \geq 0, \ \forall \mathbf{u} \in \Omega \end{aligned} \tag{3.45}$$

An equivalent minimisation to that presented in Algorithm 3 can thus be formulated as in Algorithm 4. Where $\pi_\Lambda(\cdot)$ projects the dual of the compactness energy onto the simplex $\Lambda_l(\mathbf{u}) \in [0,1] \ \mid \ \int_\Omega \Lambda_l(\mathbf{u}) = 1$ fulfilling the third constraint in Equation 3.45.

After the minimisation, a one-to-one correspondence between indicator functions and models is needed to obtain the optimal labelling. To this end, the continuous indicator function $\phi_l(\mathbf{u})$ is thresholded by selecting the maximum value of the indicator function at each data point.

In this section we have presented two different optimisation strategies for energy based multi-model fitting based on the choice to either keep a binary indicator function or relax the indicator function and thus convexify the energy. In the following section, a comparison of these two different strategies as well as justification for the proposed CORAL approach is provided.

3.2.2 Comparison of Discrete and Convex Energy Minimisation

Energy-based objective functions offer an accurate method for geometric multi-model fitting only if the energy can be minimised efficiently. In the following section we compare some of the pertinent properties of the two different energy minimisation strategies that affect its use for geometric multi-model fitting. Similar comparisons have been seen in various other domains such as image segmentation where the survey by Nieuwenhuis et al. [61] compared the performance of combinatorial and convex optimisation strategies on the energy minimisation task. For all comparison in this section the open-source implementation of α-expansion available in [62] was used together with an implementation of CORAL.

Energy Minimisation Performance

For geometric multi-model fitting, α-expansion splits the multi-label problem into a series of binary problems whereby each data point is given the choice to remain with its current label or switch to the label α. This makes it sensitive to the order of which the binary problems are solved as well as the initial conditions of the problem. In contrast, CORAL performs an optimisation over all the labels simultaneously and only introduces ambiguities after the optimisation when the solution of the relaxed problem is binarised making it more robust to label order and initial conditions as seen in Figure 3.12.

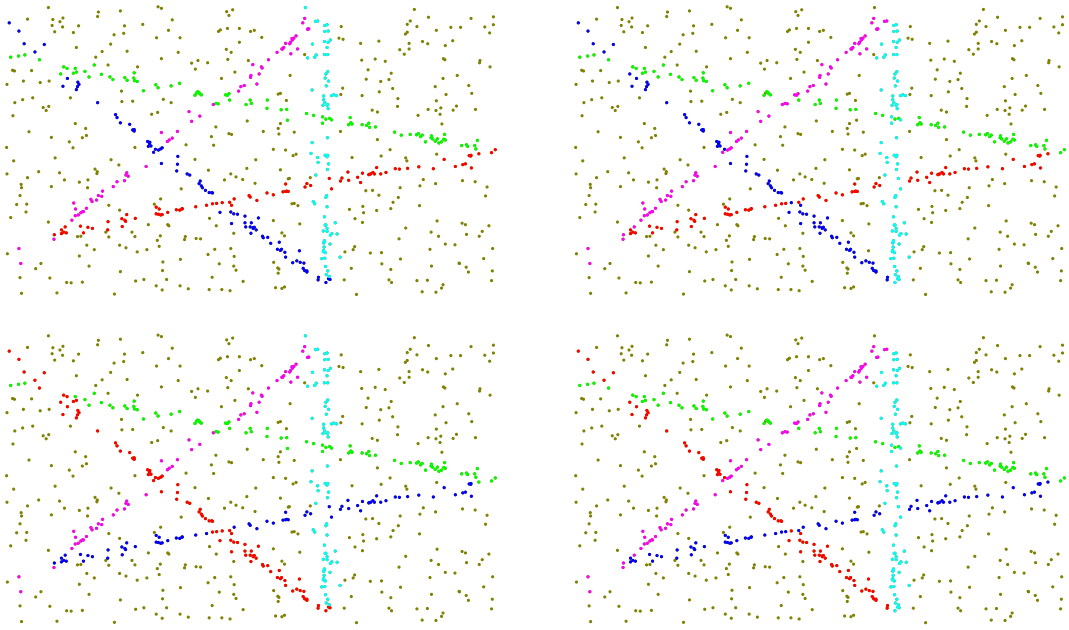


Figure 3.12: Figure illustrating the effect of the label order on the results of the geometric multi-model fitting by PEARL and CORAL. As α-expansion splits the energy minimisation into a series of binary problems it is sensitive to the label order as can be seen in the left column where points at the intersection of models have their label changed according to the label order, CORAL by performing the optimisation jointly is robust to this as seen in the right column.

This sensitivity is translated to an, on average, higher energy when α-expansion converges as compared to CORAL which is illustrated in Figure 3.13. This in turn translates to a decreased labelling performance in PEARL, which uses α-expansion for the energy minimisation, as compared to CORAL.

Run-Time Performance

The next factor considered is the speed or run-time of the energy optimisation, as the resolution of data points being considered increases this becomes more and more important.

For α-expansion, each label is given a choice as to whether it should keep its current label. While the computation of the binary choice is very efficient through graph-cuts, this is still a sequential algorithm as graph cuts cannot be parallelised easily. Goldschlager [63] posits that max-flow problems, or min-cut problems(graph

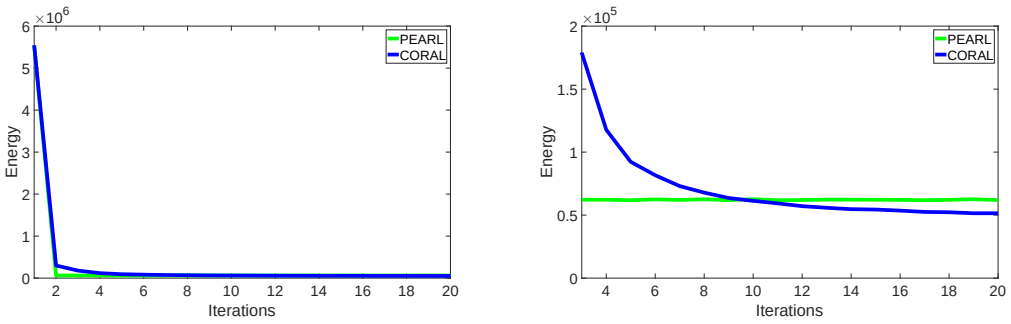


Figure 3.13: Energy evolution of the combinatorial and convex optimisation algorithms for geometrical multi-model fitting in a simulation experiment with different initialisations that are then averaged. It can be seen that both of the approaches successfully minimise the energy given, with the combinatorial approach converging faster (left). However, when the final energy is examined closer, it can be seen that the convex approach returns a lower energy. This is as α-expansion is not robust to changes in initialisations which results in a higher energy and consequently slightly worse labelling performance than CORAL.

cuts), lie in the P-complete complexity class of problems, which are not efficiently parallelisable. This is as augmenting path methods that aim to increment the flow are interdependent, as different augmenting paths share edges. Additionally, the updates on the edges have to be carried out simultaneously in each augmentation operation. Therefore, as the number of data points increases so does the time taken to perform the energy minimisation.

In addition, the number of max-flow problems needed to be solved depends on the specific multi-model fitting problem and the chosen label order. The number of augmentation steps in turn also depend on the specific graph-structure and the current labelling. This means that even on similar multi-model fitting problems with the same amount of models and data points, there can be a huge variance between the run-times.

In comparison, CORAL operates based on a first-order Primal-Dual (PD) optimisation which performs per-point evaluations making it perfectly suited for parallelisation on General Purpose Graphical Processing Units (GPGPU) hardware. Additionally, CORAL performs the same computations for every data point, thus its variance over different instances of the similar multi-model problems is lower.

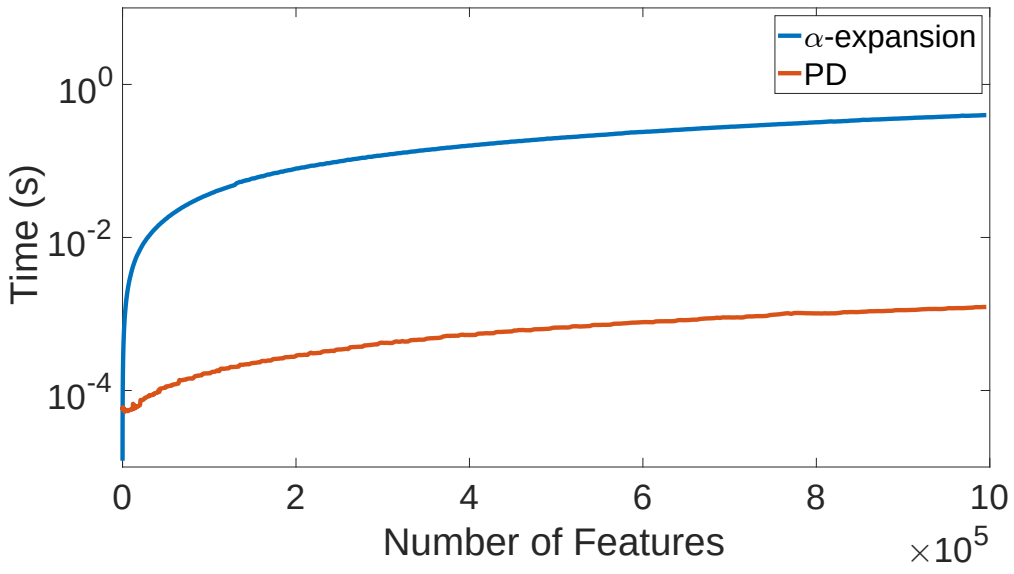


Figure 3.14: Comparison of the run-time per iteration of α-expansion and that of the primal-dual optimisation (PD) in an experiment with four models. The times are averaged over 100 runs on a machine with a 12 core Intel i7-5930K 3.50GHz CPU and a Nvidia GeForce GTX Titan Black 6048MB GPU. On this architecture, as the number of features increases so does the difference between the run times per iteration of the two methods. With the PD running at speeds two orders of magnitude (100x) faster than the α-expansion.

In Figure 3.14 a comparison of the run-time per iteration of α-expansion[2] and a Compute Unified Device Architecture (CUDA) implementation of the PD optimisation on dedicated hardware is shown. It can be seen that there is a clear reduction in run-time per iteration, of up to two orders of magnitude, through the PD optimisation as the number of features increases.

Parameters

While the α-expansion approach does not require any numerical tuning parameters, the parameters for the primal dual optimisation must be chosen based on some expert knowledge of the problem. Some optimal step-sizes α, τ can be computed automatically [59] but this only holds for a subset of problems making the use of the primal dual optimisation as a black-box algorithm less straight-forward. The addition of the regularisation for compactness into the convex optimisation as

[2]Code obtained from [62]

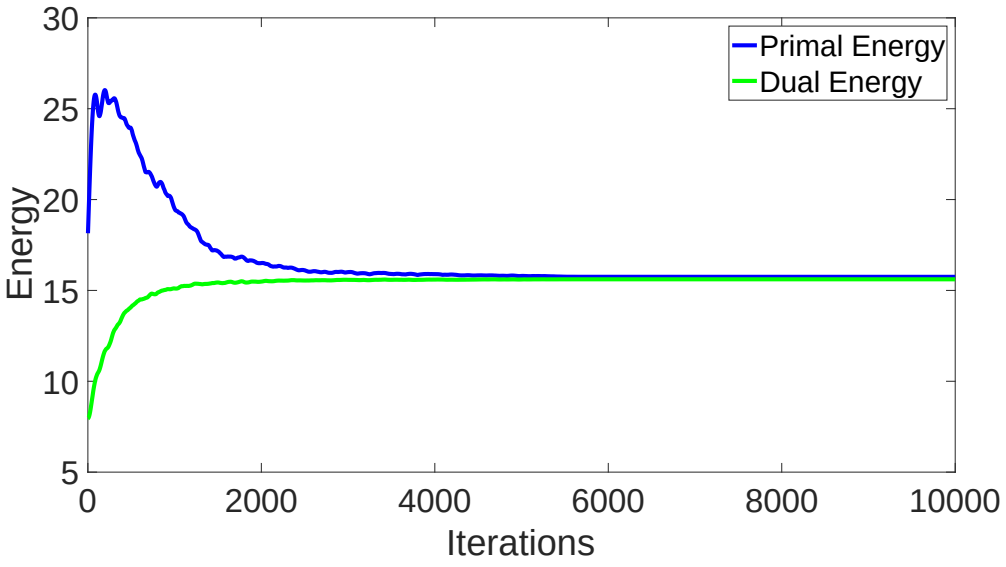


Figure 3.15: Convergence analysis of the primal dual algorithm in Algorithm 3. The optimisation is considered converged when the primal(blue)-dual(green) gap goes to zero.

in Algorithm 4 introduces an extra parameter ν that must also be tuned for limiting its general use.

Termination criterion

In general the termination criteria for the primal dual optimisation is not as well-defined as that of α-expansion as we are solving a saddle point problem. Ideally this convergence criterion is given by the primal dual gap, which gives the difference between the primal and dual energy of the optimisation. When this gap goes to zero then convergence is achieved. The computation of this is however not always straight forward. Taking the optimisation in Algorithm 3 the primal and dual energies can be calculated as below and is illustrated in Figure 3.15.

$$\begin{aligned}\mathbf{E}(\phi) &= \underbrace{\sum_{l=1}^{L}\int_{\Omega}\rho_l(\mathbf{u},\phi_l(\mathbf{u}))+\lambda|\nabla_{\mathcal{N}}\phi_l(\mathbf{u})|d\Omega}_{\text{Primal Energy}} \\ \mathbf{E}(\Psi) &= \underbrace{\int_{\Omega}\min_{l}(\rho_l(\mathbf{u})+\nabla_{\mathcal{N}}^{t}\Psi_l(\mathbf{u}))d\Omega}_{\text{Dual Energy}}\end{aligned} \tag{3.46}$$

For the convex optimisation in Algorithm 4 the calculation of the primal dual gap becomes more involved. In this case a different criterion can be used whereby

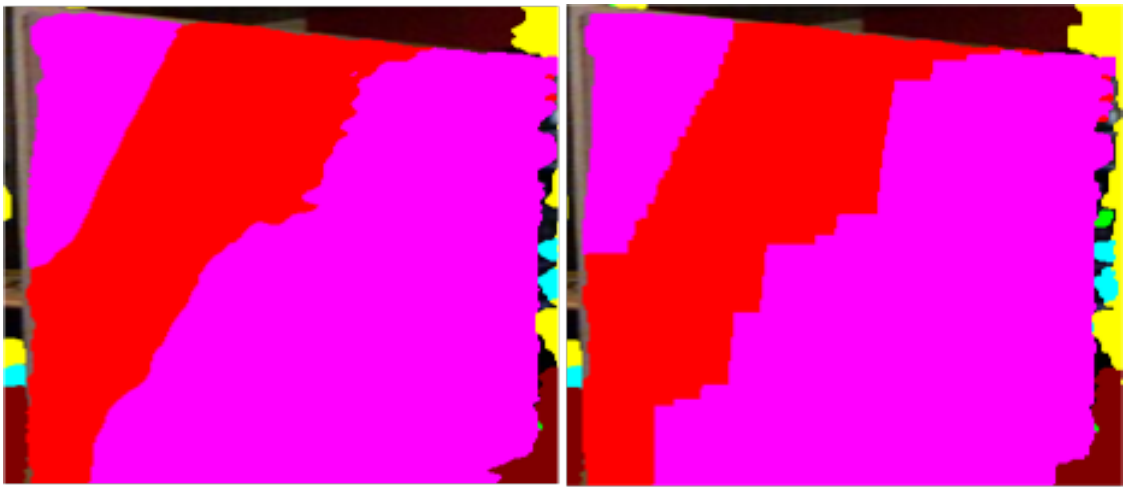


Figure 3.16: Intermediate energy minimisation result showcasing the metrication errors when the Manhattan norm is used in discrete algorithms such as α-expansion as compared to CORAL in a plane-fitting experiment from dense RGBD data. In discrete algorithms the Manhattan norm preserves edges aligned with the axes leading to a stair-case effect that is observed when dealing with dense data, whereas CORAL due to its continuous formulation is still able to return a smooth boundary.

the change in the relaxed solution between subsequent iterations is queried. If the norm of $\phi(\mathbf{u})$ between two iterations falls below a certain threshold the energy can be considered converged.

Metrication errors

Discrete combinatorial approaches such as α-expansion exhibit metrication errors as the Manhattan norm used to provide smooth object boundaries preserves edges aligned with the axes leading to a stair-case effect. CORAL on the other hand, as it employs a continuous formulation does not suffer from these discretisation effects even when the $|\cdot|_1$ norm is used as shown in Figure 3.16. In this figure, an intermediate result in a plane-fitting experiment with dense RGBD data is shown. The Manhattan norm used in PEARL can be seen to preserve edges aligned with the axes exhibiting a prominent grid bias while a smoother boundary is seen in CORAL. Moreover, the flexibility of CORAL allows for smoother norms such as $|\cdot|_2$ the norms to be chosen further boosting the smoothness.

Algorithm 5: Energy Minimisation On a Continuum Of Labels

propose initial models Θ_0;
repeat if *not converged* **then**
  Energy Minimisation to compute optimal labelling $\phi(\mathbf{u})$;
  Re-estimate model parameters to get $\Theta_{t+1}; t := t + 1$;
  (optional) Sample models from outliers $\hat{\Theta}; \Theta_t = \{\Theta_t, \hat{\Theta}\}$;
end

In summary, it can be seen that due to the ambiguities brought about by splitting the energy minimisation problem into a set of binary problems α-expansion can return a higher energy than the primal dual approach. CORAL additionally shows a faster run-time as the resolution of data increases due to its parallel nature. This makes it better equipped for dealing with geometric multi-model fitting problems where both accuracy and speed are required while also reducing the effects of metrication. All this however, comes at the cost of extra parameters that must be tuned and a looser definition of convergence as compared to α-expansion. In the following section, we show how this approach can be applied to actual geometric multi-model fitting problems.

3.2.3 Energy Minimisation On A Continuum of Labels

For geometric multi-model fitting problems, parameters of an unknown number of models are desired from noisy data. However, there is a technical hurdle with using energy minimisation algorithms for multi-model fitting. In such applications each label represents a specific model, including its parameter values, and the set of all labels L is a continuum. In line fitting, for example, $L = \mathbb{R}^2$. Practically speaking, however, the energy minimisation algorithms presented in Section 3.2.1 require a finite set L of labels (models) for their evaluation.

An option to explore the continuum of parameters is to propose a large set of models Θ_0 through Minimum Sample Sets (MSS) as used in RANSAC [27]. With the underlying logic that with a large enough model set, model parameters that approximate the true models will be present and the incorrect models will be regularised out by a combination of the smoothness and label cost energies.

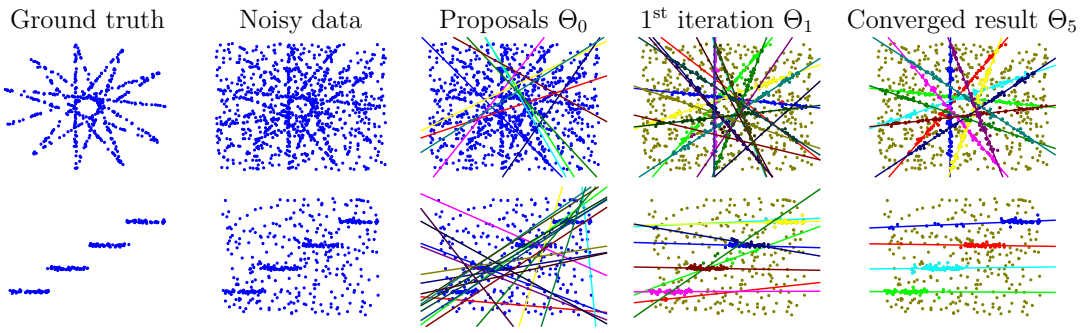


Figure 3.17: Illustrative example of energy minimisation for geometric multi-model fitting. From noisy data and a set of model proposals from random sampling, energy minimisations algorithms are able to converge on the correct underlying models through an iterative labelling and re-estimation process even in the presence of noise.

An iterative scheme over which the models are continuously improved however results in better results as shown by Isack and Boykov [43] in PEARL. The initial set of models can be improved by re-estimating its parameters based on the data points labels. It can be seen that this re-estimation step results in a further energy minimisation as the geometric error energy shown in Equation 3.4 reduces due to tighter parametrisation. We also include an optional step to sample additional models $\hat{\Theta}$ from the outliers, in this way the model set can be increased to mitigate any errors if a model is missed in the initial model set Θ_0. This gives an augmented model set $\Theta_t = \{\Theta_t, \hat{\Theta}\}$.

From these new models a new labelling can be obtained. Iterating through the optimal labelling and re-estimation of models reduces the energy further and creates a sequence of re-estimated models $\Theta_0, \Theta_1, \Theta_2, \cdots$ that eventually converge on a minimum as shown in Algorithm 5. We illustrate this in Figure 3.17 through a line-fitting example.

With this iterative scheme we can thus deploy our proposed formulation on a varied set of geometric multi-model problems. In the following section we present, in one section, all the steps needed to implement CORAL.

3.2.4 Implementation

Implementation of the CORAL formulation to geometric multi-model fitting problems consists of the application of Algorithm 5 that in turn iterates through Algorithm 4. This section provides in one place all the details needed to implement this, going through the initialisation of the various variables needed for the two algorithms. To perform the geometrical multi-model fitting we follow the next steps:

Geometric Error Computation

We begin by collecting the data points of interest $\mathbf{u} = \{\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_{N_p}\}$, where N_p is the number of data points. Then an initial set of models $\Theta_0 = \{\rho_1, \rho_2, \cdots, \rho_L\}$ is created using minimal sample sets or a sampling algorithm of choice e.g sequential RANSAC where L is the number of models. The uniform outlier model $\rho_\emptyset$ is also added to the model set such that $\Theta_0 = \{\rho_1, \rho_2, \cdots, \rho_L, \rho_\emptyset\}$.

From this we can define a data cost matrix $\Gamma(\mathbf{u})$:

$$\Gamma(\mathbf{u})_{[N_p, L+1]} = \begin{bmatrix} \rho_1(\mathbf{u}_1) & \rho_2(\mathbf{u}_1) & \cdots & \rho_L(\mathbf{u}_1) & \rho_\emptyset(\mathbf{u}_1) \\ \rho_1(\mathbf{u}_2) & \rho_2(\mathbf{u}_2) & \cdots & \rho_L(\mathbf{u}_2) & \rho_\emptyset(\mathbf{u}_1) \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \rho_1(\mathbf{u}_{N_p}) & \rho_2(\mathbf{u}_{N_p}) & \cdots & \rho_L(\mathbf{u}_{N_p}) & \rho_\emptyset(\mathbf{u}_1) \end{bmatrix}$$

Columns of this matrix correspond to the vectors $\rho_l(\mathbf{u})$, which are computed by calculating a distance between the model parameters ρ_l and the data points $\mathbf{u}$. For the outlier model a constant fidelity cost is returned, $\rho_\emptyset(\mathbf{u}) = \gamma$.

Gradient Computation

We then establish the gradient over the N_n nearest neighbours of the data points, which is encapsulated in $\nabla_\mathcal{N}$.

$$\nabla_{[(N_n * N_p), N_p]} = \begin{bmatrix} I_{p_1} & \cdots & I_{p_1 \mathcal{N}_1} & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ I_{p_1} & \cdots & I_{p_1 \mathcal{N}_n} & \cdots \\ \vdots & \vdots & \ddots & \vdots \\ \cdots & I_{p_{N_p} \mathcal{N}_1} & \cdots & I_{p_{N_p}} \\ \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & I_{p_{N_p} \mathcal{N}_{N_n}} & I_{p_{N_p}} \end{bmatrix} \tag{3.47}$$

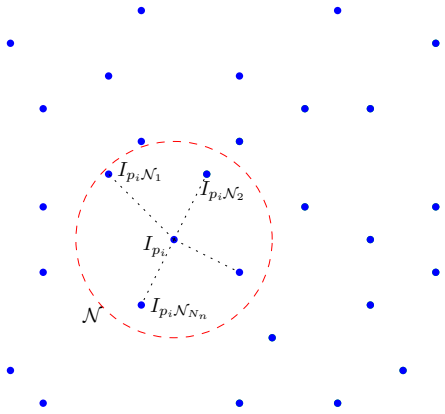


Figure 3.18: Figure illustrating the construction of the $\nabla_{\mathcal{N}}$ for i.i.d data. In this N_n nearest neighbours, the indexes of a point I_{p_i} and its neighbours are then added to $\nabla_{\mathcal{N}}$ as shown in Equation 3.47.

Each row of the matrix $\nabla_{\mathcal{N}}$ only has two non-zero terms. These correspond to the index of the current data point and that of its neighbour as illustrated in Figure 3.18. As this encodes a gradient the sum across each row must be equal to zero.

Energy Minimisation

The minimisation of the global energy in Equation 3.8 reveals an optimal assignment of data points to geometric models.

$$\min_{\phi_l(\mathbf{u})} \underbrace{\sum_{l=1}^{L} \int_{\Omega} \rho_l(\mathbf{u}, \phi_l(\mathbf{u}))d\Omega}_{\text{Geometric Error Energy}} + \lambda \underbrace{\sum_{l=1}^{L} \int_{\Omega} \omega_{\mathcal{N}} R(\nabla_{\mathcal{N}} \phi_l(\mathbf{u}))d\Omega}_{\text{Smoothness Energy}} + \underbrace{\beta \|L\|}_{\text{Compactness Energy}} \tag{3.48}$$

This is performed through the primal dual formulation of Algorithm 4. To run this, the step sizes and relaxation parameter are first initialised:

$$\tau > 0,\ \alpha > 0,\ \nu > 0, \theta \in \{0, 1\}$$

This is followed by initialisation of the regularisation parameters:

$$\lambda > 0,\ \beta > 0$$

Then the initialisation of the *dual* and *primal* variables:

$$\Psi_{[(N_n * N_p), L+1]} = \mathbf{0},\ \Lambda_{[N_p, L+1]} = \mathbf{0},\ \bar{\phi}(\mathbf{u})_{[N_p, L+1]} = \phi(\mathbf{u})_{[N_p, L+1]} = \mathbf{0}$$

Followed by the iteratively updating the *primal* and *dual* until convergence:

$$\Psi^{k+1}_{[(N_n*N_p),L+1]} = \pi_\Psi(\Psi^k + \tau\nabla\bar{\phi}^k)$$

$$\pi_\Psi(\Psi) = \frac{\Psi}{\max(1, |\Psi|)}$$

$$\Lambda^{k+1}_{[N_p,L+1]} = \pi_S(\Lambda^k + \nu\beta\bar{\phi}^k)$$

$$\phi^{k+1}_{[N_p,L+1]} = \pi_S(\phi^k - \alpha(\Gamma(\mathbf{u}) + \lambda\nabla^T\Psi^{k+1} + \beta\Lambda^{k+1}))$$

$$\bar{\phi}^{k+1} = \phi^{k+1} + \theta(\phi^{k+1} - \phi^k)$$

$\pi_s(\cdot)$ is performed through the simplex projection in Algorithm 6.

Convergence checking for the primal dual optimisation is done by via analysis of the indicator function. If the norm of the indicator function is greater than a prescribed threshold T_{PD} the primal dual algorithm has converged.

$$\text{con}_{PD} = \begin{cases} 1 & \text{if norm}(\phi_l(\mathbf{u})^k - \phi_l(\mathbf{u})^{k-1}) < T_{PD} \\ 0 & \text{otherwise} \end{cases} \tag{3.49}$$

Convergence produces the optimal assignment $\phi_l(\mathbf{u})$ after which the global energy is computed.

$$\mathbf{E}_{PD}(\mathbf{u}) = \underbrace{\sum_{l=1}^{L}\int_\Omega \rho_l(\mathbf{u}, \phi_l(\mathbf{u}))d\Omega}_{\text{Data Energy}} + \lambda\underbrace{\sum_{l=1}^{L}\int_\Omega \omega_\mathcal{N}|\nabla_\mathcal{N}\phi_l(\mathbf{u})|_1 d\Omega}_{\text{Smoothness Energy}} + \underbrace{\beta\|L\|}_{\text{Model Count Energy}} \tag{3.50}$$

Practically, this can also be run for a fixed number of iteration $N_{iterations}$.

Model Parameter Re-estimation and Resampling

To provide a unique assignment of points to models each row of the assignment function $\phi(\mathbf{u})$ is binarised. This quantises the $\phi(\mathbf{u})$ into $\{0, 1\}$ once the energy minimisation is complete. This binarised indicator function can be used to update the model parameters.

$$\hat{\rho}_l = \arg\min_{\rho_l} \sum \rho_l(\mathbf{u})\phi_l(\mathbf{u}) \tag{3.51}$$

Additionally book-keeping is done to ensure that models with no support are removed. We also include here an optional step to sample additional models $\hat{\Theta}$

Algorithm 6: Simplex projection [64]: Given a vector $\mathbf{x}(\cdot)$ of length N_p.

```
S_p Sum of positive values in x(·) ;
N_i Number of positive values in x(·);
converge = false;
while converge==false do
    S_p = 0;
    N_i = 0;
    converge = true;
    for i = 0; i < N_p; i++ do
        a = x[i];
        if a != 0 then
            N_i++;
            S_p += a;
        end
    end
    if N_i>0 then
        a = (S_p - 1) / N_i;
        for i = 0; i < N_p; i++ do
            if u[i] != 0 then
                u[i] -= a;
                if u[i] < 0 then
                    converge = false;
                    u[i] = 0;
                end
            end
        end
    else
        u[0] = 1;
    end
end
```

from the outliers to increase the model set. This creates an updated set of models $\Theta_i = \{\hat{\rho}_1, \hat{\rho}_2, \cdots, \hat{\rho}_{\hat{L}}, \rho_\emptyset\}$. After which the global energy can be computed as.

$$\mathbf{E}_{Re-est}(\mathbf{u}) = \underbrace{\sum_{l=1}^{\hat{L}} \int_\Omega \hat{\rho}_l(\mathbf{u}, \phi_l(\mathbf{u}))d\Omega}_{\text{Geometric Error Energy}} + \lambda \underbrace{\sum_{l=1}^{\hat{L}} \int_\Omega \omega_{\mathcal{N}} |\nabla_{\mathcal{N}} \phi_l(\mathbf{u})|_1 d\Omega}_{\text{Smoothness Energy}} + \underbrace{\beta \|\hat{L}\|}_{\text{Compactness Energy}} \tag{3.52}$$

Geometric Model Convergence

We then check if Algorithm 5 has converged, which suggests that the global energy and thus the geometric models have converged to a final solution. If $\mathbf{E}_{PD}(\mathbf{u})$ and $\mathbf{E}_{Re-est}(\mathbf{u})$ are compared it can be noted that if there is no change in assignment of data points the model parameters do not change which would result in the energies remaining the same. Therefore a convergence check utilising these and a threshold T_{Re-est} can be formulated:

$$\text{con}_{PD} = \begin{cases} 1 & \text{if } (\mathbf{E}_{PD}(\mathbf{u}) - \mathbf{E}_{Re-est}(\mathbf{u})) < T_{Re-est} \\ 0 & \text{otherwise} \end{cases} \tag{3.53}$$

If convergence is not achieved further energy minimisation by returning to step 4 is required. Once convergence is reached a set of geometric models and the assignment of data points to these are returned. A flowchart showing these steps in a line-fitting application is shown in Figure 3.19. Additionally, we release Matlab code for the use of the wider community functions that can be found in Appendix 9.

3.3 Conclusions

In this chapter we presented the theory for the CORAL approach to geometric multi-model fitting through a global energy that not only considers geometric errors but spatial smoothness and compactness. In this approach a convex relaxation that enables the "soft" assignment of data points to geometric models is opted for that allows for advanced optimisation techniques in the continuous domain to be pulled in. The parallelisation potential of these techniques offer CORAL an advantage over state-of-the-art combinatorial methods that require sequential evaluation and thus suffer as the resolution of data increases. CORAL intrinsically boosts run-time performance by simultaneously handling per-point evaluations making our approach more suitable for geometric model extraction in applications with real-time performance constraints.

In summary, CORAL brings in powerful optimisation machinery into the solution of geometric multi-model fitting. It offers an algorithm that is simultaneously

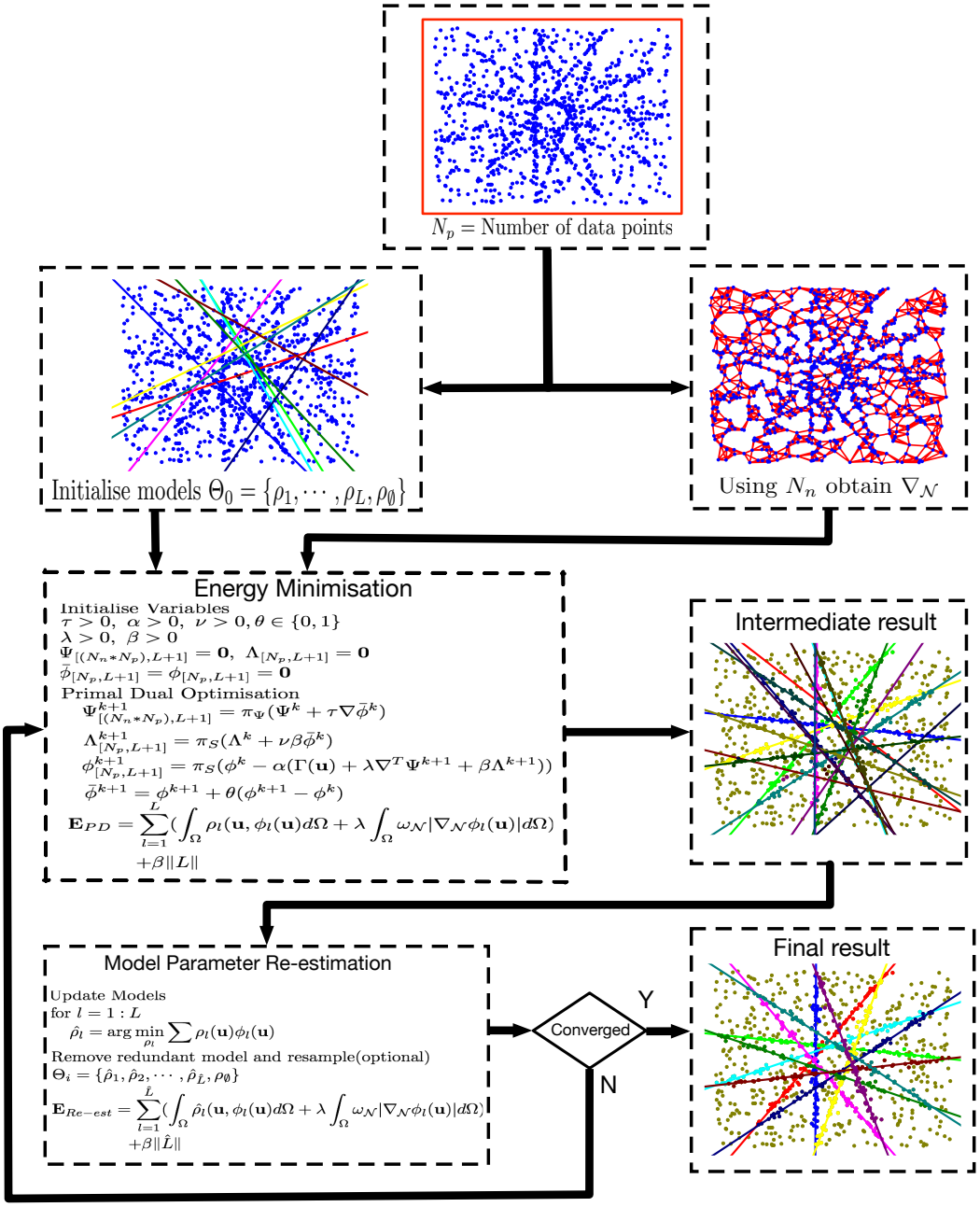


Figure 3.19: Flow-chart illustrating the use of CORAL for geometric multi-model fitting. From an initial set of N_p data points $\mathbf{u}$ supporting unknown numbers of geometric models, L models are proposed using RANSAC or minimal sample sets. From these model parameters, an initial data cost term can be retrieved. Similarly a gradient $\nabla_\mathcal{N}$ can be established through N_n nearest neighbours. This is sufficient to compute the energy in Equation 3.8 which is then minimised through Algorithm 4 to obtain an intermediate result after which the models are updated. Algorithm 5 suggests an iterative scheme of energy minimisation and model parameter estimation which continues until energy convergence when $(\mathbf{E}_{PD} - \mathbf{E}_{Re-est}) < T_{Re-est}$ giving the final result at the bottom right.

able to robustly extract accurate models in the presence of contamination and improved time performance guarantees over the state of the art. In the next chapter we qualitatively and quantitatively evaluate our CORAL formulation against state-of-the-art methods before presenting applications of CORAL in different contemporary problems in robotics.

Kinolewacho, hukata.

That which is sharpened will cut.

Swahili Proverb

4

CORAL Evaluation and Benchmarking

Abstract

In this chapter the CORAL formulation described in the previous chapter is evaluated through two differently formulated geometric multi-model fitting problems. These are firstly in multi-homography detection from sparse features matched in two camera views, followed by plane detection in dense RGBD images. The performance of CORAL is compared with different state-of-the-art multi-model fitting algorithms on the AdelaideRmf [65] and NYU[66] datasets in which ground truth is available. The performance statistics over these two problems show that CORAL is able to perform multi-model fitting swiftly and accurately on these different scenarios with results that are as good or better than the state-of-the-art even as noise and clutter in the data increases.

Contents

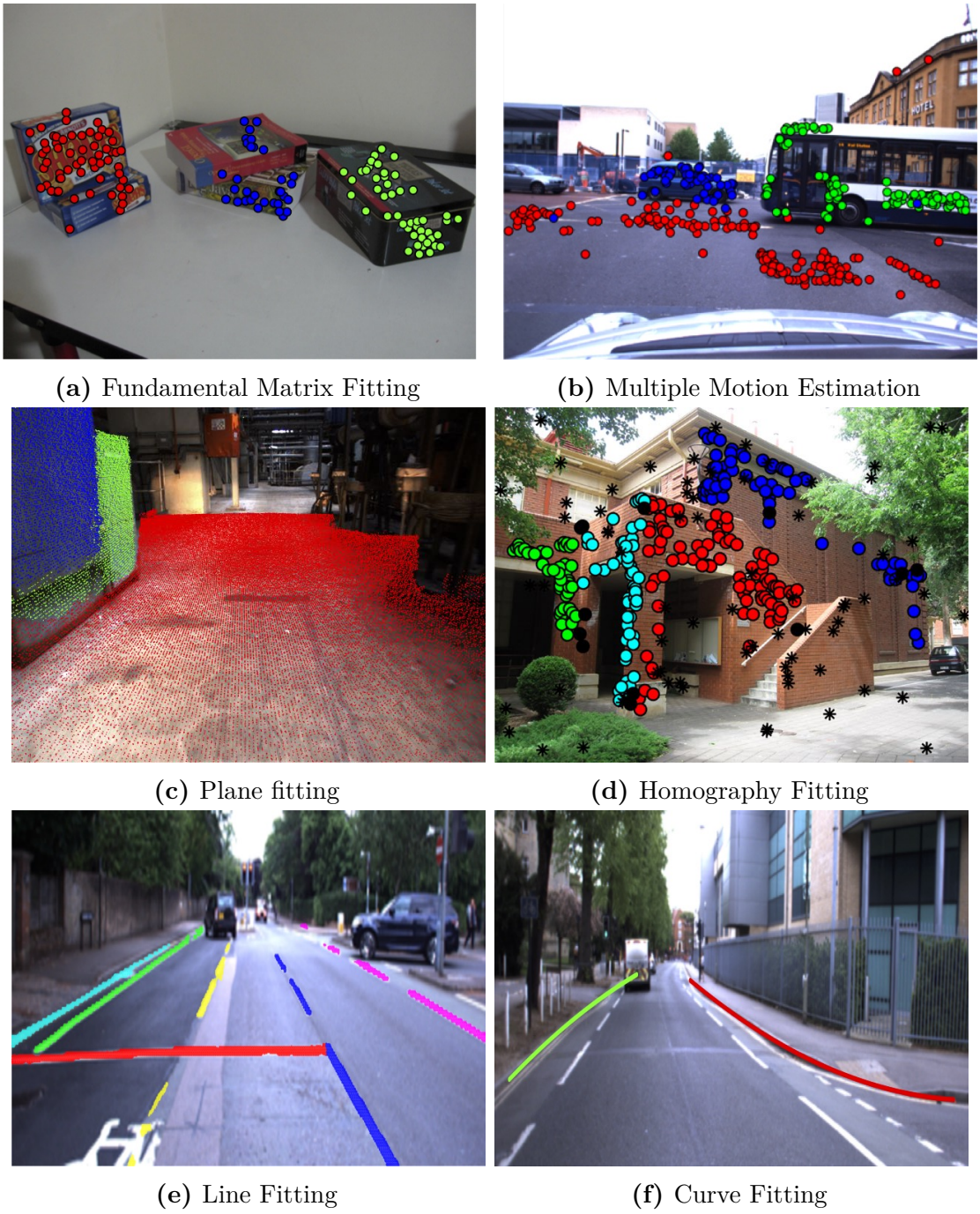
4.1 Two-view Multi-Homography Estimation 60
4.1.1 Simulation Environment 62
4.1.2 AdelaideRmf 67
4.2 Plane detection with RGBD images 68
4.3 Conclusions 71

This chapter is concerned with the performance evaluation of CORAL in various multi-model fitting problems. While CORAL is a general approach to multi-model fitting that is agnostic to a single particular application as demonstrated in Figure 4.1 we constrain the experiments in this chapter to a series of different plane fitting applications from image data. This is motivated by the fact that detection of geometric structure from images is of widespread importance to many applications in computer vision such as camera calibration, camera motion estimation and surface reconstruction. As such many multi-model fitting techniques have been benchmarked on this problem.

We evaluate the plane fitting through two different applications: the first corresponds to the multi-homography estimation from two views while the second is plane detection from a single RGBD image. In both cases we make the implicit assumption that urban scenes whether indoor or outdoor consist mostly of man-made objects (e.g. buildings, walls, screens, desks, etc.) which are by construction piece-wise planar. The image setup choice in the experiments, which is whether or not to use RGBD sensors, is informed by the particular environment. Indoor environments usually expose texture-less surfaces with low-light conditions that favour the use of RGBD sensors. In contrast, outdoor scenes are abundant in texture but the limited range of RGBD sensors makes them unsuitable for these environments. Matching features across two views followed by multi-homography estimation is a better choice for this situation. With these assumptions and different operating contexts we showcase the performance and versatility of CORAL as compared to state-of-the-art multi-model fitting algorithms.

4.1 Two-view Multi-Homography Estimation

The first application we examine considers two views of a static scene that contains multiple planes. We use the notation $(x, y, 1)$ to represent the homogeneous coordinates of a pixel of interest. Given a sparse set of n pixel correspondences between the two views $\mathbf{u}_i = (\mathbf{u}_i^1, \mathbf{u}_i^2), \mathbf{u}_i^1, \mathbf{u}_i^2 \in \mathbb{R}^2,\ i = 1 \cdots n$ where $\mathbf{u}_i = (x_i, y_i, 1)$,



(a) Fundamental Matrix Fitting

(b) Multiple Motion Estimation

(c) Plane fitting

(d) Homography Fitting

(e) Line Fitting

(f) Curve Fitting

Figure 4.1: Figure showing a sample of the geometric models that can be obtained through the CORAL formulation.

the homography $\mathbf{H}^{21} \in \mathbb{R}^{3\times3}$ establishes the mapping of pixels from the first view to the second view through an observed plane. This operation is denoted by $\mathbf{u}_i^2 = \mathbf{H}^{21}\mathbf{u}_i^1$. We aim to find the classification of pixel matches w.r.t several homographies while simultaneously rejecting outliers.

The geometric error measure between a pixel match $\mathbf{u}$ and a homography $\mathbf{H}^{21}$ can be given through the symmetric transfer error in Equation 4.1.

$$\frac{1}{2}(\|\mathbf{u}^2 - \mathbf{H}^{21}\mathbf{u}^1\| + \|\mathbf{u}^1 - \mathbf{H}^{12}\mathbf{u}^2\|) \tag{4.1}$$

where $\mathbf{H}^{12}$ is the inverse of $\mathbf{H}^{21}$. In this work we choose to enrich this error measure by introducing the propagated covariance Σ_{12} induced by the corresponding homography. This gives the Mahalanobis distance in Equation 4.2.

$$\|D(\mathbf{u}_i, \mathbf{H}^{ab})\|_{\Sigma_{ab}} = (\mathbf{u}_i^a - \mathbf{H}^{ab}\mathbf{u}_i^b)\Sigma_{ab}^{-1}(\mathbf{u}_i^a - \mathbf{H}^{ab}\mathbf{u}_i^b)^T \tag{4.2}$$

With these defined, the specific version of the energy functional in Equation 3.8 to be minimised for this multi-homography energy can be written as below:

$$\underbrace{\sum_{l=1}^{L}\frac{1}{2}\sum_{i=1}^{n}(\|D(\mathbf{u}_i, \mathbf{H}_l^{12})\|_{\Sigma_{12}} + \|D(\mathbf{u}_i, \mathbf{H}_l^{21})\|_{\Sigma_{21}})\phi_l(\mathbf{u})}_{\text{Geometric Error Energy}}$$
$$+ \underbrace{\lambda\sum_{l=1}^{L}\sum_{i=1}^{n}\omega_{\mathcal{N}}|\nabla_{\mathcal{N}}\phi_l(\mathbf{u})|_{1,1}}_{\text{Smoothness Energy}} + \underbrace{\beta\|L\|}_{\text{Compactness Energy}} \tag{4.3}$$

In the following sections, the two environments used to benchmark performance on the multi-model fitting problem are presented. This begins with a custom simulation environment created to mimic sections of indoor and outdoor scenes before proceeding to the AdelaideRmf dataset [65] that has been extensively used to benchmark multi-model fitting algorithms.

4.1.1 Simulation Environment

To characterise the performance and robustness of CORAL with respect to existing algorithms, experiments were first run in a controlled simulation environment. A simulation environment consisting of three planes, placed mutually orthogonally to each other was created. This configuration resembles, for instance, the end of a corridor or corner of a building whereby two walls and the ground are simultaneously observed as seen in Figure 4.2. Uniform sampling of the planes

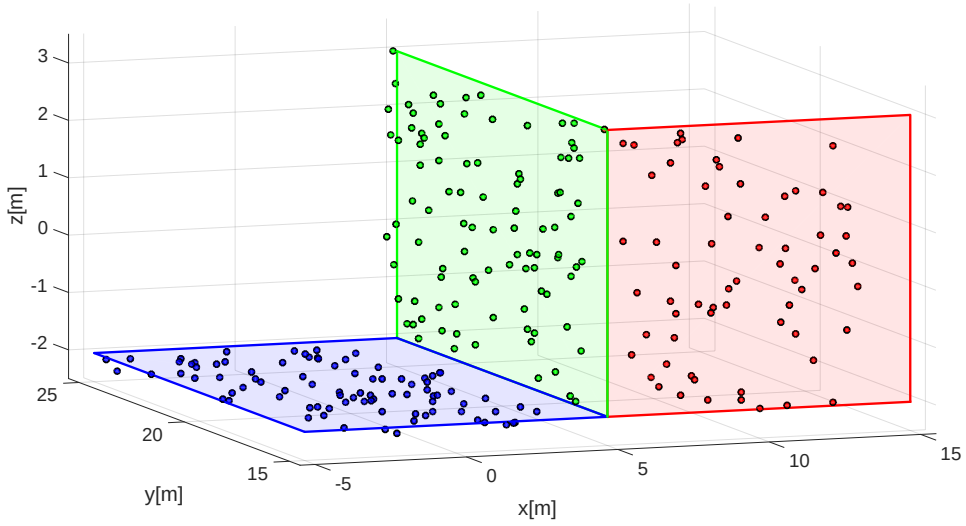


Figure 4.2: The simulated environment created to compare the performance of multi-model fitting algorithms for multi-homography fitting. This environment consisting of three mutually orthogonal planes mirrors indoor and outdoor scenes of the end of a corridor or corner of a building respectively. Observation of the scene by a camera from different views reveals the pixel correspondences needed for homography fitting.

creates the point features, which were then observed by a camera with associated noise σ_{pixel} from two frames.

In addition to points directly sampled from planes, outliers were added at different percentages to the simulation. To do so, a uniform sampling of the scene space is performed. When viewed from the camera these points accounted for pixels that do not originate from a plane creating clutter in the scene.

Our first evaluation tested CORAL under different values of σ_{pixel} in the absence of any outliers. This was compared to a implementation of sequential RANSAC and the open-source implementations of PEARL [62] and T-Linkage [67]. For the rest of this chapter we use RANSAC and sequential RANSAC interchangeably. The initial models used in T-Linkage, PEARL and CORAL were given by minimal sample sets whose parameters were obtained through the Direct Linear Transform (DLT) algorithm [7]. After evaluation under different values of σ_{pixel} in the absence of any outliers, we then injected outliers at different

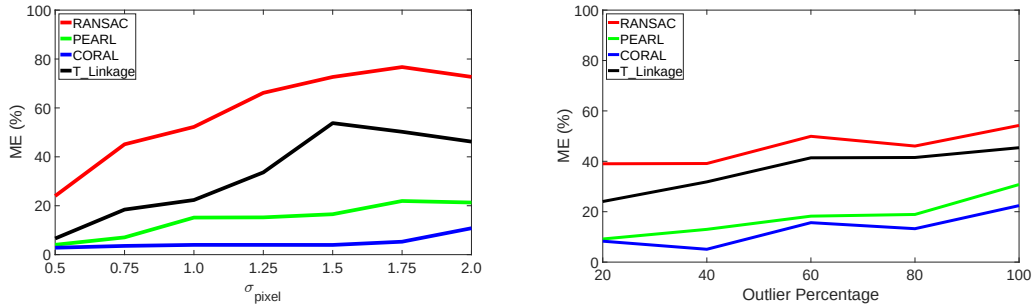


Figure 4.3: Misclassification error against sensor noise σ_{pixel} firstly in the absence of outliers (left) and then with a steady increase in outlier percentage for a fixed value of σ_{pixel}. The errors are obtained for RANSAC, PEARL, CORAL and T-Linkage. The ME is evaluated and averaged over ten different two-view pairs of the simulated environment.

percentages while keeping σ_{pixel} constant.

As a metric the proportion of wrongly classified points which we refer to as the misclassification error (ME) was used.

$$\textbf{ME} = \frac{\text{Number of Misclassified points}}{\text{Number of points}} \tag{4.4}$$

Results from these experiments are shown in Figure 4.3 where it can be seen that the global energy approaches (CORAL & PEARL) are more robust to increases in noise and clutter (outliers) as compared to RANSAC and the clustering-based T-Linkage approach. The higher ME reported by RANSAC in the presence of noise can be explained by analysing Figure 4.4. The table shows the triangulated noisy points in the simulation environment for $\sigma_{pixel} = [0.5, 1.0, 1.5]$ together with the ground truth planes from which they were originally sampled from. Colour coding of membership to a particular model is used to show the results of the multi-homography extraction for RANSAC, PEARL and CORAL.

For $\sigma_{pixel} = 1.5$, RANSAC selects models that, though geometrically valid, are not consistent with the ground truth planes. See for example, the combination of green and red points on the two vertical planes. The energy approaches are more robust to this kind of noise as the smoothness prior ensures convergence to a better solution consistent with the ground truth. This is shown in the last column of Figure 4.4. The lower ME reported by CORAL than PEARL can

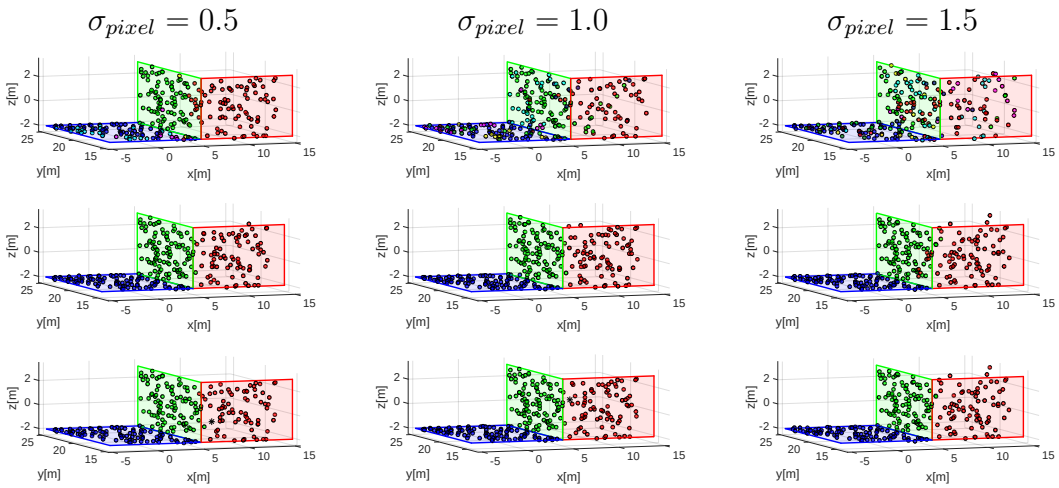


Figure 4.4: The triangulated positions of noisy pixels together with the ground truth planes (red, green and blue patches) are shown under different values of σ_{pixel}. With the membership to the different homographies indicated by the different colours. RANSAC, PEARL and CORAL results are shown in the first, second and third row respectively.

be explained by the differences between combinatorial and convex optimisation schemes for energy minimisation as was explained in 3.2.2 with ambiguities in the combinatorial energy minimisation resulting in higher energies being output that result in a higher ME in this case.

The choice of the regularisation parameters, λ and β determines the extent to which the smoothness and compactness regularisation affect the final solution of CORAL and are thus intrinsically linked to the performance. If these values are too low no regularisation takes place, while if these values get too large the geometric errors are not considered as can be seen in Figure 4.5. It can be seen from the size of the valley that there is still a range of values that give good regularisation performance, suggesting that for similar experiments these values need not be individually tuned.

Additionally the number of pixel correspondences in the simulation was varied to test the running-time of each of the energy based algorithms. In Figure 4.6 the run-times for the energy minimisation of PEARL and CORAL were then compared. The parallel implementation of CORAL results in a faster run-time as the number of features increases as compared to the sequential PEARL implementation. As the

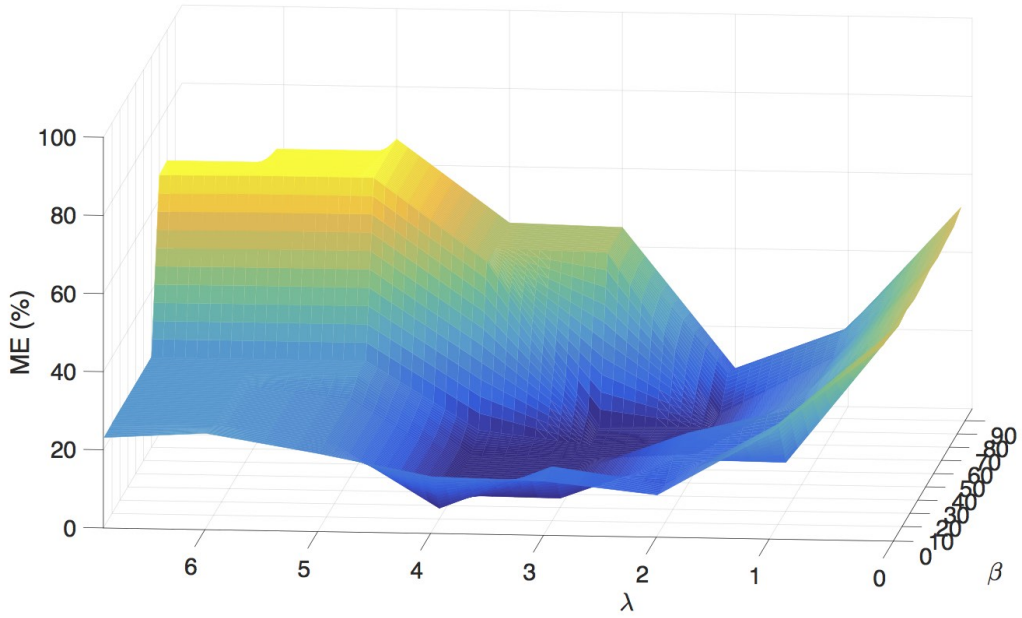


Figure 4.5: Figure illustrating the effect of the regularisation parameters λ and β on the ME averaged over ten different experiments in the simulated environment. Low values of these parameters show no regularisation while as these values get too large they over-power the geometric errors resulting in a higher ME as an over-smooth and compact solution will be found. Optimal values lie in the valley between these two, with the size of it showing that there is a range of these parameters that still give good performance.

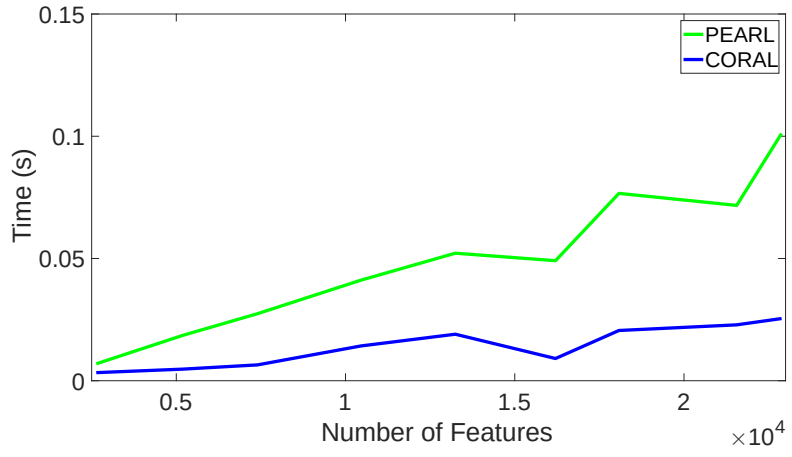


Figure 4.6: Time taken for the energy minimisation algorithms as the number of features increased in the multi-homography fitting simulation experiment. For the CORAL implementation a Nvidia GeForce GT 750M 2048MB GPU was used and for PEARL a 2.5 GHz Intel Core i7.

number of features increases so does the comparative difference in run-times, where it can be seen for twenty thousand features CORAL is up to four times faster.

4.1.2 AdelaideRmf

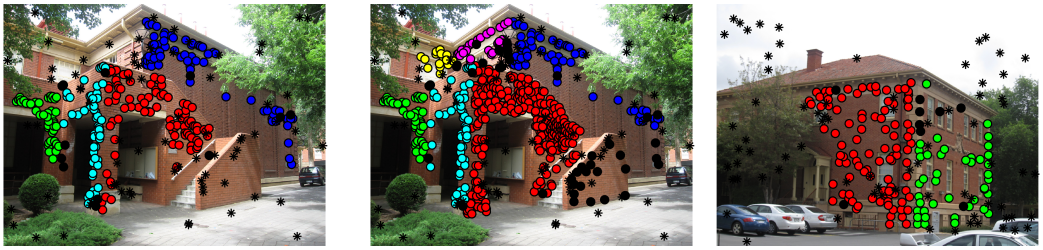
With the performance and robustness of the proposed approach verified in simulation, we benchmarked our result against the state-of-the-art on real imagery. The AdelaideRmf dataset [65] is used in this evaluation. It consists of image pairs with extracted keypoints and manually labelled ground truth. The performance of different multiple model fitting approaches on this dataset is available in [42]. From these results, we carried out a comparison of CORAL to T-linkage [39], J-linkage [38], RPA [40], SA-RCM [41], Grdy-RansaCov and ILP-RansaCov [42]. These results are shown in Table 4.1.

From Table 4.1 it can be seen that the best mean performance is achieved using our algorithm, with few cases sharing the best performance. The better median performance reported by Multi-H [68] can be explained by its specialised initialisation for homographies using affine transformations that have been shown to be superior to the DLT. This approach then subsequently uses a PEARL Formulation for energy minimisation. The proposed approach would also benefit from this specialised initialisation; however, for better comparison with the state of the art an initialisation based on the DLT for model generation is retained.

A subset of the images with the detected homographies are shown in Figure 4.7, with the membership to different models color-coded. From this we can qualitatively see that the proposed formulation is able to accurately deal with the varied range of models present in a scene.

Table 4.1: Misclassification error for two view plane segmentation in the AdelaideRmf dataset [65]. From these results it can be seen that CORAL achieves the best mean performance over the images in this dataset with a competitive median result also seen when compared to various state-of-the-art multi-model fitting alogrithms. Part of the results in this table are reported in [42].

	J-Lnkg	T-Lnkg	RPA	SA-RCM	Grdy-RansaCov	ILP-RansaCov	Multi-H	**CORAL**
mean	25.50	24.66	17.20	28.30	26.85	12.91	4.40	**4.2117**
median	24.48	24.53	17.78	29.40	28.77	12.34	**2.41**	3.48



(a) Johnsona **(b)** Johnsonb **(c)** ladysymon

Figure 4.7: Sample of images from the AdelaideRmf Dataset. With the membership to the different homographies indicated by the different colours. Crosses are used to signify points that are mislabelled while the uniform outlier label $\emptyset$ is shown in black through all figures.

4.2 Plane detection with RGBD images

The second case of structure detection that we are interested in is the fitting of planes from a single RGBD image. In this scenario, we can exploit the regularity of the image pixel grid. This idea, in contrast to a 3D point cloud representation will fully leverage the effects of the smoothness regularisation term. In other words, the individual pixel solutions will propagate along the domain of the image with a more clear definition of the local neighbourhood. In order to apply this idea, we chose to work on an inverse depth representation in a per-pixel basis. It can be shown that if two pixels $\mathbf{u}$ and $\mathbf{u}^*$ belong to the same planar surface in 3D, their inverse depths $\xi(\mathbf{u})$ and $\xi(\mathbf{u}^*)$ satisfy the following equation

$$\xi(\mathbf{u}) - \xi(\mathbf{u}^*) = \langle \mathbf{w}, \mathbf{u} - \mathbf{u}^* \rangle \tag{4.5}$$

where $\langle \cdot, \cdot \rangle$ represents the inner product between two vectors. $\mathbf{w} = (w_u, w_v)$ codifies the projection of the 3D plane normals into the image plane $w_u u_u + w_v u_v = \xi(\mathbf{u})$. The specific energy to be minimised is given by:

$$\underbrace{\sum_{l=1}^{L} \int_{\Omega} (\|\xi(\mathbf{u}) - \xi(\mathbf{u}^*) - \langle \mathbf{w}, \mathbf{u} - \mathbf{u}^* \rangle\|_{\sigma_\xi}) \phi_i(\mathbf{u}) d\Omega}_{\text{Geometric Error Energy}}$$

$$+ \underbrace{\lambda \sum_{l=1}^{L} \int_{\Omega} \omega_{\mathcal{N}}^{\alpha} |\nabla_{\mathcal{N}} \phi(\mathbf{u})|_1 d\Omega}_{\text{Smoothness Energy}} + \underbrace{\beta \|L\|}_{\text{Compactness Energy}} \tag{4.6}$$

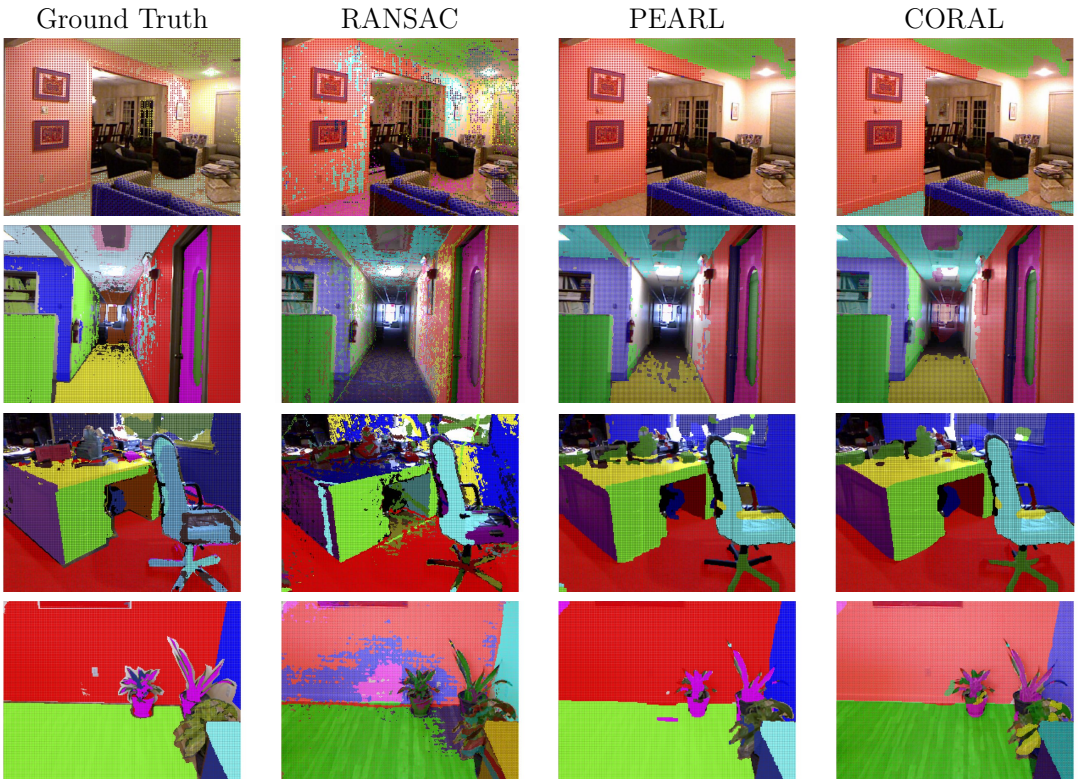


Figure 4.8: Plane segmentation on a sample of RGBD images from the NYU-Depth dataset. with the Ground Truth, RANSAC, PEARL and CORAL results presented for each image. Pixel membership to a plane model is shown by superposing on a pixel a colour-coded point that assigns it to a specific plane model.

where $\omega_{\mathcal{N}}^{\alpha} = e^{-\|\nabla I\|_{\alpha}}$ weights the smoothness term using the gradient of the corresponding image (∇I) to preserve sharp discontinuities between objects. The impact of this is controlled by parameter α.

To evaluate the performance of CORAL for plane detection using RGBD images, we use the NYU-depth dataset [66]. This contains 1449 tuples of RGB, depth and labelled images for multiple instances of objects each of resolution 420 x 560. For our evaluation, we selected a subset of 232 images containing scenes where either the walls, ceilings, desks, floor or a combination of all of these surfaces are observed. This is to ensure that our selection contains images with significant planar regions. This subset was however not explicitly created for plane extraction therefore a suitable ground truth for our problem is not available. To obtain a ground truth, we employ the labels provided and fit planes to individual instances of objects in the scene that could be planar. In addition, we merge planes with similar planar

parameters to reduce redundancy of models in the data.

Four samples of the results are shown in Figure 4.8 where each column represents the ground truth, RANSAC, PEARL and CORAL solutions respectively with a colour-coding used to show the different labels. The results show that the energy based methods produce more consistent plane models aligned with the expected ground truth. CORAL solutions show greater quality compared to PEARL solutions, in particular considering the first two images. In such cases, CORAL is able to identify more plane models than PEARL.

Table 4.2 then condenses the ME results for all three methods. It can be observed that CORAL outperforms PEARL in three of the four instances and over the whole test set.

Runtimes were also evaluated using a desktop GPU and shown in Table 4.3. CORAL runs on average around 4 times faster than PEARL on this task, even as the number of models varies.

Table 4.2: Mislabelling error for RGBD segmentation.

	RANSAC	PEARL	**CORAL**
Image 1	22.96	16.24	**13.95**
Image 2	28.70	20.59	**17.12**
Image 3	36.60	26.30	**25.30**
Image 4	15.72	**7.77**	7.83
Test set	29.38	23.04	**18.99**

Table 4.3: Time taken in seconds for the energy minimisation for PEARL and CORAL evaluated and averaged over the images shown in Figure 4.8. For the CORAL implementation a Nvidia GeForce GT 750M 2048MB GPU was used and for PEARL a 2.5 GHz Intel Core i7. The CORAL implementation was on average about four times faster even as the number of models (L) varied.

	L=4	L=8	L=16
PEARL	0.7040	2.0509	4.0199
CORAL	**0.1837**	**0.5412**	**1.1888**

4.3 Conclusions

In this chapter we evaluate and benchmark CORAL against other state-of-the-art multi-model fitting algorithms. This is through two differently structured applications which encompass plane extraction in different environments. Firstly through homography detection from sparse features matched in two camera views, followed by plane detection in dense RGBD images. In both scenarios CORAL reports performance that is as good as or better than other state-of-the-art multi-model fitting algorithms.

This chapter verifies the accuracy of CORAL on simple multi-model fitting problems however in most scenarios we are not interested in geometric models themselves. This is because geometric models are very often components of more complicated systems. In the following chapters we see how geometric models which are accurately and swiftly obtained with CORAL can be incorporated into systems for robotic mapping, perception and scene understanding.

In mathematics you don't understand things.
You just get used to them.

Johann von Neumann

5

Fast Global Labelling For Depth Map Improvement

Abstract

Dense depth map estimation techniques from cameras often struggle to accurately estimate the depth of large texture-less regions. In this chapter we present a vision-only pipeline to improve on this estimation by incorporating planar priors. In this, CORAL is employed to accurately find planar priors in a viewed scene without making any assumptions of the underlying scene layout. A fast global labelling then associates these planar priors to individual image pixels leading to more complete depth maps specifically over large, plain and planar regions that tend to dominate the urban environment.

When these improved depth maps are deployed to the creation of a vision only dense reconstruction over large scales on the KITTI dataset [69], we demonstrate reconstructions that yield significantly better results in terms of coverage (10%) while still maintaining high accuracy.

Contents

5.1 Depth Map Estimation 74
5.2 Planar Priors For Depth Map Improvement 79
5.2.1 Planar Prior Generation 83
5.2.2 Planar Prior Labelling 88

5.3 Implementation 90
5.4 Results 92
5.5 Conclusions 96

DENSE depth maps from visual sensors offer a low-cost solution for mobile robotic platforms to observe their surrounding environment. Given a particular depth map from an image, a 3D partial reconstruction of the environment can be obtained that gives the, possibly metric, location of objects in a scene as seen in Figure 5.1 at a fraction of the cost of 3D LiDAR modalities. Depth maps from passive cameras are also able to deal with the scale and lighting conditions in which many robotic platforms, such as autonomous vehicles, operate unlike active camera setups such as the Microsoft Kinect [70] that are limited to low-light indoor environments.

The quality of dense depth map estimation however suffers in the presence of large, plain and planar regions. The lack of texture in these regions gives very little information to current state-of-the-art estimation algorithms leading to noisy and sometimes erroneous depth estimation. As urban scenes where most robotic platforms operate in are generally characterised by multiple large plain surfaces, such as roads and buildings, this limits the effectiveness of depth maps for their operation where accuracy and by extension safety are critical.

In this chapter we present a pipeline to improve the dense depth estimation of urban scenes by incorporating planar priors into the estimation as seen in Figure 5.2. We employ the CORAL formulation described in Chapter 3 for fast and accurate fitting and incorporation of the planar priors into the estimation. The parallel nature of CORAL enables the proposed pipeline to run in real-time without compromising on accuracy, this makes it suited for use in robotics where speed is also tightly coupled with safety.

In Section 5.1 an overview of state-of-the-art methods, based on variational techniques, is presented. This shows the limitations when dealing with large, plain and planar surfaces. This is followed in Section 5.2 by the proposed pipeline to improve on the depth maps by obtaining and incorporating planar priors into the depth estimation using CORAL. The precise implementation is then discussed in

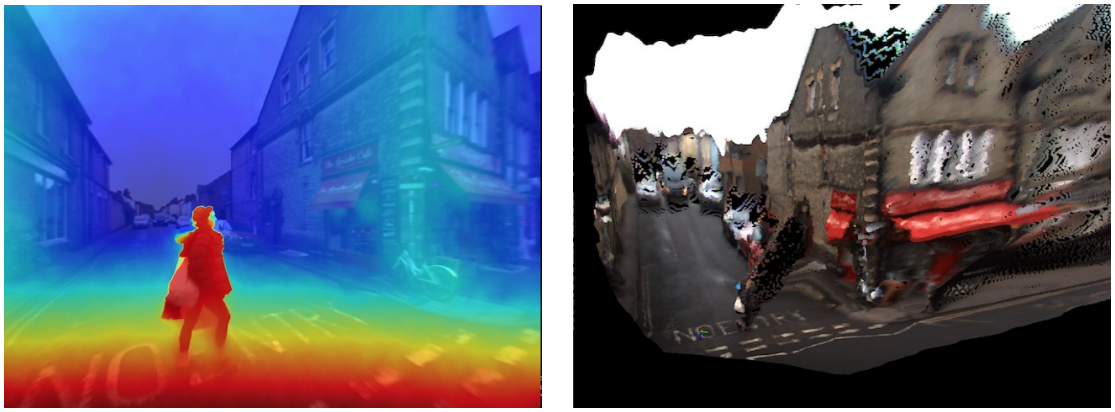


Figure 5.1: Example of a depth image (left) and its corresponding 3D reconstruction (right) from back-projecting the individual pixels using their depth measurements. The depth of the individual pixels is found by matching correspondences in multiple images followed by a subsequent optimisation. If a stereo-pair of images is used the depth obtained is also metrically valid.

more detail in Section 5.3 before results are presented and conclusions are drawn in Section 5.4 and 5.5 respectively.

5.1 Depth Map Estimation

Current state-of-the-art methods to estimate depth maps are mostly based on variational optimisation algorithms. In general these have two terms to be minimised as follows:

$$\min_{\xi} \mathbf{E}_{data}(\xi) + \mathbf{E}_{reg}(\xi) \tag{5.1}$$

For depth map estimation the data term measures the photoconsistency (over a stereo pair or a monocular sequence) of the depth estimation. In this section we focus on images obtained from a stereo camera containing a pair, left I_L and right I_R, of images. The photoconsistency term can thus be formulated as:

$$\mathbf{E}_{data}(\xi) = \iint_{\Omega} |\rho(d, x, y)| \mathrm{d}x\mathrm{d}y. \tag{5.2}$$

where (x,y) are the coordinates of a pixel in the reference image, and the function $\rho(d, x, y) = Sim^W(I_L(x + d, y), I_R(x, y))$ measures the similarity between two pixels in the stereo images using a patch window W for a candidate disparity $d \in D$.

Figure 5.2: A qualitative perspective of this chapter. A dense depth map projection (top) of a planar wall surface created by a state-of-the-art Total Generalised Variation algorithm is displayed. As expected, the algorithm results in a noisy output on the largely textureless wall and road. CORAL discovers planar regions and from there invokes a planar prior to restricted areas. This results in an improved depth map projection (bottom), showing a marked improvement in the depth estimation along the wall and road.

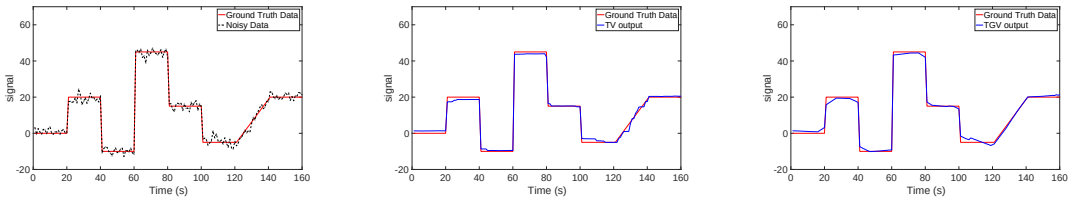


Figure 5.3: An illustrative example showing the effects of the Total Variational (TV) and Total Generalised Variational (TGV) regularisation on reconstructing a noisy signal (left). TV (middle) is able to preserve the high variation changes and reconstruct piecewise constant surfaces however in regions of constant gradient, a staircase effect is observed. The TGV term (right) also deals with the high variation changes while dealing with affine surfaces, unlike in TV piecewise constant segments are approximated by linear segments.

This is followed by a regularisation term that ideally enforces depth smoothness for homogenous surfaces while simultaneously preserving sharp discontinuities between different objects in the scene. As a scene often comprises of multiple homogenous surfaces (piecewise constant, smooth, affine) the depth estimation problem is ill-posed which makes the selection of an appropriate prior for a particular application essential. To illustrate this in more detail, a simple example probing the effect of different regularisers is presented in Figure 5.3. In this two regularisers commonly seen in the literature the Total Variational (TV) [71] and Total Generalised Variation (TGV) [72] are compared on the reconstruction of a noisy 1D signal. From this example it can be seen that the different regularisers favour different surfaces in their subsequent estimations.

The first regulariser considered is the TV which can be written as follows:

$$\mathbf{E}_{reg}(\xi) = \int_{\Omega} |\nabla \xi|_1 \tag{5.3}$$

where Ω is the domain of ξ and $\nabla\xi$ is its gradient. Regularising the TV penalises the sum of absolute infinitesimal changes over a function, encouraging piecewise constant regions as can be seen in Figure 5.3. If a region, however, has a constant gradient, it is approximated by piecewise constant regions creating a staircase effect. For depth map estimation this translates to the preservation of edges but at the same time fronto-parallel surfaces are favoured which could lead to discontinuities in surfaces at different orientations. As we have shown in the previous chapter the $|\cdot|_1$ norm

is non-smooth at its origin and therefore standard optimisation strategies cannot be used for its minimisation, however through the Legendre-Fenchel transformation described in Section 3.2.1 it can be efficiently minimised in a primal dual formulation.

To remove the staircase effect higher-order derivatives of the function need to be penalised. This is implemented in the TGV as below:

$$\mathbf{E}_{reg}(\xi) = \alpha_1 \int_\Omega |\nabla\xi - \mathbf{y}|_1 + \alpha_2 \int_\Omega |\nabla\mathbf{y}|_1 \tag{5.4}$$

where $\mathbf{y}$ is the gradient of the signal ξ. α_1 and α_2 weight the defined regularisation terms. This norm allows the signal ξ to change at a constant rate, which is what is seen in affine surfaces when Figure 5.3 is considered. Piecewise constant regions are however now approximated by linear sections. When it comes to dense depth map estimation TGV regularisation thus favours affine surfaces such as planes at any orientation.

While we desire smoothness of homogenous surfaces we also require that separation between distinct objects observed in the image persists in the depth maps. For depth map estimation this can be enforced by using the gradient of the colour image as it often follows that if there is a change in the colour image there must be a probable boundary between two distinct objects. One way of including this is through a tensor as follows:

$$\mathbf{E}_{reg}(\xi) = \alpha_1 \int_\Omega \|\mathbf{T}\nabla\xi - \mathbf{y}\|_1 + \alpha_2 \int_\Omega \|\nabla\mathbf{y}\|_1 \tag{5.5}$$

The tensor $\mathbf{T}$ is included to mitigate the tension between maintaining discontinuities between distinct objects and keeping smoothness in the energy minimisation. The tensor goes further than the appearance gradient (∇I), which simply indicates the presence of a boundary, and obtains information about the direction of these borders. This mitigates the separation of homogenous surfaces that show appearance differences due to changes in lighting i.e. those caused by shadows. The tensor can be defined as:

$$\mathbf{T} = exp(-\gamma|\nabla\mathbf{I}|^\beta)nn^T + n^\perp {n^\perp}^T \tag{5.6}$$

Figure 5.4: Comparison of three depth map regularisers: TV, TGV, and TGV-Tensor. Using the reference image (top), the TV regulariser (left) favours fronto-parallel surfaces, therefore it creates a sharp discontinuity for the shadow on the road (red rectangle) and attaches the rubbish bin to the rear of the car (red circle). TGV (centre) improves upon this by allowing planes at any orientation, but it still cannot identify boundaries between objects: the rubbish bin is again estimated as part of the car. Finally, the TGV-Tensor (right) regulariser both allows planes at any orientation and is more successful at differentiating objects by taking into account the normal of the colour image's gradient. Figure from [73].

where $n = \frac{\nabla I}{|\nabla I|}$ and $n^{\perp}$ is its orthogonal complement. $\mathbf{T}$ decomposes the disparity gradient (∇d) in directions aligned with n and $n^{\perp}$. This ensures that only components aligned with $n^{\perp}$ are penalised whereas large gradient components aligned with n, such as those appearing due to lighting changes are not penalised.

Figure 5.4 shows the effect of these three different regularisations on the depth map estimation of a scene. The TV regulariser favours fronto-parallel surfaces therefore creates discontinuities in the scene. The TGV regulariser improves on this by allowing planes at any orientation but is still unable to strongly enforce inter-object discontinuities, with the combination of the TGV and Tensor regularisation showcasing the best trade-off between inter-object smoothness and intra-object discontinuities.

Despite this strong implicit formulation, variational techniques still struggle to adequately reconstruct large, plain and planar regions. The data term becomes of little use as the lack of texture in these areas restricts the use of photoconsistency.

Figure 5.5: An example scene showcasing depth map estimation of an urban scene. From the reference image (top) it can be seen that the variational technique struggles with the large, plain and planar surfaces with the walls (bottom left, bottom right) being estimated noisily while an erroneous estimate is made for the similarly plain road (bottom middle).

Additionally, the smoothness preserving regularisers also struggle to promote smoothness from the distant barriers of these large surfaces. This results in noisy depth maps of urban scenes as can be seen in Figure 5.5. To overcome these challenges, the pipeline presented in this chapter explicitly includes the planar information into the dense depth map estimation. Leveraging the speed and accuracy of the CORAL formulation to both extract planar priors and identify pixels associated with the planes for real-time depth map improvement as is detailed in the following sections.

5.2 Planar Priors For Depth Map Improvement

In this section, a pipeline that explicitly incorporates architectural, in this case planar, priors into the estimation of dense depth maps is described. This pipeline leverages the fast and accurate planar prior discovery from CORAL to speedily create accurate depth maps for real-time use on robotic platforms such as autonomous vehicles.

There have been a variety of techniques that aim to introduce strong planar priors into dense depth map estimations with some recent approaches attempting to introduce stronger regularisation terms that enforce planarity over surfaces. Pinies et al. [74] introduced a non-local higher order regularisation term in their variational framework. This had significant improvement for large planar surfaces by allowing the propagation of depth information over distant pixels to texture-less regions of the same planar surface. However, this comes at a cost of selecting the non-local pixels which when moving from indoor environments becomes non-trivial. Concha et al. [75] similarly used a higher-order term that models Manhattan and piecewise planar structures but it requires prior estimation of the plane normals.

Looking at the field of computer vision also reveals the prevalence of geometry-only depth maps or Piecewise Planar Reconstructions (PPRs) [76–81]. These are motivated by the simplification of depth-estimation by constraining it to geometric surfaces only, where the challenges of poor texture are easily overcome through the strong planar assumption. The depth map estimation then reduces to an optimal labelling problem where each of the pixels is assigned to a particular hypothesised geometric model through a Markov Random Field (MRF) formulation.

These approaches critically neglect non-planar surfaces, which apart from indoor scenes still take up a significant amount of observed scenes. Gallup et al. [82] corrected for this by adding a non-planar model, whose depths would be obtained through a classical depth-estimation technique that creates an initial depth map to be improved. As non-planar regions tend to be high in texture, the depth estimation of these regions would not suffer in the initial depth map. The refinement of the depth map is formulated as a labelling problem that handles both planar regions through underlying geometric models and non-planar regions with the initial depth map. In this method, geometric hypotheses are obtained using RANSAC [27] over locally connected regions on the initial depth map.

While exhibiting good results, this approach relies heavily on the accuracy of the initial depth map as it is used to create the geometric hypotheses. This accuracy can not be guaranteed over larger plain textureless surfaces which hence

limited its practical use to making small corrections to medium-quality depth maps. In addition, this method used a graph-cut labelling based on α-expansion [53], limiting its real-time capabilities.

The work of [54] describes a continuous optimisation approach allowing for real-time optimal labelling. By utilising a first-order energy minimisation algorithm, this approach lends itself to a highly parallel GPU implementation. A speed-up of around 30 times is reported as compared to the discrete graph-cut approach. However, to initialise the geometric models a plane sweep algorithm is developed [83] restricting the use to layouts where the ground plane and two orthogonal facades are dominant. This is an approximation to a Manhattan world, an assumption that does not always hold leading to errors when new planar configurations are viewed. Moreover, this method does not effectively deal with non-planar regions.

In this work, an initialisation of geometric priors that does not make any assumptions about the underlying layout of the scene is opted for. This is by means of an automatic planar prior discovery solution through two-view multi-homography segmentation of the scene. To detect multiple homographies through high levels of clutter and noise is a non-trivial task and therefore the CORAL formulation presented in Chapter 3 is used in this task. The resulting homographies can be decomposed to reveal the underlying planar priors. As in [82] an extra label for non-planar models is used which is initialised with a dense depth map obtained with the Total Generalised Variation and Tensor regulariser given in Equation 5.5. Given this information the original stereo depth maps can be refined in a fast per-pixel depth labelling with parallel continuous energy minimisation.

Figure 5.6 shows an overview of the proposed depth map improvement formulation shown in this chapter. We describe the two main modules of this pipeline in the following sections, beginning with the generation of the planar priors, before the labelling is discussed.

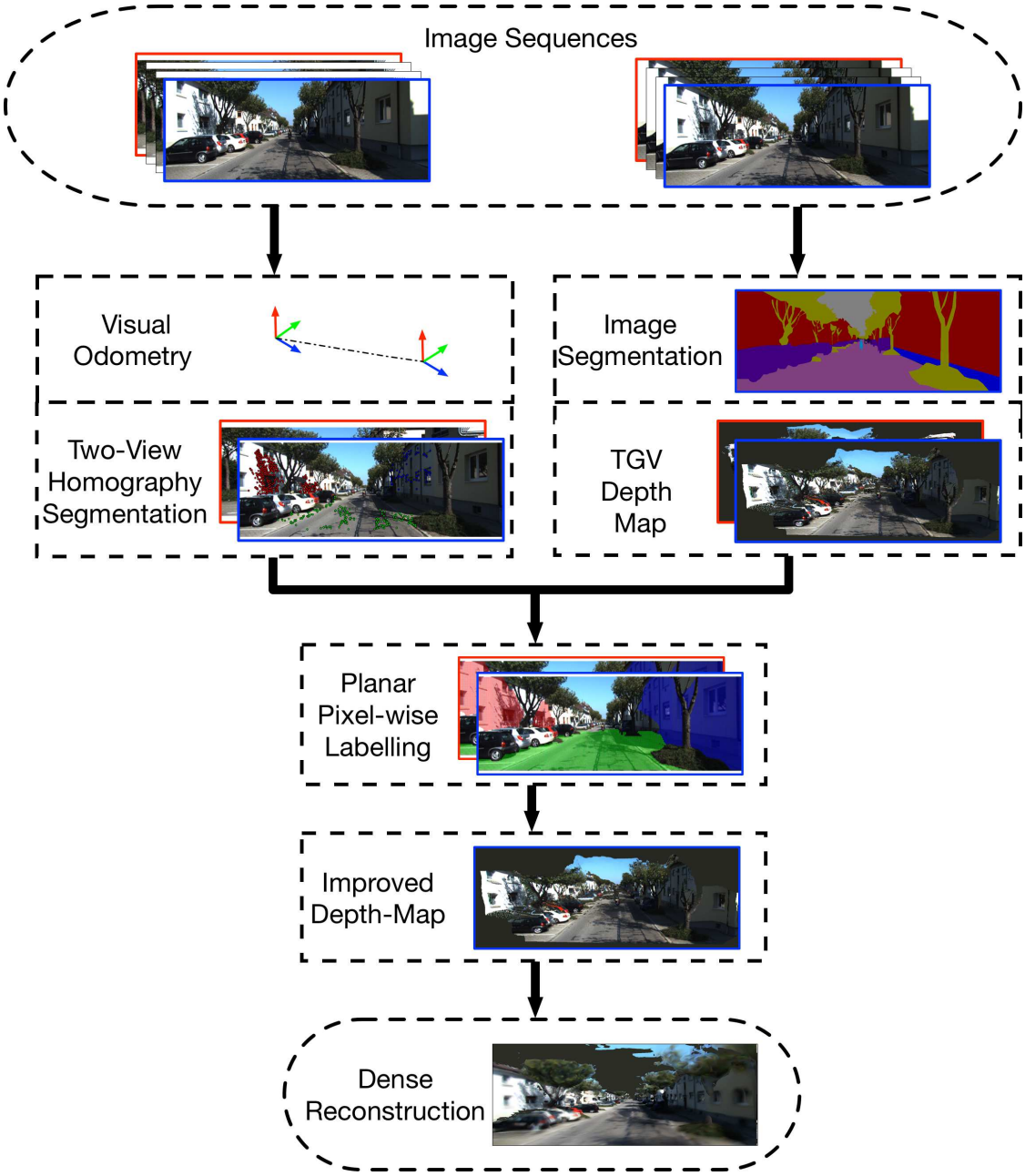


Figure 5.6: Overview of the proposed vision only depth map improvement pipeline. From image sequences (mono or stereo) a two-view homography segmentation in conjunction with visual odometry is used to create planar priors. These priors drive a fast global labelling that assigns planar regions to their corresponding prior and non-planar regions to a depth map obtained through the Total Generalised Variational (TGV) energy minimisation. From this labelling a more complete improved depth map is estimated. The fusion of consecutive improved depth maps are further fused to create a dense reconstruction.

5.2.1 Planar Prior Generation

Although planar structures in urban scenes are mainly represented by texture-less regions, we can leverage the existence of objects that can be revealed by blob detectors such as Speeded Up Robust Features (SURF) [84] to extract sparse features within these regions for their detection. Correspondences between these sparse features in two views can then reveal the underlying 3D plane using the well-established homography.

Without loss of generality, given a sparse set of n pixel correspondences in homogeneous coordinates between the two views $\mathbf{u}_i = (\mathbf{u}_i^1, \mathbf{u}_i^2) \in \mathbb{R}^2,\ i = 1 \cdots n$, a homography $\mathbf{H}^{21} \in \mathbb{R}^{3\times3}$ establishes the mapping of pixels from the first view $\mathbf{u^1}$ to the second view $\mathbf{u^2}$ through an observed plane π with normal vector $\mathbf{n}^1$ and distance d^1 [85]. With this available information, we can extract the motion between the two views $(\mathbf{R}^{21}, \mathbf{t}^{21})$.

$$\mathbf{u}^2 = \mathbf{H}^{21}\mathbf{u}^1, \tag{5.7}$$

$$\mathbf{H}^{21} = \mathbf{R}^{21} + \frac{\mathbf{t}^{21}(\mathbf{n}^1)^T}{d^1}. \tag{5.8}$$

Similar to [86] and [87] we initialise our homography fitting problem from affine transformations to achieve better performance as compared to the classical DLT [7]. For this initialisation, a non homogenous point correspondence $\mathbf{u}_i$ is augmented by the 2x2 affine matrix $\mathbf{A}$ that maps the image points surrounding $\mathbf{u}_i^1$ into those in the vicinity of $\mathbf{u}_i^2$ [86].

$$\mathbf{A} = \begin{bmatrix} a_1 & a_3 \\ a_2 & a_4 \end{bmatrix} \tag{5.9}$$

Two affine correspondences $(\mathbf{u}_i, \mathbf{A})$ and $(\mathbf{u}_j, \mathbf{B})$ then belong to the same homography if they satisfy the following conditions.

$$\begin{aligned} (\mathbf{u}_j^2 - \mathbf{u}_i^2)^T \mathbf{P}\mathbf{A}(\mathbf{u}_j^1 - \mathbf{u}_i^1) &= 0, \\ (\mathbf{u}_j^2 - \mathbf{u}_i^2)^T \mathbf{P}\mathbf{B}(\mathbf{u}_j^1 - \mathbf{u}_i^1) &= 0, \\ \begin{bmatrix} s + a_2b_3 - a_3b_2 & -(a_1b_3 - a_3b_1) \\ a_2b_4 - a_4b_2 & s - (a_1b_4 - a_4b_1) \end{bmatrix} (\mathbf{u}_j^2 - \mathbf{u}_i^2) &= \mathbf{0}. \end{aligned} \tag{5.10}$$

Where the variables s and $\mathbf{P}$ are defined as

$$s = \frac{[-a_2 + b_1 a_1 - b_1](\mathbf{u}_j^2 - \mathbf{u}_i^2) - (a_1 b_2 - a_2 b_1)(x_i^1 - x_j^1)}{(y_i^2 - y_j^2)}$$

$$\mathbf{P} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}.$$

The average error of the four conditions provides a similarity score between pairs of point correspondences. We then cluster point correspondences belonging to the same homography using affinity propagation [88]. This initialisation as can be seen in Figure 5.7 tends to over-segment the image, resulting in a large number of redundant and inaccurate models. However, this provides an initial point for CORAL, as it effectively reduces the space of labels needed to perform the energy minimisation in practice.

The global energy to be minimised by CORAL can then be defined as:

$$\underbrace{\sum_{l=1}^{L} \frac{1}{2} \sum_{i=1}^{n} (\|D(\mathbf{u}_i^1, \mathbf{H}_l^{12}\mathbf{u}_i^2)\|_{\Sigma_{12}} + \|D(\mathbf{u}_i^2, \mathbf{H}_l^{21}\mathbf{u}_i^1)\|_{\Sigma_{21}})\phi_l(\mathbf{u})}_{\text{Geometric Error Energy}}$$

$$+ \underbrace{\lambda \sum_{l=1}^{L} \sum_{i=1}^{n} \omega_{\mathcal{N}} |\nabla_{\mathcal{N}} \phi_l(\mathbf{u})|_{1,1}}_{\text{Smoothness Energy}} + \underbrace{\beta \|L\|}_{\text{Compactness Energy}} \tag{5.11}$$

The data term in Equation 5.11 accounts for the symmetric transfer of the re-projection error. Here, we refer to D as the Mahalanobis distance $\|D(\mathbf{u}_i, \mathbf{H}^{ab})\|_{\Sigma_{ab}} = (\mathbf{u}_i^a - \mathbf{u}_i^b)\Sigma_{ab}^{-1}(\mathbf{u}_i^a - \mathbf{u}_i^b)^T$ where Σ_{ab} represents the propagated covariance matrix through the mapping induced by the corresponding homography. This energy can be minimised using Algorithm 5 to obtain a compact solution that takes into account the geometric cost and also regularises for smoothness and compactness resulting in L homographies. The parameters for this energy minimisation are provided in Table 5.1.

Figure 5.8 showcases some of the results obtained from multi-homography fitting using CORAL in an urban environment. In a variety of scenes CORAL is able to fit the correct number of homographies corresponding to the planes in the scene. This is without any prior assumptions of the number or the orientation of the planes in a particular scene.

Figure 5.7: Figure showing multiple homography initialisation using affine transformations and affinity propagation clustering. With membership to a specific homography coded by colour it can be seen that this method results in an over-segmentation with a prevalence of redundant models.

The underlying planes can be revealed through a homography decomposition but prior to this, the homographies must be transformed to the correct reference

frame. As evidenced in Equation 5.11 the homographies returned by CORAL are defined as a transformation between matched image pixels in the **camera** frame. The planar models we are interested in reside in the **world** frame. A transformation between these two frames can be performed if the camera intrinsic matrix $\mathbf{K}$ is known as below;

$$\mathbf{H}_{world} = \mathbf{K}^{-1}\mathbf{H}^{\mathbf{12}}\mathbf{K} \tag{5.12}$$

After this transformation, the motion $\{\mathbf{R}, \mathbf{t}\}$ and the plane parameters $\pi = \{\mathbf{n}, d\}$ described by Equation 5.8 can be obtained through a Singular Value Decomposition (SVD)[89]. It must be noticed that SVD leads to two valid separate solutions for $\{\mathbf{R}, \mathbf{t}, \mathbf{n}\}$. In order to disambiguate between the two, we use the ego-motion estimation $\mathbf{T}_{vo}$ from visual odometry opting for the decomposition solution that is aligned with the estimate of motion.

$$\mathbf{T}_{vo} = \begin{bmatrix} \mathbf{R}_{vo} & \mathbf{t}_{vo} \\ \mathbf{0} & 1 \end{bmatrix} \tag{5.13}$$

Additionally, this decomposition only gives the translation and distance of the plane up to scale. The ego-motion translation estimate from visual odometry can be used to obtain the actual distance of the plane as follows:

$$d_a = \frac{\mathbf{t}_{vo}}{\mathbf{t}} \tag{5.14}$$

It must be noted that any discrepancy between the motion estimated through visual odometry and that obtained from the homography decomposition also provides an error measure for both the homography and its underlying planar prior. This

Table 5.1: CORAL parameters for planar prior generation.

Symbol	Value	Description
γ	5.9915	Constant cost for Outlier Model $\rho_{\emptyset}(\cdot)$
N_n	4	Number of neighbours per data point
λ	2	Regularisation parameter balancing geometric error and smoothness
β	60	Regularisation parameter balancing geometric error and compactness
τ	0.125	Gradient ascent step size for the smoothnes dual variable
ν	0.125	Gradient ascent step size for the compactness dual variable
α	0.125	Gradient descent step size for the primal variable
θ	1	Relaxation weight for the primal dual optimisation

Figure 5.8: Sample of the results from multi-homography fitting using CORAL with membership to different homographies encoded by colour-coded. CORAL can be seen to capture the large, plain and planar surfaces through the homography models in a variety of scenes and plane orientations. The planar priors can then be obtained through homography decomposition.

provides a check for homographies that are geometrically valid but do not lead to accurate planes in the scene such as the homography at infinity. In this way only accurate planes are passed on to the next step of the pipeline, which incorporates the planes into the dense estimation as described in the following section.

5.2.2 Planar Prior Labelling

With the planar priors retrieved from the two-view multi-homography segmentation, we propose a labelling of the image pixels to the corresponding underlying planes. Analogous to the homography minimisation, a similar global energy approach using the detected planar priors is used. This energy is formulated as;

$$\sum_{l=1}^{L}\left(\underbrace{\sum_{i=1}^{n}\|D(\mathbf{u})\|\phi_l(\mathbf{u})}_{\text{Data Term}}+\lambda\underbrace{\sum_{i=1}^{n}\omega_{\mathcal{N}}\|\nabla_{\mathcal{N}}\phi_l(\mathbf{u})\mathbf{S}\|_p}_{\text{Smoothness Term}}\right) \tag{5.15}$$

where L is the number of detected plane priors with an addition of the non-planar label $\emptyset$ that is provided by the original TGV stereo depth map, with the nodes $\mathbf{u} \in D$ image pixels. Borrowing from variational stereo depth map estimation approaches the data term measures photoconsistency between images. This can be expressed as:

$$D(\mathbf{u}_l) = \begin{cases} \min(\rho(\mathbf{u}_l), \rho_{max}) & l \in \pi_i, \cdots, \pi_L \\ \min(\rho(\mathbf{u}_\emptyset), \rho_{max}) + \rho_{bias} & l_\emptyset \end{cases} \tag{5.16}$$

where $\rho(\mathbf{u})$ measures the similarity between images using a patch window W. A small penalty term ρ_{bias} is added to the TGV label to account for its extra degrees of freedom, while ρ_{max} is included to handle poorly matching surfaces.

At each frame in this work there are four images available, the current stereo pair coupled with a previously viewed stereo pair from which the two-view homography segmentation is calculated. If the calibration and ego-motion is known, pixels can be projected from an image to either of the remaining three according to their corresponding depths. This depth is determined by the planar priors or given by the TGV stereo depth map in the outlier case as shown in Equation 5.16. From this projection, the photo-consistency is measured by comparing the similarity of the projected pixels and that of the second image over a window of size W.

Figure 5.9: Semantic Segmentation of the pixels of an image (left) into its respective classes (right). Buildings and roads over which planar surfaces are expected to be found are labelled in red and pink respectively.

Additionally, to ensure that non-planar regions are not wrongly assigned into planar priors, pixels that belong to non-planar classes have $\rho(\mathbf{u})$ automatically amended with respect to the semantic segmentation as shown in Equation 5.17. This semantic segmentation includes cars, trees and pedestrian classes.

$$\rho(\mathbf{u}_l) = \begin{cases} \rho_{max} + \rho_{bias} & \text{Non-Planar} \\ \rho_l(\mathbf{u}) & \text{Planar} \end{cases} \tag{5.17}$$

For this work, a full resolution residual network as outlined in [90] was used for the semantic segmentation. This approach combines outstanding recognition performance, as found in the current state-of-the-art with increased localisation accuracy. Training data was obtained from manually annotated images in the KITTI dataset [91] with an example output shown in Figure 5.9.

The smoothness term penalises depth discontinuities between neighbouring pixels based on their plane labels. A 4-connectivity pattern is used for $\mathcal{N}$ with a Euclidean norm to penalise points that belong to the same neighbourhood but have different labels. $\mathbf{S}$ is a matrix of the pixel depths corresponding to their assignment to different labels. To preserve depth discontinuities arising from different objects the smoothness term was down-weighted based on changes in the image gradient (∇I) using the weighting function $\omega_{\mathcal{N}}$ presented in Equation 5.18 where σ modifies this weighting.

$$\omega_{\mathcal{N}} = \frac{1}{\sigma \nabla I^2 + 1} \tag{5.18}$$

Observation of the global energy in Equation 5.15 reveals the similarity between this energy and the geometric multi-model fitting energy of Equation 3.8 with the only difference being the compactness energy. However, as presented in the

previous chapter, Algorithm 3 offers a minimisation of the geometric error and smoothness energies only and can thus be adopted for this energy, leveraging the speed and accuracy of CORAL for the per-pixel labelling. Sample results of the minimisation of this energy show that this method is able to correctly assign pixels to their corresponding planar priors as seen in Figure 5.10. The depth of these pixels can then be obtained using the plane equation as opposed to the noisy variation methods, offering an improved depth map. Parameters for this optimisation are presented in Table 5.2.

5.3 Implementation

We implemented and tested this pipeline using CUDA on a Nvidia GeForce GTX Titan Black 6048MB GPU. Running it one hundred times over sample images from KITTI [69], of resolution 376 x 1241, with sample results shown in Table 5.3. Bench-marking was run in scenes with three planar models mirroring the most common application scenario.

When performing per-pixel labelling using a global energy, convergence can be slow as pixels have to communicate over the entire image. To mitigate this a coarse-to-fine approach is introduced that reduces the number of iterations needed for convergence as demonstrated in [54]. This is done by performing the labelling at various levels: the original resolution, half the resolution and so on as shown in Figure 5.11.

Table 5.2: Parameters for the planar prior labelling.

Symbol	Value	Description
ρ_{max}	10	Maximum photo-consistency value
ρ_{bias}	1	Photo-consistency bias term added to account for TGV's extra degrees of freedom
λ_{TGV}	0.5	Regularisation parameter balancing data and smoothness terms in TGV depth-map
α_1, α_2	1.0, 5.0	Relative weights of the affine-smooth/piecewise-constant TGV terms
$\beta_{TGV}, \gamma_{TGV}$	1.0, 4.0	Exponent and scale factor respectively modifying the TGV-Tensor
N_n	2	Number of neighbours per pixel
σ	10	Scale factor modifying weighting of smoothness term by image gradient
λ	2	Regularisation parameter balancing photo-consistency error and smoothness
τ	0.125	Gradient ascent step size for the smoothness dual variable
α	0.125	Gradient descent step size for the primal variable
θ	1	Relaxation weight for the primal dual optimisation

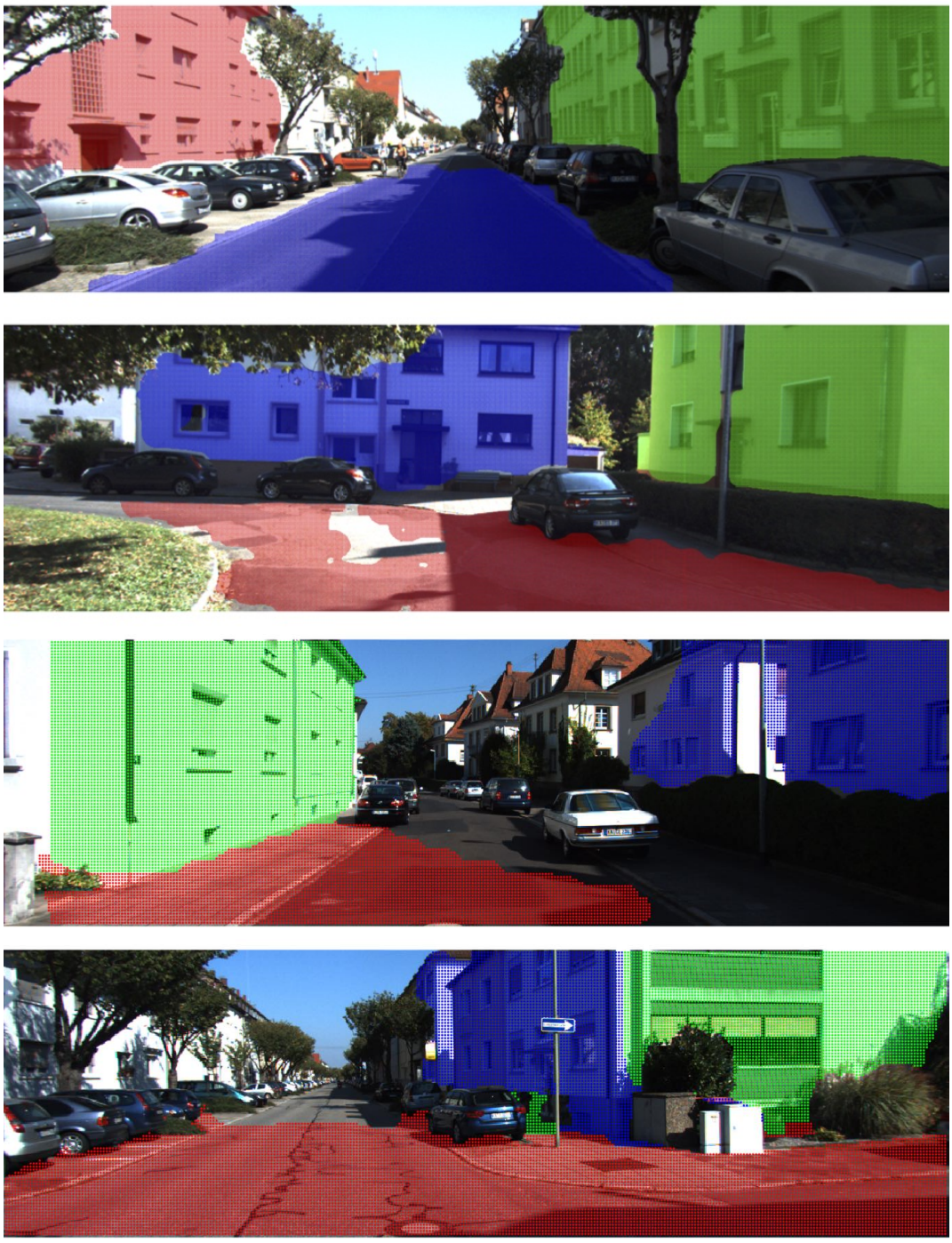


Figure 5.10: Sample results showing planar prior pixel labelling through an energy minimisation approach. This leverages planes obtained through CORAL and assigned pixels to them through a planar prior labelling. From planar priors obtained using CORAL pixels are assigned to these priors using a global energy function. The depth of these pixels can then be estimated solely using the plane hypotheses.

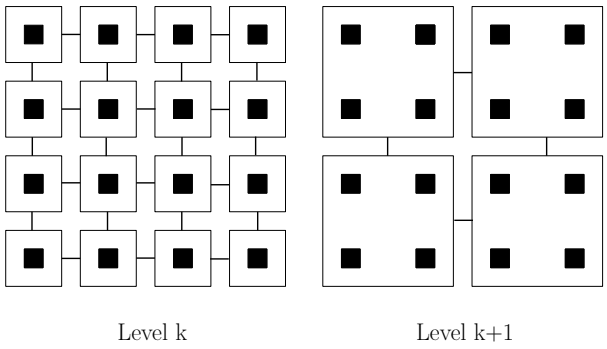


Figure 5.11: An illustration of the coarse–to-fine approach used to improve the speed of convergence in the multi-labelling problem. Groups of pixels are merged to create a coarser image in subsequent levels over which the per-pixel labelling is performed.

This approach results in faster convergence with timing results in Table 5.3 showing the proposed pipeline can improve depth maps in real-time (5Hz).

5.4 Results

This section provides an analysis of our proposed pipeline at improving depth map estimation. We are particularly interested in not only per-frame performance, as was seen in Figures 5.8 and 5.10, but whether the depth map estimation can accurately and consistently estimate large texture-less surfaces over large scales and through different view-points and occlusion levels. To this end, we chose to evaluate our system using the KITTI dataset [69] based on its length and the urban structures visible within the sequences. This dataset was captured through a stereo camera and a 3D Velodyne LiDAR sensor, additionally ground truth estimates for the camera motion are also provided.

Before performing the evaluation of the entire pipeline, we first verified the accuracy of the planar prior generation of Section 5.2.1. For this ground truth data

Table 5.3: Timing results on the depth map improvement pipeline.

Module	Time (ms)
Homography Optimisation (≈1000 correspondences)	13.7
Photo-consistency computation	3.7
Per-pixel labelling	185.3

of the plane parameters was first obtained by projecting the Velodyne LiDAR data to the left stereo image before using the segmented image to remove laser points lying in non-planar regions. CORAL was then used to create ground-truth laser planes for evaluation as shown in Figure 5.12. When these are compared to the planar priors obtained through CORAL, it can be seen that they returned accurate results, with a median angle error of 1.61 degrees and distance to plane error of 0.07 metres obtained when evaluated over a hundred metre section of the KITTI dataset. Additionally it must be noted that the homography decomposition also reveals the motion of the platform. This was also compared to the ground-truth odometry with results showing accurate estimation and a tight linkage to the plane parameters. This presents a method for filtering inaccurate plane parameters as errors in the motion are linked to poorly estimated planes.

With the accuracy of the planar priors verified, it remains to evaluate the dense depth map estimation over a large scale. To obtain this, consecutive depth maps may be back-projected to give a sparse reconstruction of the world and then composed together using the odometry source. This approach is extremely memory inefficient as the memory required grows linearly with the number of composed depth maps severely limiting the size of the potential reconstructions. Instead to evaluate the improvement in mapping in large scale areas, we use the BOR^{2}G approach outlined by [92] to create large scale dense reconstructions by fusing consecutive depth maps. In this approach, consecutive depth maps are processed through a uniform voxel grid to create a continuous surface. Each voxel stores range observations represented by their corresponding Truncated Signed Distance Function (TSDF) which is computed such that one can solve for the zero-crossing level set (isosurface) to find a continuous surface model (mesh).

Analogous to the planar prior verification, ground truth for the comparison is provided through the Velodyne LiDAR scanners with consecutive LiDAR scans consolidated into a single reference frame through the use of the odometry data. As there are inevitable errors in the KITTI's GPS/INS ground-truth poses we limited this consolidation to a relatively short distance (75m) over which a good comparison

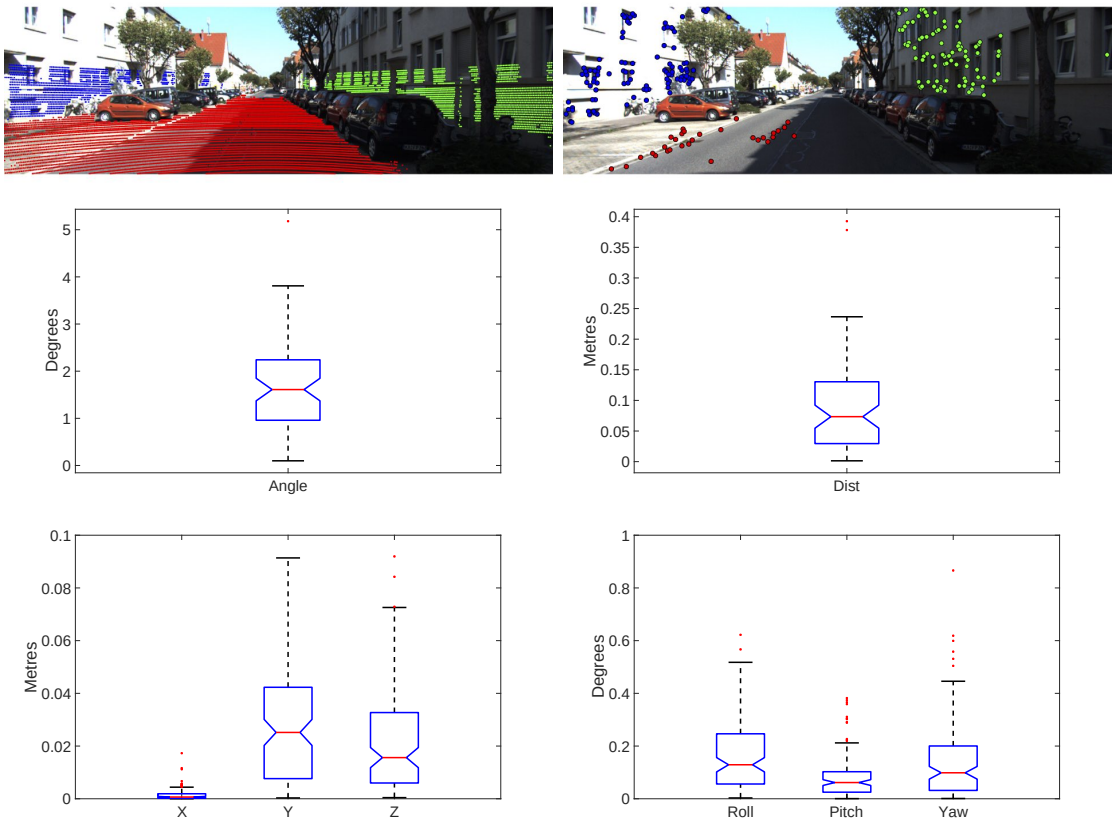


Figure 5.12: By projecting Velodyne laser data onto an image, CORAL can be used to create ground-truth plane annotations(top left), these can then be compared to the plane priors obtained through the homography decomposition (top right). A comparison of the parameters (middle row) shows a good relation between the ground truth plane parameters and those obtained by homography decomposition as shown by the small angle and distance errors in the boxplots. Additionally, as the homography decomposition provides the motion of the camera, this was compared to the ground-truth odometry (bottom row). With results showing that the decomposed homographies maintain a good estimation of the motion of the platform as seen in the boxplots.

between the two could still be made. To compute the error statistics against the sparse ground truth point cloud, the vertices of the corresponding mesh were used. The error thus reduced to the metric distance between a vertex in the mesh and its closest laser point. Two sections of the KITTI sequence 00 were chosen for evaluation in regions where the reconstruction performance was shown to be poor. Using ten centimetre voxels the reconstructions were created from the two sets of depth maps with results consolidated in Table 5.4 and Figure 5.13.

From Figure 5.13 it can be seen that the introduction of the planar priors results

Figure 5.13: Sample results of dense reconstructions from the fusion of depth maps in the two evaluated sections are shown in this figure. The left column of images corresponds to the reconstructions produced using the TGV depth maps. It can be seen that the noise on the depth maps results in some gaps in the reconstruction. By improving these depth maps with planar priors in our approach the results can be improved on, significantly, as shown in the right column. Resulting in a more accurate map of the world.

in more complete reconstructions (right column) over the evaluated sections than the raw TGV depth maps (left column). Holes in the reconstruction created as a result of the noisy depth maps in textureless regions are filled when the planar priors are used. Similarly, gaps presented through partially occluded regions are

Table 5.4: Table of the evaluated error statistics.

KITTI-VO 00	Type	Median (cm)	75% (cm)	Surface Area (m^2)	#Blocks	#Voxels 10^6	GPU Memory (MB)	Distance (m)
Case 1	TGV	15.9434	49.8064	3140.74	26367	13.49	154.494	78.22
—	Prior	18.2147	51.5005	3450.91	27405	14.03	160.576	78.22
Case 2	TGV	24.4837	99.8326	4162.02	28185	14.43	165.146	73.67
—	Prior	25.6882	98.3273	4399.76	30274	15.50	177.387	73.67

filled by this pipeline resulting in smoother and more complete reconstructions.

This is confirmed when the quantitative evaluation in Table 5.4 is viewed. For both cases, the use of the planar prior results in a significant increase in the surface area of the reconstruction, ≈ 10 % in the first case, as well as number of blocks and voxels when the planar priors are used. While the median error increases when the planar prior is used this is expected as the resulting reconstruction covers a larger area and thus accumulates more error. Additionally observe in both cases that the difference between the median errors is within the noise of the ground truth laser sensor (3 cm).

5.5 Conclusions

In this chapter, an end to end system for improving the estimation of depth maps from images is presented and evaluated. In particular focusing on large, plain and planar surfaces which have long-plagued contemporary dense depth map estimation techniques due to lack of abundant texture. We leverage an energy based model discovery technique over homographies to induce planar priors over these planar regions. This is then similarly employed to associate and label the underlying image pixels to their corresponding planar model to improve the depth map estimation. The labelling is further aided by an image segmentation that removes the pixels that belong to non-planar regions, reducing the noise and increasing the labelling accuracy.

The energy minimisations in this work rely on CORAL that allows for a parallel implementation on GPGPU hardware. Opening up this pipeline to online use in robotics applications and with steady advances in GPGPU hardware real-time implementations. This could reduce the reliance on expensive 3D sensors in favour of lower-cost visual sensors allowing for the increased development and deployment of robotic platforms. The subsequent fusion of the improved depth maps also reveals a more complete reconstruction of the environment with little change in the reconstruction error.

Nothing in life is to be feared, it is only to be understood.
Now is the time to understand more, so that we may fear less.

Marie Curie

6

Road Understanding For Autonomous Vehicles

Abstract

For autonomous vehicles, the understanding of the rules that govern allowable behaviours on a particular road are vital for safe operation. These rules mark out the allowable driving area, preview upcoming hazards such as pedestrians and intersections, as well as give the oncoming lane. Whilst most of these remain fairly static and can be either manually encoded or recorded offline, it is still important for the autonomous vehicles to include mechanisms for the live detection of these rules, especially if autonomous vehicle technology desires to remain scalable.

In this chapter we offer a hybrid approach to obtain the rules of the road even under severe occlusions, with a two-step approach that combines deep neural networks and geometric multi-model fitting through CORAL. We show that these techniques allow for the extraction of multiple road marking and road boundaries at real-time when evaluated on the Oxford RobotCar Dataset.

Contents

6.1 Road Marking Classification 103
6.1.1 Road Marking Segmentation 103
6.1.2 Road Marking Geometric Primitive Fitting 104
6.1.3 Road Marking Geometric Primitive Clustering 107

6.1.4 Road marking tracking 109
6.2 Road Boundary Classification 112
6.2.1 Road Boundary Segmentation 112
6.2.2 Road Boundary Fitting 114
6.3 Implementation 117
6.4 Conclusion 118

Over the past few years, deep learning methods have reshaped the way many complex problems are approached. This is by offering a data driven approach that was in contrast to pure geometry approaches that had dominated the literature. This recent dominance has been brought about largely through two major contributions [93], firstly the changes in the way neural networks are initialised proposed by Glorot and Bengio [94] allowed for neural networks to be trained more effectively, secondly the increase in computational power available through GPGPU's has significantly reduced the time needed to train deep networks making them feasible for complex tasks from object segmentation, localisation [95], single view depth predictions [96–98], to autonomous navigation [99].

However to obtain well-generalised high performance deep networks, a large amount of training samples are required. The training data must capture both the variation in the scenarios encountered and the prevailing conditions over which they are observed for the specific task at hand. This in most cases requires time-consuming annotations of data by humans. Additionally, many existing deep methods do not explicitly infer geometry especially in scenarios with a large amount of noise. Liu et al. [100] present one of the first deep-learning approaches that estimate underlying geometry but their approach is limited to low-noise situations. This approach was used to fit planes in RGBD images in scenes where the planes dominate the scene.

These two limitations make end-to-end deep learning approaches non-trivial to apply in certain applications. In this chapter one such scenario is explored, the understanding of the road by autonomous vehicles through painted road markings and road boundaries. A complementary two-step approach that utilises the well-established segmentation capabilities of deep networks and combines it with the accurate multi-model fitting formulation of CORAL is presented. Allowing for

more efficient accumulation of training data while also ensuring that the underlying geometry that gives the road markings and boundaries is inferred.

For autonomous vehicles (or any robotic platforms), safe operation and deployment is intrinsically tied to their understanding of the current work-space. This extends from the knowledge of its location and objects in its surroundings which were explored in Chapter 5 to the allowable behaviour at that particular location. The latter is mainly encoded into painted markings on the road surface as well as their limits (road boundaries) which guide vehicles into acceptable behaviour and serve as warning for different hazards.

While there has been substantial effort put into high-definition offline mapping services such as Google Maps, HERE, OpenStreetMap among others to encode this road information, these offline systems do not fully negate the need for autonomous vehicles to be able to directly detect and interpret road markings and their boundaries in real-time through their live sensors for several reasons. This is as roads are constantly changing, increasing and undergoing maintenance creating discrepancies between the current road and when it was mapped. These off-line methods cannot directly compensate for this, leading to safety concerns for autonomous operation especially when regions that receive less traffic are considered, as autonomous vehicles move from large well-understood cities to more rural areas where the offline maps are updated less frequently.

Early approaches into the online detection of road markings using cameras, the modality of choice in this chapter, proposed matching of features/shapes to obtain the road marking classes [101, 102]. These approaches were however very sensitive to changes in orientation, scale, lighting, occlusions and weather conditions. On the other hand for road boundary detections many camera-based approaches [103–108] used the assumption that road boundaries, or curbs, could be identified by a change in height. Thus they can be modelled as 2D step functions in the image but only after the 3D structure of the world has been obtained through dense depth map techniques. This makes these approaches not only sensitive to the shortcomings of dense depth map estimation techniques, highlighted in the

previous chapter, but as curbs are very often occluded, mostly by parked cars, this leads to limited success in many complex scenes.

Deep learning approaches could potentially offer a solution to some of these challenges, however as highlighted previously this comes at the cost of obtaining adequate training data. For road marking classification, annotating road marking classes is not only labour intensive [109] but there is a large change in similar road marking classes between regions/countries that makes generalisation of these deep learning methods difficult. This translates to having to do an additional expensive annotation process for every new region encountered which limits scalability. For road boundary (curbs) detection obtaining the training data is simpler, however, as the underlying geometry is not inferred, the performance of deep networks at identifying curbs deteriorates greatly in the presence of occlusions. These challenges as was stated before make the application of an end-to-end deep learning pipeline non-trivial for this task.

If the problem is split however, the well-established segmentation capabilities of deep networks can be utilised, where segmentation here refers to determining which pixels of a target image belong either to the road markings or boundaries as compared to the classification approach which end-to-end networks would give. Shifting the focus to this binary segmentation opens up the possibility of using semi-supervised methods for creating the ground truth data greatly reducing the human annotation effort. From the corresponding segmentations it can be seen that by obtaining the underlying geometric primitives the road markings and boundaries can be revealed. This is as the majority of road markings consist of either linear segments or a collection of linear segments while road boundaries consist of cubic splines.

This motivates the approach taken in this chapter, whereby a deep segmentation network is used to obtain the road markings and boundary pixels. For the segmentation network used in this chapter we adopted the U-Net architecture [110] shown in Figure 6.1 which is a Convolutional Neural Network (CNN). CNNs were designed based on studies of the visual cortex that has shown that many neurons only react to visual stimuli in a particular region which were then coded

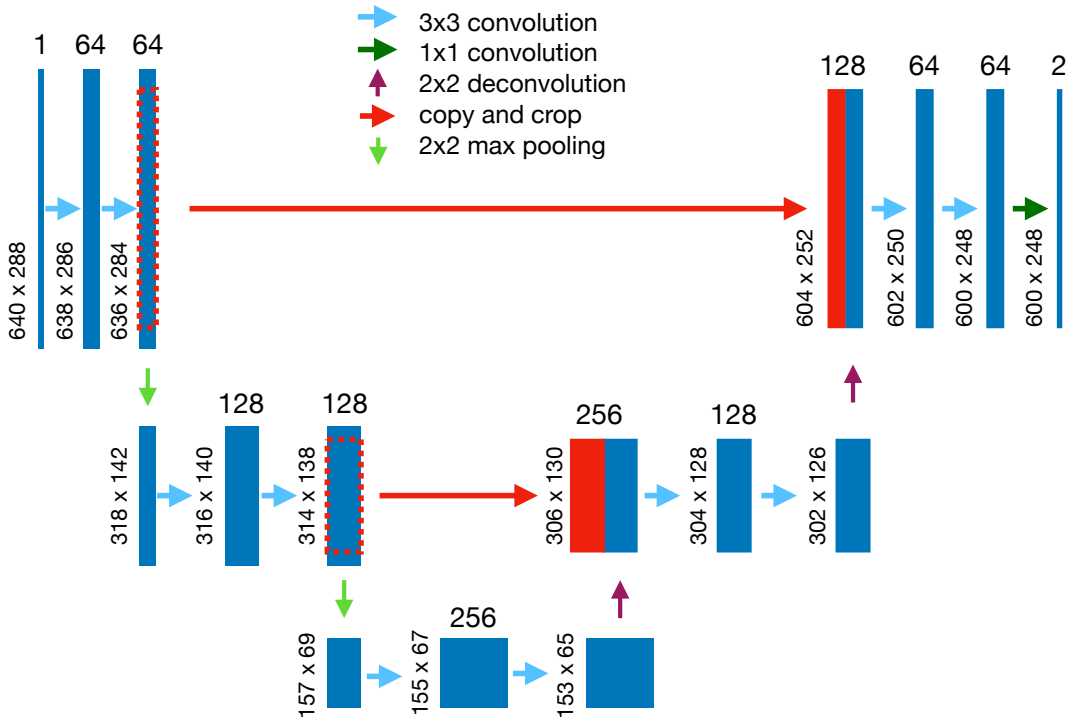


Figure 6.1: Example of the three layers deep fully convolutional U-net architecture [110]. In this architecture neurons in the first layer are only connected to pixels in their receptive fields, giving the networks the ability to concentrate on lower level features in the first layer, then aggregate them into higher level features in the next hidden layers. This makes them particularly suited for visual tasks. Additionally there is a pooling stage, that subsamples images to reduce the computational load and the number of parameters [93].

into the neocongitron [111]. In this architecture neurons in the first layer of a CNN are only connected to pixels in their receptive fields, giving the networks the ability to concentrate on lower level features in the first layer, then aggregate them into higher level features in the next hidden layers. This makes them particularly suited for visual segmentation tasks.

Shifting to binary segmentation networks also greatly reduces the annotation effort. For these two applications, LiDAR data can be pulled in to further reduce the image annotation load as can be seen in Figure 6.2. This is by annotating a section of a LiDAR point-cloud followed by projecting these annotations to the images captured in that section, this way image annotations can be swiftly obtained. In addition if multiple runs of the same environment are performed, a high-fidelity

Figure 6.2: Figure showing a LiDAR point cloud of a road surface and the corresponding camera image of the same scene. It can be seen that the high-reflectance pixels correspond to road markings which can be related to the image pixels using a Conditional Random Field.

localiser can be used to project annotations to the different runs. In so doing ensuring the training data captures changes in prevailing conditions, lighting and weather which is critical to obtain a high-performance deep network. In this work we used the Oxford RobotCar Dataset [10], in this dataset a hundred different runs of a ten kilometre route around Oxford that highlight different lighting and weather conditions are presented thus this was used to obtain the annotations.

After the pixels have been obtained, geometric models are fit to the segmented pixels using CORAL. The observed scenes contain an unknown number of geometric models while the output of the networks will still inevitably contain some level of noise. The CORAL formulation is able to deal with both of these scenarios while still returning the geometric models in real-time, which is crucial for the on-line understanding of the road.

In the following sections, the specifics for the two different road understanding tasks are explored in more detail. Section 6.1 details the road marking classification with Section 6.2 detailing the road boundary classification. This is followed by implementation details from both methods and conclusions in Section 6.3 and 6.4 respectively.

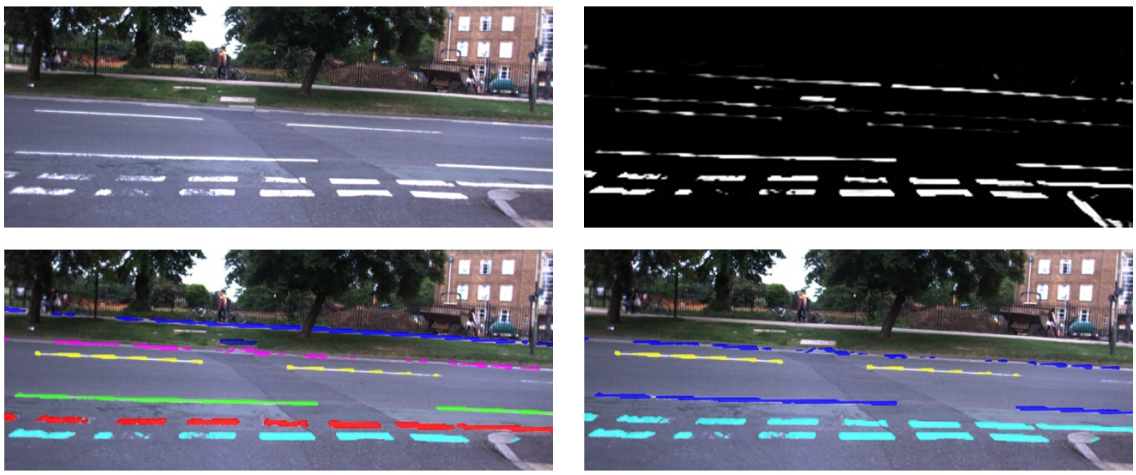


Figure 6.3: Semantic classification of road markings in an urban environment. From an image taken by the front-facing camera of an autonomous vehicle (top left), pixels of potential road markings can be identified by a trained deep neural network (top right) under various lighting conditions. A two-step CORAL optimisation retrieves the road marking classes. The first optimisation step reveals geometric primitives (lines) which encode road marking segments (bottom left), before a further optimisation step classifies these primitives into semantically meaningful road markings as seen in the bottom right image.

6.1 Road Marking Classification

Road markings in many ways encode the rules of the road. Giving visual cues around acceptable behaviour such as lane-following while also signalling the presence of potential hazards such as oncoming intersections. For autonomous vehicles to operate safely on the roads they must also be able to interpret and understand the road markings in view. In this section we show how our proposed approach enables this, this is summarised in Figure 6.3 and the following sections.

6.1.1 Road Marking Segmentation

The first step of our proposed approach to road marking classification is to employ a deep semantic segmentation network to identify image pixels $\mathbf{u} \in \{\mathbf{u}_1, \cdots, \mathbf{u}_{n_{rm}}\}$ that belong to road markings, n_{rm} is the number of identified pixels. Bruls et al. [112] showed that in the presence of adequate training data this network will outperform other techniques as it is able to exploit the global scene context making it more

robust to lighting changes, spatial deformations, degradation, and partial occlusions.

The creation of the training data is made easier by bootstrapping LiDAR point-clouds to the images as shown in Figure 6.2. This is as road markings in LiDAR point-clouds often appear as high-reflectance LiDAR points, which are not affected by lighting changes. Correlating these with the monocular images can provide an adequate source of training data. These point-clouds are however littered with multiple high reflectance sources from cars and bicycles to lamp posts. By limiting the region of interest to the approximately planar road surface, a majority of these noisy high-reflectance sources can be removed.

The classification of the image pixels then occurs after the projection of the road surface points. A Conditional Random Field (CRF) approach [113] can be used to associate image pixels with high-reflectance points on the point-clouds within the region of interest as was shown in [112]. This allows for the automatic generation of a large set of road marking annotations under various conditions which are used for training purposes.

This data is then used to train a U-Net inspired deep network as described in [112] which at run-time outputs a road marking mask over the images. Figure 6.4 shows that the network is robust to both appearance and lighting changes in the scene, providing a robust set of pixels from which individual road markings can be obtained.

6.1.2 Road Marking Geometric Primitive Fitting

After the identification of the road marking pixels by the trained deep segmentation network, we fit geometric primitives that encapsulate road marking segments. The number of road marking segments in a specific scene is *a priori* unknown, and apart from the trivial case of lane markings, they occur in different orientations and lengths with various levels of occlusion. Additionally the network output also contains some level of noise, as objects of high reflectance (i.e. curbs) not corresponding to road markings can sometimes be wrongly segmented. As this work concentrates on road markings from linear segments, fitting the geometric primitives can be seen as a multi-line fitting problem. While most of the algorithms

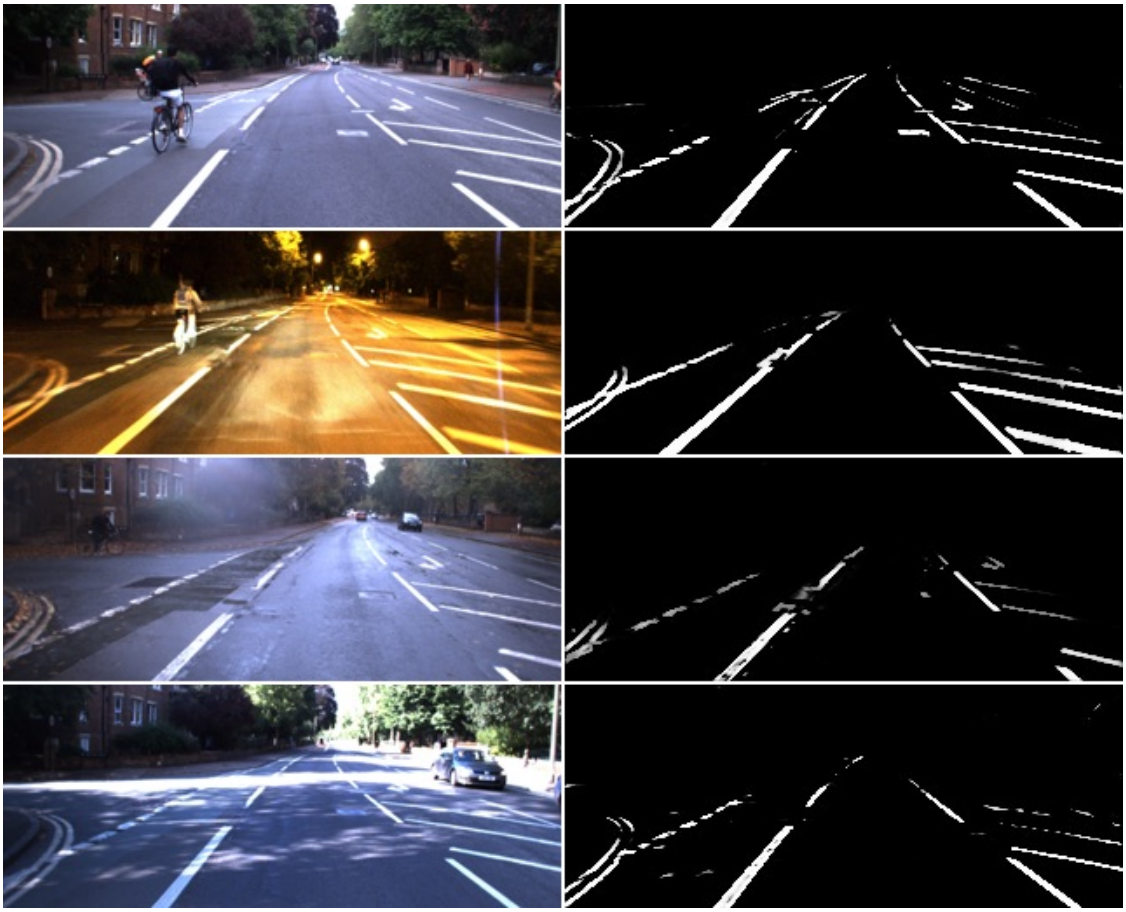


Figure 6.4: Road marking detection under different lighting conditions (overcast, night, rainy, and sunny). Despite the large changes in the prevailing conditions the trained deep segmentation network [112] is able to accurately identify the pixels belonging to the road markings from the monocular image.

commonly used for multi-line fitting such as the Hough Transform and sequential ransac will be able to show good performance in many scenarios particularly when viewing a scene where the road markings are primarily lane markers there is a marked deterioration in performance when approaching different scenes as can be seen in Figure 6.5. With this in mind we adopt CORAL to perform the multi-line fitting. We define the following global energy function:

$$\underbrace{\sum_{l=1}^{L}\sum_{i=1}^{n_{rm}}(\|D(\mathbf{A}_l,\mathbf{u}_i)\|)\phi_l(\mathbf{u})}_{\text{Geometric Error Term}} + \lambda\underbrace{\sum_{l=1}^{L}\sum_{i=1}^{n_{rm}}|\nabla_{\mathcal{N}}\phi_l(\mathbf{u})|_{1,1}}_{\text{Smoothness Term}} + \underbrace{\beta\|L\|}_{\text{Compactness Term}} \tag{6.1}$$

Here $\mathbf{A}_l = (a_l, b_l, c_l)$ is the line equation $a_l\mathbf{x} + b_l\mathbf{y} = c_l$ and we refer to D as the shortest Euclidean distance between a pixel $\mathbf{u}_i = (x_i, y_i)$ and the line $\mathbf{A}_l$.

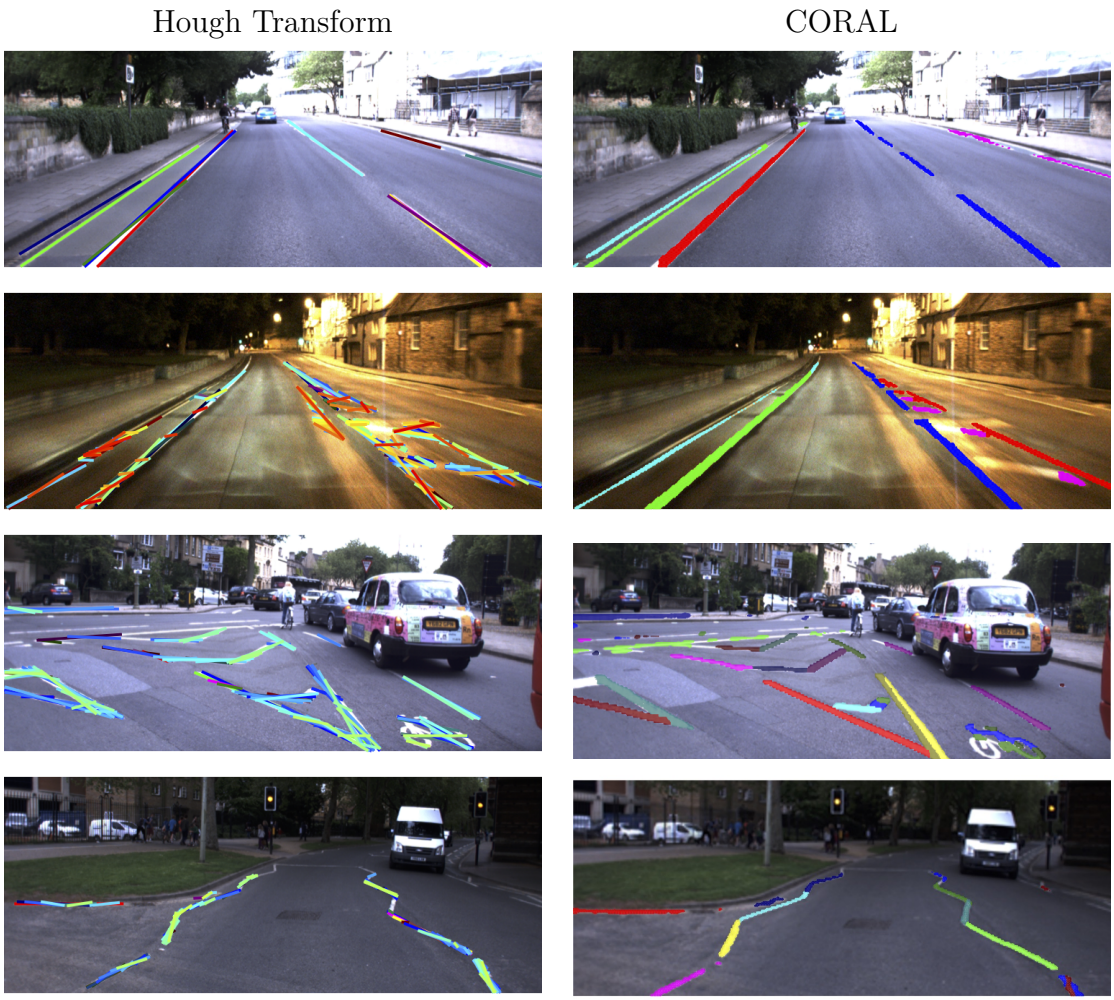


Figure 6.5: A comparison of the results from road marking geometric primitive extraction under different prevailing conditions using the Hough Transform and CORAL approaches. It can be seen that our proposed energy optimisation approach is able to accurately extract linear segments in a diversity of scenes while the Hough Transform suffers in the presence of noise and in more complex scenes result in an over-parametrisation of the scene. CORAL is thus able to accurately reveal the underlying primitives for a large number of road markings present in the urban scene.

Minimisation of the energy in Equation 6.1 through the primal dual optimisation in Algorithm 4 reveals the underlying geometric models.

The initialisation of models needed in Algorithm 5 was performed by the Hough Transform, which provides a large number of models with low accuracy for iterative refinement through the CORAL formulation. A refinement that converges on L road marking segments with sample results in Figure 6.5 showing that these correspond to the underlying primitives for a large number of road markings in a diversity of

scenes. The parameters needed for the energy minimisation are shown in Table 6.1.

While the fitting of geometric models to the segmentation data is far from trivial, when it comes to road markings we are more interested in their classes as these encode the rules of the road. What we can observe is that road marking classes consist of collections of geometric primitives arranged in a particular spatial configuration. This in itself constitutes a geometric model albeit a hybrid one. We can therefore apply a subsequent global optimisation step to obtain the road marking classes, details of which are presented in the next section.

6.1.3 Road Marking Geometric Primitive Clustering

In this section we seek a clustering of the detected road marking segments to obtain semantically meaningful road marking classes. We leverage the idea that collections of geometric primitives, however complicated, are still geometric models albeit with more constraints ($\theta(\cdot)$). Here $\theta(\cdot)$ represent the intra-class constraints induced within these collections. Moreover, a set of fixed rules govern the spatial relationships between road marking classes which can be encoded into inter-class constraints $\Omega(\cdot)$. These constraints allow for the definition of a data fidelity term, similar to the geometric error energy in Equation 6.1 but over the detected road segments rather than the road marking pixels.

Before the clustering can be performed, the effect of the camera perspective must be removed. In this work as we utilise images from front-facing cameras mounted on autonomous vehicles, the perspective of the camera significantly distorts the length, orientation and position of the road marking segments. By positioning

Table 6.1: CORAL parameters for the road marking geometric primitive fitting.

Symbol	Value	Description
γ	2	Constant cost for Outlier Model $\rho_{\emptyset}(\cdot)$
N_n	4	Number of neighbours per data point
λ	2	Regularisation parameter balancing geometric error and smoothness
β	60	Regularisation parameter balancing geometric error and compactness
τ	0.125	Gradient ascent step size for the smoothness dual variable
ν	0.125	Gradient ascent step size for the compactness dual variable
α	0.125	Gradient descent step size for the primal variable
θ	1	Relaxation weight for the primal dual optimisation

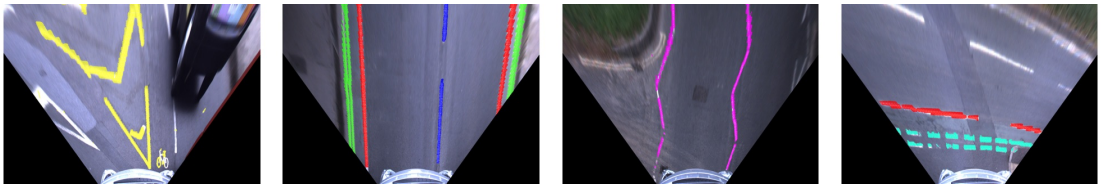


Figure 6.6: Figure showing manually annotated road markings after inverse perspective mapping. These show a junction (left), lane markings (second left), zig-zag lines (second right) and a road intersection (right) scene. In this birds eye view of the scene the effects of the camera perspective have been removed, road markings thus observed retain their configurations even as the autonomous vehicles traverse urban scenes.

a virtual camera above the observed scene this effect can be removed providing consistency not only in the detected road marking segments but also in their spatial configurations. This repositioning is performed through a homography warping of the scene, referred to as the Inverse Perspective Mapping (IPM) [114], which despite assuming that the road surface is planar works well in practice as seen in Figure 6.6.

We can then focus on the clustering of six road-marking classes commonly seen in the urban settings. These are single lane boundaries, double lane boundaries, lane separators, intersection markings, zig-zag and then junction road markings. Detection of these classes is important for the safe use of autonomous vehicles as they denote the allowable drivable area (lanes) and also preview upcoming road situations (e.g. a pedestrian crossing or a junction). With these classes in mind we introduced two constraints, the angle between road marking segments and the corresponding distance between them. While simple, these constraints encompass the possible configurations of the road marking classes and can be defined as:

$$\theta_{angle}(\mathbf{A}_i, \mathbf{A}_j) = |\arctan(a_i/b_i) - \arctan(a_j/b_j)| \tag{6.2}$$

$$\theta_{dist}(\mathbf{A}_i, \mathbf{A}_j, \mathbf{u}_i^m, \mathbf{u}_j^m) = (D(\mathbf{A}_l, \mathbf{u}_i^m) + D(\mathbf{A}_l, \mathbf{u}_j^m))/2 \tag{6.3}$$

where $\mathbf{A}_i = (a_i, b_i, c_i)$ is the equation, $a_i\mathbf{x} + b_i\mathbf{y} = c_i$, of the road marking segment i and $\mathbf{u}_i^m$ is the midpoint of the pixels assigned to it.

By observing the classes it can be seen that the lane boundaries, lane separators and intersection road markings all consist of singular infinite line models. There is however, a strong prior that these classes appear parallel to each other allowing

for the definition of an inter-class constraint that penalises a collection of these if they are not parallel through the angle constraint. Similarly, the double lane boundary consists of two parallel singular infinite line models within close proximity of each other, providing inter-class constraints based on the angle and distance. Lastly, the angle is preserved in the zig-zag and junction crossing, producing a similar inter-class constraint. The different classes and the constraints between them are summarised in Table 6.2, constraints which are used to formulate the geometric error term in Equation 6.4. We can thus propose several instances n_{rc} of these semantic classes through a targeted search that aims to find collection of lines that are parallel as well as those that fulfil the angle constraints of the zig-zag and junction classes. This results in the following energy as below:

$$\underbrace{\sum_{l=1}^{L}\sum_{i=1}^{n_{rc}}(\|\theta(\mathbf{A}_l,\mathbf{A})\|)\phi_l(\mathbf{A})}_{\text{Geometric Error Term}}+\lambda\underbrace{\sum_{l=1}^{L}\sum_{i=1}^{n_{rc}}|\nabla_{\mathcal{N}}\phi_l(\mathbf{A})|_{1,1}}_{\text{Smoothness Term}}+\underbrace{\beta\|L\|}_{\text{Compactness Term}} \tag{6.4}$$

This is then are fed into the CORAL formulation in Algorithm 4, the iterative optimisation of which reveals a compact set of semantic class instances as shown in Figure 6.7. It can be seen qualitatively from this Figure that accurate classification is achieved over different classes and prevailing conditions through this approach. To differentiate between the three classes of singular infinite line models in these results, the contiguity of their associated pixel inliers is used.

6.1.4 Road marking tracking

While the previous section has described a framework for obtaining road marking classes from a single frame, tracking of road markings through consecutive image

Table 6.2: CORAL geometric clusters.

	Single Lane	**Double Lane**	**Separator**	**Give-way**	**Zig-zag**	**junction**
Single Lane		$\theta_{angle}(\cdot)$	$\theta_{angle}(\cdot)$	$\theta_{angle}(\cdot)$		
Double Lane	$\theta_{angle}(\cdot)$	$\theta_{dist}(\cdot)$	$\theta_{angle}(\cdot)$	$\theta_{angle}(\cdot)$		
Separator	$\theta_{angle}(\cdot)$	$\theta_{angle}(\cdot)$		$\theta_{angle}(\cdot)$		
Give-way	$\theta_{angle}(\cdot)$	$\theta_{angle}(\cdot)$	$\theta_{angle}(\cdot)$	$\theta_{dist}(\cdot)$		
Zig-zag					$\theta_{angle}(\cdot)$	
Junction						$\theta_{angle}(\cdot)$

Figure 6.7: Sample of results from the semantic classification of road marking pixels under different weather and lighting conditions (overcast/rain/night). Our proposed CORAL approach is able to detect multiple road marking segments in images from detected road marking pixels. These segments are aggregated through another energy minimisation into their semantic classes. Thereby we reveal the underlying meaning of the road markings in complex urban environments providing important cues for autonomous vehicles. These include indicating for upcoming road situations, which could require specific behaviour. For instance, the zig-zag markers (*purple*) indicate an upcoming pedestrian crossing and the give-way dashes (*cyan*) indicate a junction.

frames improves the robustness of the classification, in most cases road markings are seldom only seen in one frame and persist from frame to frame. This persistence increases our confidence of correct classification when a road marking is viewed over multiple scenes as illustrated in Figure 6.8. Additionally some road markings when

Figure 6.8: Given a fully observed road marking (left) our proposed approach is able to correctly detect the underlying semantic class of the road markings on a image. However when this becomes partially observed (middle) the interaction of the underlying geometric primitives with others in the scene can cause mis-classification, in this case the road junction was labelled as a zig-zag line. By introducing road marking tracking (right) even under partial observation the correct underlying can be arrived at, making the classifications more robust.

traversed are not fully observable, i.e. a road junction, making their classification ambiguous. By tracking classes we can ensure that the correct assignment is made even as road markings become partially observed.

In this work, the road marking classes are collections of geometric primitives. These primitives can be tracked between subsequent frames if the motion $\mathbf{T} = \{\mathbf{R}, \mathbf{t}\}$ between the frames is known. We use the homography transformation between the subsequent frames to project a line model $\mathbf{A}_i$ into the next frame where the plane equation refers to the road surface.

$$\begin{aligned} \mathbf{H} &= \mathbf{R} + \frac{\mathbf{t}\mathbf{n}^t_{plane}}{d_{plane}} \\ \mathbf{A}_{i+1} &= \mathbf{H}\mathbf{A}_i. \end{aligned} \tag{6.5}$$

These give an initial set of projected models for the subsequent frame, obtained from the initial frame, however, this set does not include all possible models, as road marking segments can appear for the first time in a particular frame. To cope with this, road marking pixels that are inliers to the projected models are first removed before a further Hough Transform (HT) initialisation is done to the outliers availing new road marking geometric primitives as summarised in Algorithm 7. Putting this into a sliding-window pipeline further increases robustness as it naturally deals with over-exposed scenes and fully occluded road markings. This allows detected semantic classes to persist further.

Algorithm 7: Multi-frame road marking tracking.

{Initialisation};
if *First Frame* **then**
  Propose $\hat{\Theta}_0$ models with HT ;
else
  Calculate Homography ;
  Warp Lines $\boldsymbol{\Theta}_{i-1}$ to $\breve{\Theta}_i$;
  Remove inliers;
  Propose Θ_{HT} models from outliers using HT;
  $\hat{\Theta}_i = \{\breve{\Theta}_i, \Theta_{HT}\}$;
end
while *not converged* **do**
  Primal Dual Optimisation;
  Merge Lines;
  Re-estimate Lines;
end
L road marking segments Θ_i;
Semantic Road markings ;

6.2 Road Boundary Classification

In the context of autonomous driving, road boundaries or curbs play an important role as they delimit, legally and intentionally, driveable space thus providing important information that can be used for mapping, path-planning and navigation. For their classification we use a similar two-step approach that was taken for road marking classification. Firstly employing a deep neural network to identify pixels of curbs, followed by geometric multi-model fitting to obtain the curbs.

6.2.1 Road Boundary Segmentation

In a similar manner to road marking classification, the first step in road boundary classification is to identify image pixels $\mathbf{u} \in \{\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_{n_{rb}}\}$ that belong to road boundaries, where n_{rb} is the number of identified image pixels. However as was pointed to in the previous section, road boundaries are usually occluded to various degrees. Concentrating on only the *visible* boundaries or curbs would therefore lead to an incomplete picture of the scene. Even for an adequately trained deep segmentation network, occlusions lead to blurry outputs as the deep network does

Figure 6.9: Given an input image (left), an adequately trained deep segmentation network is able to correctly identify the *visible* road marking pixels. However it produces a blurry output through occlusions (red rectangle) as it is unable to infer the underlying geometry that persists through the occlusions which would bias it to correctly detect the *occluded* road boundaries.

not infer the underlying geometry of the curb that would allow it to detect pixels through occlusions. Blurry outputs which can be seen in Figure 6.9.

This suggests that the detection should be split, with one component revealing the *visible* curbs and the other the *occluded* curbs. In this work we opt for two deep networks to achieve this. Before presenting the details of these networks, adequate training data must be obtained. Again as in the road marking case, LiDAR data is used to bootstrap the ground-truth data annotations. Hand-annotating is done on LiDAR point-clouds before these labelled points are projected back to the camera images to obtain pixel-wise annotations. Localisation between different traversals of the annotated route also allows the number of annotated images to be increased and crucially presents occluded road marking labels. This is as occlusions, such as parked cars do not persist in the scene and can thus be obtained through different traversals. In this work we used the Oxford RobotCar Dataset [10] to obtain the annotations and then evaluated it on different runs of this dataset.

For the detection of *visible* curbs, the ground truth annotations were used to train a U-net deep segmentation network such as that presented in Figure 6.1 which has been shown to be robust to both appearance and lighting changes.

For the detection of *occluded* curbs, Sulemanyov et al. [115] suggested this should be approached as an object detection rather than a segmentation problem. With the objects to be detected in this case a set of discrete (anchor) lines set at different angles as shown in Figure 6.10. These discrete lines would thus point directly

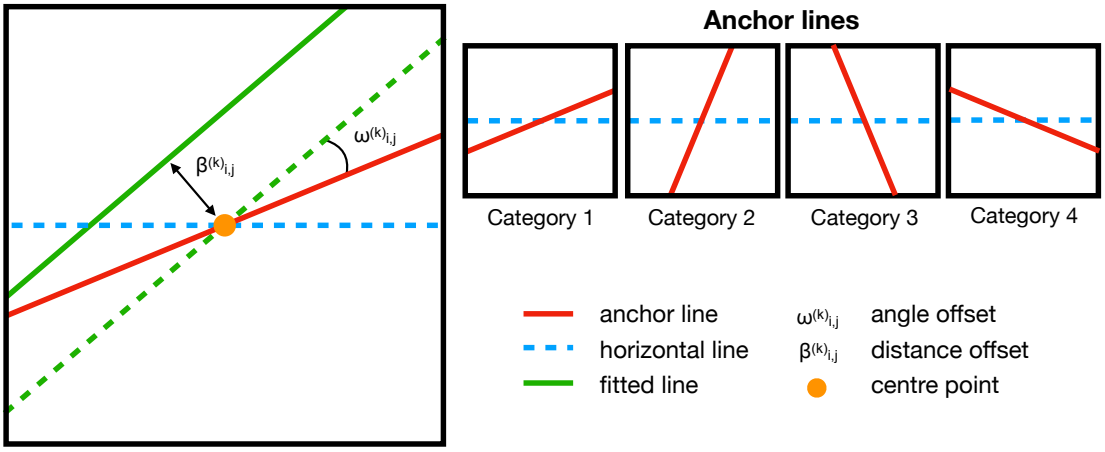


Figure 6.10: Illustration of the line objects used to parametrise the *occluded* road boundaries. In a particular cell, four anchor line classes are provided each at a different angle to the centre point of the cell. *Occluded* road boundaries are to be assigned to one of these four classes after which offsets to both the angle ($\omega^k_{i,j,gt}$), and distance ($\beta^k_{i,j,gt}$) of the lines to the centre-point computed for finer localisation.

to the pixels of the *occluded* road boundaries leading ultimately to a pixel-wise segmentation. To better localise the curbs, offsets were also allowed to the angle and distances of the lines from a defined centre point.

Before training of this object detection network could take place, the pixel-wise curb labels were converted to this new parametrised form. This was by dividing the image into grids of squares over which lines were fitted to the pixel-wise curb labels as illustrated in Figure 6.11. Suleymanov et al. [115] showed that by training a CNN that produces outputs at different scales with these parametrised labels, the anchor lines and their offsets corresponding to each cell of the grid could be estimated. This would then reveal the underlying *occluded* pixels.

By combining these two networks, road boundary pixels can be obtained under various levels of occlusion as demonstrated in Figure 6.12. These pixel-wise outputs however do not carry the semantic information on the road boundaries as this is encoded in the geometric models. A subsequent model-fitting described in the following section is thus required.

6.2.2 Road Boundary Fitting

From the output of the network, we need to transform pixel-wise output into a

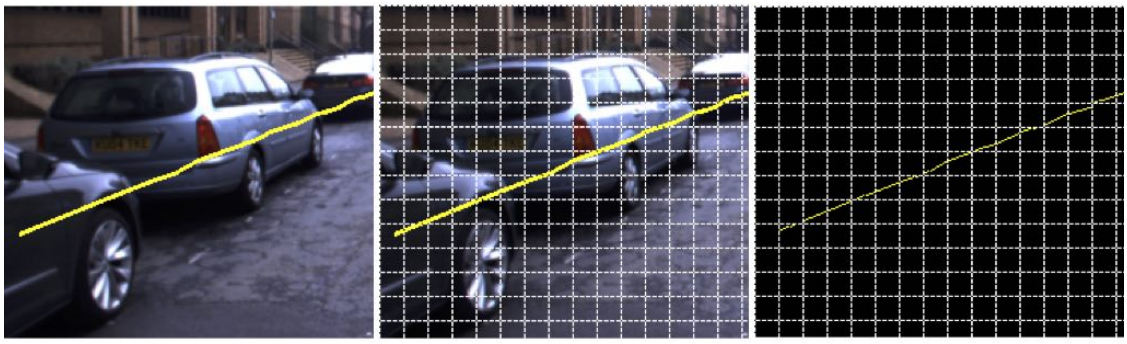


Figure 6.11: Examples of parameterisation of *occluded* road boundary labels. In the left column pixel-wise labels are shown followed by the division of pixel-wise masks into a grid of squares. The final *occluded* road boundary masks are drawn based on parameterised labels in the right column.

Figure 6.12: Sample outputs from the combined networks under different levels of road boundary occlusion. These are able to seamlessly deal with and predict the position of the road boundaries in scenarios of low to no occlusion (left), some occlusion (middle) and full occlusion (right). The blue pixels are the visible road boundaries while the yellow pixels represent the occluded boundaries.

set, with unknown cardinality, of semantically and geometrically meaningful road boundaries. When the motion of the vehicle is parallel to the road boundaries, simple approaches such as using the pixels to the left and right to form two separate boundaries would lead to usable road boundary classes. However, the motion of the vehicle in comparison to the road boundaries set is not known *a priori* and of course road junctions present a larger number of road boundaries where this simple approach as well as other more sophisticated RANSAC approaches fail as seen in Figure 6.13.

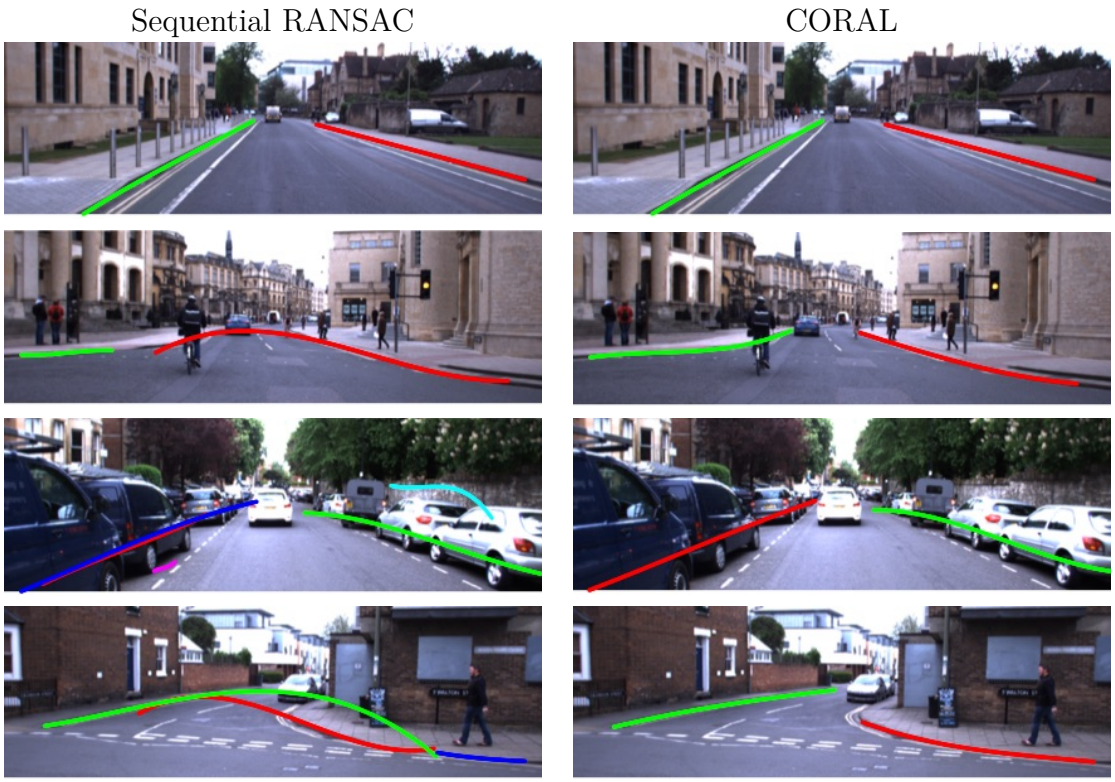


Figure 6.13: Comparison of results of geometric road boundary fitting from the output of the combined networks using Sequential RANSAC and CORAL. While CORAL is able to obtain the minimum set of cubic curves that represent the road boundaries, without any prior information of the number of models, in a diversity of viewed scenes the performance of RANSAC deteriorates in the presence of noise and in more complex scenes returning inaccurate road boundary models.

As before in this thesis we opt for the CORAL formulation given in Chapter 3 to detect the multiple geometric primitives. This allows in our case for a minimal number of best-fit cubic curves (boundaries) to be associated to the network output with the global energy shown in Equation 6.6:

$$\underbrace{\sum_{l=1}^{L}\sum_{i=1}^{n_{rb}}(\|D(\mathbf{A}_l,\mathbf{u}_i)\|)\phi_l(\mathbf{u})}_{\text{Geometric Error Term}} + \lambda \underbrace{\sum_{l=1}^{L}\sum_{i=1}^{n_{rb}}|\nabla_{\mathcal{N}}\phi_l(\mathbf{u})|}_{\text{Smoothness Term}} + \underbrace{\beta\|L\|}_{\text{Compactness Term}} \tag{6.6}$$

Here $\mathbf{A}_l = (a_l, b_l, c_l, d_l, e_l)$ is the equation of the cubic $a_l\mathbf{x}^3 + b_l\mathbf{x}^2 + c_l\mathbf{x} + d_l\mathbf{y} = e_l$ and we refer to D as the shortest distance between a pixel $\mathbf{u}_i = (x_i, y_i)$ and the curve $\mathbf{A}_l$. Minimisation of the energy in Equation 6.6 through the primal dual optimisation in Algorithm 4 reveals the underlying geometric models.

The initialisation of models needed in Algorithm 5 was performed through sequential RANSAC. For this initialisation we do not require RANSAC to run to completion, which would present for each run a single model with some guaranteed probability, rather we only execute it for a fixed few iterations and obtain the best estimate at the last iterations. In this way we can obtain a large number of models for refinement through CORAL without a time-expensive model initialisation. A refinement that converges on L road boundaries with sample results in Figure 6.13 showing that these correspond to the underlying primitives in a diversity of scenes. The parameters needed for the energy minimisation are shown in Table 6.3.

6.3 Implementation

The deep networks in this chapter were then implemented in Python with the tensorflow library while CORAL was in turn implemented using CUDA and deployed on a NVIDIA TITAN GPU. To obtain the running times over both the road marking and road boundary fitting, we averaged the times taken for one hundred different images present in the Oxford RobotCar dataset [10]. The timing results for both of these are presented in Table 6.4 with the two-step approach we adopt in this chapter showing real-time performance (5Hz) in both the road marking and road boundary classification tasks.

Table 6.3: CORAL parameters for the road boundary fitting.

Symbol	Value	Description
γ	2	Constant cost for Outlier Model $\rho_{\emptyset}(\cdot)$
N_n	4	Number of neighbours per data point
λ	2	Regularisation parameter balancing geometric error and smoothness
β	60	Regularisation parameter balancing geometric error and compactness
τ	0.125	Gradient ascent step size for the smoothness dual variable
ν	0.125	Gradient ascent step size for the compactness dual variable
α	0.125	Gradient descent step size for the primal variable
θ	1	Relaxation weight for the primal dual optimisation

Table 6.4: Timing results for the road marking and road boundary classification.

Module	Time (ms)
Road marking pixel detection(U-Net)	16
CORAL (Road marking geometric primitive fitting)	110.8
CORAL (Road marking geometric primitive clustering)	38.4
Visible Curb detection (U-Net)	50
Occluded Curb detection	65
Combined Curb Detection+Post Processing	120
CORAL (Road boundary fitting)	90

6.4 Conclusion

This chapter presents a hybrid approach that aims to detect road markings and boundaries in real-time for interpretation by autonomous vehicles. Unlike many contemporary approaches seen in the literature that require heavily human-annotated data to run end-to-end deep networks, we opt for a two-step approach that allows for the quick semi-supervised accumulation of annotations to train deep segmentation networks that identify pixels belonging to the road markings and boundaries. After which CORAL is used to obtain the geometric models; lines and curves for the road markings and boundaries respectively.

We show that using this approach different road marking and road boundary classes can be classified under vastly different prevailing conditions and levels of occlusions, with qualitative results showing repeatable performance in a number of scenarios.

The semantic classification, needed for road markings, also foreshadows the idea that semantic information is not only encoded in geometric models themselves but also in their relationships with each other. In the next chapter we formalise this through a unified representation for the relationships between different geometric entities in the context of scene understanding and reconstruction.

Alles Gescheite ist schon gedacht worden.
Man muss nur versuchen, es noch einmal zu denken.

All intelligent thoughts have already been thought;
what is necessary is only to try to think them again.

Johann Wolfgang von Goethe

7

A Unified Representation for Scene Reconstruction

Abstract

In many applications utilising geometric models, such as those explored in the previous chapters, most of the salient information sought is usually encoded within the geometric models. For instance, we have seen planes as proxies for walls and curves as proxies for road boundaries. Nonetheless, there are multiple scenarios in which it is the relationships between different geometric models that hold the information, as was foreshadowed in the road marking clustering process. In this chapter, we present a unified representation that formalises the relationships between different geometric models in the context of sparse but large-scale LiDAR reconstructions of the environment.

With this representation, errors caused by drift in estimation over these scales can be compensated for by leveraging the relationships between geometric models that represent architectural objects i.e walls. We show that even with low-cost platforms consisting of a stereo camera and 2D LiDAR, by using this representation we can create large scale reconstructions of comparable accuracy, within the laser noise (0.03cm), to high-fidelity professional surveys.

Contents

7.1 Graph Based Optimisation For Architectural Constraints 122
7.2 Symmetries & Perturbations Model 124
7.2.1 Optimisation 127
7.3 Implementation 130
7.3.1 Constraint Generation 130
7.3.2 Data Association 132
7.4 Results 133
7.5 Conclusion 135

In the previous two chapters, two different applications of geometrical models as components of complex systems have been presented in which we have seen that most of the desired information has been encoded within the geometric models themselves. However, in many cases it is the relationship between different geometric models that is of importance. This chapter presents a unified representation that formalises the different relationships between different geometric models, an aspect which is explored within sparse reconstructions with a 2D laser and a stereo camera that doubles as an odometry source. This combination is able to, at a low-cost, effectively create large-scale reconstructions of the environment as shown in Figure 7.1.

However, odometry sources generally accumulate error and "drift" over time leading to significant errors over large-scale reconstructions. In this work we make use of Visual Odometry, which consists of frame-to-frame motion estimation from a stereo camera, through a geometric hypothesise-and-test architecture followed by least-squares optimisation [23]. This is accurate over small regions but similarly accumulates drift.

To ensure that an accurate large-scale reconstruction can be recovered these errors due to drift must be compensated for. In this chapter, we leverage the relationships between different architectural constraints to improve the motion estimation and subsequent accuracy of the reconstructions. Architectural constraints in this context would refer to man-made objects that can be described through

Figure 7.1: A large scale reconstruction obtained as solution of the graph optimisation, proposed in this chapter, over a 2km trajectory of an urban environment (top). By combining two different sensor modalities, a push broom laser and a stereo camera with a known cross-calibration, we can create low-cost reconstructions of the environment, these are however subject to "drift" that impacts their accuracy. In this chapter we offer a unified framework to handle multiple architectural constraints driven by the different geometric features discovered in the environment (e.g. points, planes, etc), these constraints can then be employed to improve the motion estimation resulting in accurate reconstructions over large-scales (bottom).

a geometric model i.e. walls (planes) and edges (lines). These can be obtained accurately through CORAL.

In the following sections we first present an overview of methods that have been used to improve on reconstructions using architecture as a guide in Section 7.1. Before presenting our proposed formulation in Section 7.2. The particular implementation details are given in Section 7.3 before results are presented and conclusions are drawn in Section 7.4 and 7.5 respectively.

7.1 Graph Based Optimisation For Architectural Constraints

Attempts to improve incremental motion estimation, i.e. reduce odometry drift, have been widely reported with regards to the Simultaneous Localisation And Mapping (SLAM) problem whose main objective was to localise robots within a created map that was almost always sparse. While this is a distinctly different problem from the large-scale reconstruction considered here, the techniques used in the SLAM problem can be leveraged to provide a better motion estimation that can be used to create more accurate reconstructions over large-scales.

Within the literature there are two common approaches used to find this solution. The first method being a filtering solution. In this approach online estimation of the robot's current position or state is carried out, with the estimate being continuously augmented and refined by new measurements as they arrive. This has been implemented using Kalman and information filters [116] as well as particle filters in fastSLAM [117].

More recent and state-of-the-art methods approached the problem via a graph-based formulation which utilises all the measurements that have been taken. This involves the construction of a graph whose nodes represent robot poses or features/landmarks and the edges represents a constraint between the two nodes [118]. With a constructed graph, the solution of this problem is the retrieval of the maximal configuration of nodes that explain the measurements and is a non-linear optimisation problem.

This approach was first proposed by Lu and Milios [119] however it was only widely adopted later when insights in the structure of the SLAM problem were found that resulted in the reduction of the complexity of the optimisation problem. Thrun and Montomerlo [120] applied techniques from variable elimination to the reduction of the dimensionality of this problem. By marginalising over the graph they were able to efficiently optimise over large areas for the construction of urban maps.

In addition, within the SLAM literature multiple attempts to include architectural constraints to the localisation problem have been reported. Early work into this was reported by Weingarten and Siegwart [11] who incorporated planes into a filter-based approach to the SLAM problem. Using a 3D laser scanner planes were generated through a RANSAC algorithm [27] and included into an Extended Kalman Filter (EKF)-formulation via a Symmetry and Perturbations (SP) model [121]. A similar EKF approach was also taken in [12], [13] and [14]. While the inclusion of these features improved the solution of the problem, the intrinsically large computational cost associated with using an EKF severely limited the application of this approach.

Planar features have also been incorporated to graph-based SLAM approaches within the literature with the major reported difference being the representation of the planes and the effect it has on the optimisation process. Lee et al. [122] utilised planes obtained from a RGBD camera for indoor mapping by representing them within the spherical coordinate system prior to graph optimisation. The use of this spherical coordinate system however leaves the optimisation subject to singularities as it is a minimal representation of orientation [16]. Kaess [15] included planes obtained from a RGBD camera to the graph-formulation by defining a minimal representation for the planes similar to that used to represent quaternions. Trevor et al. [17] and Taguchi et al. [18] represented planar features using the plane equations (normal $\mathbf{n}$, distance d). While these different representations showed good results when applied to the SLAM problem they lacked generality, and a new representation had to be derived to cope with geometric entities other than planes.

To overcome this lack of generality, in this work we take advantage of the Symmetries and Perturbations (SP) model which allows for the representation of any geometric entity through an attached coordinate frame. This allows us to encode relationships between different geometric entities as we detail in the following section.

7.2 Symmetries & Perturbations Model

In the SP model a geometric entity e is represented by a reference coordinate frame $\mathbf{T}^e$ attached to it. This reference encapsulates both the entity's position and orientation as expressed below.

$$\mathbf{T}^e = \begin{pmatrix} \mathbf{R}^e & \mathbf{t}^e \\ \mathbf{0} & 1 \end{pmatrix} \in SE(3) \tag{7.1}$$

Where $\mathbf{R}^e \in SO(3)$ is the rotation matrix and $\mathbf{t}^e \in \mathbb{R}^3$ is the 3 dimensional translation vector of the transformation. The SP model is based on the following two principles:

- The error in the position of a geometric entity is represented locally
- A geometric entity is not allowed to move locally along its underlying symmetries

In our case to accomplish the first requirement we denote the current estimate of the location of an entity by $\hat{\mathbf{T}}^e$ whereas its local error is represented by the differential $\mathbf{\Delta}^e$. The current estimation $\hat{\mathbf{T}}^e$ belongs to the Special Euclidean $SE(3)$ Lie Group while the error $\mathbf{\Delta}^e$ is defined in the Lie algebra $\mathfrak{se}(3)$ which allows for manifold optimisation (no constraints required). The true position of the entity is then given by:

$$\mathbf{T}^e = \hat{\mathbf{T}}^e \oplus \mathbf{\Delta}^e \tag{7.2}$$

with $\oplus$ representing the composition of the transformations described later.

The Lie algebra $\mathfrak{se}(3)$ is the tangent space of $SE(3)$ at the identity. The elements of this algebra are the 6-vectors $(\mathbf{w}, \mathbf{v})^T$ where $\mathbf{w} = (w_x, w_y, w_z)$ is the axis-angle representation of rotation and $\mathbf{v}$ is a rotated version of the translation $\mathbf{t}$. An

exponential mapping can be used to map the elements of the $\mathfrak{se}(3)$ algebra back to the $SE(3)$ group as shown in Equation 7.3.

$$\exp_{SE(3)}(\mathbf{w}, \mathbf{v}) = \begin{pmatrix} \exp_{SO(3)}(\mathbf{w}) & \mathbf{V}\mathbf{v} \\ \mathbf{0} & 1 \end{pmatrix} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix} \tag{7.3}$$

with

$$\exp_{SO(3)}(\mathbf{w}) = \mathbf{I} + \frac{sin(\theta)}{\theta}[\mathbf{w}]_\times + \frac{1 - cos(\theta)}{\theta^2}[\mathbf{w}]_\times \tag{7.4}$$

$$\mathbf{V} = \mathbf{I} + \frac{1 - cos(\theta)}{\theta^2}[\mathbf{w}]_\times + \frac{\theta - sin(\theta)}{\theta^2}[\mathbf{w}]_\times \tag{7.5}$$

$$\theta = ||\mathbf{w}||^2 \tag{7.6}$$

and $[\cdot]_\times$ is an operator which maps a 3-vector to its skew-symmetric matrix.

The components of the error $\mathbf{\Delta}^e$ are given by $d\mathbf{v}_\bullet$ for the translation and $d\mathbf{w}_\bullet$ for the rotation where the elements in the subscript describe the allowed axis (x, y, z) along which we can translate or rotate to correct the error.

To fulfil the second requirement of the model, the axis along which a particular entity can be locally moved depends on the symmetries and real Degrees Of Freedom (DOF) of the entity. Figure 7.2 shows the corresponding coordinate frames attached to a point, line, plane and robot pose entities. The corresponding differential movements allowed are also defined in the table with the motion composition given by:

$$\hat{\mathbf{T}}^e \oplus \mathbf{\Delta}^e \stackrel{def}{=} \hat{\mathbf{T}}^e \exp(\mathbf{\Delta}^e) \tag{7.7}$$

For the purpose of this work we focus primarily on the inclusion of plane entities, which can be used to generate architectural constraints in the context of the large data sets being analysed. As can be seen from Figure 7.2, the coordinate frame attached to the plane has its $\mathbf{x}$ and $\mathbf{y}$ vectors lying along the plane with the $\mathbf{z}$ vector being the normal of the plane. From this it can be seen that any translation in the $\mathbf{x}$ and $\mathbf{y}$ direction as well as rotation around the $\mathbf{z}$ axis of this reference will not change the plane equation thus the plane entity has three DOF. By applying the respective rotations sequentially we also retain full control of the symmetry of

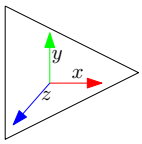
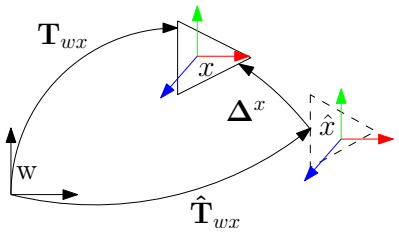
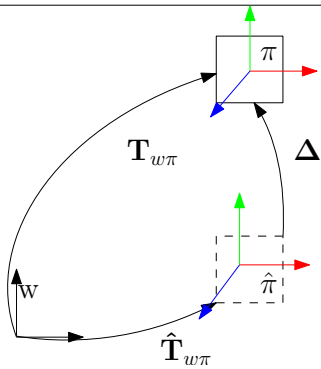
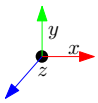
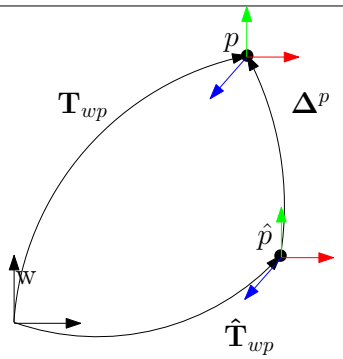
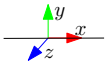
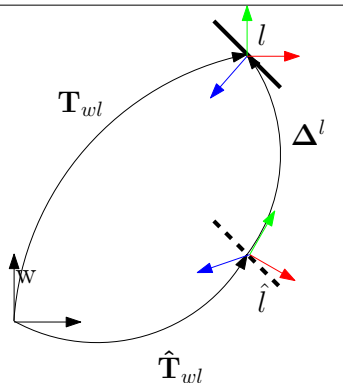
Entity	Reference	Graph Constraint	$\exp(\Delta^e)$	DOF
Pose			$\exp(d\mathbf{v}_{xyz} d\mathbf{w}_{xyz})$	6
Plane			$\exp(d\mathbf{v}_z)\exp(d\mathbf{w}_y)\exp(d\mathbf{w}_x)$	3
Point			$\exp(d\mathbf{v}_{xyz})$	3
Line			$\exp(d\mathbf{v}_{yz})\exp(d\mathbf{w}_z)\exp(d\mathbf{w}_y)$	4

Figure 7.2: Description of the proposed geometric entities and constraints: a pose T^x, plane T^π, point T^p and line T^l with their respective representation within the proposed Lie algebra framework. The attached reference frame as well as the graphical representation of the entities are described together with the construction of $\exp(\mathbf{\Delta}^e)$, the exponential mapping of the differential to preserve the symmetry of each of the individual entities.

the entities. In this way the $\exp(\Delta^e)$ fully reflects the degrees of freedom available to the specific entity as shown in Figure 7.2.

7.2.1 Optimisation

The optimisation problem solves for a configuration of parameters $(\mathbf{T}^e)^*$ that explains all the measurements by minimising the following cost function:

$$\mathbf{F}(\mathbf{T^e}) = \sum_{i,j \in C} \mathbf{e}_{ij}(\mathbf{T}_i^e, \mathbf{T}_j^e, \mathbf{z}_{ij})^T \Omega_{ij} \mathbf{e}_{ij}(\mathbf{T}_i^e, \mathbf{T}_j^e, \mathbf{z}_{ij}) \tag{7.8}$$

where $\mathbf{e}_{ij}(\mathbf{T}_i^e, \mathbf{T}_j^e, \mathbf{z}_{ij})$ is an error function that measures how well the parameter blocks $\mathbf{T}_i^e$ and $\mathbf{T}_j^e$ satisfy the constraint whose mean is given by $\mathbf{z}_{ij}$ and covariance by Ω_{ij}. The set C represents the pairs of geometric entities for which a constraint exists.

The error function $\mathbf{e}_{ij}$ used for all constraints in this framework is defined as:

$$\mathbf{e}_{ij} \overset{def}{=} \mathbf{e}_{ij}(\mathbf{T}_i^e, \mathbf{T}_j^e, \mathbf{z}_{ij}) \tag{7.9}$$

$$\mathbf{e}_{ij} = \log(\mathbf{E}^{ee}) \tag{7.10}$$

The $\log(\cdot)$ function is the inverse of the exponential map defined in Equation 7.3 and maps the Lie group $\mathbb{SE}(3)$ to its corresponding Lie algebra $\mathfrak{se}(3)$. In our implementation of our SP model we utilise the residual matrix $\mathbf{E}^{ee}$ to represent the error between different geometric entities. Where as in Figure 7.2 we use a superscript x for pose entities, π for plane entities, p for point entities and l for line entities.

Pose-Pose residual

The first constraint used in this framework is that between two poses where the residual matrix is given as:

$$\mathbf{E}^{xx} = (\mathbf{T}_j^x)^{-1} \mathbf{T}_i^x \mathbf{z}_{ij} \tag{7.11}$$

with $\mathbf{z}_{ij}$ defined as the relative transformation measurement made between the poses.

Point-Plane residual

The first architectural constraint considered is that between a point i and a plane j whose corresponding transformations are given by:

$$\mathbf{T}^{\pi_j} = \begin{pmatrix} \mathbf{n}_x^{\pi_j} & \mathbf{n}_y^{\pi_j} & \mathbf{n}_z^{\pi_j} & \mathbf{p}^{\pi_j} \\ 0 & 0 & 0 & 1 \end{pmatrix} \in SE(3) \tag{7.12}$$

$$\mathbf{T}^{p_i} = \begin{pmatrix} \mathbf{I} & \mathbf{p}^i \\ \mathbf{0} & 1 \end{pmatrix} \in SE(3) \tag{7.13}$$

These arise when a point lies on a plane or the distance z_{ij} between a point and a plane is measured. The corresponding residual matrix is given as:

$$\mathbf{E}^{\pi p} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \mathbf{n}_z^{\pi_j} \cdot (\mathbf{p}^i - \mathbf{p}^{\pi_j}) - z_{ij}) \\ 0 & 0 & 0 & 1 \end{pmatrix} \in SE(3) \tag{7.14}$$

Plane-Plane Angle residual

Another architectural constraint arises when the angle z_{ij} between two planes is known, for example for two parallel or orthogonal walls indoors. The corresponding residual matrix between the planes $\mathbf{T}^{\pi_i}$ and $\mathbf{T}^{\pi_j}$ is:

$$\mathbf{E}^{\pi\pi} = \begin{pmatrix} cos(\phi) & -sin(\phi) & 0 & 0 \\ sin(\phi) & cos(\phi) & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \in SE(3) \tag{7.15}$$

$$\phi = z_{ij} - acos(\mathbf{n}_z^{\pi_i}.\mathbf{n}_z^{\pi_j}) \tag{7.16}$$

Error minimisation

As compared to the other optimisation problems we have encountered so far, the error minimisation here requires a non-linear optimisation and thus we cannot apply the techniques explored in Chapter 3.2.1. Instead, a numerical solution to perform least squares optimisation using the Gauss-Newton (GN) or Levenberg-Marquardt (LM) algorithms can be derived [123]. In this section we refer to $\mathbf{\Delta}^e$ as the state vector consisting of all the differentials with $\mathbf{\Delta}_i^e$ referring to the individual differential of entity i. Similarly we define the shorthand:

$$\mathbf{e}_{ij}(\hat{\mathbf{T}}^e \oplus \mathbf{\Delta}^e) \stackrel{def}{=} \mathbf{e}_{ij}(\hat{\mathbf{T}}_i^e \oplus \mathbf{\Delta}_i^e, \hat{\mathbf{T}}_j^e \oplus \mathbf{\Delta}_j^e, \mathbf{z}_{ij}) \tag{7.17}$$

The idea employed by LM and GN is to approximate the error function by linearising around the current estimate using a first order Taylor expansion.

$$\mathbf{e}_{ij}(\hat{\mathbf{T}}^e \oplus \boldsymbol{\Delta}^e) \approx \hat{\mathbf{e}}_{ij} + \mathbf{J}_{ij}\boldsymbol{\Delta}^e \tag{7.18}$$

The Jacobian $\mathbf{J}_{ij}$ is given by

$$\mathbf{J}_{ij} = \left.\frac{\partial \mathbf{e}_{ij}(\mathbf{T}^e \oplus \boldsymbol{\Delta}^e)}{\partial \boldsymbol{\Delta}^e}\right|_{\boldsymbol{\Delta}^e=0} \tag{7.19}$$

Building up from Equation 7.8.

$$\begin{aligned} F_{ij}(\mathbf{T}^e) &= \mathbf{e}_{ij}(\hat{\mathbf{T}}^e \oplus \boldsymbol{\Delta}^e)\Omega_{ij}\mathbf{e}_{ij}(\hat{\mathbf{T}}^e \oplus \boldsymbol{\Delta}^e) \\ &\approx (\mathbf{e}_{ij} + \mathbf{J}_{ij} \oplus \boldsymbol{\Delta}^e)\Omega_{ij}(\mathbf{e}_{ij} + \mathbf{J}_{ij} \oplus \boldsymbol{\Delta}^e) \\ &= \underbrace{\mathbf{e}_{ij}^T\Omega_{ij}\mathbf{e}_{ij}}_{c_{ij}} + 2\underbrace{\mathbf{e}_{ij}^T\Omega_{ij}\mathbf{J}_{ij}}_{b_{ij}}\boldsymbol{\Delta}^e + (\boldsymbol{\Delta}^e)^T\underbrace{\mathbf{J}_{ij}^T\Omega_{ij}\mathbf{J}_{ij}}_{H_{ij}}\boldsymbol{\Delta}^e \\ &= c_{ij} + 2b_{ij}\boldsymbol{\Delta}^e + (\boldsymbol{\Delta}^e)^T H_{ij}\boldsymbol{\Delta}^e \end{aligned}$$

Applying this to Equation 7.8

$$\begin{aligned} \mathbf{F}(\mathbf{T}^e) &= \sum_{\langle i,j\rangle \in C} F_{ij}(\mathbf{T}^e) \\ &= \sum_{\langle i,j\rangle \in C} c_{ij} + 2b_{ij}\boldsymbol{\Delta}^e + \boldsymbol{\Delta}^e H_{ij}\boldsymbol{\Delta}^e \\ &= \mathbf{c} + 2\mathbf{b}^T\boldsymbol{\Delta}^e + (\boldsymbol{\Delta}^e)^T\mathbf{H}\boldsymbol{\Delta}^e \end{aligned}$$

This form is obtained by setting $\mathbf{c} = \sum c_{ij}$,$\mathbf{b} = \sum b_{ij}$ and $\mathbf{H} = \sum H_{ij}$. The solution can be found as below

$$\mathbf{H}\boldsymbol{\Delta}^e = -\mathbf{b} \tag{7.20}$$

The update is obtained by composing the solution $\boldsymbol{\Delta}^e$ obtained in the current iteration k with the previous estimate $\hat{\mathbf{T}}_k^e$.

$$\hat{\mathbf{T}}_{k+1}^e = \hat{\mathbf{T}}_k^e \oplus \boldsymbol{\Delta}^e \tag{7.21}$$

The linearisation, solution and update step can be iterated through until convergence. In our implementation we use the Ceres solver [124] with automatic differentiation to calculate the Jacobians in Equation 7.19.

7.3 Implementation

With the optimisation procedure for the geometric features described it remains to generate and incorporate features, in this thesis we concentrate on planes, into the framework. The generation of planes is done through segmentation of a 3D point-cloud that represents the platform's environment using CORAL.

To generate the 3D point-cloud the trajectory is divided into sections of user-defined distance with the poses at the start of the sections defined as Key Poses K_i $i \in \{1, \cdots, n_s\}$ where n_s is the number of sections in the trajectory. Within these sections the error produced by the effect of drift on the VO is small enough such that the point cloud generated within the section can be considered accurate. Plane segmentation is therefore carried out over the section's point cloud followed by data association between different sections.

7.3.1 Constraint Generation

With the 3D point cloud over a section defined, the next step is segmenting the point cloud into planes. To do this we use the CORAL formulation to perform the plane extraction. We define a global energy function to be minimised as below:

$$\underbrace{\sum_{l=1}^{L}\sum_{i=1}^{n_p}(\|D(\mathbf{P}_l, \mathbf{u}_i)\|)\phi_l(\mathbf{u})}_{\text{Geometric Error Term}} + \lambda \underbrace{\sum_{l=1}^{L}\sum_{i=1}^{n_p}|\nabla_{\mathcal{N}}\phi_l(\mathbf{u})|_1}_{\text{Smoothness Term}} + \underbrace{\beta\|L\|}_{\text{Compactness Term}} \tag{7.22}$$

Here $\mathbf{P}_l = (n_x, n_y, n_z, d)$ is the Plane equation $n_x\mathbf{x} + n_y\mathbf{y} + n_z\mathbf{z} = d$ and we refer to D as the shortest Euclidean distance between a point $\mathbf{u}_i = (x_i, y_i, z_i)$ and the plane $\mathbf{P}_l$, n_p is the number of points in the 3D point cloud. Minimisation of the energy in Equation 7.22 through the primal dual optimisation in Algorithm 4 reveals the underlying planes. The initialisation of models needed in Algorithm 4 is performed using a curvature-based algorithm [125]. This utilises the relationship between a point in the point cloud and the small local region neighbouring it. If the local region is planar the curvature of the region is expected to be low and the normals of the points within the region will also point in the same direction. By comparing the residual and normals of a point to certain user-defined thresholds

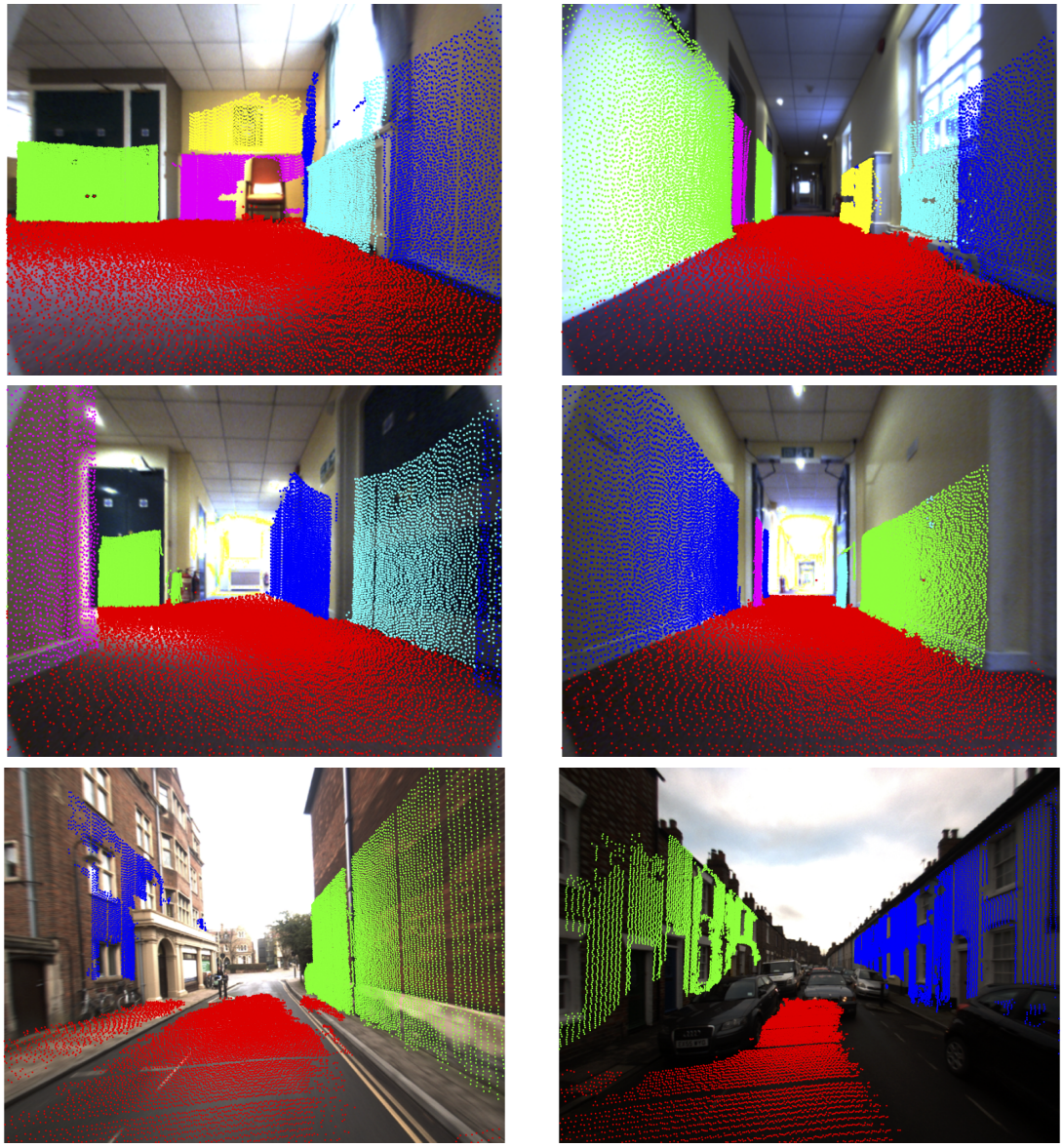


Figure 7.3: Sample of results for plane-fitting through CORAL on small sections of 3D LiDAR point-clouds projected onto camera images. It can be seen that CORAL can in a variety of cases, including indoor and outdoors, fit planes accurately to the LiDAR points which in this image are colour-coded according to their assigned models.

it is possible to 'grow' a plane from the local regions. The energy minimisation as has been shown in previous applications converges to L planes which as can be seen in Figure 7.3 correspond to the underlying planes. The parameters for this energy minimisation using CORAL are given in Table 7.1.

The addition of the reference coordinate frame used to incorporate the plane into the SP model requires the definition of two orthogonal vectors that lie on the

plane $\mathbf{n}_x^\pi$, $\mathbf{n}_y^\pi$, a normal vector $\mathbf{n}_z^\pi$ and a point $\mathbf{p}^\pi$ belonging to the plane as shown in Figure 7.2 and defined in Equation 7.12. To do this two points that belong to the plane are chosen from which the vector between them is computed. This vector is in turn orthogonal to the normal vector from the plane equation $\mathbf{n}_l = (n_x, n_y, n_z)$

7.3.2 Data Association

In large-scale reconstructions there are two elements of data association that are considered, that between planes found in successive sections and those between planes in sections that are revisited resulting in a loop closure that can be detected by techniques such as FABMAP [126]. In both cases we only associate planes if they are coplanar.

All the planes generated in a section are represented in the frame of their corresponding Key Pose. To compare two planes represented in different sections with Key Poses K_a and K_b a transformation $\mathbf{T}_{K_aK_b} \in SE(3)$ is used to represent the planes in section b in the frame of Key Pose a. For successive sections we use VO to calculate $\mathbf{T}_{K_aK_b}$ while for the loop closures we use pairs of stereo images matched by FABMAP to compute the transformation.

The coplanar test between two planes $\mathbf{T}^{\pi_a}$ and $\mathbf{T}^{\pi_b}$ can be performed by comparing the angle θ_{ab} and distance d_{ab} between the planes with user-defined angle and distance thresholds respectively.

$$\theta_{ab} = \arccos((\mathbf{n}_z^{\pi_a})^T.(\mathbf{R}_{K_aK_b}.\mathbf{n}_z^{\pi_b})) \tag{7.23}$$

$$d_{ab} = \mathbf{n}_z^{\pi_a}(\mathbf{c}^{\pi_a} - (\mathbf{R}_{K_aK_b}.\mathbf{c}^{\pi_b} + \mathbf{t}_{K_aK_b})) \tag{7.24}$$

Table 7.1: CORAL parameters for the plane fitting in a 3D pointcloud.

Symbol	Value	Description
γ	0.03m	Constant cost for Outlier Model $\rho_\emptyset(\cdot)$
N_n	4	Number of neighbours per data point
λ	0.1	Regularisation parameter balancing geometric error and smoothness
β	30	Regularisation parameter balancing geometric error and compactness
τ	0.125	Gradient ascent step size for the smoothness dual variable
ν	0.125	Gradient ascent step size for the compactness dual variable
α	0.125	Gradient descent step size for the primal variable
θ	1	Relaxation weight for the primal dual optimisation

$\mathbf{c}^{\pi} \in \mathbb{R}^3$ is the centroid of a plane. If θ_{ab} and d_{ab} are less than their respective thresholds the planes are coplanar and can be associated as one.

7.4 Results

To test this proposed framework four independent surveys were carried out. We generated planes from the dense laser map and added them as constraints for the optimisation. The results of the two surveys of an indoor environment, the Acland building in Keble College Oxford, and the two of an outdoor environment, a triangular loop in the Jericho area Oxford are shown in Figure 7.4. The first column of the figure shows the reconstructions created using the motion estimation provided by VO, with the reconstructions created after the optimisation being shown in the second column.

For the indoor surveys taken, the ground truth is available via a professional survey taken of the same environment. The comparison of these surveys to the professional survey after alignment via Iterative Closest Point (ICP) algorithm is shown in the second column of Figure 7.4 coupled with the histogram of the error. For the first survey a Root Mean Square (RMS) error of 0.1125 metres was observed for the points with the platform travelling for approximately 200 metres. More importantly the median can be seen to lie below the laser noise threshold of 0.03 metres thus through the optimisation a reconstruction comparable in accuracy to the professional survey can be obtained.

In the second denser survey, as the platform was moved at a slower speed, reported a higher RMS error of 0.2257 metres over a similar distance travelled was reported. In addition, several windows and doors, which were open when the professional survey was taken, were closed when taking this survey which contributed greatly to the increase in error.

In the outdoor environment however ground truth data was not available, for the comparison the two outdoor surveys were compared against each other with the RMS error of this comparison coming to 0.943 metres with the platform covering two loops of approximately 0.9 kilometres each. While this is a significant reduction

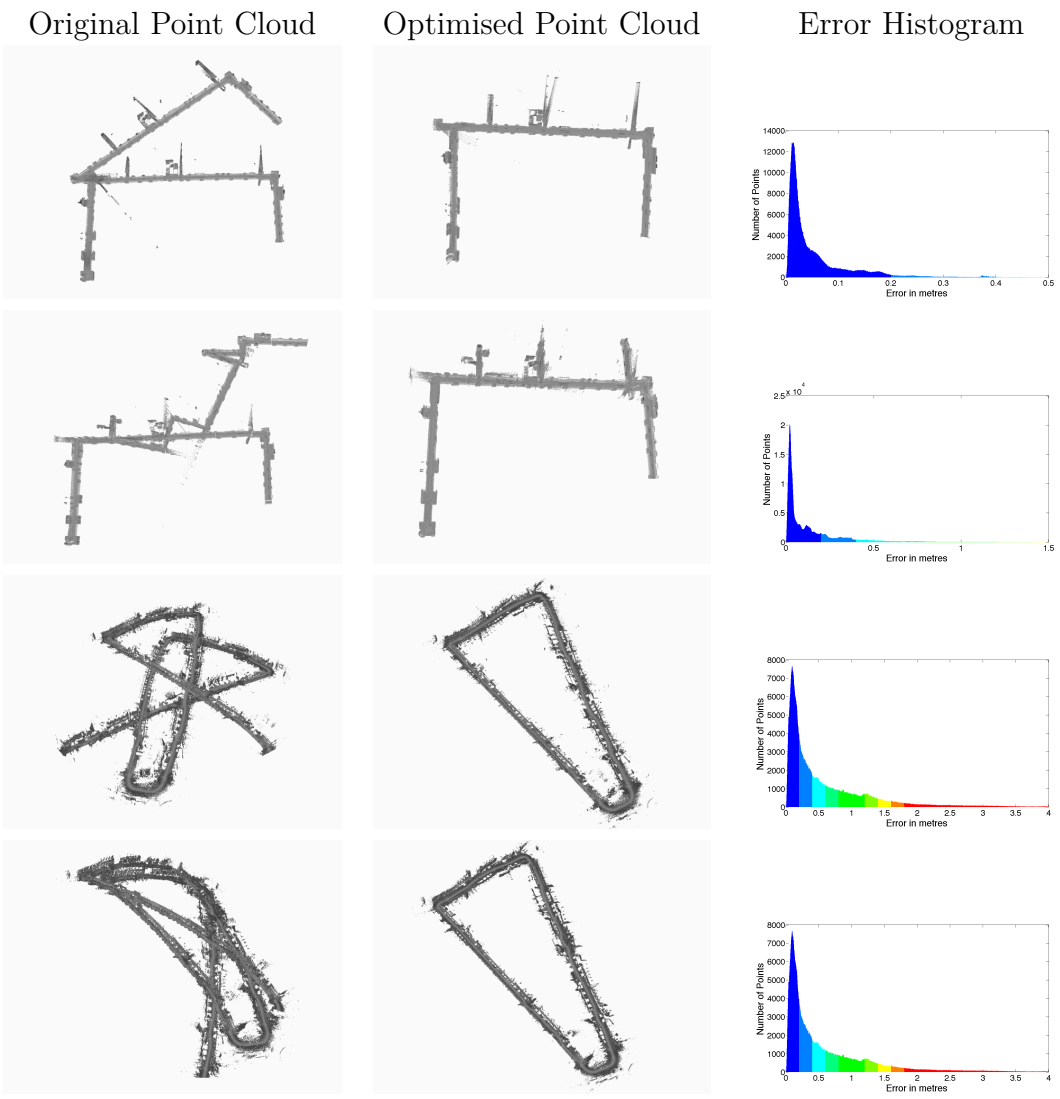


Figure 7.4: Figure showing the results of 4 surveys taken and optimised with this framework. The original surveys created using VO are shown in column 1, while a comparison of the surveys after optimisation with architectural constraints together with their subsequent histograms is shown in columns 2 and 3 respectively. Rows 1 and 2 present results from 2 independent indoor surveys of the Acland building in Oxford where the ground truth dense map is available courtesy of a professional survey and used for the comparisons. In row 3 two different independent outdoor maps are presented (grey and blue) of the Jericho triangle in Oxford. Since ground truth is not available the comparison is done between the 2 point clouds optimised independently.

in error of the reconstructions, it highlights the difference between indoor and outdoor reconstructions. In the indoor case the entire environment can be described by a configuration of planes i.e two sets of orthogonal planes found in corridors as shown in Figure 7.5 making the final result very accurate while this is not the case

Table 7.2: Number of constraints used in each of the surveys utilising the proposed unified optimisation framework.

Experiment	Pose-Pose	Plane-Point	Plane-Plane	Num of Planes	Time per iteration (s)
Acland 1	24523	84524	13	186	0.9
Acland 2	62421	252124	9	194	3.168
Jericho 1	114816	20860	0	125	3.163
Jericho 2	112164	21869	0	133	2.895

in the outdoor environment limiting the final accuracy of the model.

Table 7.2 provides details of the number of architectural constraints used during each of the surveys carried out in this unified optimisation framework. Additionally, we list the time required for each iteration in each of the evaluated datasets.

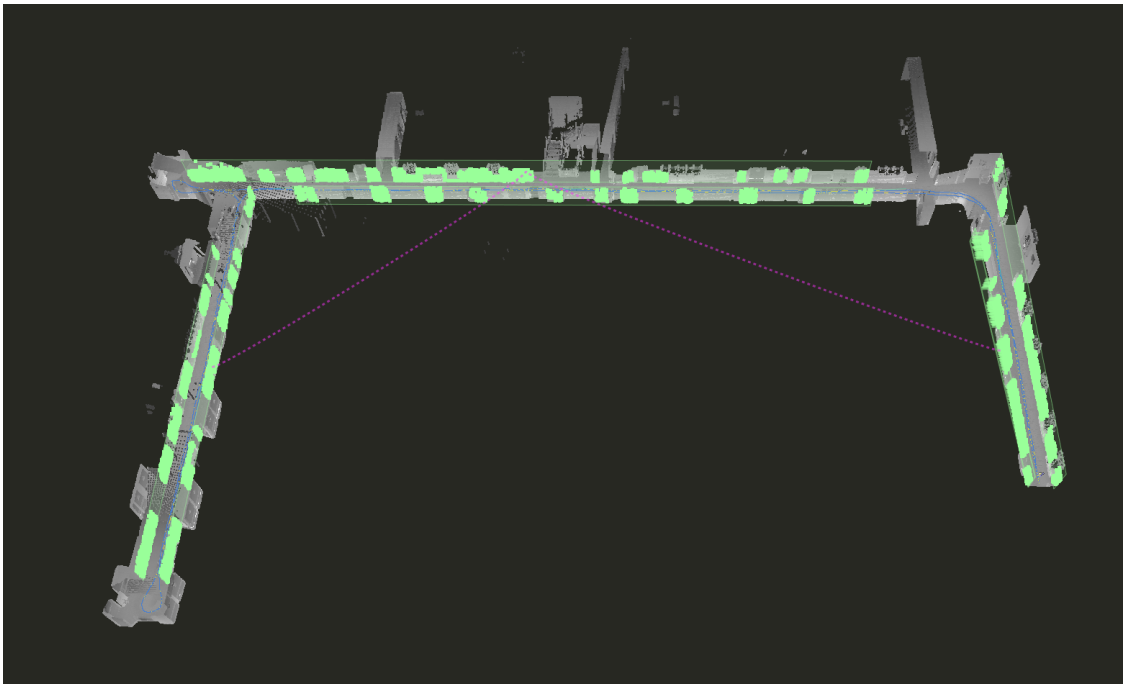


Figure 7.5: This figure shows the configuration of planes from the survey of the Acland building. The structure of this environment can be described by the automatically generated orthogonal relationships (in purple) between the three sets of planes (in green) forming the corridors.

7.5 Conclusion

In this chapter we have presented a unified framework that encompasses the relationships between different geometric models and applied it to the understanding

and reconstruction of scenes captured by a low-cost platform consisting of a 2D LiDAR sensor in the push-broom orientation and an associated calibrated camera. By coupling the motion estimation from the camera and the 2D LiDAR scans, 3D reconstructions of the environment could be made. Visual Odometry which provides the motion estimates however, drifts over time which leads to inaccuracy over the resulting reconstructions when large-scale environments are considered.

However by leveraging relationships between geometric models, most notably those between planes that were detected by CORAL, or architectural constraints as we refer to them we are able to directly compensate for this drift. With experiments showing that we obtain final reconstructions with accuracy comparable to high-resolution static professional surveys of the same scenes.

In the beginning the Universe was created.
This has made a lot of people very angry and been widely regarded as a bad move.

Douglas Adams

8 Conclusions

Contents

8.1 Future Work 140
8.2 Closing Remarks 142

This thesis aims to introduce a fast, accurate and robust geometric multi-model fitting algorithm. Geometry as we show in Chapter 1 has played a defining role in how we as humans understand the world around us and has been used from early models of the universe to more contemporary technological fields such as computer vision and robotics. However, geometry is seldom directly observed making the fitting of geometric models to observed data a crucial task. The fitting of multiple geometric models is however not trivial: as it is a chicken and egg problem where data must be clustered according to the parameters of geometric models that must also be estimated. Further, the fitting of geometric models is complicated by noise and clutter which inevitably appear in observed data.

While the multi-model fitting algorithm proposed in this thesis is agnostic to a specific application, we are particularly interested in incorporating geometric models into contemporary mobile robotics platforms. In these platforms, the trade-off between accuracy and speed of geometric model extraction is critical for safety of the platform and those it interacts with. We thus introduced in Chapter 2 some

of the components that underpin modern robotic platforms as they traverse the world, with a view towards evaluating our proposed multi-model fitting algorithm on these. The reference frames and transformations described in this chapter enable us to denote the precise location of a particular sensor or platform in time. As cameras and LiDAR are the main sensor modalities present in these robotic platforms, we reviewed each of these and the importance of accurate calibration between the sensors mounted on the platforms.

Chapter 3 then began with a comprehensive review of multi-model fitting techniques and their differences before presenting the energy based approach that we formulated in this thesis. In this review it can be seen that a large number of contemporary approaches rely on greedily maximising the inliers of a particular model while ignoring the overall classification of data points to models. This makes them ill-suited to multi-model fitting especially in the presence of noise and clutter. In contrast, energy-based approaches can incorporate not only the geometric model errors but a spatial smoothness and compactness component as in the energy function below:

$$\mathbf{E}(\phi) = \underbrace{\sum_{l=1}^{L} \int_{\Omega} \rho_l(\mathbf{u}, \phi_l(\mathbf{u})) d\Omega}_{\text{Geometric Error Energy}} + \lambda \underbrace{\sum_{l=1}^{L} \int_{\Omega} \omega_{\mathcal{N}} R(\nabla_{\mathcal{N}} \phi_l(\mathbf{u})) d\Omega}_{\text{Smoothness Energy}} + \underbrace{\beta \|L\|}_{\text{Compactness Energy}}$$

Minimising this energy thus reveals an optimal labelling of data points to models that is robust to noise and clutter. Additionally, whereas existing approaches had opted for a "hard" assignment of data points to models, making this energy function non-convex, we opt for a relaxation to a "soft" assignment that convexifies this energy creating the **CO**nvex **R**elaxation **AL**gorithm (CORAL). This relaxation allows for advanced optimisation techniques in the continuous domain to be used as opposed to the combinatorial approaches that have been seen in the literature. It can be observed that the parallelisation potential of these techniques offers CORAL an advantage over the combinatorial techniques that require sequential evaluation and thus suffer as the resolution of the data increases. CORAL intrinsically boosts run-time performance by simultaneously handling per-point evaluations making

our approach more suitable for geometric model extraction in applications with real-time performance constraints. Thus by bringing in powerful optimisation machinery into the solution of geometric multi-model fitting CORAL offers an algorithm that is simultaneously able to robustly extract accurate models in the presence of contamination and improve time performance over the state-of-the-art.

In Chapter 4 the CORAL formulation is evaluated and its performance benchmarked against other state-of-the-art multi-model fitting algorithms. This is through two differently structured applications, which both encompass plane extraction in different environment. Firstly through homography detection from sparse features matched in two camera views, followed by plane detection in dense RGBD images. In both scenarios CORAL reports performance that is as good as or better than other state-of-the-art multi-model fitting algorithms.

Chapter 5 presents the first application of CORAL as a component of a robotic system; in this case CORAL is used for improving the estimation of depth-maps from images. In particular we focus on depth-estimation of large, plain and planar surfaces which have long plagued contemporary dense depth-map estimation techniques due to lack of abundant texture. We leverage CORAL's model discovery technique over homographies to induce planar priors over these planar regions. This is then similarly employed to associate and label the underlying image pixels to their corresponding planar model to improve the depth-map estimation. The labelling is further aided by an image segmentation that removes the pixels that belong to non-planar regions, reducing the noise and increasing the labelling accuracy. As the energy minimisations in this work rely on CORAL that allows for a parallel implementation on GPGPU hardware, this proposed pipeline is open to online use in robotics platforms with real-time requirements. This could reduce the reliance on expensive 3D sensors in favour of lower-cost visual sensors allowing for the increased development and deployment of robotic platforms. The subsequent fusion of the improved depth-maps also reveals a more complete reconstruction of the environment with little change in the reconstruction error.

Chapter 6 then presents a hybrid approach that aims to detect road markings and boundaries in real-time for interpretation by autonomous vehicles. Unlike many contemporary approaches seen in the literature that require heavily human-annotated data to run end-to-end deep networks, we opt for a two-step approach that allows for the quick semi-supervised accumulation of annotations to train deep segmentation networks to identify pixels belonging to the road markings and boundaries. After which CORAL is used to obtain the geometric models, lines and curves for the road markings and boundaries respectively. We show that with this approach that we can classify different road marking classes under vastly different prevailing conditions. Similarly, road boundaries are also reliably detected under different levels of occlusion in various scenarios.

In Chapter 7 we present a unified framework that encompasses the relationships between different geometric models and apply it to the understanding of a scene traversed by a platform consisting of a 2D LiDAR sensor and a camera. As the motion estimation given through VO drifts in time this leads to inaccuracy over the resulting reconstructions when large-scale environments are considered. However by leveraging relationships between geometric models, most notably those between planes that were detected by CORAL, we are able to directly compensate for the drift where experiments show that we are able to obtain final reconstructions with accuracy comparable to high-resolution static professional surveys, within laser noise, of the same scenes.

8.1 Future Work

While this thesis has presented theory and experiments that contribute to various communities, there are a number of ways in which it can be improved. We present four general suggestions that could be pursued.

Expand CORAL to fit different models simultaneously.

In this thesis, we have concentrated on the fitting of one type of geometric model while there are multiple scenarios where different types of geometric models need to

be fit simultaneously. Expanding our formulation to simultaneously accommodate models of different complexity would enable CORAL to find application over more problems.

Consolidate energy minimisations over photoconsistency.

In the pipeline proposed in Chapter 5 for the improvement of depth map estimation it can be seen that there are multiple energy minimisations over a photoconsistency data term and a spatial smoothness term. This is seen firstly to provide the initial depth map through TGV regularisation and then again to perform the per-pixel labelling to planar priors. These two optimisations in the proposed pipeline are done independently while they operate over the same variables, by consolidate these into a singular energy minimisation not only will this reduce the run-time of the algorithm but could improve the accuracy of the depth-map estimation as planar and non-planar pixels would be optimised simultaneously.

Annotations through CORAL

In the hybrid approach to understanding the road presented in Chapter 6 we are motivated, especially in the road marking case, by the difficulty of obtaining ground-truth annotations for the training of deep networks. The output of the road marking semantic classification is however, a classification of image pixels to different road-marking classes. These could be utilised to train a deep-learning network for the classification task rather than just the segmentation task. In this way this hybrid-approach can be expanded to serve as a means for the swift collection of annotations for deep learning networks.

Expand relationships over the SP model

In Chapter 7 we presented a uniform formulation that represents the relationships between geometric models. In this chapter we probe the relationships between lines, planes and points and showed that they are sufficient for accurate optimisation in large-scale reconstructions. In smaller-scale reconstructions more complicated

models such as generalised cylinders, spheres and their relationships might be of more importance than the infinite line and plane models. By expanding the SP model to include these entities more accurate reconstructions over multiple scales.

8.2 Closing Remarks

This thesis sought to contribute to the state-of-the-art on geometric multi-model fitting. Our primary contirbution is the CORAL formulation for fast, parrallel and robust multi-model fitting that was presented in Chapter 3 and evaluated in Chapter 4. After which we presented different applications of this algorithm tied to robotic mapping in Chapter 5, perception in Chapter 6 and scene reconstruction and understanding in Chapter 7.

The algorithms and systems presented, requiring more than 200k lines of CUDA, C++ and Matlab (which we provide in the Appendix) code. With the result being a fast and accurate multi-model fitting algorithm for use in multiple scenarios which we hope will be a valuable resource for the community.

Machines take me by surprise with great frequency.

Alan Turing

9 CORAL Matlab Implementation

In this Chapter we present Matlab Functions needed to implement CORAL for geometric multi-model fitting. These follow the procedures presented in Section 3.2.4. A script that performs this is presented below, followed by its constituent functions.

```
%% ============================== MRG Copyright Header ==============================
%%
%% Copyright (c) 2003-2018 University of Oxford. All rights reserved.
%% Authors: Paul Amayo, Mobile Robotics Group, University of Oxford
%%          http://mrg.robots.ox.ac.uk
%%
%% This file is the property of the University of Oxford.
%% Redistribution and use in source and binary forms, with or without
%% modification, is not permitted without an explicit licensing
%% agreement (research or commercial). No warranty, explicit
%% or implicit, provided.
%%
%% ============================== MRG Copyright Header ==============================
%-------------------------------------------------------------------------------------
% CORAL formulation for geometric multi-model fitting using a global
% energy functional that is then minimised.
%-------------------------------------------------------------------------------------
% Input: set of data points
% Output: set of geometric models of known type


%% Collect the target data points into a set
u=[u_1,u_2,...,u_n];

%% Create a gradient function defined over the num_neighbours nearest
    neighbours of the points
nabla=CalculateGradient( u, num_points, num_neighbours );

%% Create a set of initial models from the data set
```

```
[theta_init, num_models]=InitialiseModels(u);

%% Define the constant fidelity cost for the outlier model
gamma=...;

%% Initialise the variables for the energy minimisation
alpha=...;
tau=...;
nu=...;
theta=...;
beta=...;
lambda=...;
thresh_pd=...;
thresh_mod=...;

%% Perform the energy minimisation to obtain the optimised geometric
    models
[phi, theta_model]=EnergyMinimisation(u, theta_model_init, nabla,...
                                      alpha, tau, nu, lambda,...
                                        beta, theta, gamma,...
                                      num_points, num_labels,...
                                      num_neighbours, thresh_pd,...
                                      thresh_mod);
```

Listing 9.1: Script that implements CORAL to perform geometrical multi-model fitting and returns a set of optimised models.

```
function [ nabla ] = CalculateGradient( u, num_points,num_neighbours )
  % CalculateGradient- Calculates the gradient operator defined over
    the neighbours of the data points
  % INPUTS:
  %   u: Set of data points
  %   num_points: Number of data points in the set u
  %   num_neighbours: Number of neighbours for each data point
  % OUTPUTS:
  %   nabla: Gradient operator defined over the neighbourhood of data
    points

  % Search the points for the nearest neighbours
  [index_neighbors, distances_neighbors] = knnsearch(u,u,'K',...
                              num_neighbours+1);

  % Remove the indexes that belong to the point itself
  index_neighbors(:,1) = [];
  distances_neighbors(:,1) = [];

  % Fill the nabla matrix with the nearest neighbours
  i_values=[];
  j_values=[];
  z_values=[];
  for i = 1:num_points
      for j = 1:num_neighbours
          index_point = i + (j-1)*num_points;
          i_values = [i_values index_point index_point];
          j_values = [j_values i  index_neighbors(i,j)];
```

```
            z_values = [z_values -1  1];
        end
    end
    nabla = sparse(i_values,j_values,z_values);
```

Listing 9.2: Function that calculates the gradient operator nabla.

```
function [ theta_model_init, num_models ] = InitialiseModels( u )
    % InitialiseModels- Create an initial set of models from the data
      points
    % INPUTS:
    %   u: Set of data points
    % OUTPUTS:
    %   nabla: Gradient operator defined over the neighbourhood of data
      points

    % Define a strategy for obtaining initial geometric models e.g.
    % through MSS or RANSAC
    theta_model_init=...;
    % Get the number of models
    num_models=length(theta_model_init);
```

Listing 9.3: Function that creates the initial geometric models needed for the energy minimisation.

```
function [phi, theta_model]=EnergyMinimisation(u, theta_model_init,
    nabla, alpha, tau, nu, lambda, beta, theta, gamma, num_points,
    num_labels, num_neighbours, thresh_pd, thresh_mod)
    % EnergyMinimisation - Performs the primal dual optimisation
    % INPUTS:
    %   u: Set of data points
    %   theta_model_init: Set of geometric models parameters
    %   nabla: Gradient function defined over the neighbourhood of the
    %       data points
    %   alpha: Step size for the smoothness dual maximisation
    %   nu: Step size for the compactness dual maximisation
    %   tau: Step size for the primal minimisation
    %   theta: Relaxation parameter for the primal variable
    %   lambda: Regularisation parameter that balances the smoothness
    %       and geometric error energies
    %   beta: Regularisation parameter that balances the compactness
    %       and geometric error energies
    %   gamma: Constant fidelity cost for the outlier model
    %   num_points: Number of data points in the set u.
    %   num_labels: Number of labels being optimised.
    %   thresh_pd: Threshold used to determine if the primal dual
    %       optimisation has converged
    %   thresh_mod: Threshold used to determine if the global energy and
    %       thus the geometric models has converged
    % OUTPUTS:
    %   phi: Indicator function with assigned membership of data points
    %       to geometric models
    %   Theta: Set of optimised geometric model parameters


```

```
29  % Calculate the initial model costs
30  model_cost  = CalculateGeometricCost( u, theta_model_init,...
31                                         num_labels, num_points,...
32                                         gamma);
33  converge_models=false;
34  while(~converge_models)
35    % Initialise the Convex Optimisation Variables
36    phi=zeros(num_points,num_labels);
37    phi_bar=zeros(num_points,num_labels);
38    lambda_compact=zeros(num_points,num_labels);
39    psi=zeros(num_neighbours*num_points,num_labels);
40    converge_pd=false;
41    %Primal Dual Optimisation
42      while(~converge_pd)
43        % Update the primal and dual variables
44          psi=UpdateSmoothnessDual(psi,phi_bar,nabla,lambda,alpha);
45          lambda_compact=UpdateCompactnessDual(lambda_compact,...
46                                               phi_bar,...
47                                               beta, nu,...
48                                               num_points);
49          phi_prev=phi;
50          [phi,phi_bar]=UpdatePrimal(phi, phi_bar, model_cost,...
51                                     nabla,psi, lambda_compact,...
52                                     lambda, beta, tau,...
53                                     num_labels);
54         % Check for convergence
55          if norm(phi-phi_prev)<thresh_pd
56            converge_pd=true;
57          end
58      end;
59      % Calculate the final energy after the primal dual optimisation
60      energy_pd=CalculateEnergy(model_cost,nabla,phi,lambda,beta);
61
62      % Binarise the indicator function to retrieve a unique labelling
63      phi=BinarisePhi(phi);
64
65      % Book-keeping to remove models that no longer have any support
66      [phi,num_labels]=ReduceLabels(phi,num_labels);
67
68      % Re-estimate the geometric models based on the new parameters
69      theta=RecalculateModels(theta,phi,u);
70      model_cost  = CalculateGeometricCost( u, Theta, num_labels,...
71                                            num_points, gamma );
72
73      % Calculate the energy after the model re-parameterisation and
74      % check for energy convergence
75      energy_re_est=CalculateEnergy(model_cost, nabla, phi, lambda,...
76                                    beta);
77      if abs(energy_pd-energy_re_est)<thresh_pd
78          converge_pd=true;
79      end
```

```
80   end
```

Listing 9.4: Function that performs the energy minimisation by iterating through Algorithms 4 and 5 respectively to return an optimal labelling of data points to models and the parameters of these models.

```
function [ model_cost ] = CalculateGeometricCost( u, theta_model,
    num_labels, num_points, gamma )
  % CalculateGeometricCost- Calculates the geometric errors of data to
     a set of models
  % INPUTS:
  %   u: Set of data points
  % theta_model: Set of geometric models
  % num_labels: Number of labels being optimised.
  %   num_points: Number of data points in the set u.
  %   gamma: Constant fidelity cost for the outlier model
  % OUTPUTS:
  % model_cost: matrix containing the geometric errors of data to a
    set of models

  %Initialise the model cost matrix with the value of the outlier
    model
   model_cost=ones(num_points,num_labels)*gamma;

  %Update the model cost with the geometric error between a point j
    and a model i
    for j=1:num_points
        for i=1:num_labels-1
          rho=theta_model(i);
          model_cost(j,i)=GeometricError(rho,u(j));
        end
    end
end

function cost=GeometricError(rho,u_j)
  % GeometricError- Calculates geometric error between a point and a
  % geometric model
  % INPUTS:
  %   u_j: Data point
  %   rho: Parameters of a geometric model
  % OUTPUTS:
  %   cost: geometric error between the point and the model

   cost=...;
end
```

Listing 9.5: Functions that perform the geometric cost computation from a set of data points and geometric models.

```
function psi=UpdateSmoothnessDual(psi,phi_bar,nabla,lambda,alpha);
  % UpdatePsi-Performs the maximisation of the smoothness dual
  % variable, psi
  % INPUTS:
  %   psi: Dual variable for the smoothness energy
  %   phi_bar: Relaxed version of the indicator function phi
```

```
%   nabla: Gradient function defined over the neighbourhood of the
%       data points
%   lambda: Regularisation parameter that balances the smoothness
%       and geometric error energies
%   alpha: Step size for the smoothness dual maximisation
% OUTPUTS:
%   psi: Dual variable for the smoothness energy

% Perform the gradient ascent
psi=psi+lambda*alpha*nabla*phi_bar;

% Project this updated variable back to the feasible set
l_1=L1Norm(psi,num_points,num_labels,num_neighbours);
l_1=max(1,l_1);
psi=psi./l_1;
```

Listing 9.6: Function that performs the maximisation through gradient ascent of the smoothness dual variable psi.

```
function l1_mat  = L1Norm(psi, num_points, num_labels, num_neighbours)
% L1Norm - Calculates the L1 norm of the psi matrix
% INPUTS:
%   psi: Dual variable for the smoothness energy
%   alpha: Step size for the smoothness dual maximisation
%   num_points: Number of data points in the set u.
%   num_labels: Number of labels being optimised.
%   num_neighbours: Number of neighbours for each data point
% OUTPUTS:
%   l1_mat: L1 norm of the dual variable psi

% Initialise the l1 val
l1_val=zeros(num_points,num_labels);
neighbour_steps=0:1:num_neighbours;
init_index=1;
% Compute the L1 norm through the following operation:
% |psi|_n1 +|psi|_n2 + .... |psi|_num_neighbours
for i=1:num_neighbours
    final_index=neighbour_steps(i+1)*num_points;
    l1_val=abs(l1_val)+abs(Psi(init_index:final_index,:));
    init_index=final_index+1;
end
% Replicate this for the other neighbours
l1_mat=repmat(l1_val,num_neighbours,1);
```

Listing 9.7: Function that computes the L1 norm of the smoothness dual variable psi.

```
function lambda_compact=UpdateCompactnessDual(lambda_compact, phi_bar,
    beta, nu, num_points)
% UpdateCompactnessDual - Performs the maximisation of the
% compactness dual variable, lambda_compact
% INPUTS:
%   lambda_compact: Dual variable for the compactness energy
%   phi_bar: Relaxed version of the indicator function phi
%   beta: Regularisation parameter that balances the compactness
%       and geometric error energies
```

```
  %   nu: Step size for the smoothness dual maximisation
  %   num_points: Number of data points in the set u.
  % OUTPUTS:
  %   lambda_compact: Dual variable for the compactness energy

  % Perform the gradient ascent
  lambda_dual=lambda_dual+beta*nu*phi_bar;
  % Project this back to the feasible set
  lambda_dual=Simplexprojection(Lambda',num_points)';

end
```

Listing 9.8: Function that performs the maximisation through gradient ascent of the compactness dual variable lambda_compact.

```
function [phi,phi_bar]=UpdatePrimal(phi, phi_bar, model_cost, nabla,
    psi, lambda_compact, lambda, beta, tau, num_labels)
  % PrimalUpdate - Performs the minimisation of the primal variable
  % phi
  % INPUTS:
  %   phi: Indicator function
  %   phi_bar: Relaxed version of the indicator function phi
  %   model_cost: matrix containing the geometric errors of data to
  %       a set of models
  %   nabla: Gradient function defined over the neighbourhood of the
  %       data points
  %   psi: Dual variable for the smoothness energy
  %   lambda_compact: Dual variable for the compactness energy
  %   lambda: Regularisation parameter that balances the smoothness
  %       and geometric error energies
  %   beta: Regularisation parameter that balances the compactness
  %       and geometric error energies
  %   nu: Step size for the primal minimisation
  %   num_labels: Number of labels being optimised.
  % OUTPUTS:
  %   phi: Indicator function
  %   phi_bar: Relaxed version of the indicator function phi

  % Save and then perform gradient descent on the primal variable phi
  phi_prev=phi;
  phi=phi-tau*(model_cost+lambda*nabla'*Psi+beta*Lambda);

  % Project this back to the feasible set and update the relaxed
    phi_bar variable
  phi=Simplexprojection(phi,num_labels);
  phi_bar=phi+theta*(phi-phi_prev);
```

Listing 9.9: Function that performs the minimisation through gradient descent of the primal variable phi.

```
function [ delta ] = Simplexprojection( delta, num_rows )
  % Simplexprojection - Projects a variable delta onto the simplex
  % INPUTS:
  %   delta: Variable to be projected to the simplex
  %   num_rows: Number of rows in the variable delta .
```

```
  % OUTPUTS:
  %   delta: Variable projected to the simplex

  delta_size=size(delta);
  for j=1:delta_size(1)
      % Get the current row
      delta_vec=delta(j,:);
      converge = false;
      while (converge==false)
          sum = 0;
          ni = 0;
          converge = true;
          for i=1:num_labels
              a = delta_vec(i);
              if (a ~= 0)
                  ni=ni+1;
                  sum =sum+ a;
              end
          end
          if (ni)
              a = (sum - 1) / ni;
              for  i=1:num_rows
                  if (delta_vec(i) ~= 0)
                      delta_vec(i) =delta_vec(i)- a;
                      if (delta_vec(i) < 0)
                          converge = false;
                          delta_vec(i) = 0;
                      end
                  end
              end
          else
              delta_vec = 1/num_labels;
          end
      end
      delta(j,:)=delta_vec;
  end
```

Listing 9.10: Function that projects variables onto the simplex.

```
function phi=BinarisePhi(phi,num_points,num_labels)
  % BinarisePhi- Performs a binarisation of the indicator function.
  % INPUTS:
  %   phi: Indicator function
  %   num_points: Number of data points in the set u.
  %   num_labels: Number of labels being optimised.
  % OUTPUTS:
  %   phi: Indicator function

  % get the indices of the largest values of the indicator function
    phi.
  [~,phi_labels]=max(phi,2);
  phi_binary=zeros(num_points,num_labels);

  % Assign a value of 1 to these indices
  for i=1:num_points
```

```
    point_label=phi_labels(i);
    phi_binary(i,point_label)=1;
  end
  phi=phi_binary;
```

Listing 9.11: Function that binarises the *soft* indicator function and in so doing creates a unique assignment of points to models.

```
function theta_model=RecalculateModels(u,phi,num_labels)
  % RecalculateModels- Recalculate the parameters of the geometric
  % models based on the assignment of points to models given by the
  % indicator function phi.
  % INPUTS:
  %   u: Set of data points
  %   phi: Indicator function
  %   num_points: Number of data points in the set u
  %   num_labels: Number of labels being optimised
  % OUTPUTS:
  %   theta_model: Set of geometric model parameters

  % get the indices of the largest values of the indicator
  % function phi.
  [~,phi_labels]=max(phi,2);

  for i=1:num_labels-1
      % get the indices of the points assigned to the model i
      current_label=find(phi_labels==i);
      % collect a set of points assigned to the model i
      u_curr=u(current_label);
      % Compute the geometric parameters of the model from this set
      Theta(i)=FitModel(u_curr);
  end
end

function rho=FitModel(u_input)
  % FitModel - Fits a model of known type to the input data
  % INPUTS
  %   u_input: Set of input data points
  % OUTPUTS
  %     rho: parameters of fit model
  rho=...;
end
```

Listing 9.12: Functions that recalculate the parameters of the geometric models given a new assignment through the indicator function phi.

```
function [phi, num_labels]=ReduceLabels(phi,num_labels)
  % ReduceLabels - Removes models with no support from the energy
  %  minimisation problems
  % INPUTS
  %   phi: Indicator function
  %   num_labels: Number of labels being optimised
  % INPUTS
  %   phi: Indicator function
  %   num_labels: Number of labels  being optimised
```

```
10  i=1;
11  while i<params.ContOpt_num_labels
12      % query a row of phi to see if it has support
13      phi_curr=find(phi(:,i)==1);
14      i=i+1;
15      % Replace rows  and reduce labels if a model with no support
16      % is found.
17      if(isempty(phi_curr))
18          for j=(i-1):num_labels-1
19              phi(:,j)=phi(:,j+1);
20          end
21          num_labels=num_labels-1;
22          i=1;
23        end
24    end
```

Listing 9.13: Function that performs book-keeping to make sure that models with no support are removed from subsequent energy minimisation steps

The first kind of intellectual and artistic personality belongs to the hedgehogs, the second to the foxes ...

— Sir Isaiah Berlin

References

[1] Physics of the universe. *Aristotle's Universe*. https://www.physicsoftheuniverse.com/cosmological.html. [Online; accessed 17-June-2018]. 2018.

[2] Encyclopaedia Brittanica. *Kepler's laws of planetary motion*. https://www.britannica.com/science/Keplers-laws-of-planetary-motion. [Online; accessed 17-June-2018]. 2018.

[3] Upper Canada Stretchers. *Fine Art Education*. https://www.ucsart.com/fine-art-education/learn-the-golden-ratio. [Online; accessed 17-June-2018]. 2018.

[4] Sameer Agarwal et al. "Building rome in a day". In: *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE. 2009, pp. 72–79.

[5] Kevin M Judd, Jonathan D Gammell, and Paul Newman. "Multimotion Visual Odometry (MVO): Simultaneous Estimation of Camera and Third-Party Motions". In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Madrid, Spain, Jan. 2018.

[6] Dennis Zill, Warren S Wright, and Michael R Cullen. *Advanced engineering mathematics*. Jones & Bartlett Learning, 2011.

[7] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[8] Terry Scott et al. "Exploiting known unknowns: Scene induced cross-calibration of lidar-stereo systems". In: *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE. 2015, pp. 3647–3653.

[9] Terry Scott et al. "Choosing a time and place for calibration of lidar-camera systems". In: *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE. 2016, pp. 4349–4356.

[10] Will Maddern et al. "1 Year, 1000 km: The Oxford RobotCar dataset". In: *The International Journal of Robotics Research* 36.1 (2017), pp. 3–15. eprint: http://dx.doi.org/10.1177/0278364916679498. URL: http://dx.doi.org/10.1177/0278364916679498.

[11] Jan Weingarten and Roland Siegwart. "3D SLAM using planar segments". In: *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*. IEEE. 2006, pp. 3062–3067.

[12] Fabien Servant et al. "Visual planes-based simultaneous localization and model refinement for augmented reality". In: *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*. IEEE. 2008, pp. 1–4.

[13] Andrew P Gee et al. “Discovering higher level structure in visual SLAM”. In: *Robotics, IEEE Transactions on* 24.5 (2008), pp. 980–990.

[14] José Martínez-Carranza and Andrew Calway. “Unifying Planar and Point Mapping in Monocular SLAM.” In: *BMVC*. 2010, pp. 1–11.

[15] Michael Kaess. “Simultaneous localization and mapping with infinite planes”. In: *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE. 2015, pp. 4605–4611.

[16] F Sebastian Grassia. “Practical parameterization of rotations using the exponential map”. In: *Journal of graphics tools* 3.3 (1998), pp. 29–48.

[17] Alexander JB Trevor, JG Rogers, and Henrik I Christensen. “Planar surface SLAM with 3D and 2D sensors”. In: *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE. 2012, pp. 3041–3048.

[18] Yuichi Taguchi et al. “Point-plane SLAM for hand-held 3D sensors”. In: *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE. 2013, pp. 5182–5189.

[19] G-Q Wei and SD Ma. “A complete two-plane camera calibration method and experimental comparisons”. In: *Computer Vision, 1993. Proceedings., Fourth International Conference on*. IEEE. 1993, pp. 439–446.

[20] Zhengyou Zhang. “Flexible camera calibration by viewing a plane from unknown orientations”. In: *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*. Vol. 1. Ieee. 1999, pp. 666–673.

[21] Zhengyou Zhang. “A flexible new technique for camera calibration”. In: *IEEE Transactions on pattern analysis and machine intelligence* 22 (2000).

[22] Peter F Sturm and Stephen J Maybank. “On plane-based camera calibration: A general algorithm, singularities, applications”. In: *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.* Vol. 1. IEEE. 1999, pp. 432–437.

[23] David Nistér, Oleg Naroditsky, and James Bergen. “Visual odometry”. In: *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*. Vol. 1. IEEE. 2004, pp. I–652.

[24] M. Dzitsiuk et al. “De-noising, Stabilizing and Completing 3D Reconstructions On-the-go using Plane Priors”. In: *International Conference on Robotics and Automation (ICRA)*. May 2017.

[25] Yizhong Zhang et al. “Online structure analysis for real-time indoor scene reconstruction”. In: *ACM Transactions on Graphics (TOG)* 34.5 (2015), p. 159.

[26] Ricardo Cabral and Yasutaka Furukawa. “Piecewise planar and compact floorplan reconstruction from images”. In: *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. IEEE. 2014, pp. 628–635.

[27] Martin A Fischler and Robert C Bolles. “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography”. In: *Communications of the ACM* 24.6 (1981), pp. 381–395.

[28] Michael J Black and P Anandan. “A framework for the robust estimation of optical flow”. In: *Computer Vision, 1993. Proceedings., Fourth International Conference on*. IEEE. 1993, pp. 231–236.

[29] George W Brown, Alexander M Mood, et al. "On median tests for linear hypotheses". In: *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*. The Regents of the University of California. 1951.

[30] Peter Meer et al. "Robust regression methods for computer vision: A review". In: *International journal of computer vision* 6.1 (1991), pp. 59–70.

[31] Philip HS Torr. "Geometric motion segmentation and model selection". In: *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* 356.1740 (1998), pp. 1321–1340.

[32] Etienne Vincent and Robert Laganiére. "Detecting planar homographies in an image pair". In: *Image and Signal Processing and Analysis, 2001. ISPA 2001. Proceedings of the 2nd International Symposium on*. IEEE. 2001, pp. 182–187.

[33] Marco Zuliani, Charles S Kenney, and BS Manjunath. "The multiransac algorithm and its application to detect planar homographies". In: *Image Processing, 2005. ICIP 2005. IEEE International Conference on*. Vol. 3. IEEE. 2005, pp. III–153.

[34] Yasushi Kanazawa and Hiroshi Kawakami. "Detection of planar regions with uncalibrated stereo using distributions of feature points." In: *BMVC*. Citeseer. 2004, pp. 1–10.

[35] Lei Xu, Erkki Oja, and Pekka Kultanen. "A new curve detection method: randomized Hough transform (RHT)". In: *Pattern recognition letters* 11.5 (1990), pp. 331–338.

[36] Dorin Comaniciu and Peter Meer. "Mean shift: A robust approach toward feature space analysis". In: *IEEE Transactions on pattern analysis and machine intelligence* 24.5 (2002), pp. 603–619.

[37] Wei Zhang and Jana Kosecka. "Nonparametric estimation of multiple structures with outliers". In: *Dynamical Vision*. Springer, 2007, pp. 60–74.

[38] Roberto Toldo and Andrea Fusiello. "Robust multiple structures estimation with j-linkage". In: *European conference on computer vision*. Springer. 2008, pp. 537–547.

[39] Luca Magri and Andrea Fusiello. "T-linkage: A continuous relaxation of j-linkage for multi-model fitting". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2014, pp. 3954–3961.

[40] Luca Magri and Andrea Fusiello. "Robust Multiple Model Fitting with Preference Analysis and Low-rank Approximation." In: *BMVC*. 2015, pp. 20–1.

[41] Trung T Pham et al. "The random cluster model for robust geometric fitting". In: *IEEE transactions on pattern analysis and machine intelligence* 36.8 (2014), pp. 1658–1671.

[42] Luca Magri and Andrea Fusiello. "Multiple Model Fitting as a Set Coverage Problem". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 3318–3326.

[43] Hossam Isack and Yuri Boykov. "Energy-based geometric multi-model fitting". In: *International journal of computer vision* 97.2 (2012), pp. 123–147.

[44] Jin Yu, Tat-Jun Chin, and David Suter. "A global optimization approach to robust multi-model fitting". In: *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE. 2011, pp. 2041–2048.

[45] Andrew Delong et al. “Fast approximate energy minimization with label costs”. In: *International journal of computer vision* 96.1 (2012), pp. 1–27.

[46] Stan Birchfield and Carlo Tomasi. “Multiway cut for stereo and motion with slanted surfaces”. In: *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*. Vol. 1. IEEE. 1999, pp. 489–495.

[47] Song Chun Zhu and Alan Yuille. “Region competition: Unifying snakes, region growing, and Bayes/MDL for multiband image segmentation”. In: *IEEE transactions on pattern analysis and machine intelligence* 18.9 (1996), pp. 884–900.

[48] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. “Grabcut: Interactive foreground extraction using iterated graph cuts”. In: *ACM transactions on graphics (TOG)*. Vol. 23. 3. ACM. 2004, pp. 309–314.

[49] Ramin Zabih and Vladimir Kolmogorov. “Spatially coherent clustering using graph cuts”. In: *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*. Vol. 2. IEEE. 2004, pp. II–II.

[50] Philip HS Torr and David W Murray. “Stochastic motion clustering”. In: *European Conference on Computer Vision*. Springer. 1994, pp. 328–337.

[51] Hongdong Li. “Two-view motion segmentation from linear programming relaxation”. In: *Computer Vision and Pattern Recognition, 2007. CVPR’07. IEEE Conference on*. IEEE. 2007, pp. 1–8.

[52] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[53] Yuri Boykov, Olga Veksler, and Ramin Zabih. “Fast approximate energy minimization via graph cuts”. In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 23.11 (2001), pp. 1222–1239.

[54] Christopher Zach et al. “Fast Global Labeling for Real-Time Stereo Using Multiple Plane Sweeps.” In: *VMV*. 2008, pp. 243–252.

[55] Julian Besag. “On the statistical analysis of dirty pictures”. In: *Journal of the Royal Statistical Society. Series B (Methodological)* (1986), pp. 259–302.

[56] Stuart Geman and Donald Geman. “Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images”. In: *IEEE Transactions on pattern analysis and machine intelligence* 6 (1984), pp. 721–741.

[57] H Bauschke and Yves Lucet. “What is a Fenchel conjugate”. In: *Notices of the AMS* 59.1 (2012).

[58] Ankur Handa et al. “Applications of Legendre-Fenchel transformation to computer vision problems”. In: *Department of Computing at Imperial College London. DTR11-7* 45 (2011).

[59] Thomas Pock and Antonin Chambolle. “Diagonal preconditioning for first order primal-dual algorithms in convex optimization”. In: *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE. 2011, pp. 1762–1769.

[60] Jing Yuan and Yuri Boykov. “TV-Based Multi-Label Image Segmentation with Label Cost Prior.” In: *BMVC*. Vol. 2. 3. 2010, p. 4.

[61] C. Nieuwenhuis, E. Toeppe, and D. Cremers. "A Survey and Comparison of Discrete and Continuous Multi-label Optimization Approaches for the Potts Model". In: *Int. Journal of Computer Vision* 104.3 (2013), pp. 223–240.

[62] Olga Veksler and Andrew. Delong. *Multi-label optimization.* http://vision.csd.uwo.ca/code. [Online; accessed 19-November-2016]. 2016.

[63] Leslie M Goldschlager, Ralph A Shaw, and John Staples. "The maximum flow problem is log space complete for P". In: *Theoretical Computer Science* 21.1 (1982), pp. 105–111.

[64] Christian Michelot. "A finite algorithm for finding the projection of a point onto the canonical simplex of R^n". In: *Journal of Optimization Theory and Applications* 50.1 (1986), pp. 195–200.

[65] Hoi Sim Wong et al. "Dynamic and hierarchical multi-structure geometric model fitting". In: *2011 International Conference on Computer Vision.* IEEE. 2011, pp. 1044–1051.

[66] Pushmeet Kohli Nathan Silberman Derek Hoiem and Rob Fergus. "Indoor Segmentation and Support Inference from RGBD Images". In: *ECCV.* 2012.

[67] Luca Magri and Andrea. Fussielo. *T-Linkage.* http://www.diegm.uniud.it/fusiello/demo/jlk/. [Online; accessed 17-June-2018]. 2018.

[68] Daniel Barath, Levente Hajder, and Jiri Matas. "Multi-H: Efficient Recovery of Tangent Planes in Stereo Images". In: *BMVC, 27th British Machine Vision Conference.* 28 (2016), p. 32.

[69] Andreas Geiger, Philip Lenz, and Raquel Urtasun. "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite". In: *Conference on Computer Vision and Pattern Recognition (CVPR).* 2012.

[70] Zhengyou Zhang. "Microsoft kinect sensor and its effect". In: *IEEE multimedia* 19.2 (2012), pp. 4–10.

[71] Leonid I Rudin, Stanley Osher, and Emad Fatemi. "Nonlinear total variation based noise removal algorithms". In: *Physica D: nonlinear phenomena* 60.1-4 (1992), pp. 259–268.

[72] Kristian Bredies, Karl Kunisch, and Thomas Pock. "Total generalized variation". In: *SIAM Journal on Imaging Sciences* 3.3 (2010), pp. 492–526.

[73] Michael Tanner et al. "Keep Geometry in Context: Using Contextual Priors for Very-Large-Scale 3D Dense Reconstructions". In: *Robotics: Science and Systems, Workshop on Geometry and Beyond: Representations, Physics, and Scene Understanding for Robotics.* June 2016.

[74] Pedro Piniés, Lina Maria Paz, and Paul Newman. "Dense and Swift Mapping with Monocular Vision". In: *Int. Conf. on Field and Service Robotics (FSR). Toronto, ON, Canada.* 2015.

[75] Alejo Concha et al. "Manhattan and Piecewise-Planar Constraints for Dense Monocular Mapping." In: *Robotics: Science and systems.* 2014.

[76] Carolina Raposo, Michel Antunes, and Joao P Barreto. "Piecewise-planar stereoscan: structure and motion from plane primitives". In: *European Conference on Computer Vision.* Springer. 2014, pp. 48–63.

[77] Jozsef Molnar, Rui Huang, and Zoltan Kato. "3D reconstruction of planar surface patches: A direct solution". In: *Asian Conference on Computer Vision*. Springer. 2014, pp. 286–300.

[78] Michel Antunes, João P Barreto, and Urbano Nunes. "Piecewise-planar reconstruction using two views". In: *Image and Vision Computing* 46 (2016), pp. 47–63.

[79] Sudipta N Sinha, Drew Steedly, and Richard Szeliski. "Piecewise planar stereo for image-based rendering". In: *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE. 2009, pp. 1881–1888.

[80] Yasutaka Furukawa et al. "Manhattan-world stereo". In: *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE. 2009, pp. 1422–1429.

[81] Yasutaka Furukawa et al. "Reconstructing building interiors from images". In: *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE. 2009, pp. 80–87.

[82] David Gallup, Jan-Michael Frahm, and Marc Pollefeys. "Piecewise planar and non-planar stereo for urban scene reconstruction". In: *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE. 2010, pp. 1418–1425.

[83] David Gallup et al. "Real-time plane-sweeping stereo with multiple sweeping directions". In: *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*. IEEE. 2007, pp. 1–8.

[84] Herbert Bay et al. "Speeded-up robust features (SURF)". In: *Computer vision and image understanding* 110.3 (2008), pp. 346–359.

[85] Anubhav Agarwal, CV Jawahar, and PJ Narayanan. "A survey of planar homography estimation techniques". In: *Centre for Visual Information Technology, Tech. Rep. IIIT/TR/2005/12* (2005).

[86] Carolina Raposo and Joao P Barreto. "Theory and Practice of Structure-from-Motion using Affine Correspondences". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 5470–5478.

[87] Daniel Barath and Levente Hajder. "Novel ways to estimate homography from local affine transformations". In: (2016).

[88] Brendan J Frey and Delbert Dueck. "Clustering by passing messages between data points". In: *science* 315.5814 (2007), pp. 972–976.

[89] Olivier D Faugeras and Francis Lustman. "Motion and structure from motion in a piecewise planar environment". In: *International Journal of Pattern Recognition and Artificial Intelligence* 2.03 (1988), pp. 485–508.

[90] Tobias Pohlen et al. "Full-Resolution Residual Networks for Semantic Segmentation in Street Scenes". In: *arXiv preprint arXiv:1611.08323* (2016).

[91] German Ros et al. "Vision-based offline-online perception paradigm for autonomous driving". In: *Applications of Computer Vision (WACV), 2015 IEEE Winter Conference on*. IEEE. 2015, pp. 231–238.

[92] Michael Tanner et al. “BOR2G: Building Optimal Regularised Reconstructions with GPUs (in cubes)”. In: *International Conference on Field and Service Robotics (FSR)*. Toronto, ON, Canada, June 2015.

[93] Aurélien Géron. *Hands-on machine learning with Scikit-Learn and TensorFlow: concepts, tools, and techniques to build intelligent systems*. " O'Reilly Media, Inc.", 2017.

[94] Xavier Glorot and Yoshua Bengio. “Understanding the difficulty of training deep feedforward neural networks”. In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. 2010, pp. 249–256.

[95] Alex Kendall, Matthew Grimes, and Roberto Cipolla. “Posenet: A convolutional network for real-time 6-dof camera relocalization”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 2938–2946.

[96] David Eigen, Christian Puhrsch, and Rob Fergus. “Depth map prediction from a single image using a multi-scale deep network”. In: *Advances in neural information processing systems*. 2014, pp. 2366–2374.

[97] Fayao Liu, Chunhua Shen, and Guosheng Lin. “Deep convolutional neural fields for depth estimation from a single image”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 5162–5170.

[98] Iro Laina et al. “Deeper depth prediction with fully convolutional residual networks”. In: *3D Vision (3DV), 2016 Fourth International Conference on*. IEEE. 2016, pp. 239–248.

[99] Alex Kendall et al. “Learning to Drive in a Day”. In: *arXiv preprint arXiv:1807.00412* (2018).

[100] Chen Liu et al. “PlaneNet: Piece-wise Planar Reconstruction from a Single RGB Image”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 2579–2588.

[101] Georg Maier, Sebastian Pangerl, and Andreas Schindler. “Real-time detection and classification of arrow markings using curve-based prototype fitting”. In: *Intelligent Vehicles Symposium (IV), 2011 IEEE*. IEEE. 2011, pp. 442–447.

[102] Xinxin Du and Kok Kiong Tan. “Comprehensive and practical vision system for self-driving vehicle lane-level localization”. In: *IEEE transactions on image processing* 25.5 (2016), pp. 2075–2088.

[103] Veronique Prinet et al. “3D road curb extraction from image sequence for automobile parking assist system”. In: *Proceedings - International Conference on Image Processing, ICIP* (2016), pp. 3847–3851.

[104] M. Kellner et al. “Multi-cue, Model-Based Detection and Mapping of Road Curb Features Using Stereo Vision”. In: *2015 IEEE 18th International Conference on Intelligent Transportation Systems*. Sept. 2015, pp. 1221–1228.

[105] L. Wang et al. “Multi-cue road boundary detection using stereo vision”. In: *2016 IEEE International Conference on Vehicular Electronics and Safety (ICVES)*. July 2016.

[106] F. Oniga, S. Nedevschi, and M. M. Meinecke. "Curb Detection Based on a Multi-Frame Persistence Map for Urban Driving Scenarios". In: *2008 11th International IEEE Conference on Intelligent Transportation Systems*. Oct. 2008, pp. 67–72.

[107] M. Kellner, M. E. Bouzouraa, and U. Hofmann. "Road curb detection based on different elevation mapping techniques". In: *2014 IEEE Intelligent Vehicles Symposium Proceedings*. June 2014, pp. 1217–1224.

[108] J. Siegemund, U. Franke, and W. Förstner. "A temporal filter approach for detection and reconstruction of curbs and road surfaces based on Conditional Random Fields". In: *2011 IEEE Intelligent Vehicles Symposium (IV)*. June 2011, pp. 637–642.

[109] Seokju Lee et al. "VPGNet: Vanishing Point Guided Network for Lane and Road Marking Detection and Recognition". In: *Computer Vision (ICCV), 2017 IEEE International Conference on*. IEEE. 2017, pp. 1965–1973.

[110] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *CoRR* abs/1505.04597 (2015). arXiv: 1505.04597. URL: http://arxiv.org/abs/1505.04597.

[111] Kunihiko Fukushima and Sei Miyake. "Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition". In: *Competition and cooperation in neural nets*. Springer, 1982, pp. 267–285.

[112] Tom Bruls et al. "Mark Yourself: Road Marking Segmentation via Weakly-Supervised Annotations from Multimodal Data". In: *Robotics and Automation (ICRA), 2018 IEEE International Conference on*. IEEE. 2018, in press.

[113] Philipp Krähenbühl and Vladlen Koltun. "Efficient inference in fully connected crfs with gaussian edge potentials". In: *Advances in neural information processing systems*. 2011, pp. 109–117.

[114] Bonolo Mathibela, Paul Newman, and Ingmar Posner. "Reading the Road: Road Marking Classification and Interpretation". In: *IEEE Trans. Intelligent Transportation Systems* 16.4 (2015), pp. 2072–2081. URL: http://dx.doi.org/10.1109/TITS.2015.2393715.

[115] Tarlan Suleymanov, Paul Amayo, and Paul Newman. "Inferring Road Boundaries Through and Despite Traffic". In: *Proceedings of the IEEE International Transport and Safety Conference (ITSC)*. Hawaii, USA, Nov. 2018.

[116] Randall Smith, Matthew Self, and Peter Cheeseman. "Estimating uncertain spatial relationships in robotics". In: *Autonomous robot vehicles*. Springer, 1990, pp. 167–193.

[117] Michael Montemerlo et al. "FastSLAM: A factored solution to the simultaneous localization and mapping problem". In: *AAAI/IAAI*. 2002, pp. 593–598.

[118] Giorgio Grisetti et al. "A tutorial on graph-based SLAM". In: *Intelligent Transportation Systems Magazine, IEEE* 2.4 (2010), pp. 31–43.

[119] Feng Lu and Evangelos Milios. "Globally consistent range scan alignment for environment mapping". In: *Autonomous robots* 4.4 (1997), pp. 333–349.

[120] Sebastian Thrun and Michael Montemerlo. "The graph SLAM algorithm with applications to large-scale mapping of urban structures". In: *The International Journal of Robotics Research* 25.5-6 (2006), pp. 403–429.

[121] Jose Castellanos et al. "The SPmap: A probabilistic framework for simultaneous localization and map building". In: *Robotics and Automation, IEEE Transactions on* 15.5 (1999), pp. 948–952.

[122] Tae-kyeong Lee et al. "Indoor mapping using planes extracted from noisy RGB-D sensors". In: *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE. 2012, pp. 1727–1733.

[123] Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[124] Sameer Agarwal, Keir Mierle, et al. *Ceres Solver*. `http://ceres-solver.org`.

[125] Tahir Rabbani, Frank van den Heuvel, and G Vosselmann. "Segmentation of point clouds using smoothness constraint". In: *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences* 36.5 (2006), pp. 248–253.

[126] Mark Cummins and Paul Newman. "FAB-MAP: Probabilistic localization and mapping in the space of appearance". In: *The International Journal of Robotics Research* 27.6 (2008), pp. 647–665.