Local Surface Models for Stereo Vision

by

Shahnawaz Ahmed

Bachelor of Science (Mathematics Honours) 2006

Master of Science (Mathematics) 2008

A dissertation submitted to

The School of Electronic Engineering and Computer Science

in partial fulfilment of the requirements for the Degree of

Doctor of Philosophy

in the subject of

Computer Science

Queen Mary University of London

Mile End Road

E1 4NS, London, UK

September, 2018

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my primary supervisor, Dr. Miles Hansard, for getting me excited about stereo vision. This thesis would not have been possible without his invaluable comments and constant support. In addition, I am deeply grateful to my secondary supervisor, Prof. Andrea Cavallaro, for his continuous suggestions and encouragement during these four quick years. I also owe a great debt of gratitude to my independent assessor, Dr. Alex Henshaw, for his generous comments during various stages of this work. I want to extend my thanks to Prof. James Brasington and Mr. Gabriel Max Connor Streich for helping me with the laser scanner. I would also like to express my very profound gratitude to my former colleagues and teachers who have exceedingly enabled me to grow in every aspect of life.

I am highly indebted to my fellow lab mates for the unforgettable moments spent together inside and outside the lab. I will always cherish the stimulating discussions and also the social activities.

I am also grateful to my friends and family for their continual love and moral support, and especially to my elder brother, Dr. Suman Ahmed, for his guidance, inspiration and motivation.

Finally, an exceptional thanks to my friend Mr. Sandip Mahanta, his wonderful wife Mrs. Nabanita Das, and their lovely daughter Ms. Suhana Mahanta for their continuous support during this endeavour. I will forever cherish our precious memories. Thanks again for making me a part of the family and providing a home away from home.

Declaration

I, Shahnawaz Ahmed, confirm that the research included in this thesis is my work, that is duly acknowledged, and my contributions are indicated. I have also acknowledged previously published materials.

I attest that reasonable care has been exercised to ensure the originality of this work, and to the best of my knowledge does not break any UK law, infringe any third party's copyright or other intellectual property rights, or contain any confidential material.

I accept that the college has the right to use plagiarism detection software to check the electronic version of the thesis.

I confirm that this thesis has not been previously submitted for the award of a degree to any other university.

The copyright of this thesis rests with the author and no quotation from it or information derived from it may be published without the prior written consent of the author.

Signature: Shahnawaz Ahmed

Date: September 14, 2018

In ever loving memory of my brother, Sharif

Primary supervisor
Dr. Miles Hansard

Secondary supervisor
Prof. Andrea Cavallaro

Author
Shahnawaz Ahmed

Local Surface Models for Stereo Vision

Abstract

This thesis develops new stereo vision methods for the reconstruction and analysis of complex surfaces, such as riverbeds. Depth and surface orientation estimates are crucial to the understanding of 3D scene geometry, from calibrated stereo images.

In this work, we propose new visibility and disparity magnitude constraints, for slanted patches in the scene. These constraints can be used to associate geometrically feasible planes with each point, in a 3D disparity space representation. The new constraints are validated in the PatchMatch Stereo framework of Bleyer et al. (BMVC, 2011). In order to estimate the plane parameters in this algorithm, we modify the original spatial propagation procedure, and introduce a gradient-free non-linear optimiser. These improvements allow us to achieve accurate disparity maps, with sub-pixel precision, according to the Middlebury stereo benchmark.

In addition to surface orientation, curvature information is needed for a full understanding of the local surface structure. The PatchMatch surface model is planar, and does not directly estimate the local curvature. We propose a local quadric surface model, which uses both the spatial position and the estimated surface normals, in the disparity space. We also propose principal curvature and principal direction constraints, which ensure that the local quadric model is geometrically feasible.

Finally, we describe the design and capture of a new photogrammetric dataset, which can be used to study topographic changes in riverbed morphology, over time. The dataset is challenging for conventional stereo matching algorithms, because the visible surface consists of sand, which lacks large-scale image features. This laboratory dataset comprises thirty-nine calibrated stereo pairs, plus fifteen ground-truth depth maps, obtained by a laser scanner. We used this dataset to validate the stereo vision methods developed in the thesis, in relation to potential geomorphological applications.

Contents

Acknowledgements **ii**

Declaration **iii**

Abstract **v**

Published work **xi**

Notations **xii**

1 Introduction **1**

1.1 Fluvial processes 1

1.2 Research problem 2

1.3 Challenges 2

1.4 Laboratory model 4

1.5 Depth measuring techniques 5

1.6 Time-of-Flight 6

1.6.1 Terrestrial laser scanning 7

1.6.2 Kinect 8

1.7 Photogrammetry 9

1.7.1 Stereo triangulation 10

1.7.2 Structure-from-Motion 11

1.8 Contributions 12

1.9 Organisation of the thesis 13

2 State of the art **14**

2.1 Introduction 14

2.2 Imaging geometry 15

2.2.1 Projective transformation 15

2.2.2 Pin-hole camera model . . . 16
2.2.3 Camera parameters . . . 16
2.2.4 Camera calibration . . . 19
2.2.5 Rectification . . . 19
2.2.6 Stereo pairs . . . 20
2.3 Disparity space . . . 20
2.3.1 Rigid transformation in disparity space . . . 22
2.3.2 Disparity gradient . . . 24
2.4 The stereo matching problem . . . 25
2.4.1 Local methods . . . 28
2.4.2 Global methods . . . 28
2.5 Dense local stereo algorithms . . . 29
2.5.1 Patch size . . . 31
2.5.2 Patch shape . . . 32
2.5.3 Adaptive support weight . . . 33
2.5.4 Cost function . . . 34
2.5.5 Sub-pixel disparity . . . 36
2.5.6 Occlusion . . . 37
2.6 PatchMatch Stereo . . . 37
2.7 Limitations of the PMS framework . . . 41
2.8 Summary . . . 41

3 Constrained Optimisation for Plane-Based Stereo 43
3.1 Introduction . . . 43
3.2 General framework . . . 44
3.2.1 Point-normal plane representation in disparity space . . . 44
3.2.2 Cost function . . . 47
3.2.3 Matching strategy . . . 49
3.3 Constrained plane initialisation . . . 52
3.3.1 Visibility constraint in the disparity space . . . 53
3.3.2 Disparity bound constraint on support window . . . 55
3.4 Constrained optimisation . . . 58

3.5 Results . . . 59
3.5.1 Experimental set-up . . . 59
3.5.2 Comparison with PMS . . . 60
3.5.3 Comparison with other methods . . . 64
3.6 Surface reconstruction . . . 66
3.6.1 Imaging set-up . . . 66
3.6.2 Sand images . . . 68
3.6.3 Crumpled paper surface reconstruction . . . 68
3.7 Summary . . . 68

4 Quadric surface model for PatchMatch stereo framework 69
4.1 Introduction . . . 69
4.2 Disparity models . . . 70
4.2.1 Planar disparity model . . . 71
4.2.2 Quadric disparity model . . . 71
4.3 Quadric transformations . . . 74
4.3.1 Scene to image space conversion . . . 75
4.3.2 Scene to disparity space conversion . . . 75
4.3.3 Transformation of a quadric from scene to disparity space . . . 75
4.3.4 Transformation of a quadric among views in disparity space . . . 75
4.4 Quadric PatchMatch Stereo (QPMS) . . . 76
4.5 Propagation . . . 80
4.5.1 Spatial propagation . . . 80
4.5.2 View propagation . . . 81
4.6 Constrained optimisation . . . 83
4.6.1 Quadric initialisation . . . 83
4.6.2 Principal curvature constraint . . . 84
4.6.3 Principal direction constraint . . . 85
4.6.4 Disparity constraint . . . 86
4.6.5 Surface normal constraint . . . 86
4.7 Local shape measures . . . 87
4.8 Results . . . 88

4.8.1 Experimental set-up . . . 88
4.8.2 Comparison with IPMS . . . 90
4.8.3 Curvature analysis . . . 93
4.9 Summary . . . 94

5 Riverbed dataset **95**
5.1 Introduction . . . 95
5.2 Riverbed set-up . . . 95
5.2.1 Water flow speed . . . 96
5.2.2 Lights . . . 96
5.2.3 Stereo pair . . . 97
5.2.4 Ground truth depth . . . 97
5.2.5 Amount of sand added after each capture . . . 98
5.2.6 Flume set-up . . . 99
5.3 Post-processing . . . 100
5.3.1 Image registration . . . 100
5.3.2 Disparity map visualisation . . . 101
5.3.3 Matching TLS and camera coordinates . . . 101
5.4 Results . . . 101
5.4.1 Stereo pair generation . . . 103
5.4.2 Disparity maps with different patch size with/without support weight . . . 103
5.4.3 Comparing laser scan with stereo depth map . . . 103
5.4.4 Comparison of 1D laser and stereo point cloud slices . . . 107
5.5 Surface curvature analysis . . . 107
5.6 Summary . . . 119

6 Conclusion **121**
6.1 Summary . . . 121
6.2 Future work . . . 123

Appendix A Two view geometry **125**
A.1 Coordinate systems . . . 125
A.1.1 Homogeneous coordinates . . . 126

A.2 Lens distortion 127

A.2.1 Epipolar geometry 129

Appendix B Differential geometry **131**

B.1 Differential geometry on a Monge patch 131

B.1.1 Surface differential properties 131

B.1.2 Orthonormal basis in tangent space 133

B.1.3 Shape operator 134

B.1.4 Curvatures 135

B.1.5 Fundamental forms 137

B.1.6 Alternative formulas 139

Bibliography **140**

Published work

Journal papers

[J1] Shahnawaz Ahmed, Miles Hansard and Andrea Cavallaro. Constrained Optimization for Plane-Based Stereo. *IEEE Transactions on Image Processing*, 27-8:3870–3882, August 2018.

Notations

List of Abbreviations

IPMS Initialised PatchMatch Stereo, page 12

PMS PatchMatch Stereo, page 12

QPMS Quadric PatchMatch Stereo, page 12

DEM Digital Elevation Model, page 2

GPS Global Positioning System, page 8

IMU Inertial Measurement Unit, page 8

LIDAR Light Detection and Ranging, page 8

NCC Normalized Cross Correlation, page 34

RMSE Root mean square error, page 102

SAD Sum of Absolute Differences, page 34

SFM Structure-from-Motion, page 11

SSD Sum of Squared Differences, page 34

TLS Terrestrial Laser Scanning, page 7

ToF Time of Flight, page 6

List of symbols

$(o_x, o_y)^\mathsf{T}$ Pixel coordinates of the principal point, page 18

$(U, V, W)^\mathsf{T}$ World coordinate system, page 125

$(\tilde{x}, \tilde{y})^\mathsf{T}$	Image coordinate system, page 126
$(x, y)^\mathsf{T}$	Pixel coordinate system, page 126
$(X, Y, Z)^\mathsf{T}$	Camera coordinate system, page 125
$\mathbf{\Gamma}$	Projective transformation, page 21
c	Curvedness, page 88
$\mathbf{K}$	Camera intrinsic, page 18
C	Left camera, page 129
C'	Right camera, page 129
$\mathbf{E}$	Camera extrinsic, page 19
$\mathcal{D}$	Disparity space / Left disparity space, page 21
$d_{\min}$	Minimum disparity, page 31
$d_{\max}$	Maximum disparity, page 31
$\mathbf{l}'$	Epipolar line in I', page 129
$\mathbf{e}$	Left epipole, page 129
$\mathbf{e}'$	Right epipole, page 129
$\mathbf{F}$	Fundamental matrix, page 129
$\mathbf{H}$	Homography, page 15
$\mathbf{I}$	Identity matrix, page 19
I'	Search image, page 129
κ_1, κ_2	Principal curvatures, page 87
λ	Scalar, page 15
λ	Scale factor, page 18

$(a,b,c)^\mathsf{T}$	Plane parameters of $\mathbf{f}$, page 39
$\lvert\cdot\rvert$	L_1 norm, page 34
$\mathbf{B}$	Blue colour channel, page 4
$\mathbf{G}$	Green colour channel, page 4
$\mathbf{R}$	Red colour channel, page 4
$\mathcal{W}(\cdot)$	Support region, page 34
$\mathbf{O}_\mathsf{C}$	Origin of the camera coordinate system, page 125
$\mathbf{O}_\mathsf{I}$	Principal point, page 126
$\mathbf{O}_\mathsf{P}$	Origin of pixel coordinate system, page 126
$\mathbf{O}_\mathsf{W}$	Origin of the world coordinate system, page 125
d	Disparity, page 9
$\tilde{\mathbf{p}}$	Image point in image coordinate system, page 126
$\bar{\mathbf{p}}$	Image point in pixel coordinate system, page 126
$\bar{\mathbf{p}}_s$	Pixel coordinates centered at the principal point, page 17
$\bar{\mathbf{p}}_{\boldsymbol{\pi}}$	Point on plane $\boldsymbol{\pi}$, page 15
$\bar{\mathbf{p}}'$	Corresponding point of $\bar{\mathbf{p}}$ in the other view, page 129
$\mathbf{P}$	Scene point, page 125
$\mathbf{f}$	Plane in the $\mathcal{D}$, page 39
$\boldsymbol{\pi}$	Plane, page 15
x	x coordinate of a image point in I, page 10
x'	x coordinate of a image point in I', page 10
$\mathbf{q}$	Corresponding point of $\bar{\mathbf{q}}$ in $\mathcal{D}$, page 37

$\tilde{\mathbf{Q}}$	Quadric in scene space, page 74
$\mathbf{Q}$	Quadric in disparity space, page 71
$\mathbf{R}$	Rotation matrix, page 18
$\mathscr{S}$	Approximated local surface in disparity space, page 70
$\simeq$	equality up to a scale factor, page 127
s	Shape index, page 87
$\mathcal{S}$	Scene space, page 125
s_x	Size of pixels per metric unit along the x axis, page 17
s_y	Size of pixels per metric unit along the y axis, page 17
$\mathbf{t}$	Translation vector, page 18
$\tilde{\mathcal{S}}$	Approximated local surface in scene space, page 70
$\mathbf{V}_{\mathsf{LR}}$	Projection matrix from left to right disparity space, page 76
$\mathbf{V}_{\mathsf{RL}}$	Projection matrix from right to left disparity space, page 76
x_s	x coordinate in pixel coordinates with origin at the principal point, page 17
y_s	y coordinate in pixel coordinates with origin at the principal point, page 17
$\mathbf{0}$	Zero vector, page 19
B	Baseline, page 9
f	Focal length, page 126

Chapter 1

Introduction

Geomorphology is the scientific study of landforms and the processes that shape them. Geomorphologists seek to understand the formation of landscapes and try to predict future changes through a combination of field observations, physical experiments and numerical modelling.

The study of geomorphology can be subdivided into various interconnected geomorphologic processes such as aeolian (*e.g.,* air), biological, fluvial (*e.g.,* water), glacial etc. Broadly each process consists of three categories, (a) production of regolith[1] by weathering and erosion, (b) transportation of regolith, and eventually (3) deposition of regolith. Wind, waves, chemical dissolution, mass wasting, groundwater movement, surface water flow, glacial action, tectonism, and volcanism are primarily responsible for most topographic changes. In this thesis, we are interested in learning how fluvial process changes its associated topography.

1.1 Fluvial processes

Fluvial geomorphologic processes are those related to rivers and streams, both of which are good carriers of water and sediment. As water flows over the channel bed, the channel bed assembles and transports sediments, either as bed load, suspended load or dissolved load. The rate of transportation of sediments depends on the availability of sediments in the channel bed and the flow of the channel bed.

Besides, rivers and streams are also capable of eroding rocks and creating new sediments,

[1] Regolith is a layer of loose, heterogeneous material covering solid rock. It includes dust, soil, broken rock, and other related materials.

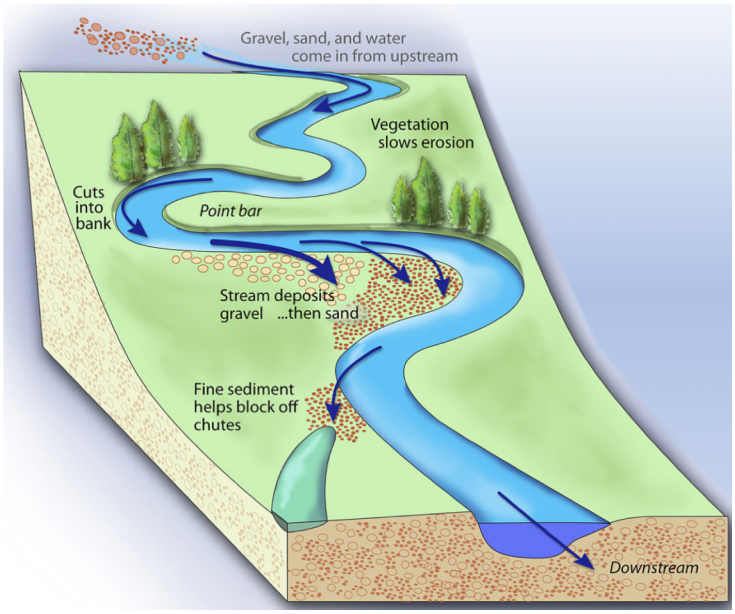


Figure 1.1: Fluvial Processes (Image courtesy National Science Foundation)

both from their beds and their surroundings. The flow of water gives rivers and streams more power to erode as there is more friction in the moving water. Deposition of the sediments occurs during flooding or flowing into an open plain. Thus, rivers and streams play a major role in large-scale landscape evolution in non-glacial environments. So both rivers and streams have key connections with the formation of different landscapes (Fig. 1.1).

1.2 Research problem

The overall aim of the project is to create a physically valid 3D digital elevation model (DEM) of dynamic objects, which will help answer certain geomorphological questions such as their formations, erosion and deposition of sediments over the channel bed, etc. To get the DEMs, we need to capture a high quality calibrated dense 3D model of the channel bed of a river or stream. The idea is to build a working model inside a laboratory with constrained parameters and then extend the model to an outside environment.

1.3 Challenges

There are only a few techniques that can acquire high-quality 3D data of a dynamic scene. All the techniques mentioned in Sec. 1.6 have some limitations. Most of them can not work with a dynamic scene. The following are the primary challenges of this project.

- **Dynamic scene modelling.** As water flows over a channel bed, it erodes sediment at some places and also deposits the sediment at other places as well, so the surface of the channel bed is continuously changing with time.
- **Fine texture.** It will be hard for stereo matching algorithms to find structures in the sand images as there is not enough variation of textures. Moreover, the textures changes with conditions, *e.g.,* when the sand is wet or dry.
- **Calibration.** From a geomorphic perspective, the greatest topographic difference in the active area is likely to remain at the millimetre scale. Precise calibration is required to achieve this accuracy.
- **Ambient light and reflectivity.** The lighting condition is also a key issue, *e.g.,* Microsoft Kinect will not work in sunlight. On a natural channel bed, the material reflectivity will vary greatly. For the laboratory model, we need diffused light to reduce sensor noise and reflection.
- **Field condition.** The rate of change of erosion and sedimentation will depend on the slope, landscape formation of the riverbed and volume of flow. The model needs to be robust enough to detect the changes while the field condition varies.
- **Refraction.** If we put a coin in a bowel and put some water in it, it seems that the coin has come a little bit upwards. The true depth of the coin is hard to compute just from seeing it. This is known as the refraction problem. Such problem will occur if water is considered while capturing the stereo pair.
- **Colour aliasing.** When a scene is captured digitally, any patterns or colours that did not exist in the original scene, but are present in the reproduced image are generally referred to as 'aliasing'. In single sensor electronic imaging systems, scene colour is acquired by sub-sampling in three colour planes to capture colour image data simultaneously for red, green and blue colour components. Usually, this is accomplished by placing a mosaic of red, green and blue filters over a 2D single sensor array. One way of arranging red, green and blue pixels to form a mosaic pattern (e.g. Bayer pattern) is shown in Fig. 1.2. Since each pixel is filtered to record only one of three colours, the data from each pixel cannot fully specify each of the red, green, and blue values on its own. To obtain a full-colour image, various demosaicing algorithms can be used to interpolate a set of complete red, green, and blue values for each pixel. These algorithms make use of the surrounding pixels

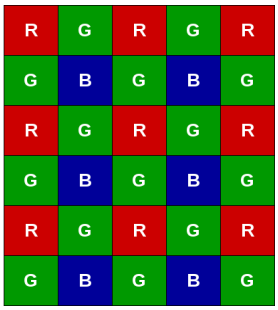


Figure 1.2: RGB color filter array with Bayer pattern

(a) Colour aliasing of the sand in the sandbox using PointGrey Cameras. Note the un-physical red and green hues.

(b) Images of the same sand in the sandbox using a Nikon D7100.

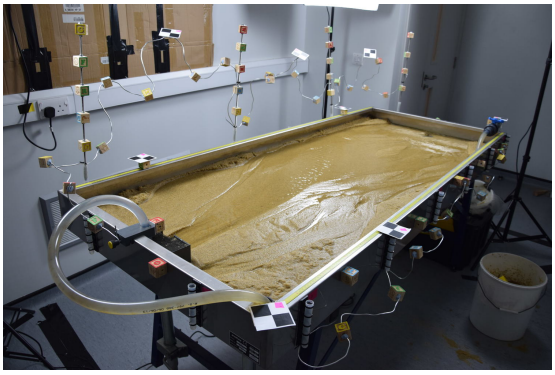
Figure 1.3: Images from different cameras

of all colours to estimate the values for a particular pixel. One significant characteristic of this type of sensors is horizontally adjacent pixels appear to contribute significantly to the response of their neighbours. Some colours exhibit a large variation between the **G** (green) values depending on the dominance of **B** (blue) and **R** (red) channel values. This discrepancy gives rise to a blocking effect on the colour interpolated (and zoomed) image, and also spreading of false colours into a detailed structure, due to **B** and **R** channel induced errors. Fig. 1.3a shows the colour aliasing effect on the image of the sand taken by the PointGrey camera whereas Fig. 1.3b is the same image taken by Nikon D7100 DSLR Camera with 18-55 mm AF-S lens, where the colour aliasing is not visible.

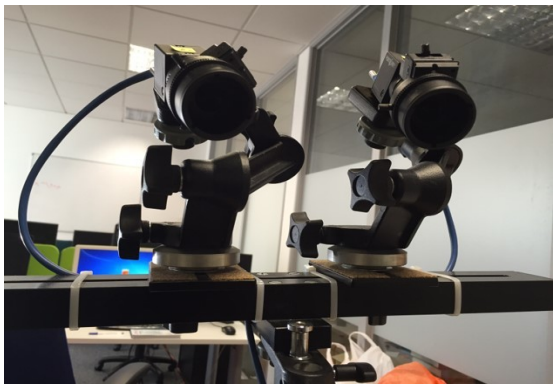
1.4 Laboratory model

We have two laboratory models to replicate the topography of a channel bed. The first model is a small sandbox (75cm $\times$ 56 cm) with markers all around (Fig. 1.4a). On top of the sandbox,

(a) Sandbox (70 cm × 50 cm) with markers attached from the top view.



(b) Riverflow Simulator



(c) PointGrey stereo camera

(d) Nikon D7100

Figure 1.4: Laboratory model

we have a synchronised PointGrey stereo pair (Fig. 1.4c). In this setting, we assume that the sand is static and use a patch based stereo matching to reconstruct the 3D model. The first model does not involve water. The other model is a river flow simulator (Fig. 1.4b) which is a bigger sandbox with water running through it. It replicates the flow of a river, so the sand here is no longer rigid. We first to model the static sand and later the dynamic one. As the resolution of the PointGrey camera is inferior, we later used a DSLR camera (Nikon D7100) (Fig. 1.4d) to capture the images. Markers were used around the sandbox to rectify (Sec. 2.2.5) the images as the sand images do not have enough texture variation to establish keypoints.

1.5 Depth measuring techniques

The increasing availability of high-performance computers, complex surface models and sophisticated optimisation techniques have significantly impacted the field of digital elevation modelling and geomorphological terrain analysis. Based on new developments in remote sensing technology, there has been a notable transformation in the acquisition of topographic data. Time-

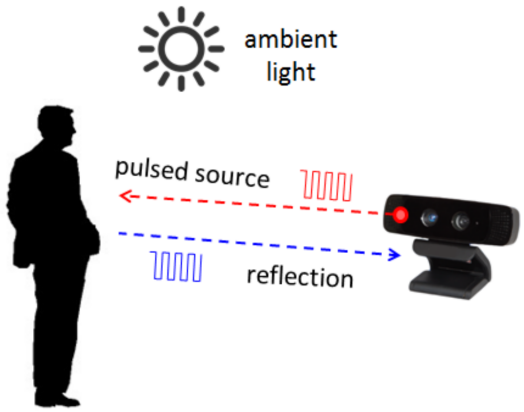


Figure 1.5: Time-of-Flight camera operation (Image courtesy Texas Instruments)

of-Flight and Photogrammetry, in particular, have revolutionised the quality of digital elevation models and the 3D representation of a terrain's surface by extending their spatial extent, resolution, and accuracy [89, 68]. The Time-of-Flight principle uses active sensors (*for ex.,* LED or laser transmitters) to measure the depth of the scene, whereas photogrammetry relies on passive sensors (*e.g.,* CCD, CMOS).

1.6 Time-of-Flight

Time-of-Flight (ToF) principle is used for long and short range laser scanning. The time-of-flight range finder estimates the distance of a surface by calculating the round-trip time of a pulse of laser (ultraviolet or infra-red) light (Fig. 1.5). The laser emits a pulse and the amount of time before the detector sees the reflected light is measured. Since the speed of light is known, the round-trip time determines the travel distance of the light, which is twice the distance between the scanner and the surface. The accuracy of a time-of-flight camera depends on how precisely we can measure the time.

A Time-of-flight camera [22] uses infra-red (IR) light to probe the subject and produce a depth image where each pixel encodes the distance to the corresponding point in the scene by measuring the phase delay of the reflected IR light (Fig. 1.6). The range finder can only detect the distance of one point at a time in its direction of view. Thus, to scan the entire field of view, one should change the range finder's direction of view. It can be done either by rotating the range finder itself or by using a system of rotating mirrors. The latter method is commonly used because mirrors are much lighter, thus makes rotation much faster with greater accuracy. The

Figure 1.6: Time-of-Flight camera (Image courtesy vision-systems.com)

following are the devices that use ToF principle to measure the depth of a scene.

1.6.1 Terrestrial laser scanning

Terrestrial laser scanning (TLS) is a ground-based technique to measure the position and dimension of objects for surveying tasks. This active imaging method can rapidly acquire accurate, dense 3D point clouds of object surfaces by laser range finder (Fig. 1.7). It uses an impulse based time of flight technique to measure the distance between the scanner and the object.

The horizontal field of view of the most advanced TLS devices is 360° with almost 310° in the vertical direction. State of the art laser scanners offers an accuracy of about 10 mm. The data is measured by the scanners either in polar or cylindrical coordinates along with additional information on the intensity of the received signal. These data describe a 3D cloud of detected points in space, which is known as 'point cloud' data.

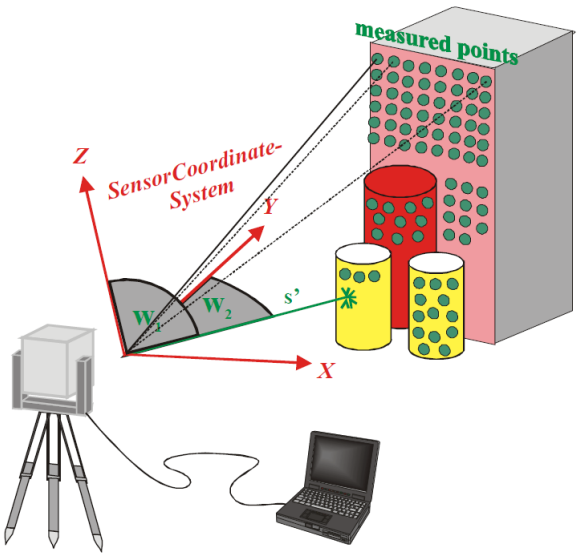


Figure 1.7: The principle of terrestrial laser scanning [82]

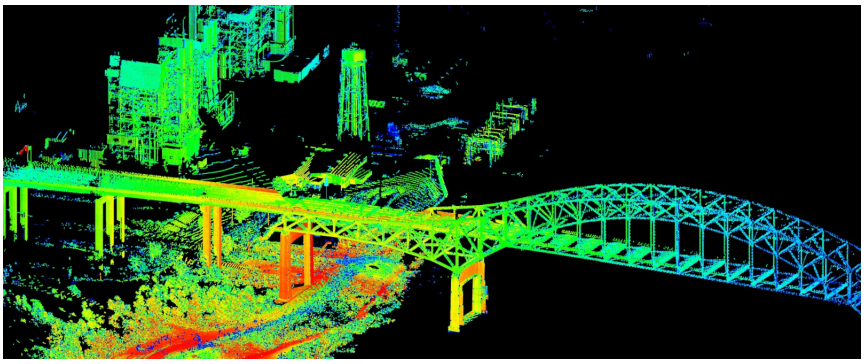


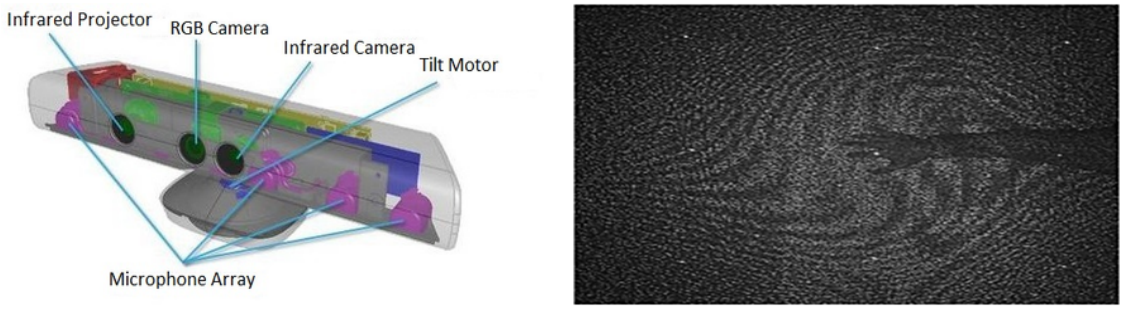
Figure 1.8: A 3D terrestrial LIDAR scan of the Interstate-510 bridge in New Orleans, LA, USA. (Image courtesy USGS Multimedia Gallery)

LIDAR, which stands for Light Detection and Ranging, is a remote sensing method that uses ultraviolet light in the form of a pulsed laser to measure depth. The light pulses combined with other data, such as the inertial measurement unit (IMU) and the global positioning system (GPS) is recorded by the system, which generates precise three-dimensional information about the shape of the object and its surface characteristics (Fig. 1.8).

The LIDAR system pulses a laser beam onto a mirror and projects it over the survey area, measuring between 20,000 to 100,000 points per second. When the laser beam hits an object, it is reflected to the mirror. The time interval between the pulse leaving the platform and its return to the LIDAR sensor is measured. During post-processing, the time interval measurements are converted to distance and corrected using the IMU and GPS data. GPS can accurately determine the position in terms of latitude, longitude and altitude, which are also known as the X, Y and Z coordinates. IMU aids the GPS measurements by calculating the angular rates, linear velocity and position relative to a global reference frame. The LIDAR sensor collects a huge amount of data, and a single survey can easily generate millions of points totalling several terabytes.

1.6.2 Kinect

The first version of Microsoft Kinect uses structured light to measure depth [96]. It uses an infrared (IR) projector to project a known IR pattern on the subject. There is also an IR camera which then captures the reflected pattern and based on the known pattern reflected from the subject, and it can calculate the depth. Fig. 1.9b shows the projection of an IR pattern on the hand and the wall [59].



(a) Left image (b) Infra-red pattern [59]

Figure 1.9: Kinect (Image courtesy kinectingforwindows.com)

1.7 Photogrammetry

Photogrammetry is a technique that estimates the 3D position of surface points using two or more overlapping images of a single physical object taken from different viewpoints. The fundamental principle of photogrammetry is the triangulation of corresponding image points and the camera centres. The intersection of the rays obtained from individual camera centre and image point produces the 3D coordinates of the image point.

Stereo vision requires two or more cameras for understanding the 3D geometry of a scene. Cameras should be as similar as possible for avoiding unnecessary difficulties (*e.g.,* parameters of a lens and image sensor). Calibration (Sec. 2.2.4) is essential to recover 3D quantitative measures about the observed scene from 2D images. It estimates the parameters defining the camera model. The baseline distance (B) of a stereo rig is the fixed translational distance between the projection centres of two cameras. Any point in the scene that is visible in both cameras will be projected to a pair of 'conjugate' image points in the two images also known as corresponding points. A stereo camera set-up allows computing the physical depth of an image point using the concept of disparity (Sec. 2.3), which is the image distance between the corresponding points. The disparities of all the corresponding points when two images are stored in the coordinates of the chosen reference image is known as the disparity map. To find the physical depth in terms of disparity, we first need to rectify[2] (Sec. 2.2.5) the images. In the rectified case (Sec. 2.2.5), while viewing a 3D point through the two cameras of a stereo rig, the point in the right image will have a different horizontal coordinate than the point in the left image, but same vertical coordinates. This apparent horizontal shift of a corresponding point in between the images is the disparity d,

[2]Virtual rotation around the optical centre to reproject image planes onto a common plane parallel to the line between optical centres. Pixel motion becomes horizontal after the rectification process.

which is inversely proportional to the depth of a scene point [12].

1.7.1 Stereo triangulation

The 3D position a scene point can be reconstructed from its projection onto the image planes, once the relative position and orientation of the cameras are known. Consider the simplest case where there are only two cameras and the left camera coordinate system is aligned with the world coordinate system (App. A.1). We also assume that the optical axes of the two cameras are parallel (no rotation), and the translation of the right camera is only along the X axis.

Consider such a stereo camera model with the left camera C and the right camera C′ (Fig. 1.10). Let B be the baseline between C and C′ and f be the focal length of each camera. We also assume that a scene point $\mathbf{P} = (X,Y,Z)^\top$ projects to $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$ in the left image I and the right image I', respectively. The image points $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$ will have the same y coordinate but different x coordinate. Let x and x' be the x coordinate of $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$, respectively. Thus the disparity in this case is $d = x - x'$. Using the concept of similar triangles, the depth of the scene point $\mathbf{P}$ (Z coordinate of $\mathbf{P}$ in the left camera coordinate system, App. A.1) is governed by the expression : $\frac{d}{B} = \frac{fs_x}{Z}$ and hence depth of $\mathbf{P}$ is:

$$Z = B\frac{fs_x}{d}, \tag{1.1}$$

where s_x denotes the size of the pixels per metric unit along the x-axis. This process of finding

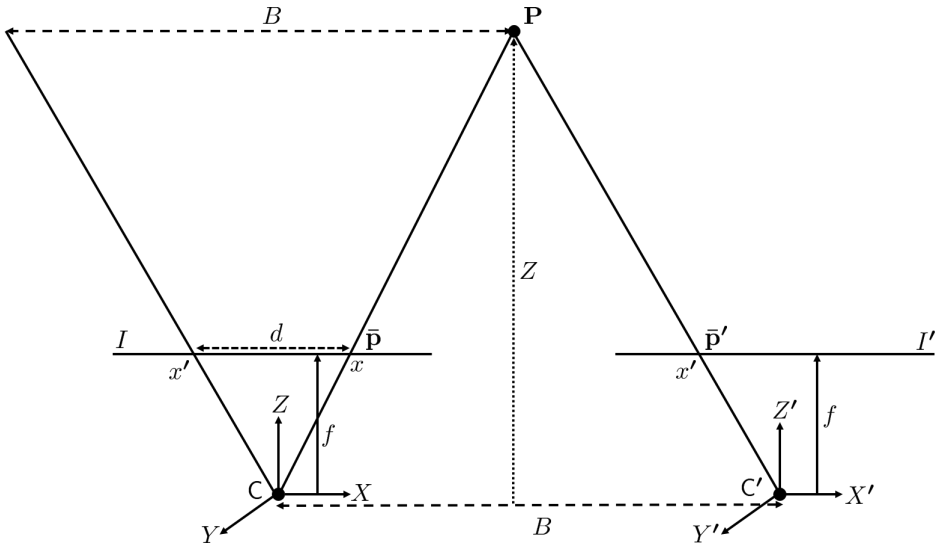


Figure 1.10: Stereo triangulation. Figure reproduced from [12].

depth by triangulating the scene point and the image points is called "stereo triangulation" [84]. Conversely, we can also find the disparity when the depth is known as:

$$d = B\frac{fs_x}{Z} \ . \tag{1.2}$$

1.7.2 Structure-from-Motion

Structure-from-Motion (SFM) [89] uses multi-view approach to estimate the 3D structures from 2D image sequences. Instead of using a single stereo pair, the SFM technique requires multiple, overlapping images and camera calibration information (intrinsic parameters) as input to extract features and 3D reconstruction algorithm (Fig. 1.11).

Feature points are detected in each image and matched between image pairs and then the SFM framework computes the scene structure and camera motion [56]. To find correspondence between images, features such as corner points (edges with image gradients in multiple directions) are tracked from one image to the next. The trajectories of the feature points are then used to reconstruct the 3D positions and the camera's motion. SFM computes an external camera pose per image (the motion) and a 3D point cloud (the structure) representing the scene.

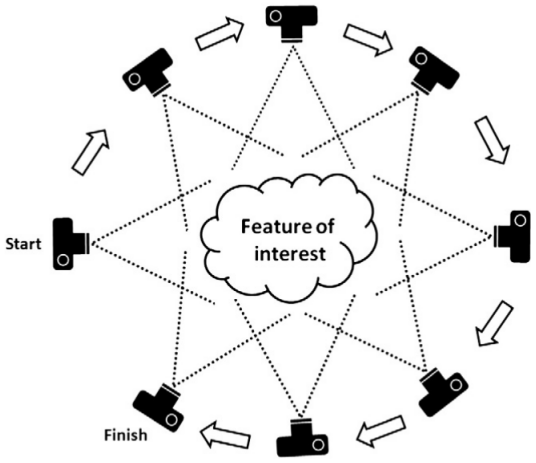


Figure 1.11: Structure-from-Motion (SFM). Instead of a single stereo pair, the SFM technique requires multiple, overlapping images as input to extract and match features. Figure reproduced from [89].

1.8 Contributions

We develop new stereo vision methods for the reconstruction and analysis of complex surfaces, such as riverbeds. Depth and surface orientation estimates are crucial to the understanding of 3D scene geometry, from calibrated stereo images. We propose a stereo matching algorithm "Initialised PatchMatch Stereo" (*IPMS*) , which consists two constraints, namely visibility and disparity magnitude constraints, that associate feasible planes with each point in the disparity space. The new constraints are validated in the PatchMatch Stereo (*PMS*) [9] framework, which is a dense local stereo framework that uses a slanted support plane at each pixel and creates a sub-pixel precision disparity map. We use these new constraints not only for initialisation but also in the local plane refinement step of this iterative algorithm. The proposed constraints increase the probability of estimating correct plane parameters and lead to an improved 3D reconstruction of the scene. Furthermore, the proposed constrained initialisation reduces the number of iterations before convergence to the optimal plane parameters. In addition, as most stereo image pairs are not perfectly rectified, we modify the view propagation of the *PMS* framework by assigning the plane parameters to the neighbours of the candidate pixel. To update the plane parameters in the plane refinement step, we use a gradient-free non-linear optimiser.

Both *PMS* and *IPMS* fail to smoothly reconstruct curved surfaces in the disparity space as the framework uses a planar model to estimate the disparity of each pixel in the image. Moreover, the model does not directly provide the local curvature information. We further propose a local quadric surface model which successfully handles both curved and planar surfaces in the disparity space and, also provides the curvature information at every pixel. As the spatial position of each point in the disparity space is known, a local quadric can be estimated over a patch at each pixel in the disparity space. The estimation depends on the patch size and the distribution of disparities over the patch and is highly affected by outliers. Moreover, such estimation does not use the surface normal information. In the proposed method, Quadric PatchMatch Stereo (*QPMS*), both spatial and normal information are used to fit local quadric at each pixel. We further address the false matching problem by introducing disparity guided spatial propagation, where a non-linear disparity dissimilarity function weights the aggregated cost. Disparity guided spatial propagation prevents false matches from growing and also fill them with the correct disparity value, provided there is at least one good surface approximation of the neighbours. Moreover, the proposed modifications will work with any stereo algorithm that can be cast in the PatchMatch Stereo

framework.

We also designed and captured a new photogrammetric dataset, which can be used to study topographic changes in riverbed morphology, over time. The dataset is challenging for conventional stereo matching algorithms because the visible surface consists of sand, which lacks large-scale image features. This laboratory dataset comprises thirty-nine calibrated stereo pairs, plus fifteen ground-truth depth maps, obtained by a laser scanner. We used this dataset to validate the stereo vision methods developed in the thesis, in relation to potential geomorphological applications.

1.9 Organisation of the thesis

The thesis is organised as follows. Chapter 2 reviews basic stereo vision concepts and focus on state-of-the-art dense local stereo methods that approximate the local scene surface in the disparity space to measure depth. In Chapter 3, we introduce the Initialised PatchMatch stereo algorithm which mainly describes constrained tangent support plane generation in the disparity space. The Quadric PatchMatch stereo algorithm is discussed in Chapter 4, which generates feasible local quadric surfaces for local stereo matching. This helps us to retrieve the curvature information directly from the associated quadric. In Chapter 5, we present the riverbed dataset along with ground-truth depth. We also analyse the depth and the curvature estimates of the generated surfaces. Finally, conclusion and future research direction on dense local stereo matching are drawn in Chapter 6.

Chapter 2

State of the art

2.1 Introduction

Accurate depth estimation from stereo images is important in many applications, such as augmented reality [67], terrain estimation [16], mapping [78], navigation [79], scene segmentation [40], object recognition [35] and 3D reconstruction [8]. Stereo matching algorithms estimate the depth of a scene by finding corresponding points between two views [66], [8]. The algorithms usually assume that the images are rectified, which reduces the stereo correspondence problem to a 1-D search problem, where matching pixels lie along the horizontal scan-line of the rectified images. A dense stereo matching algorithm produces a complete disparity map by estimating individual disparity for every image point of the rectified stereo pair, which is used to retrieve the depth information.

The rest of the chapter is organised as follows. We discuss the imaging geometry along with the camera parameters in Section 2.2. As the pixel matching occurs in the disparity space, we analyse the geometric properties of the disparity space in Section 2.3. The stereo matching problem is introduced in Section 2.4. Section 2.5 presents state of the art dense local stereo matching models. We introduce the PatchMatch stereo framework and its limitations in Section 2.6 and Section 2.7, respectively. Finally, Section 2.8 concludes the review and outlines future directions. In addition, some concepts on two-view geometry are discussed in the Appendix A.

2.2 Imaging geometry

A digital image is a two dimensional intensity array consisting one or more channels [58, 50], *e.g.,* in a grayscale image there is only one channel but for a colour image, we have three channels, red (**R**), green (**G**), and blue (**B**).

2.2.1 Projective transformation

A homography is a projective transformation that maps points from one plane to another plane (*e.g.,* a transformation that maps points on a planar surface in the world to the image plane) [33, 84]. Fig. 2.1 illustrates the geometry involved in this process.

A 2D homography is represented by a 3×3 homogeneous matrix $\mathbf{H}$, which relates any point $\bar{\mathbf{p}}$ (in homogeneous coordinates App. A.1.1) on a plane $\boldsymbol{\pi}$ to its corresponding point $\bar{\mathbf{p}}'$ lying on a different plane $\boldsymbol{\pi}'$.

$$\bar{\mathbf{p}}' = \mathbf{H}\bar{\mathbf{p}}'.$$

The homography has 8 degrees of freedom (9 entries in the $\mathbf{H}$ matrix, but the common scale factor is not relevant). Hence, to determine the homography, we require 4 pairs of corresponding points (two coordinates per point). It is important that no 3 points are collinear, in order for the solutions to be well-defined in general.

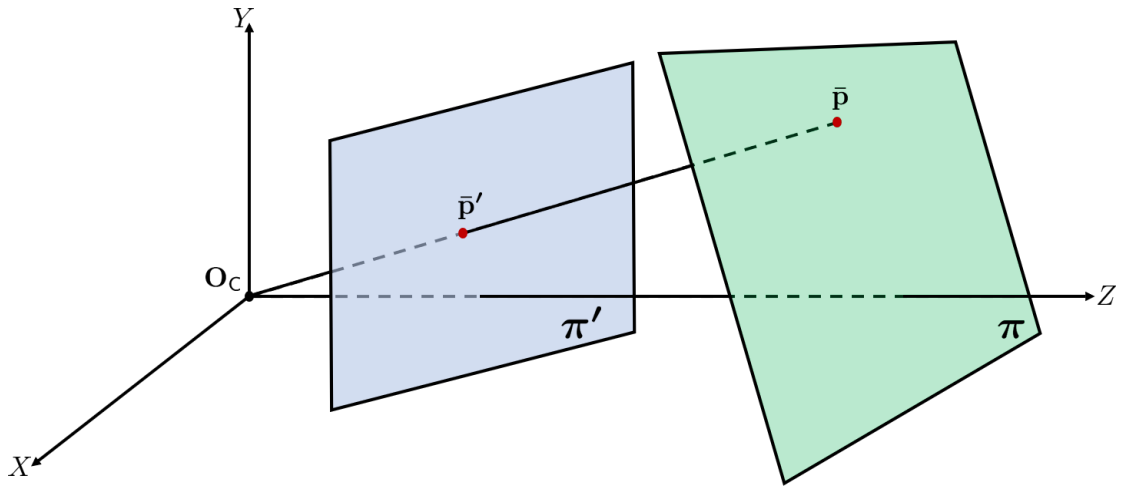


Figure 2.1: Central projection maps points on one plane to points on another plane. The projection also maps lines to lines, which can be seen by considering a plane through the projection centre that intersects with the two planes $\boldsymbol{\pi}$ and $\boldsymbol{\pi}'$. Since lines are mapped to lines, the central projection is a projectivity and is represented by a homography $\mathbf{H}$, where $\bar{\mathbf{p}}' = \mathbf{H}\bar{\mathbf{p}}$ [33].

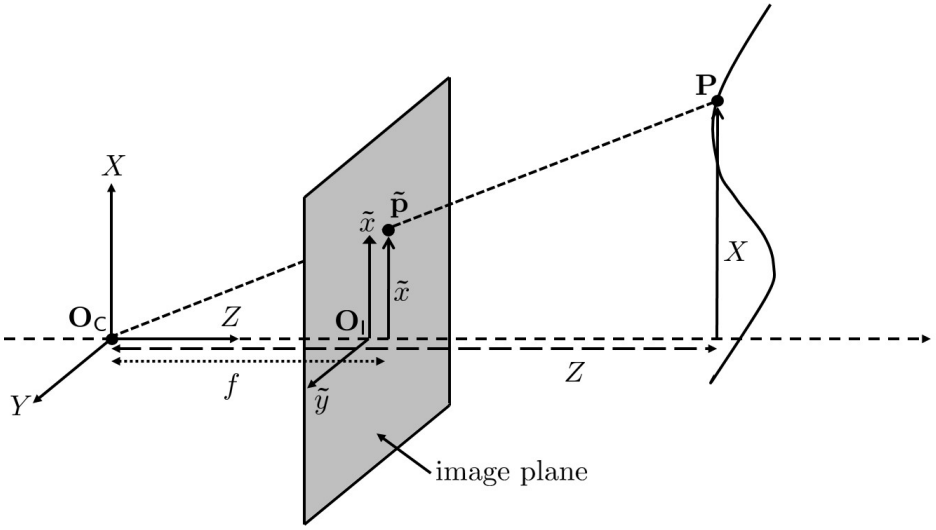


Figure 2.2: Frontal pinhole imaging model. Image point $\tilde{\mathbf{p}}$ is the projection of the scene point $\mathbf{P}$ on the image plane at a distance f from the optical centre $\mathbf{O}_\mathsf{C}$. The image point is registered where the ray passing through $\mathbf{O}_\mathsf{C}$ and $\mathbf{P}$ intersects the image plane.

2.2.2 Pin-hole camera model

For simplicity, let us assume the camera coordinate system is aligned with the world coordinate system, with the optical centre $\mathbf{O}_\mathsf{C}$ as the common origin. When the aperture of a lens decreases to zero, then all the rays are forced to go through the optical centre $\mathbf{O}_\mathsf{C}$. Consequently, the only points that contribute to the irradiance at the image point $\tilde{\mathbf{p}}$ lie on the line through $\mathbf{O}_\mathsf{C}$ and $\tilde{\mathbf{p}}$. Let $\mathbf{P} = (X,Y,Z)^T$ (in camera coordinates, App. A.1) be a scene point that is mapped to $\tilde{\mathbf{p}} = (\tilde{x},\tilde{y})^T$ (in image coordinates), relative to a reference frame centred at the optical center $\mathbf{O}_\mathsf{C}$, with its Z-axis being the optical axis [58]. The relation that maps the point $\mathbf{P}$ in the physical world to a point $\tilde{\mathbf{p}}$ in the image plane is called a projective transformation [11]. Let f be the focal length of the lens, which is measured in metric units (usually millimetre). Then it is immediate to see from similar triangles in Fig. 2.2, that $\mathbf{P}$ and its image point $\tilde{\mathbf{p}}$ are related by the so-called ideal perspective projection.

$$\tilde{x} = f\frac{X}{Z}, \qquad \tilde{y} = f\frac{Y}{Z}, \tag{2.1}$$

which is a mapping from 3D to 2D and is defined as $\tilde{\mathbf{p}} = \boldsymbol{\pi}\mathbf{P}$.

2.2.3 Camera parameters

The pin-hole camera model in Eq. 2.1 is highly specified to a very particular choice of reference frame, the 'canonical retinal frame', centred at the optical centre with one axis aligned with the

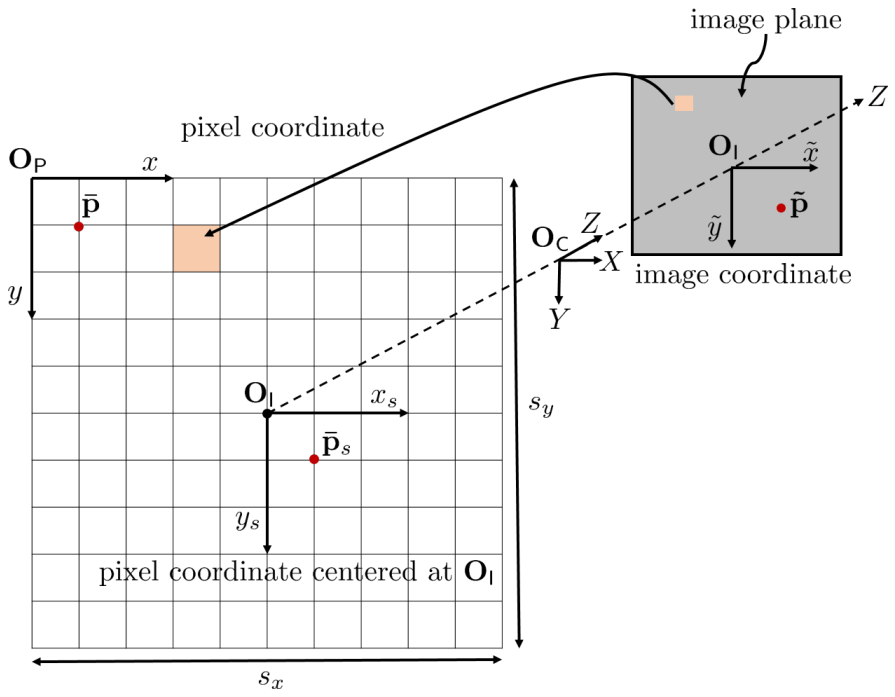


Figure 2.3: Transformation from image coordinates to pixel coordinates

optical axis. In practice, when an image is captured, the measurements are obtained in terms of pixels $(x,y)^\intercal$ (in pixel coordinates, App. A.1) with the origin of the pixel coordinates frame typically in the upper left corner. To reuse the model in Eq. 2.1, we need to specify the relationship between the retinal plane coordinate frame and the pixel arrays [58]. For this transformation, we first need to specify the units along the x and y axes. In general, $\tilde{\mathbf{p}} = (\tilde{x},\tilde{y})^\intercal$ is specified in terms of metric units (e.g., millimetres). We now need to find a correspondence that maps $\tilde{\mathbf{p}}$ to pixel unit, which depends on the size of the pixel (in metric units) along the x and y axis. Let s_x and s_y denote the size of pixels per metric unit along the x and y axis, respectively, as shown in Fig. 2.3. The transformation can be defined by the following scaling matrix [58] :

$$\bar{\mathbf{p}}_s = \begin{bmatrix} x_s \\ y_s \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \tilde{\mathbf{p}} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} \tilde{x} \\ \tilde{y} \end{bmatrix},$$

where, $\bar{\mathbf{p}}_s$ is the pixel coordinates of $\tilde{\mathbf{p}}$ centered at the principal point $\mathbf{O}_\mathrm{I}$ (where the $z-$ axis intersects the image plane). When $s_x = s_y$, the pixels are square, otherwise rectangular. However, x_s and y_s are still specified relative to $\mathbf{O}_\mathrm{I}$ (in pixel coordinate), whereas the origin of the pixel coordinate system is conventionally specified relative to the upper-left corner, and is indicated by positive numbers. We translate the origin of the reference frame to the upper-left corner as

shown in Fig. 2.3 by the following transformation.

$$x = x_s + o_x,$$
$$y = y_s + o_y,$$

where, $\bar{\mathbf{p}} = (x,y)^\mathsf{T}$ are the pixel coordinates of $\bar{\mathbf{p}}_s$ and $\mathbf{O}_\mathsf{I} = (o_x, o_y)^\mathsf{T}$ are the pixel coordinates of the principal point $\mathbf{O}_\mathsf{I}$ relative to the image reference frame. The actual coordinates of the scene point $\mathbf{P} = (X,Y,Z)^\mathsf{T}$ are given by the pixel coordinates $\bar{\mathbf{p}} = (x,y)^\mathsf{T}$ instead of the ideal image coordinates $\tilde{\mathbf{p}} = (\tilde{x},\tilde{y})^\mathsf{T}$.

The above steps of image transformation can be nicely expressed in a homogeneous representation as the following :

$$\bar{\mathbf{p}} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & o_x \\ 0 & 1 & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & o_x \\ 0 & s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ 1 \end{bmatrix}$$

Combining the projection model in eq. 2.1 with scaling and translation yields a more realistic model of a transformation between $\mathbf{P}$ and $\bar{\mathbf{p}}$ in homogeneous coordinates.

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \frac{1}{\lambda} \begin{bmatrix} 1 & 0 & o_x \\ 0 & 1 & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix},$$

where λ is a scale factor. Multiplication of the first three matrices give an upper triangular matrix (3×3), which is called the camera intrinsic, or the calibration matrix and is denoted by $\mathbf{K}$.

$$\mathbf{K} = \begin{bmatrix} 1 & 0 & o_x \\ 0 & 1 & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} fs_x & 0 & o_x \\ 0 & fs_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.2}$$

In cases where the world coordinates do not merge with the camera coordinates, apart from the camera intrinsic $\mathbf{K}$, the geometric relationship between $\mathbf{P}$ and $\bar{\mathbf{p}}$ also depends on a rigid body motion $(\mathbf{R},\mathbf{t})$, where $\mathbf{R}$ is a rotation matrix (3×3) and $\mathbf{t}$ is a three dimensional translation vector

(3×1). The camera extrinsic $\mathbf{E}$ is defined as follow:

$$\mathbf{E} = \left[\begin{array}{c|c} \mathbf{R} & \mathbf{t} \\ \hline \mathbf{0}^{\mathsf{T}} & 1 \end{array}\right], \tag{2.3}$$

where $\mathbf{0} = (0,0,0)^{\mathsf{T}}$. The overall model for image formation can be captured by the following transformation.

$$\bar{\mathbf{p}} = \frac{1}{Z}\,\mathbf{K}\,[\,\mathbf{I} \mid \mathbf{0}\,]\,\mathbf{E}\,\mathbf{P}. \tag{2.4}$$

2.2.4 Camera calibration

Camera calibration is a necessary step in 3D computer vision to extract metric information from 2D images and correspondences. Camera calibration determines the parameters of the transformation between an object in 3D space and the 2D image observed by the camera from images. The transformation includes camera intrinsic $\mathbf{K}$ and extrinsic $\mathbf{E}$. There are five parameters in the intrinsic matrix. The rotation matrix $\mathbf{R}$ consists of 9 elements but only has 3 degrees of freedom (3 angles). The translation vector $\mathbf{t}$ has also 3 parameters making a total of six parameters for the extrinsic matrix.

Apart from the intrinsic and extrinsic parameters of the camera, we also finds the radial and tangential lens distortion coefficients (App. A.2). We use a regular chessboard pattern to calibrate the camera. The black and white alternative pattern ensures that there is no bias towards any sides during measurement [11]. It takes in the chessboard image and the size of the pattern (internal corners along with width and height) to calculate the camera parameters [95].

2.2.5 Rectification

In stereo vision, two cameras are used to get a pair of images from different viewpoints. The rectification process reprojects the two non-coplanar image planes onto a common plane parallel to the line between the optical centres. The process resamples the stereo pair such that all epipolar lines (App. A.2.1) are horizontal and the pixel motion is constrained to horizontal motion only (corresponding points have identical y-coordinate). The transformation is based on a homography transformation computed from the fundamental matrix (App. A.2.1). We fix one image plane, and the rectification process determines a transformation of the other image plane such that pairs of epipolar lines become collinear and parallel to the axes of the fixed image plane. Since

many projective transformations can achieve this, a typical rectification method attempts to find a pair of transformation that undergoes minimal image distortion. The important advantage of rectification is that it reduces the stereo correspondence problem to a 1-D search problem along the horizontal scan line of the rectified images. As most stereo matching algorithms assume that the images are taken from a parallel stereo rig, image rectification is often performed before the stereo matching method.

2.2.6 Stereo pairs

We take advantage of the static scene and use only one camera to capture the scene from two substantially overlapping but different viewpoints. The camera intrinsic and the lens distortion parameters (App. A.2) remain the same for the image pair. We first find out the camera intrinsics and the lens distortion coefficients using a regular chequerboard method. Using the lens distortion coefficients, both images are then undistorted. Later, the undistorted image pairs and the camera intrinsics are passed to an OpenCV [45] routine where it uses scale-invariant feature transform (SIFT) [56] algorithm to detect some matching points. It further uses epipolar constraints to discard the outliers. Based on the refined matching points, it then calculates the extrinsic parameters. Once both parameters are known, we rectify the images using Bouguet's algorithm [10] in OpenCV.

2.3 Disparity space

Consider a rectified stereo rig, i.e. epipolar lines are parallel to the x-axis. There is no loss of generality as once the epipolar geometry (App. A.2.1) is known, we can rectify the images from the stereo rig [33]. We also assume that both cameras have the same camera intrinsic. Let f be the focal length, $(o_x, o_y)^\top$ be the principal point coordinates associated with the stereo rig in pixel coordinates and B be the baseline of the stereo rig. Let $\bar{\mathbf{p}} = (x, y)^\top$ and $\bar{\mathbf{p}}' = (x', y')^\top$ be the two corresponding points (in pixel coordinates) of a 3D point $\mathbf{P} = (X, Y, Z)^\top$ (in world coordinates, App. A.1) in a rectified image pair. Since the corresponding image points must lie on the epipolar lines, we get the following relation:

$$\begin{cases} x' = x - d \\ y' = y \end{cases}$$

where d (in pixels) is defined as the disparity of $\bar{\mathbf{p}}$ [85].

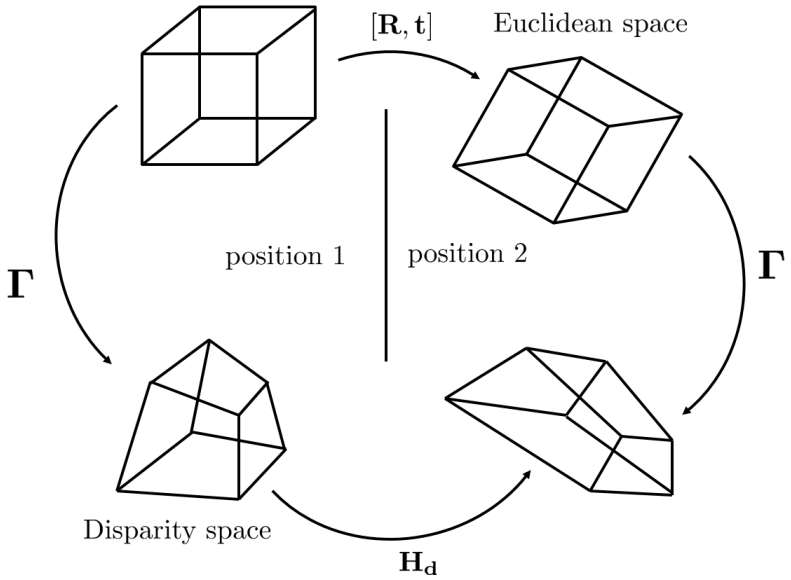


Figure 2.4: Euclidean reconstruction and motion of a cube vs. reconstruction and motion in the disparity space. Figure reproduced from [18].

In homogeneous coordinates (App. A.1.1), using eq. 2.2.3 and incorporating eq. 1.2, we can write,

$$\begin{pmatrix} x \\ y \\ d \\ 1 \end{pmatrix} = \frac{1}{Z} \begin{pmatrix} 1 & 0 & 0 & o_x \\ 0 & 1 & 0 & o_y \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_x & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & Bf & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

which simplifies to,

$$\begin{pmatrix} x \\ y \\ d \\ 1 \end{pmatrix} = \simeq \mathbf{\Gamma} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}, \quad \text{where } \mathbf{\Gamma} = \begin{pmatrix} fs_x & 0 & o_x & 0 \\ 0 & fs_y & o_y & 0 \\ 0 & 0 & 0 & Bfs_x \\ 0 & 0 & 1 & 0 \end{pmatrix}. \tag{2.5}$$

The 3D space generated by (x,y,d) is called the disparity space $\mathcal{D}$. The projective transformation $\mathbf{\Gamma}$ between the world coordinates $(X,Y,Z,1)^\intercal \in \mathcal{S}$ in the 3D Euclidean space and the pixel coordinates $(x,y,d,1)^\intercal \in \mathcal{D}$ is demonstrated by eq. 2.5. Therefore $(x,y,d,1)^\intercal$ is a projective reconstruction of the scene. Hence the disparity space is a projective space [18].

2.3.1 Rigid transformation in disparity space

The rigid transformation that maps two reconstructions of a rigid moving object in the disparity space is called d-motion (Fig. 2.4). As disparity images have a well-behaved noise, d-motion can be used instead of the 3-D Euclidean space for motion estimation.

Let us consider a fixed stereo rig observing a moving point. Let $\mathbf{P}_1 = (X_1, Y_1, Z_1)^\mathsf{T} \in \mathcal{S}$ and $\mathbf{P}_2 = (X_2, Y_2, Z_2)^\mathsf{T} \in \mathcal{S}$ be the respective 3D Euclidean coordinates of the same point before and after the rigid motion. Let $\mathbf{R}$ be the rotation matrix and $\mathbf{t}$ be translation vector of this rigid motion. Let $\bar{\mathbf{p}}_1 = (x_1, y_1, d_1)^\mathsf{T} \in \mathcal{D}$ and $\bar{\mathbf{p}}_2 = (x_2, y_2, d_2)^\mathsf{T} \in \mathcal{D}$ be the corresponding pixel coordinates of $\mathbf{P}_1$ and $\mathbf{P}_2$ in disparity space respectively. Using homogeneous coordinates we can model this motion as :

$$\begin{pmatrix} \mathbf{P}_2 \\ 1 \end{pmatrix} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \begin{pmatrix} \mathbf{P}_1 \\ 1 \end{pmatrix} \tag{2.6}$$

From eq. 2.5, we get:

$$\begin{aligned} \begin{pmatrix} \bar{\mathbf{p}}_1 \\ 1 \end{pmatrix} &= \frac{1}{Z_1} \mathbf{\Gamma} \begin{pmatrix} \mathbf{P}_1 \\ 1 \end{pmatrix} \\ \begin{pmatrix} \bar{\mathbf{p}}_2 \\ 1 \end{pmatrix} &= \frac{1}{Z_2} \mathbf{\Gamma} \begin{pmatrix} \mathbf{P}_2 \\ 1 \end{pmatrix} \end{aligned} \tag{2.7}$$

Replacing $(\mathbf{P}_1, 1)^\mathsf{T}$ and $(\mathbf{P}_2, 1)^\mathsf{T}$ from eq. 2.7 in eq. 2.6, we get,

$$Z_2 \mathbf{\Gamma}^{-1} \begin{pmatrix} \bar{\mathbf{p}}_2 \\ 1 \end{pmatrix} = Z_1 \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \mathbf{\Gamma}^{-1} \begin{pmatrix} \bar{\mathbf{p}}_1 \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} \bar{\mathbf{p}}_2 \\ 1 \end{pmatrix} = \frac{Z_1}{Z_2} \mathbf{\Gamma} \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \mathbf{\Gamma}^{-1} \begin{pmatrix} \bar{\mathbf{p}}_1 \\ 1 \end{pmatrix} \simeq \mathbf{\Gamma} \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^\mathsf{T} & 1 \end{pmatrix} \mathbf{\Gamma}^{-1} \begin{pmatrix} \bar{\mathbf{p}}_1 \\ 1 \end{pmatrix}$$

Let $\mathbf{H_d} = \mathbf{\Gamma} \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{pmatrix} \mathbf{\Gamma}^{-1}$. Then $\begin{pmatrix} \bar{\mathbf{p}}_2 \\ 1 \end{pmatrix} \simeq \mathbf{H_d} \begin{pmatrix} \bar{\mathbf{p}}_1 \\ 1 \end{pmatrix}$. After simplification, we find $\mathbf{H_d}$ has the

following matrix

$$\mathbf{H_d} = \begin{pmatrix} r_{11} + \frac{o_x}{fs_x} r_{31} & \frac{s_x}{s_y} r_{12} + \frac{o_x}{fs_y} r_{32} & \frac{1}{B} t_1 + \frac{o_x}{Bfs_x} t_3 & -\left(r_{11} + \frac{o_x}{fs_x} r_{31}\right) o_x - \left(\frac{s_x}{s_y} r_{12} + \frac{o_x}{fs_y} r_{32}\right) o_y + fs_x r_{13} + o_x r_{33} \\ \frac{s_y}{s_x} r_{21} + \frac{o_y}{fs_x} r_{31} & r_{22} + \frac{o_y}{fs_y} r_{32} & \frac{s_y}{Bs_x} t_2 + \frac{o_y}{Bfs_x} t_3 & -\left(\frac{s_y}{s_x} r_{21} + \frac{o_y}{fs_x} r_{31}\right) o_x - \left(r_{22} + \frac{o_y}{fs_y} r_{32}\right) o_y + fs_y r_{23} + o_y r_{33} \\ 0 & 0 & 1 & 0 \\ \frac{1}{fs_x} r_{31} & \frac{1}{fs_y} r_{32} & \frac{1}{Bfs_x} t_3 & -\frac{o_x}{fs_x} r_{31} - \frac{o_y}{fs_y} r_{32} + r_{33} \end{pmatrix}$$

Let us assume the pixels are square. Then s_x=s_y, and we call it s. Also define $\alpha = fs$, where α denotes the focal length in pixel unit. Then $\mathbf{H_d}$ can be further simplified as follows :

$$\mathbf{H_d} = \begin{pmatrix} r_{11} + \frac{o_x}{\alpha} r_{31} & r_{12} + \frac{o_x}{\alpha} r_{32} & \frac{1}{B} t_1 + \frac{o_x}{B\alpha} t_3 & -\left(r_{11} + \frac{o_x}{\alpha} r_{31}\right) o_x - \left(r_{12} + \frac{o_x}{\alpha} r_{32}\right) o_y + \alpha r_{13} + o_x r_{33} \\ r_{21} + \frac{o_y}{\alpha} r_{31} & r_{22} + \frac{o_y}{\alpha} r_{32} & \frac{1}{B} t_2 + \frac{o_y}{B\alpha} t_3 & -\left(r_{21} + \frac{o_y}{\alpha} r_{31}\right) o_x - \left(r_{22} + \frac{o_y}{\alpha} r_{32}\right) o_y + \alpha r_{23} + o_y r_{33} \\ 0 & 0 & 1 & 0 \\ \frac{1}{\alpha} r_{31} & \frac{1}{\alpha} r_{32} & \frac{1}{B\alpha} t_3 & -\frac{o_x}{\alpha} r_{31} - \frac{o_y}{\alpha} r_{32} + r_{33} \end{pmatrix} \quad (2.8)$$

We can express the transformation in a more compressed form. Let $\mathbf{R_1} = \begin{pmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{pmatrix}$, $\mathbf{r} = \begin{pmatrix} r_{13} \\ r_{23} \end{pmatrix}$, $\mathbf{s} = \begin{pmatrix} r_{31} \\ r_{32} \end{pmatrix}$, $\lambda = R_{33}$, $\bar{\mathbf{t}} = \begin{pmatrix} t_1 \\ t_2 \end{pmatrix}$, and $\mu = t_3$. Then

$$\mathbf{R} = \begin{pmatrix} \mathbf{R_1} & \mathbf{r} \\ \mathbf{s^T} & \lambda \end{pmatrix} \quad \text{and} \quad \mathbf{t} = \begin{pmatrix} \bar{\mathbf{t}} \\ \mu \end{pmatrix}$$

Let,

$$\mathbf{T} = \begin{pmatrix} 1 & 0 & \frac{o_x}{\alpha} \\ 0 & 1 & \frac{o_y}{\alpha} \end{pmatrix}, \quad \mathbf{O} = \begin{pmatrix} o_x \\ o_y \end{pmatrix}, \quad \bar{\mathbf{R}} = \begin{pmatrix} \mathbf{R_1} \\ \mathbf{s^T} \end{pmatrix}, \quad \mathbf{R_{\bullet 3}} = \begin{pmatrix} r \\ \lambda \end{pmatrix}$$

Then $\mathbf{H_d}$ can be expressed as the following:

$$\mathbf{H_d} = \begin{pmatrix} \mathbf{T}\bar{\mathbf{R}} & \frac{1}{B}\mathbf{T}\mathbf{t} & \mathbf{T}\bar{\mathbf{R}}\mathbf{O} + \alpha\mathbf{T}\mathbf{R_{\bullet 3}} \\ 0 & 1 & 0 \\ \frac{1}{\alpha}\mathbf{s^T} & \frac{\mu}{\alpha B} & -\frac{1}{\alpha}\mathbf{s^T}\mathbf{O} + \lambda \end{pmatrix} \quad (2.9)$$

The transformation $\bar{\mathbf{p}}_2 = \mathbf{H_d}\bar{\mathbf{p}}_1$ is called the *d-motion* [18].

2.3.2 Disparity gradient

The stereo vision problem is usually formulated in the disparity space $d(x,y)$, where d is the positional disparity (measured in pixels) at pixel $\bar{\mathbf{p}}$. Using the left (reference) camera coordinate system as the world coordinate system, let $\mathbf{P}$ be the corresponding scene point of $\bar{\mathbf{p}}$. The depth at $\bar{\mathbf{p}}$ is $Z = \dfrac{\alpha B}{d(x,y)}$, where α is the focal length in the pixel unit and B is the baseline of a rectified stereo pair.

Disparity gradients are used to constrain the surface orientation for stereo correspondence [55]. To find the relation between the depth gradients (in Euclidean space) and the disparity gradients (in disparity space), we need to relate the partial derivatives of Z with respect to X and Y, and the partial derivatives of d with respect to x and y.

The image coordinates x and y are in pixel unit whereas X and Y are in the physical unit. So $\dfrac{\partial x}{\partial X}$ and $\dfrac{\partial y}{\partial Y}$ can be determined by quantisation of the image sensor, which turns out to be a constant. Let s_x and s_y denote the size of a pixel in the metric unit along the X and Y axis respectively. We further assume that the pixels are square. In that case $s_x = s_y = s$, which follows:

$$\frac{\partial x}{\partial X} = s_x = s, \quad \frac{\partial y}{\partial Y} = s_y = s, \quad \alpha = fs, \tag{2.10}$$

where f is the focal length in physical unit. The relation between pixel unit and physical unit is given by eq. 2.10. Next we find the partials Z_X and Z_Y.

$$\begin{aligned} Z_X &= \frac{\partial Z}{\partial X} = \frac{\partial}{\partial X}\left(\frac{\alpha B}{d}\right) = \alpha B \frac{\partial}{\partial x}\left(\frac{1}{d}\right)\frac{\partial x}{\partial X} = -\frac{\alpha B}{d^2}\frac{\partial d}{\partial x}\frac{\partial x}{\partial X} = -\frac{\alpha s_x B}{d^2}\frac{\partial d}{\partial x} = -\frac{\alpha s B}{d^2}\frac{\partial d}{\partial x} \\ Z_Y &= \frac{\partial Z}{\partial Y} = \frac{\partial}{\partial Y}\left(\frac{\alpha B}{d}\right) = \alpha B \frac{\partial}{\partial y}\left(\frac{1}{d}\right)\frac{\partial y}{\partial Y} = -\frac{\alpha B}{d^2}\frac{\partial d}{\partial y}\frac{\partial y}{\partial Y} = -\frac{\alpha s_y B}{d^2}\frac{\partial d}{\partial y} = -\frac{\alpha s B}{d^2}\frac{\partial d}{\partial y} \end{aligned} \tag{2.11}$$

The relation between the depth gradient and the disparity gradient is given by eq. 2.11 [53]. From eq. 2.5 we get :

$$\begin{pmatrix} x \\ y \\ d \end{pmatrix} = \begin{pmatrix} \alpha\frac{X}{Z} + o_x \\ \alpha\frac{Y}{Z} + o_y \\ \alpha\frac{B}{Z} \end{pmatrix} \tag{2.12}$$

Let $\bar{\mathbf{p}} = (x, y, d)^\top$. Taking partial derivatives of each component of $\bar{\mathbf{p}}$ with respect to x we get,

$$1 = \frac{\partial}{\partial x}\left(\alpha\frac{X}{Z} + o_x\right) = \frac{\alpha}{Z}\frac{\partial X}{\partial x} - \frac{\alpha X}{Z^2}\frac{\partial Z}{\partial x} = \frac{fs}{Z}\frac{1}{s} - \frac{fsX}{Z^2}\frac{\partial Z}{\partial X}\frac{\partial X}{\partial x} = \frac{f}{Z} - \frac{fX}{Z^2}Z_X = \frac{f}{Z}\left(1 - \frac{X}{Z}Z_X\right)$$

$$0 = \frac{\partial}{\partial x}\left(\alpha\frac{Y}{Z} + o_y\right) = -\frac{\alpha Y}{Z^2}\frac{\partial Z}{\partial x} = -\frac{fsY}{Z^2}\frac{\partial Z}{\partial X}\frac{\partial X}{\partial x} = -\frac{fY}{Z^2}Z_X$$

$$d_x = \frac{\partial}{\partial x}\left(\alpha\frac{B}{Z}\right) = -\frac{\alpha B}{Z^2}\frac{\partial Z}{\partial x} = -\frac{fsB}{Z^2}\frac{\partial Z}{\partial X}\frac{\partial X}{\partial x} = -\frac{fB}{Z^2}Z_X$$

$$\implies \bar{\mathbf{p}}_x = \begin{pmatrix} 1 \\ 0 \\ d_x \end{pmatrix} = -\frac{1}{Z}\begin{pmatrix} f\left(\frac{Z_X}{Z} - \frac{1}{X}\right) & 0 & 0 \\ 0 & f\frac{Z_X}{Z} & 0 \\ 0 & 0 & fB\frac{Z_X}{Z^2} \end{pmatrix}\begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$$

Similarly, taking partial derivatives of each component of $\bar{\mathbf{p}}$ with respect to y we get, by analogy,

$$\bar{\mathbf{p}}_y = \begin{pmatrix} 1 \\ 0 \\ d_y \end{pmatrix} = -\frac{1}{Z}\begin{pmatrix} f\frac{Z_Y}{Z} & 0 & 0 \\ 0 & f\left(\frac{Z_Y}{Z} - \frac{1}{Y}\right) & 0 \\ 0 & 0 & fB\frac{Z_Y}{Z^2} \end{pmatrix}\begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$$

2.4 The stereo matching problem

Detecting conjugate pixels in a stereo image pair, *i.e.,* for each pixel in the reference image, finding the corresponding pixel in the search image, is a fundamental problem known as the stereo matching or correspondence problem. One of the important tasks of a computer vision system is to determine the depth of a scene point relative to the camera position. Extracting depth information from a single image is extremely hard. However, given a pair of images acquired by a synchronised stereo rig, we can extract the depth information using stereo algorithms. The pixel-by-pixel matching between the images gives us the depth map or the so-called disparity map. For rectified images, the matching process is reduced to one dimensional where corresponding pixels lies on a horizontal scan line, as shown in Fig. 2.5. A point $\bar{\mathbf{p}} \in I$ may arise from any scene point on $\mathsf{C}\bar{\mathbf{p}}$ and will appear in the right image I' at any point on the epipolar line $\mathbf{l}'$. Stereo matching algorithms can now be used to find the disparity of each pixel.

Let us consider a rectified stereo image pair I and I' captured from the left (C) and right (C') cameras respectively, as shown in Fig. 2.6 [8]. In the stereo matching box in Fig. 2.6, we have put I on top of I' to understand why disparity allows depth reasoning. The background pixel

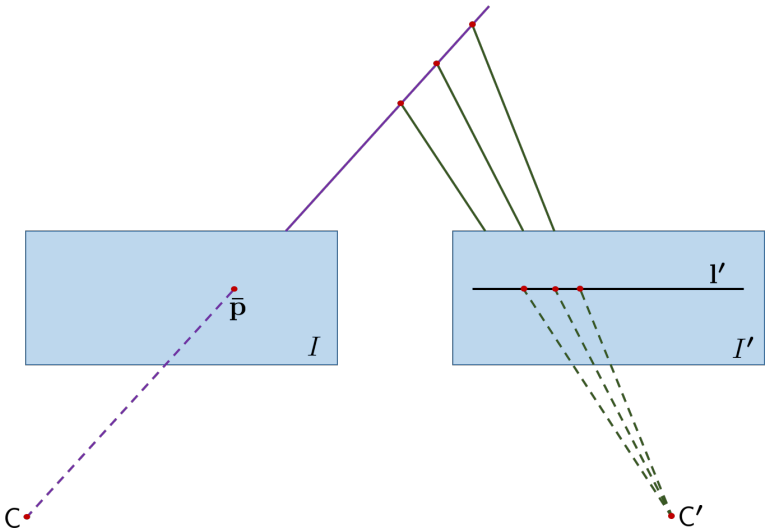


Figure 2.5: Geometry of epipolar lines, where C and C′ are left and right camera centres respectively. Point $\bar{\mathbf{p}}$ in the left image I may arise from any points on the extended C$\bar{\mathbf{p}}$ and will appear in the right image I' at any point on the epipolar line $\mathbf{l}'$.

$\bar{\mathbf{p}} \in I$ and its matching pixel $\bar{\mathbf{p}}' \in I'$ are displaced in the horizontal direction due to the different perspective under which they have been captured. The amount of displacement is the disparity d_{p}. Observe that the disparity d_{q} of the foreground pixel $\bar{\mathbf{q}} \in I$ and its matching pixel $\bar{\mathbf{q}}' \in I'$ is larger than d_{p}. This can be easily justified as we have seen in Sec. 1.7.1 that the disparity is inversely proportional to the distance of the point from the camera. Note that the human brain also perceives depth using disparity information [73].

After finding the disparity of each pixel, the disparity map is stored as an intensity image

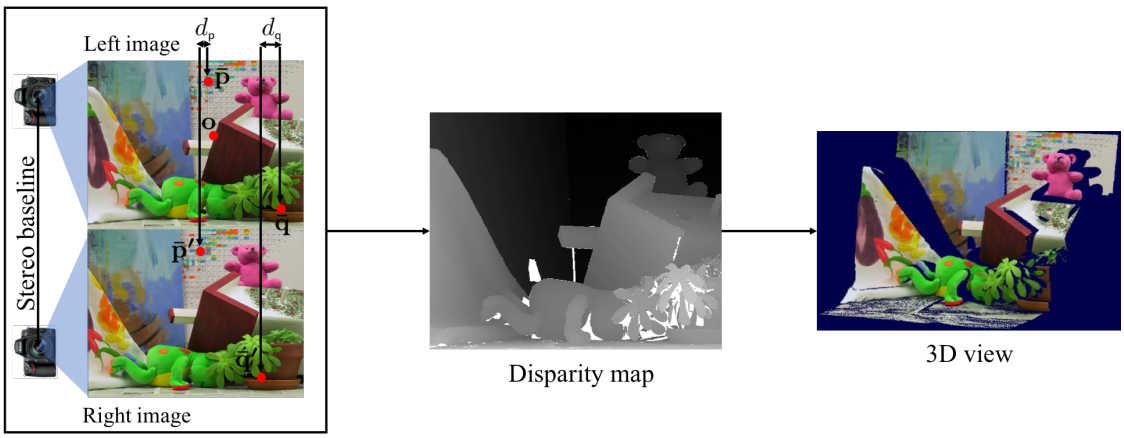


Figure 2.6: Depth reconstruction via stereo matching. Two slightly displaced cameras capture the scene from two different viewpoints. The stereo matching module finds dense correspondence between the images and produces a disparity map, which latter reconstructs the 3D view of the scene.

where darker pixels encode low disparity value (furthest from the camera), and brighter pixels encode large disparities (closest to the camera) (Fig. 2.6). The disparity map is sufficient to reconstruct a metric 3D model of the image, which is the final goal in the shape from stereo approach.

Stereo matching algorithms must overcome a variety of challenges resulting from scene properties and image acquisition processes. Challenges related to scene properties include depth discontinuities and occlusions (see the marked pixel '**o**' in Fig. 2.6), low-textured areas and repetitive patterns, reflections and illumination differences. Challenges arising from the image acquisition process include the effects of quantisation, noise and image blur. General stereo matching challenges can be found in Fig. 2.7.

Stereo correspondence algorithms can be grouped into those producing sparse output and those giving dense output. Our work mainly focuses on dense output as contemporary applications demand dense output. According to disparity assignment to pixels, stereo matching algorithms can be broadly classified as local or global methods [75]. Local methods (area based) trade accuracy for speed. They are also known as window-based methods as the disparity computation at a given pixel depends only on the intensity values within the support window. Global methods (energy-based) on the other hand are time-consuming but very accurate. Their goal is to minimise a global cost function for the whole image combining data and smoothness term. There are also some methods other than the local and global ones. A comparison of the current stereo methods can be found on the Middlebury stereo evaluation web-page [76].

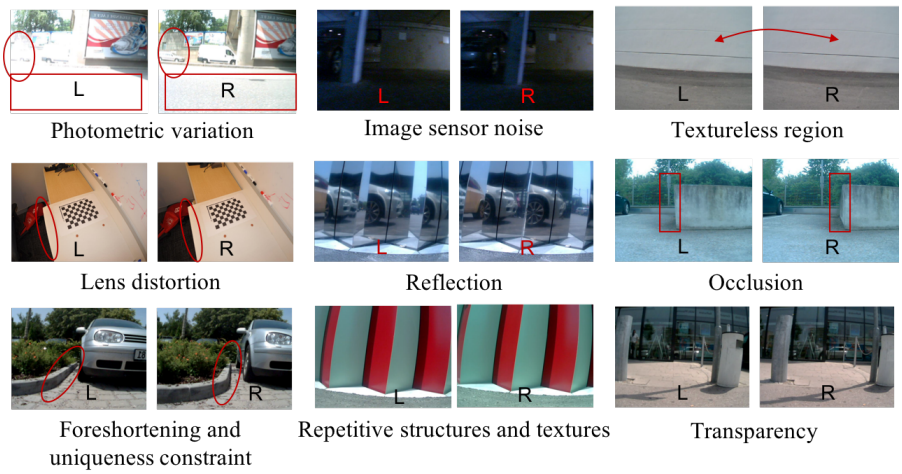


Figure 2.7: Stereo matching challenges. (L) Left image, (R) Right image.

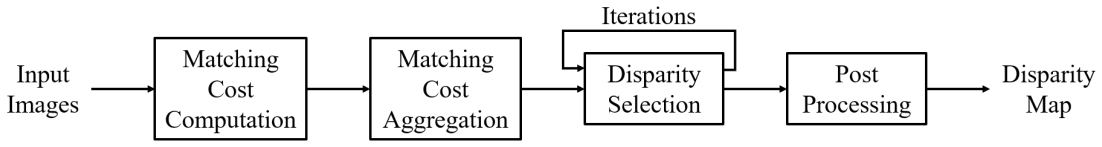


Figure 2.8: Generalized block diagram of a dense local stereo correspondence algorithm [75].

2.4.1 Local methods

Local methods are driven by the compatibility of individual matches, which assign disparities to pixels according to the information provided by its local, neighbouring pixels. Local methods are usually fast and produces relatively crude disparity maps.

Most local methods involve four stages for computing the disparity map: matching cost computation, cost aggregation, disparity computation and disparity refinement [75]. A cost function is used to measure the matching cost, *i.e.,* the apparent dissimilarity across views. Common pixel-based matching costs are based on the absolute or squared difference of colour values. Since the colour values may be misleading, image gradients are often combined with the cost function [49]. The disparity is selected by comparing the aggregated matching cost at different disparity values and using the winner-takes-all (WTA) strategy. Finally, some post-processing is performed to fill in the occluded pixels and mismatches via a left/right consistency check [75]. Filling the occluded regions often generates artefacts in the disparity map. To solve this problem, filters (e.g. weighted median) are applied on the disparity map for smoothing the artefacts [37], [74]. A general structure of the majority of dense local stereo algorithms is shown in Fig. 2.8. Note that most local stereo matching algorithm implicitly assumes the standard stereo photo consistency, which means matching objects have similar colours across the two input views.

2.4.2 Global methods

Global methods assign disparities to pixels depending on information derived from the whole image [77, 90]. Global methods make explicit piecewise smoothness assumptions and minimise a global cost function. The extra smoothness assumption helps global methods to produce a more accurate disparity map compared to local methods, but they often require longer execution time [66]. Global algorithms define an energy function $E(D)$ that measures the quality of the disparity map with an explicit smoothness term. The goal is to find the optimum disparity map $\mathcal{D}$, which minimises $E(D)$. Typically $E(D)$ is the combination of data and smoothness term and

can be expressed as

$$E(d) = E_{\text{data}}(d) + \lambda \cdot E_{\text{smooth}}(d), \tag{2.13}$$

where, λ is a user defined regularisation parameter that balances the influence of E_{data} and E_{smooth}. The data term E_{data} measures colour similarity and is defined as

$$E_{\text{data}} = \sum_{\bar{\mathbf{p}} \in I} \text{cost}\left(\bar{\mathbf{p}}, \bar{\mathbf{p}} - d_{\bar{\mathbf{p}}}\right), \tag{2.14}$$

where, I is the left image and $d_{\bar{\mathbf{p}}}$ represents the disparity of $\bar{\mathbf{p}}$. The cost function $\text{cost}\left(\bar{\mathbf{p}}, \bar{\mathbf{q}}\right)$ computes the pixel dissimilarity between $\bar{\mathbf{p}} \in I$ and $\bar{\mathbf{q}} \in I'$ (e.g., absolute difference of intensity values), where I' is the right image.

We now focus on the E_{smooth} term. It is responsible for preferring disparities that are spatially smooth by assigning a lower energy. E_{smooth} is defined as

$$E_{\text{smooth}} = \sum_{\langle \bar{\mathbf{p}}, \bar{\mathbf{q}} \rangle \in \mathcal{N}} \text{s}\left(d_{\bar{\mathbf{p}}}, d_{\bar{\mathbf{q}}}\right), \tag{2.15}$$

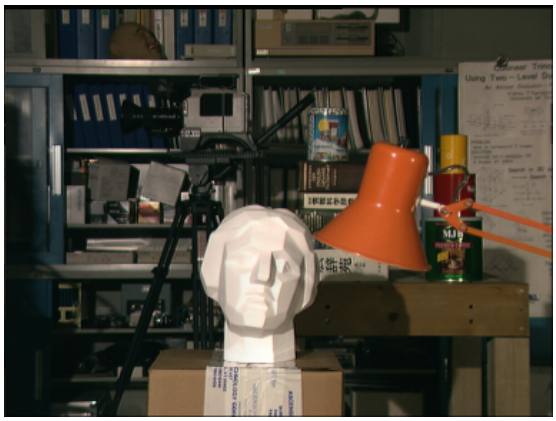
where, $\mathcal{N}$ denotes all pairs of spatially neighbouring pixels in the left image. It is often referred as a pairwise term as it is defined on pairs of pixels whereas the data term is referred to as unary term. The function $\text{s}\left(d_{\bar{\mathbf{p}}}, d_{\bar{\mathbf{q}}}\right)$ assigns a penalty if $\bar{\mathbf{p}}$'s disparity is different from that of $\bar{\mathbf{q}}$. For example in Potts model [70], it is defined as

$$\text{s}\left(d_{\bar{\mathbf{p}}}, d_{\bar{\mathbf{q}}}\right) = \begin{cases} 0, & \text{if } |d_{\bar{\mathbf{p}}} - d_{\bar{\mathbf{q}}}| < \tau, \\ 1, & \text{otherwise,} \end{cases} \tag{2.16}$$

where τ is a user defined threshold.

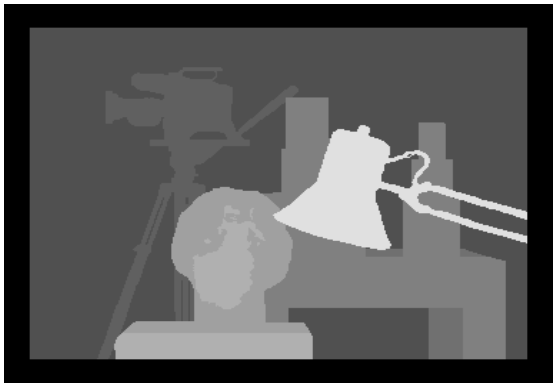
2.5 Dense local stereo algorithms

We start by describing the simplest possible dense local stereo algorithm. The photo consistency assumption tells us that corresponding pixels in the left (I) and right (I') image have similar colours. If the images are rectified, we can also say that the corresponding pixels lie on the same horizontal scan line. For each pixel $\bar{\mathbf{p}} \in I$, a local method searches along the corresponding scan line in I'. We then select the pixel $\bar{\mathbf{p}}' \in I'$ whose colour is most similar to $\bar{\mathbf{p}}$. Fig. 2.9d



(a) Left image

(b) Right image



(c) Ground-truth disparity

(d) Disparity map from naive local algorithm

Figure 2.9: Disparity results on the Middlebury Tsukuba stereo pair

shows the noisy disparity map of this naive approach. The problem is the high ambiguity of the data, i.e., the correspondence of a red pixel in Fig. 2.9a has a relatively large candidate set of red pixels in the right image (Fig. 2.9b). The common approach is to regularise this problem by imposing a smoothness assumption to cope with this ambiguity [8]. This smoothness assumption means that the neighbouring spatial pixels are likely to have similar disparities. This assumption is implemented differently in various stereo algorithms and is also the main difference between local and global methods [75].

The naive approach can be improved by matching small image areas rather than taking a single pixel. We use the smoothness assumption by assuming every pixel inside the area has the same disparity. We refer this small image area as patch or window. Local methods find the disparity of a pixel by considering a support window (patch) centred at that pixel in the reference image. The support window is then projected in the search image. Local algorithms choose disparity that minimises the dissimilarity between the two support windows. Based on this idea, we can reformulate the corresponding algorithm. For each pixel $\bar{\mathbf{p}} \in I$, we compute its

disparity as

$$d_{\bar{\mathbf{p}}} = \underset{d_{\min} \leq d \leq d_{\max}}{\arg\min} \sum_{\bar{\mathbf{q}} \in \mathcal{W}(\bar{\mathbf{p}})} \text{cost}\,(\bar{\mathbf{q}}, \bar{\mathbf{q}} - d)\,, \tag{2.17}$$

where, $d_{\min}$ and $d_{\max}$ are parameters defining the minimum and maximum allowed disparity respectively. $\mathcal{W}(\bar{\mathbf{p}})$ denotes a patch centred at $\bar{\mathbf{p}}$. The function $\text{cost}(\bar{\mathbf{p}}, \bar{\mathbf{q}})$ computes the colour dissimilarity between $\bar{\mathbf{p}} \in I$ and $\bar{\mathbf{q}} \in I'$. We write $\bar{\mathbf{q}} - d$ to denote the pixel coordinate of q in the other view by subtracting the disparity d from $\bar{\mathbf{q}}$'s x-coordinate.

2.5.1 Patch size

The projection matrix which transfers the support window in the search image holds the key to find corresponding pixels in all local stereo algorithms. The fronto-parallel planar model (planes parallel to the image plane) assumes that all the pixels inside the support window have constant disparity (that of the centre pixel) [92, 64, 63]. The translation matrix[1] between two rectified images induced by the fronto-parallel plane in the disparity space is given by:

$$\mathbf{M} = \begin{pmatrix} 1 & 0 & 0 & d_0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

where d_0 is the disparity of the centre pixel inside the support window. This assumption makes the patch size $|\mathcal{W}|$ an important parameter in any local algorithm that highly influences disparity accuracy and computational time.

The size needs to be proportional to the image resolution and the viewpoint separation and, inversely proportional to the depth of the scene, which is unknown. From a computational point of view, the algorithm's runtime complexity is $\mathcal{O}(N \cdot (d_{\max} - d_{\min}) \cdot |\mathcal{W}|)$, where N is the number of pixels in the image. Hence larger patch size leads to higher computational run time. However using sliding window technique [21], [64], the runtime complexity can be reduced to $\mathcal{O}(N \cdot (d_{\max} - d_{\min}))$. If we use a small patch, the algorithm will be faster but other problems arise, such as the patch may not capture enough texture variation to resolve the matching ambiguities. In particular, the algorithm fails in untextured regions and repetitive image regions. However,

[1] The translation matrix will be generalized later to incorporate non-fronto-parallel planes.

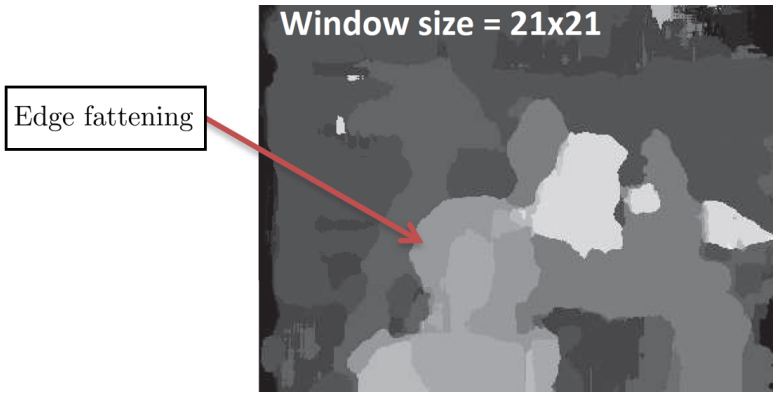


Figure 2.10: Edge fattening problem [8]

disparity discontinuities are well preserved as a consequence of small patch size. Large patch produces a smoother disparity map, but object borders are badly preserved. This is because of our implicit smoothness assumption, i.e., pixels within the patch have constant disparity. In the proximity of depth discontinuities, this assumption is broken as the patch captures a mixture of foreground and background disparities. The disparity of the foreground and background pixels depends on the lower colour dissimilarity of the patch, which depends on the texture of the object. In many cases, the foreground's texture is dominant or both foreground and background regions contain entirely untextured surfaces, which leads to the well-known edge fattening problem [8] (Fig. 2.10).

Fusiello *et al.* [23], Hirschmuller *et al.* [38] and Veksler [87] proposed the adaptive window approach where each pixel selects an individual patch size such that the support region of that pixel remains large enough to capture enough intensity variation but does not overlap a disparity discontinuity in order to avoid the edge fattening problem. In practice, none of the above methods has been able to compete with the quality of global methods.

2.5.2 Patch shape

Conventional area-based stereo matching employs a "square window" to measure the similarity between two patches, which implicitly assumes that the intensity patterns surrounding the corresponding points have no deformations between different views. However, in general, the local intensity patterns deform according to the view and surface orientations. As the surface orientations vary with image regions, the patch must be locally deformed depending on the surface orientations. Kanade *et al.* [47] proposed a method that uses the local variation of the intensity and the disparity to employ a statistical disparity distribution within a patch. The method

searches for a window that produces the estimate of disparity with the least uncertainty for each pixel of an image: the method controls not only the size but also the shape (rectangle) of the patch. Hattori *et al.* [34] proposed an algorithm that estimates the depth and surface orientations simultaneously. The algorithm locally deforms the patch according to the surface orientation which is directly recovered from intensity gradients within the patch.

2.5.3 Adaptive support weight

The adaptive support weight [91] was proposed to overcome the edge fattening problem, where the influence of the pixels inside the patch depends on the colour and spatial similarity with the centre pixel of the patch. This strategy reduces the edge fattening problem while using a large patch size.

The adaptive support window [91] assumes that spatially close pixels that are of the same colour are likely to originate from the same scene object. Hence, they are also likely to share the same disparity. The key idea is to assign an individual weight to each pixel inside the patch. The weight function $A(\bar{\mathbf{p}}, \bar{\mathbf{q}})$ is defined as

$$A(\bar{\mathbf{p}}, \bar{\mathbf{q}}) = \exp\left\{ -\left(\frac{\text{color}(\bar{\mathbf{p}}, \bar{\mathbf{q}})}{\gamma_c} + \frac{\text{spatial}(\bar{\mathbf{p}}, \bar{\mathbf{q}})}{\gamma_s} \right) \right\}, \tag{2.18}$$

where γ_c and γ_s are user defined parameters and $\text{color}(\bar{\mathbf{p}}, \bar{\mathbf{q}})$ computes the colour dissimilarity of $\bar{\mathbf{p}} \in I$ and $\bar{\mathbf{q}} \in I'$ as the Euclidean distance between $\bar{\mathbf{p}}$'s and $\bar{\mathbf{q}}$'s colour values in RGB space. The

Figure 2.11: Adaptive support weight approach [91]

function spatial$(\bar{\mathbf{p}},\bar{\mathbf{q}})$ computes the Euclidean distance between $\bar{\mathbf{p}}$'s and $\bar{\mathbf{q}}$'s pixel coordinates. A pixel $\bar{\mathbf{q}}$ obtains high weight if it is similar to the centre pixel $\bar{\mathbf{p}}$ in term of colour and spatial position. Incorporating the weight function in eq. 2.17, we get

$$d_{\bar{\mathbf{p}}} = \underset{d_{\min}\leq d\leq d_{\max}}{\arg\min} \sum_{\bar{\mathbf{q}}\in\mathcal{W}(\bar{\mathbf{p}})} A(\bar{\mathbf{p}},\bar{\mathbf{q}})\cdot \text{cost}\,(\bar{\mathbf{q}},\bar{\mathbf{q}}-d)\ . \tag{2.19}$$

The adaptive support weight produces a good disparity map (Fig. 2.11). However, it comes with at the price of considerably increased runtime, which depends on the patch size. Hosni *et al.* [41] investigated different strategies for computing the adaptive support weights for local stereo algorithms.

2.5.4 Cost function

To determine the quality of matching, different algorithms use different measures e.g., absolute intensity difference (AD), squared intensity difference (SD), normalized cross correlation (NCC). Scharstein *et al.* have evaluated various cost functions in [75]. In general costs are aggregated over a support window which can be square or rectangular, fixed-size or adaptive ones around a pixel. The cost functions around a pixel $\bar{\mathbf{p}}$ can be expressed as follows:

Sum of Absolute Differences (SAD)

$$SAD(\bar{\mathbf{p}},d) = \sum_{(x,y)\in\mathcal{W}(\bar{\mathbf{p}})} |I(x,y)-I'(x,y-d)|, \tag{2.20}$$

Sum of Squared Differences (SSD)

$$SSD(\bar{\mathbf{p}},d) = \sum_{(x,y)\in\mathcal{W}(\bar{\mathbf{p}})} \big(I(x,y)-I'(x,y-d)\big)^2, \tag{2.21}$$

Normalized Cross Correlation (NCC)

$$NCC(\bar{\mathbf{p}},d) = \frac{\sum_{(x,y)\in\mathcal{W}(\bar{\mathbf{p}})} I(x,y)\cdot I'(x,y-d)}{\sqrt{\sum_{(x,y)\in\mathcal{W}(\bar{\mathbf{p}})} I^2(x,y)\cdot \sum_{(x,y)\in\mathcal{W}(\bar{\mathbf{p}}')} I'^2(x,y-d)}}, \tag{2.22}$$

where $I(x,y)$ and $I'(x,y)$ are the intensity values at (x,y) pixel location in the left and right image, $|\cdot|$ denotes the absolute value difference, d is the disparity value under consideration, and $\mathcal{W}(\bar{\mathbf{p}})$ is the support region around $\bar{\mathbf{p}}$.

The simpler algorithms make use of a winner-takes-all (WTA) strategy for selecting the disparity.

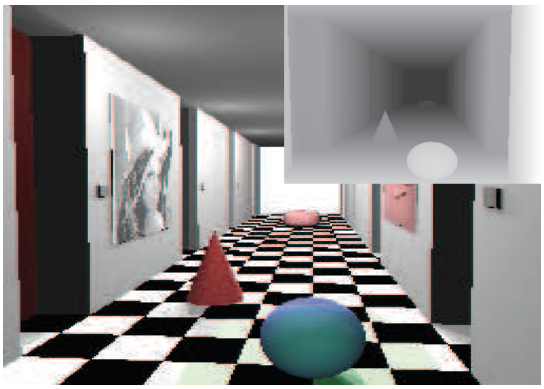
For *e.g.,* when the cost function is SAD, disparity d at $\bar{\mathbf{p}}$ is chosen as follows:

$$d^{\star} = \arg\min_{d} \ \mathrm{SAD}(\bar{\mathbf{p}}, d) \tag{2.23}$$

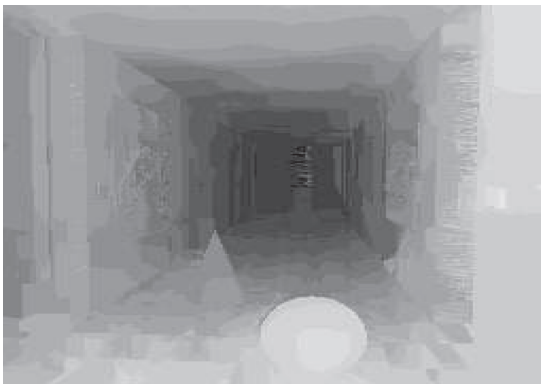
i.e., for every pixel $\bar{\mathbf{p}}$ and for a constant value of disparity d the minimum cost is selected. In many cases, the selection of disparity is an iterative process as each pixel's disparity also depends on its neighbouring disparity. As a result, we end up with more than one iteration to find the best disparity. An additional refinement step is frequently used to rule out the outliers, filling the occluded regions and smoothing out the disparity map.

Apart from these cost functions, many researchers have proposed alternative cost functions that are more efficient to handle low texture areas and repetitive patterns in an image. De-Maeztu *et al.* [17] proposed a dissimilarity measure based on the gradient fields of the images. Gradient-based cost function produces a superior disparity map because gradient measure is more robust than the classical intensity based measure. Gu *et al.* [31] used adaptive support-weight and rank transform to obtain an initial disparity and added a disparity calibration process to refine the final disparity map. Hosni *et al.* [42] presented a fast algorithm using adaptive support weight and guided image filtering that produced disparity maps comparable to global algorithms. In [62], Min *et al.* proposed a joint histogram based cost aggregation using per-pixel likelihood function. Their algorithm reduced the complexity of window-based matching while keeping a similar accuracy through the reference pixel-independent sampling of the matching window. Mei *et al.* [61] proposed the AD-Census algorithm which is a GPU-based stereo matching system with a good performance in both accuracy and speed. AD-Census effectively combines the absolute differences (AD) measure and the census transform which provides a robust measure for cost aggregation.

SAD and SSD are not robust to photometric changes but extremely fast to compute. On the contrary, NCC and census transform is robust to photometric changes but computationally expensive. Gradient-based cost function produces superior disparity map compared to the intensity based cost measures and it is computationally cheap as well. Gradient and intensity based cost functions are often combined to extend the robustness of cost aggregation. However, none of them is very successful at handling strong local radiometric changes caused by changing the location of the light sources.



(a) Left image and ground truth disparity of the corridor pair that contains highly slanted surfaces.



(b) Stair-case effect. Disparity map computed using fronto-parallel windows and integer-valued disparity.

Figure 2.12: Left image, ground-truth and disparity map [8]

2.5.5 Sub-pixel disparity

Most local algorithms measure the disparity values as integers whereas ideally matching should be done at continuous sub-pixel disparity values. Using fronto-parallel windows and assuming integer-valued disparity results in a discontinuity in the disparity values which is known as the 'stair-case effect' (Fig. 2.12b). To overcome this problem, we need to use locally continuous support regions matched at continuous sub-pixel disparity values.

Some local methods obtain the sub-pixel precision by fitting a parabola in the cost volume, which is often done in the post-processing step. This leads to noisy sub-pixel information as the scene surface is not always a paraboloid. A better solution is to compute sub-pixel displacements directly in the matching process. This can be accomplished by extending the label space, i.e., some fractional disparities are considered along with the integer-valued disparities [26]. It still remains a discrete approach and hence increases the runtime. We can cope with the slanted windows in the same fashion by including a set of slanted windows along with the fronto-parallel ones [25]. This is known as plane sweeping in literature. As mentioned before, this strategy also leads to longer run times.

Sinha *et al.* [80] removed the full search by deriving local plane hypothesis from a sparse feature correspondence. Local plane sweeps are then performed around each slanted plane to produce out-of-plane parallax and matching-cost estimates. This has the advantage of handling highly slanted surfaces without requiring many disparity hypotheses and without any bias towards fronto-parallel orientations. The local plane sweep not only performs sub-pixel registration, but it also deals with curved surfaces, which a single plane would fail to model.

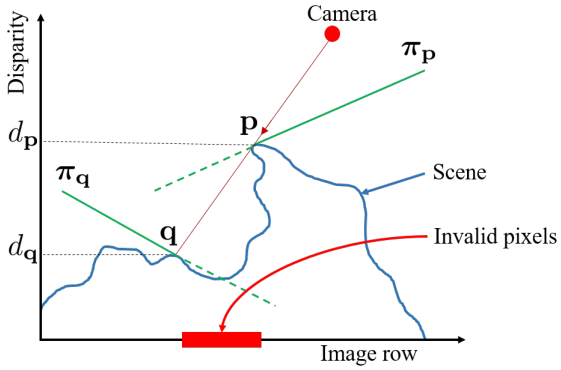


Figure 2.13: Occlusion handling. Holes are filled by computing the disparity of the invalid pixel according to left and right closest valid neighbouring pixel. Let $\mathbf{p}$ and $\mathbf{q}$ be the closest left and right points in the disparity space lying on planes $\boldsymbol{\pi}_{\mathbf{p}}$ and $\boldsymbol{\pi}_{\mathbf{q}}$ with disparity $d_{\mathbf{p}}$ and $d_{\mathbf{q}}$, respectively. The disparity of the invalid pixel is computed according to the plane $\boldsymbol{\pi}_{\mathbf{p}}$ and $\boldsymbol{\pi}_{\mathbf{q}}$ and the lowest disparity is selected [9].

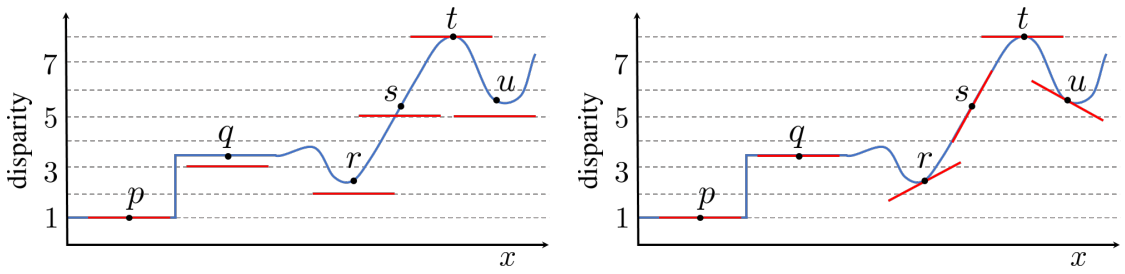
2.5.6 Occlusion

Local algorithms are unable to handle occlusion directly in the matching process. Typically occlusions are treated in a post-processing step by left-right consistency check [75]. This requires computation of the disparity map of the right image along with the left image. Ideally, a matching pair should have the same disparity magnitude. The left-right consistency check then invalidates pixels of the left image whose disparity value differs beyond a threshold in the right image. In the final step, these invalidated pixels are filled by replicating the lowest disparity of spatially neighbouring valid pixels in the horizontal direction (Fig. 2.13) [74], [44]. Selecting the lower disparity is motivated by the fact that occlusion occurs in the background. However, this strategy generates artefacts in the disparity map. To weaken this problem, Hirschmuller [37] proposed a weighted median filter to smooth the disparity map.

2.6 PatchMatch Stereo

In addition to depth, surface orientation information is important for understanding the scene geometry. The PatchMatch Stereo (*PMS*) framework [9] can simultaneously estimate the depth and the surface orientation at every pixel in an image. Algorithms [36, 5, 24] use the *PMS* framework to reconstruct the visible surfaces in the disparity space [75] defined by the pixel coordinates and the possible disparities.

Fig. 2.14a illustrates the problems of using fronto-parallel planes. The model only works where the surface segment coincides with the fronto-parallel plane at the whole-valued disparity



(a) Planar (Fronto-parallel) surface model at integer disparity.

(b) Planar (slanted) surface model of *PMS*.

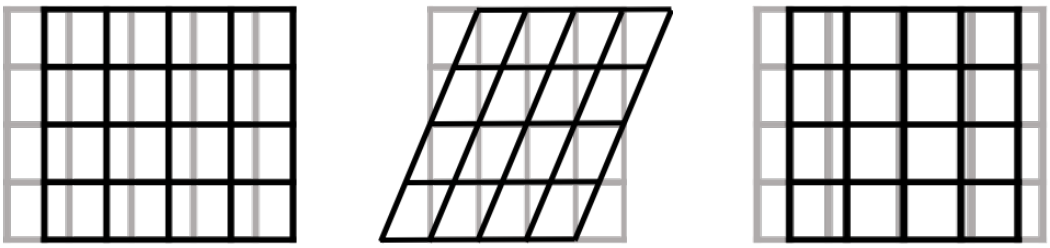
Figure 2.14: Planar surface model in 1D. The blue curve represents a 1D slice of the surface and the surface models are shown in red. Figure partially reproduced from [9].

(Fig. 2.14a, point p). The smoothness assumption in local methods does not hold in scenarios where, 1. the support window contains pixels that lie on a different surface than that of the center pixel, 2. the surface segment lies at a sub-pixel disparity (Fig. 2.14a, point q), 3. the support window captures a non-fronto-parallel surface, such as a slanted (Fig. 2.14a, point s) or curved surface (Fig. 2.14a, points r, t, u). The adaptive support weight provides the solution for the first problem and has already been implemented [42, 74]. For the latter, Gallup *et al.* [25] presented a model using slanted planes in disparity space instead of fronto-parallel ones and proposed a real time multi-view global method generating a sparse disparity map. Bleyer *et al.* [9] combined the slanted support and PatchMatch [3] in *PMS*. *PMS* overparametrizes the disparity space by computing individual slated planar surface at each pixel in the disparity space onto which the support region is projected (Fig. 2.14b). *PMS* is an iterative algorithm that relies on random sampling and propagation of good plane parameter estimates. Heise *et al.* [36] reformulated the projection matrix of *PMS* as a linear transformation:

$$\mathbf{M} = \begin{pmatrix} 1+a & b & 0 & c \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

The transformation between two rectified images induced by a plane $\boldsymbol{\pi}$ with parameters (a,b,c) in the disparity space has only three degrees of freedom, where a, b and c regulates the scaling, shearing and translation, respectively. Fig 2.15 illustrates the change of a support window when projected to the other view.

The reconstruction is based on associating a slanted plane with each candidate match in the



(a) Translation (b) Shearing (c) Scaling

Figure 2.15: Illustration of translation, shearing and scaling transformation of the support window induced by planes in the disparity space. The black patch is the projection of the grey patch in the other view. Figure reproduced from [36].

disparity space [75], in contrast with methods that evaluate the complete disparity space image, either explicitly, or by searching over the range of disparities at each pixel. PatchMatch [3] is extended to overcome this problem by finding an approximate nearest neighbour according to a plane. The key ideas of the *PMS* framework are that neighbouring pixels have coherent matches, and large numbers of random samples will yield some good initial estimates of the plane parameters. The framework randomly assigns the plane parameters to each pixel of both images, and later uses two kinds of propagation scheme; *spatial* and *view* to propagate the estimated plane parameters within and across images, followed by an optimisation scheme called *plane refinement*. The spatial propagation propagates the plane parameters among spatial neighbours whereas view propagation propagates the plane parameters within and across views. Plane parameters are also locally optimised at individual pixels in the plane refinement step. The *PMS* framework produces two separate disparity maps for the stereo pair. Many local methods derive the sub-pixel information in the post-processing step by fitting a parabola in the cost volume whereas *PMS* computes the sub-pixel information directly from the support plane. As a result, it is more effective to reconstruct highly slanted and round surfaces (Fig. 2.16). *PMS* reformulates eq. 2.19 such that the optimisation is performed over the set of all possible 3D planes as follows,

$$d_{\bar{\mathbf{p}}} = \underset{\mathbf{f}\in\mathscr{F}}{\arg\min} \sum_{\bar{\mathbf{q}}\in\mathcal{W}(\bar{\mathbf{p}})} A(\bar{\mathbf{p}},\bar{\mathbf{q}}) \cdot \mathrm{cost}\left(\bar{\mathbf{q}}, \bar{\mathbf{q}} - (a\ \bar{\mathbf{q}}_x + b\ \bar{\mathbf{q}}_y + c)\right), \tag{2.24}$$

where, $\mathscr{F}$ is the set of all planes in the disparity space and a, b, and c are the three parameters of the plane $\mathbf{f} \in \mathscr{F}$, $\bar{\mathbf{q}}_x$ and $\bar{\mathbf{q}}_y$ are the x and y coordinates of $\bar{\mathbf{q}} \in \mathcal{W}(\bar{\mathbf{p}})$. Hence the approach of checking all possible labels is no longer valid here.

The basic assumption of *PMS* is that large regions of an image can be modelled by approx-

Figure 2.16: Diaparity map using PatchMatch Stereo. The disparity map is much smoother due to slanted support windows and continuous sub-pixel disparities [8].

imately the same plane. In the initialisation step of the algorithm, each pixel is assigned to a plane with a random parameter. Because of the random assignment of plane parameters most initialised planes will be wrong, but the hope is that at least one pixel of a region carries a plane that is close to the optimal one. This assumption is very likely to hold since we have many guesses. A single correct guess is sufficient for this algorithm to work as the correct plane is then propagated to neighbouring pixels. *PMS* is among the best performing local algorithms when sub-pixel precision is considered (Middlebury error threshold 0.5).

Besse *et al.* [5] unified the Particle Belief Propagation (PBP) and PatchMatch and formulated a global stereo algorithm PMBP. PMBP uses the same unary term as *PMS*. However, it also has a smoothness term that measures the deviation between two local planes. Heise *et al.* [36] presented an explicit variational smoothness model for *PMS* using quadratic relaxation [43] and the same smoothness term used in [5]. Li *et al.* [54] also proposed a PatchMatch based superpixel cut (PMSC) algorithm to assign the plane parameters at each pixel. PMSC uses a bilayer matching cost for 3D labels (plane parameters) by combining CNN-based and PatchMatch based similarity measurements on the superpixel proposals and later optimise directly the pixel grid MRF energy with 3D labels using α-expansion graph cut. State of the art stereo methods using planar support are summarised in Table 2.1.

Li and Zucker [53] presented a general surface model which fits curved surfaces at each pixel in the scene space. The scene space was used because the disparity space has numerical sensi-

Table 2.1: Comparison of stereo matching algorithms using surfaces in disparity space.

Type	Algorithm	Cost function		Adaptive support weight		Smoothness	Disparity model	No. of iterations
		Intensity	Gradient	Reference image weight	Search image weight			
Global	Gallup *et al.* [25]	✓				✓	planar	–
	PMBP [5]	✓	✓	✓		✓	planar	3
	PM-Huber [36]	✓	✓	✓		✓	planar	3
	PMSC [54]	✓	✓	✓		✓	planar	5
Local	PMS [9]	✓	✓	✓			planar	3
	IPMS (proposed) [1]	✓	✓	✓	✓		planar	2
	QPMS (proposed)	✓	✓	✓	✓		quadric	3

tivity problems with higher order derivatives. The planar model can be regarded as a particular case of this general model which works in the disparity space. Planar models deal with only first-order derivatives, so slanted planar surfaces are stable in the disparity space.

2.7 Limitations of the PMS framework

The basic *PMS* framework has six limitations. First, its initialisation process does not guarantee the association of a geometrically feasible plane at each pixel, as it randomly selects the plane parameters for each pixel in both reference and search images. Second, the plane refinement process uses a variant of the Luus-Jaakola optimisation [57] to minimise the cost function, which is inefficient to find a local minimum of the given cost function. Third, the framework assumes that the stereo images are perfectly rectified which is not the case for typical stereo pairs. Fourth, it generates false matches in low textured areas [32]. The model also fails to smoothly reconstruct curved surfaces in the disparity space as the framework is based on a planar model. Finally, the planar disparity model does not provide the curvature information of the associated surface, which is useful to understand the local surface structure.

2.8 Summary

In this chapter, we have reviewed the necessary concepts of stereo vision. Camera parameters help to extract metric information from 2D images and correspondences. As stereo matching produces a disparity map, we also discussed the geometric features of the disparity space. Although local algorithms have recently become very popular, they are still not capable of handling large untextured regions. They operate on windows that are displaced in the right image to find the matching point of lowest colour dissimilarity in the left image. The implicit assumption that every pixel within a patch has constant disparity breaks at scenarios where the pixel lies on the slant or curved surface and also at the disparity borders leading to edge fattening problem. Adaptive weight can solve the edge fattening problem but increases the runtime. Typically oc-

clusions are treated in a post-processing step by left-right consistency check. We also reviewed state-of-the-art dense local stereo matching algorithms to find out recent advancements and remaining challenges. We then focused on the *PMS* framework which uses slanted planes in the disparity space to model the scene surface. Finally, some limitations of the *PMS* framework were identified, which will be addressed in the following chapters.

Chapter 3

Constrained Optimisation for Plane-Based Stereo

3.1 Introduction

The PatchMatch Stereo (*PMS*) framework [9] can simultaneously estimate the depth and the surface orientation at every pixel in an image. Algorithms [36, 5, 24] use the *PMS* framework to reconstruct the visible surfaces in the disparity space (Sec. 2.3) defined by the pixel coordinates and the possible disparities. The reconstruction is based on associating a slanted plane with each candidate match, in contrast with methods that evaluate the complete disparity space image, either explicitly, or by searching over the range of disparities at each pixel (Sec. 2.6).

In this work, we present a constrained initialisation scheme that works with any algorithm that can be cast in the *PMS* framework. We introduce two new constraints to restrict the initialisation scheme by generating only geometrically feasible planes such that the disparity of every pixel inside a patch must lie between the maximum and minimum allowed disparity. The proposed constraints are also imposed during the optimisation process. For the plane refinement problem, the usual *PMS* cost function cannot be minimised by standard gradient descent methods due to the presence of discontinuous thresholds in the pixel dissimilarity function [9]. We avoid this problem by using the "Bound Optimisation BY Quadratic Approximation" (BOBYQA) algorithm [71]. BOBYQA is a gradient-free non-linear trust region based constrained optimiser [93] which uses our geometric constraints to ensure that the plane parameters remain feasible. Moreover, we relax the view propagation (Sec. 2.6) by assigning the plane parameters to the immediate 4-neighbours (left, right, upper and lower) of the candidate pixel to tackle the imperfectly rectified

image pairs. More generally, this strategy is useful because most plane parameters are incorrect in the earlier propagations. To make the cost function more robust to false matches in occluded regions, we change the support weight function so that it considers both search and reference image weights. We finally update the pixel dissimilarity function from the truncated sum of absolute colour difference (TSAD) to the truncated sum of square difference (TSSD), which is more compatible with BOBYQA.

The rest of the chapter is organised as follows. The general framework is proposed in Section 3.2. We introduce our constrained initialisation in Section 3.3 and the constrained optimisation in Section 3.4. Section 3.5 provides information on the experimental set-up, parameters used and the results. We discuss some disparity results of the sand images in Section 3.6. Finally, conclusions are drawn in Section 3.7.

3.2 General framework

Here we formulate a general setting for *PMS*, as well as for our contributions in Section 3.3 and 3.4. We begin by defining the relationships between planes in disparity space. We also present modifications to the cost function and the propagation procedure in Section 3.2.2 and 3.2.3.

3.2.1 Point-normal plane representation in disparity space

To find corresponding pixels, *PMS* starts with a rectified colour stereo pair, comprising I and I', where I is the reference image and I' is the search image (which will be exchanged during the course of the algorithm). Let $\mathcal{S}$ be the visible surface. Let $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$ be corresponding pixels projected from a scene point $\mathbf{P} \in \mathcal{S}$ to the image plane I and I', respectively (Fig. 3.1). Let $\tilde{\mathbf{n}}$ be the outward unit surface normal of the plane $\tilde{\mathbf{f}}$ in the scene space containing $\mathbf{P}$. We can find the depth of $\mathbf{P}$ provided the plane $\tilde{\mathbf{f}}$ is known, *i.e.,* both $\mathbf{P}$ and $\tilde{\mathbf{n}}$ are known. As disparity is inversely proportional to depth, disparity can also be found by the analogous plane representation of $\tilde{\mathbf{f}}$ in the disparity space. Note that the real world planes are related to the disparity space planes by a projective transformation [53].

Let $\bar{\mathbf{p}} = (x,y)^{\mathsf{T}} \in I$ and $\bar{\mathbf{p}}' = (x',y)^{\mathsf{T}} \in I'$. As I and I' are rectified and $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$ are matching pixels, $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$ have the same y coordinate and the same disparity magnitude d. Then $\mathbf{p} = (x,y,d)^{\mathsf{T}} \in \mathcal{D}$ and $\mathbf{p}' = (x',y,d)^{\mathsf{T}} \in \mathcal{D}'$ are two different projective transformations of $\mathbf{P}$ [36], where $\mathcal{D}$ and $\mathcal{D}'$ denote the disparity spaces (Sec. 2.3) generated by the refer-

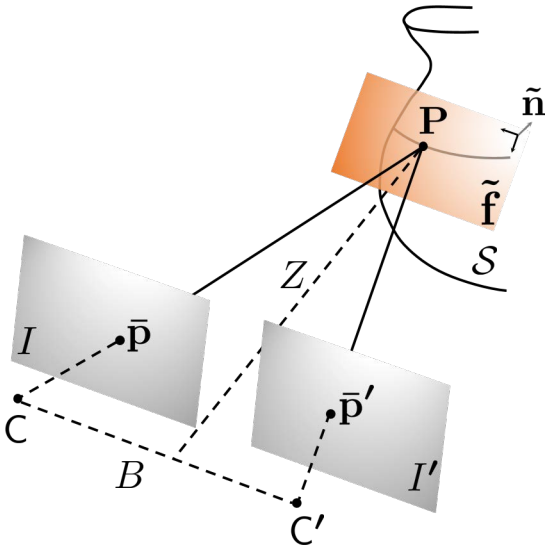


Figure 3.1: Pinhole camera model. Image points $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$ are the projections of a scene point $\mathbf{P} \in \mathcal{S}$ on the reference image I and the search image I' from two different views obtained by the left camera C and the right camera C' respectively, where $\mathcal{S}$ is the visible surface. The baseline distance between C and C' is B. The scene point $\mathbf{P}$ is at a distance Z from B. The plane $\tilde{\mathbf{f}}$ at $\mathbf{P}$ has a unit surface normal $\tilde{\mathbf{n}}$ in the outwards direction. Our objective is to find Z and $\tilde{\mathbf{n}}$ from the plane $\tilde{\mathbf{f}}$.

ence and search image pixels, respectively. We have used different notations for the disparity spaces to highlight the projective transformations of $\mathbf{P}$ generating from two different views. As $\bar{\mathbf{p}} \in I$, $x' = x - d$, otherwise $x' = x + d$. The relation between $\mathbf{p}$ and $\mathbf{p}'$ in the disparity space is:

$$\mathbf{p}' = \mathbf{M}\mathbf{p}, \quad \text{where} \quad \mathbf{M} = \begin{pmatrix} 1 & 0 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} . \tag{3.1}$$

Disparity spaces $\mathcal{D}$ and $\mathcal{D}'$ will differ due to the visibility effects. *i.e.,* eq. 3.1 is not true for all points in practice. Using the projective transformations of $\mathbf{P}$, the scene surface normals can be found via the surface normals in the disparity space.

A plane $\mathbf{f}$ in the disparity space is defined by a point $\mathbf{p}$ and a surface normal $\mathbf{n}$ at $\mathbf{p}$ (Fig. 3.2). When $\mathbf{p}$ and $\mathbf{n}$ are known, $\mathbf{f}$ can be represented by three plane parameters $\mathbf{f} := (a,b,c)^{\mathsf{T}}$. The disparity d of $\bar{\mathbf{p}}$ with respect to the plane $\mathbf{f}$ is given by:

$$d = ax + by + c \; . \tag{3.2}$$

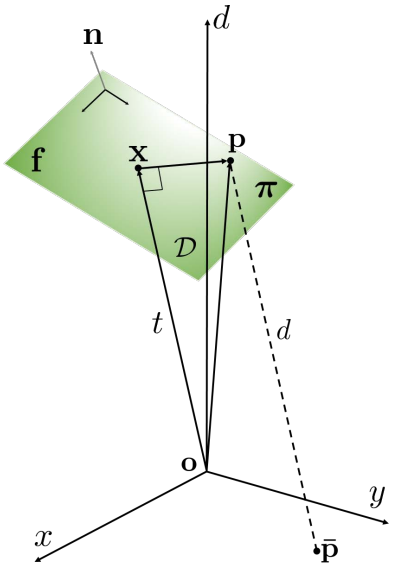


Figure 3.2: Representation of a plane in the disparity space. Point **p** is the corresponding point in the disparity space $\mathcal{D}$ of an image point $\bar{\mathbf{p}}$ with disparity d. The plane **f** at **p** has a unit surface normal **n** in the outward direction. The plane $\boldsymbol{\pi} = (\mathbf{n}, -t)$ is the point-normal representation on **f**, where t is the projection of **op** on **n**. The value of t can be positive or negative depending on the orientation of **f**.

Therefore, disparity is *over parametrized*[1] by the plane parameters of **f**. If $(a,b,c)^\mathsf{T}$ is known, we can find the disparity δ of any pixel $(\xi,\eta)^\mathsf{T}$ with respect to the plane **f**. The disparity δ is given by

$$\delta = a\xi + b\eta + c\ .$$

To find boundary constraints on the disparity and the surface normal, our proposed method, *Initialised PatchMatch Stereo* (IPMS), works with the point-normal parametrisation of planes (Fig. 3.2); unlike *PMS* which directly uses the plane parameters.

Let $\mathbf{n} = (u,v,w)^\mathsf{T}$ be the unit surface normal at **p** (Fig. 3.2). In homogeneous coordinates, the plane passing though **p** with normal **n** can be defined as

$$\boldsymbol{\pi} := (\mathbf{n}, -t)\cdot(\mathbf{x}, 1) = 0, \qquad \text{where } t = \mathbf{n}\cdot\mathbf{p}\ .$$

Geometrically, t defines the perpendicular distance of the plane $\boldsymbol{\pi}$ from the origin. The value of t can be positive or negative depending on the orientation of $\boldsymbol{\pi}$. We use the notation **f** and $\boldsymbol{\pi}$ when the plane is parametrised by plane parameters and point-normal, respectively. The relation

[1] Disparity is single valued function, which is estimated here by three parameters.

between the plane parameters $\mathbf{f} := (a,b,c)^\mathsf{T}$ and the unit normal $\mathbf{n} = (u,v,w)^\mathsf{T}$ at $\mathbf{p}$ is:

$$\begin{pmatrix} a \\ b \\ c \end{pmatrix} = \frac{1}{w} \begin{pmatrix} -u \\ -v \\ t \end{pmatrix} , \quad \mathbf{n} = \frac{1}{\sqrt{1+a^2+b^2}} \begin{pmatrix} -a \\ -b \\ 1 \end{pmatrix} .$$

3.2.2 Cost function

The performance of a stereo matching algorithm depends on matching cost for measuring the similarity of an image region across views [39]. The intensities of a matching region can differ because of certain radiometric changes and/or noise. Therefore, a matching cost function has to be robust to such variations. We use a pixel-based matching cost function (Sec. 2.5.4) along with adaptive support weight (Sec. 2.5.3).

Let $\mathbf{p}$ be a point in the disparity space which corresponds to an image point $\bar{\mathbf{p}}$. Let Π be the set of all candidate planes passing through $\mathbf{p}$, we want to find a plane $\boldsymbol{\pi}$ that minimises the aggregated matching cost:

$$\boldsymbol{\pi} = \underset{\boldsymbol{\omega} \in \Pi}{\arg\min}\ \mathrm{cost}\big(\bar{\mathbf{p}}, \boldsymbol{\omega}\big) .$$

The aggregated cost (Sec. 2.5.4) of $\bar{\mathbf{p}}$ according to $\boldsymbol{\pi}$ is computed as

$$\mathrm{cost}(\bar{\mathbf{p}}, \mathscr{S}) = \frac{\sum\limits_{\bar{\mathbf{q}} \in \mathcal{W}(\bar{\mathbf{p}})} \sum\limits_{\bar{\mathbf{q}}' \in \mathcal{W}(\bar{\mathbf{p}}')} A\big(\bar{\mathbf{p}}, \bar{\mathbf{q}}\big) \cdot A\big(\bar{\mathbf{p}}', \bar{\mathbf{q}}'\big) \cdot E\big(\bar{\mathbf{q}}, \bar{\mathbf{q}}'\big)}{\sum\limits_{\bar{\mathbf{q}} \in \mathcal{W}(\bar{\mathbf{p}})} \sum\limits_{\bar{\mathbf{q}}' \in \mathcal{W}(\bar{\mathbf{p}}')} A\big(\bar{\mathbf{p}}, \bar{\mathbf{q}}\big) \cdot A\big(\bar{\mathbf{p}}', \bar{\mathbf{q}}'\big)} , \tag{3.3}$$

where $\mathcal{W}(\bar{\mathbf{p}})$ denotes a square patch centred at $\bar{\mathbf{p}}$, $\bar{\mathbf{p}}'$ denotes the matching pixel of $\bar{\mathbf{p}}$ with respect to $\boldsymbol{\pi}$ and $\mathcal{W}(\bar{\mathbf{p}}')$ denotes the projection of $\mathcal{W}(\bar{\mathbf{p}})$ with respect to $\boldsymbol{\pi}$. Point $\bar{\mathbf{q}}'$ is the matching point of $\bar{\mathbf{q}}$ in the other view with respect to the plane $\boldsymbol{\pi}$. Let $\bar{\mathbf{q}} = (x,y)^\mathsf{T} \in I$. The disparity d of $\bar{\mathbf{q}}$ is given by eq. 3.2. Then $\bar{\mathbf{q}}' = (x-d,y)^\mathsf{T}$. If $\bar{\mathbf{q}} \in I'$, then $\bar{\mathbf{q}}' = (x+d,y)^\mathsf{T}$.

The weight function $A(\bar{\mathbf{p}}, \bar{\mathbf{q}})$ in eq. 3.3 is used to overcome the edge fattening problem. It implements the adaptive support weight (Sec. 2.5.3) [91] by computing the affinity for $\bar{\mathbf{p}}$ and $\bar{\mathbf{q}}$ (lying on the same plane), from the associated colour and spatial distances:

$$A(\bar{\mathbf{p}}, \bar{\mathbf{q}}) = \exp\left(-\frac{\Delta c_{\bar{\mathbf{p}}\bar{\mathbf{q}}}}{\gamma_\mathrm{c}}\right) \exp\left(-\frac{\Delta g_{\bar{\mathbf{p}}\bar{\mathbf{q}}}}{\gamma_\mathrm{g}}\right) , \tag{3.4}$$

where $\Delta c_{\bar{\mathbf{p}}\bar{\mathbf{q}}}$ measures the colour similarity and $\Delta g_{\bar{\mathbf{p}}\bar{\mathbf{q}}}$ denotes the geometric proximity between $\bar{\mathbf{q}}$ and the center pixel $\bar{\mathbf{p}}$ of $\mathcal{W}(\bar{\mathbf{p}})$. The user-defined parameters γ_c and γ_g are scale parameters

and their values are discussed in Section 3.5. The colour similarity term $\Delta c_{\bar{\mathbf{p}}\bar{\mathbf{q}}}$ between $\bar{\mathbf{p}}$ and $\bar{\mathbf{q}}$ is defined as:

$$\Delta c_{\bar{\mathbf{p}}\bar{\mathbf{q}}} = ||I_{\bar{\mathbf{p}}} - I_{\bar{\mathbf{q}}}|| \,,$$

where $||I_{\bar{\mathbf{p}}} - I_{\bar{\mathbf{q}}}||$ computes the Euclidean distance of the colours of $\bar{\mathbf{p}}$ and $\bar{\mathbf{q}}$ in the CIELAB space, which approximates the perception of colour [91]. The geometric proximity term $\Delta g_{\bar{\mathbf{p}}\bar{\mathbf{q}}}$ is defined as the Euclidean distance between the coordinates of $\bar{\mathbf{p}}$ and $\bar{\mathbf{q}}$:

$$\Delta g_{\bar{\mathbf{p}}\bar{\mathbf{q}}} = ||\bar{\mathbf{p}} - \bar{\mathbf{q}}|| \,.$$

The performance of a local stereo method depends to a large extent on the support weights that are used in the aggregation step [41]. It has also been shown in [41] that the adaptive support weight proposed in [91] works best in all considered scenarios including occluded regions [9]. The support weight $A(\bar{\mathbf{p}}', \bar{\mathbf{q}}')$ in the other view is also computed similarly.
The error function $E(\bar{\mathbf{q}}, \bar{\mathbf{q}}')$ is defined as the pixel dissimilarity between $\bar{\mathbf{q}}$ and $\bar{\mathbf{q}}'$:

$$E\big(\bar{\mathbf{q}}, \bar{\mathbf{q}}'\big) = (1-\alpha) \cdot \min\big(||I_{\bar{\mathbf{q}}}, I_{\bar{\mathbf{q}}'}||^2, \tau_{\text{col}}\big) + \alpha \cdot \min\big(|\nabla I_{\bar{\mathbf{q}}} - \nabla I_{\bar{\mathbf{q}}'}|, \tau_{\text{grad}}\big), \tag{3.5}$$

where $||I_{\bar{\mathbf{q}}}, I_{\bar{\mathbf{q}}'}||^2$ denotes the sum of squared distance (SSD) of $\bar{\mathbf{q}}$ and $\bar{\mathbf{q}}'$ in RGB space and $|\nabla I_{\bar{\mathbf{q}}} - \nabla I_{\bar{\mathbf{q}}'}|$ denotes the absolute difference of grey-value gradients computed at $\bar{\mathbf{q}}$ and $\bar{\mathbf{q}}'$. Since the x-coordinate of $\bar{\mathbf{q}}'$ lies in the continuous domain, we derive its colour and gradient values by linear interpolation. The user-defined parameter α balances the influence of the colour and gradient terms. Other user-defined parameters τ_{col} and τ_{grad} are thresholds that truncate costs for robustness in occluded regions. Their values are discussed in Section 3.5.

We updated the colour similarity term of *PMS* to Truncated Sum of Squared intensity Differences (TSSD). Sum of Absolute intensity Differences (SAD) is more robust to noise and outliers than SSD, in the sense that the influence function[2] is bounded [7]. However, both Truncated Sum of Absolute intensity Differences (TSAD) and TSSD are robust, as their influence functions are zero beyond the *outlier* threshold. Because of the sharp nature of SAD around the origin, SAD tends to lock on to a small number of very good matches in [9], which are then propagated. However, the smooth nature of SSD helps BOBYQA to improve the good matches more effectively than the original Luus-Jaakola scheme, as discussed in Section 3.4. Additionally, when the noise

[2] An influence function shows the effect of a change in one observation on an estimator.

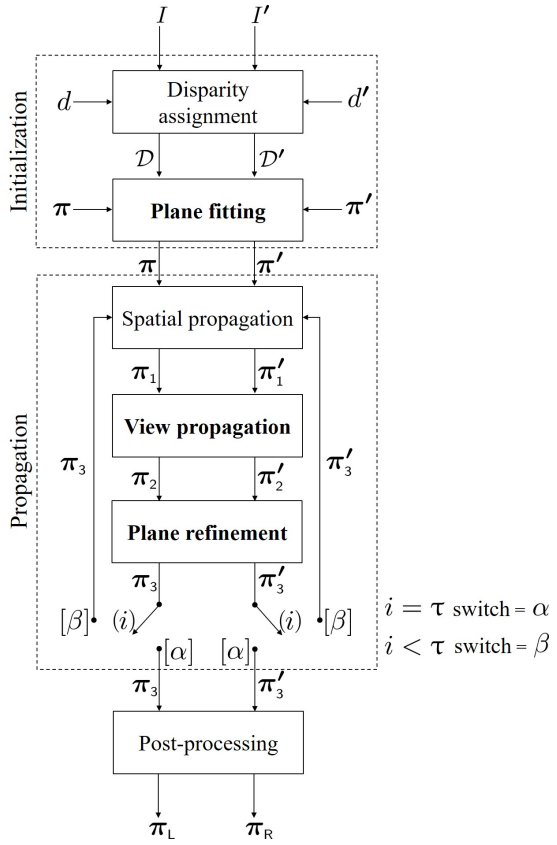


Figure 3.3: Initialised PatchMatch Stereo (*IPMS*) flowchart. Input I and I' are the rectified reference and search images, respectively. Left and right disparity spaces $\mathcal{D}$ and $\mathcal{D}'$ are generated by selecting disparities d and d' from the disparity constraints for every pixel in I and I', respectively. The surface normals $\mathbf{n}$ and $\mathbf{n}'$ are selected from the normal constraints, which are used along with the point in the disparity space to generate the planes $\boldsymbol{\pi}$ and $\boldsymbol{\pi}'$ for every pixel in I and I', respectively. The total number of iterations is a user defined parameter τ, while the iteration number is denoted by i. After each operation, the updated planes are represented by $\boldsymbol{\pi}_j$ and $\boldsymbol{\pi}'_j$. *IPMS* converges in two iterations in contrast to *PMS*, which takes three. After the post processing the final plane parameters for the left and right image are denoted by $\boldsymbol{\pi}_\mathrm{L}$ and $\boldsymbol{\pi}_\mathrm{R}$, respectively.

is Gaussian [6], SSD is optimal for a maximum-likelihood estimate, in the neighbourhood of a true match [48]. Experimental results in Fig. 8 also support this claim.

3.2.3 Matching strategy

To assign the disparity at each pixel of I and I', we need to find a plane that minimises $\mathrm{cost}(\bar{\mathbf{p}}, \mathscr{S})$. *IPMS* follows the same framework of *PMS* with modifications highlighted in bold in Fig. 3.3. We now discuss the three stages of the algorithm.

Initialisation

We use a uniform distribution U that randomly assigns the disparities of each pixel in I and I' between the minimum allowed disparity, $d_{\min}$, and the maximum allowed disparity, $d_{\max}$:

$$d \sim \mathsf{U}(d_{\min}, d_{\max}) \; . \tag{3.6}$$

The initialisation process hypothesis the left $(\mathcal{D})$ and right $(\mathcal{D}')$ disparity spaces. Then, we assign a unit normal at each pixel of I and I' to find the plane parameters as discussed in Section 3.3. We use the same initialisation process at each pixel of both images.

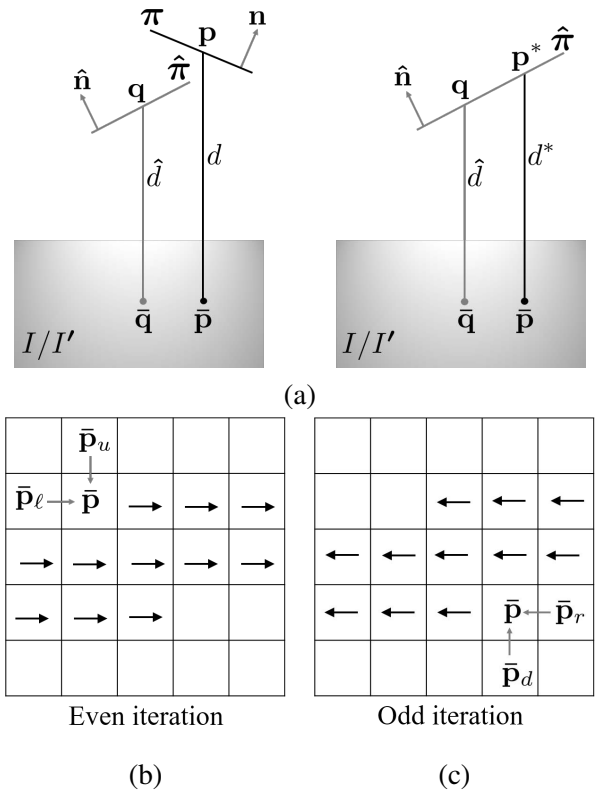


Figure 3.4: Spatial propagation. (a) Image point $\bar{\mathbf{q}}$ is a spatial neighbour of $\bar{\mathbf{p}}$. Points $\mathbf{p}$ and $\mathbf{q}$ are corresponding points of $\bar{\mathbf{p}}$ and $\bar{\mathbf{q}}$ in the disparity space lying on the plane $\boldsymbol{\pi}$ and $\tilde{\boldsymbol{\pi}}$ with unit normal $\mathbf{n}$ and $\tilde{\mathbf{n}}$, and disparity d and $\hat{d}$, respectively. Image point $\bar{\mathbf{p}}$ corresponds to a new point $\mathbf{p}^*$ in the disparity space with respect to $\tilde{\boldsymbol{\pi}}$. In spatial propagation we aggregate the cost of the patch $\mathcal{W}(\bar{\mathbf{p}})$ centred at $\bar{\mathbf{p}}$ with respect to $\boldsymbol{\pi}$ and $\tilde{\boldsymbol{\pi}}$ if the new disparity d^* of $\bar{\mathbf{p}}$ with respect to $\tilde{\boldsymbol{\pi}}$ is between $d_{\max}$ and $d_{\min}$. We update the plane of $\mathbf{p}$ to $\tilde{\boldsymbol{\pi}}$ if the aggregated cost gets reduced by $\tilde{\boldsymbol{\pi}}$. (b) and (c) show the direction of spatial propagation for odd and even iterations. The four immediate neighbours of $\bar{\mathbf{p}}$ are denoted by $\bar{\mathbf{p}}_\ell$, $\bar{\mathbf{p}}_r$, $\bar{\mathbf{p}}_u$, $\bar{\mathbf{p}}_d$ (left, right, upper and lower). In even iterations we consider the left and upper neighbours as spatial neighbours, whereas in odd iterations the right and lower neighbours are verified.

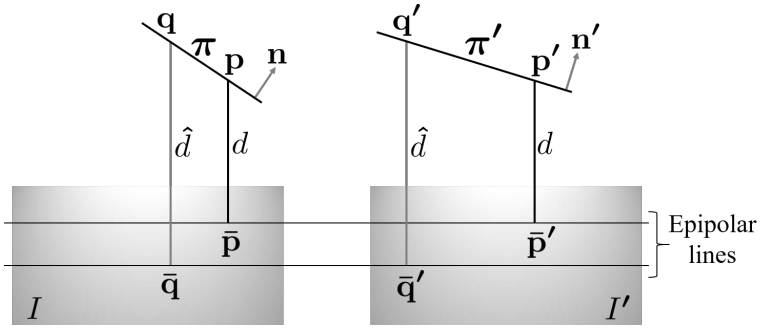


Figure 3.5: Change of plane normals in view propagation. Image points $\bar{\mathbf{p}} \in I$ and $\bar{\mathbf{p}}' \in I'$ are two matching points. Points $\mathbf{p}$ and $\mathbf{p}'$ are corresponding points of $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$ in the disparity space lying on the plane $\boldsymbol{\pi}$ and $\boldsymbol{\pi}'$ with disparity d, and surface normal $\mathbf{n}$ and $\mathbf{n}'$, respectively. Let $\bar{\mathbf{q}}$ be a neighbouring point of $\bar{\mathbf{p}}$ and $\bar{\mathbf{q}}'$ be the matching pixel of $\bar{\mathbf{q}}$ in the other view. By transforming the plane normals in the other view, we can show that the disparity $\hat{d}$ of $\bar{\mathbf{q}}$ and $\bar{\mathbf{q}}'$ are equal with respect to $\boldsymbol{\pi}$ and $\boldsymbol{\pi}'$, respectively. In the figure, $\mathbf{q}$ and $\mathbf{q}'$ are corresponding points of $\bar{\mathbf{q}}$ of $\bar{\mathbf{q}}'$ in the disparity space with respect to $\boldsymbol{\pi}$ and $\boldsymbol{\pi}'$.

Propagation

We use the *PMS* iterative scheme to propagate the plane parameters in two different directions considering both views. In every iteration, each pixel runs through three independent stages: spatial propagation, view propagation, and plane refinement. We use the same *spatial propagation* approach as mentioned in *PMS* (Fig. 3.4). As for *view propagation* , we exploit the strong coherency that exists between left and right disparity maps so that a pixel and its matching pixel in the other view have the same disparity. However, the surface normals in the disparity space change across views due to different view points. Let $\mathbf{n}'$ be the unit normal of the plane $\boldsymbol{\pi}'$ at $\mathbf{p}' \in \mathcal{D}'$. The relation between $\mathbf{n}$ and $\mathbf{n}'$ (eq. A.1) is:

$$\mathbf{n}' = \frac{\mathbf{M}^{-\mathsf{T}}\mathbf{n}}{||\mathbf{M}^{-\mathsf{T}}\mathbf{n}||} , \tag{3.7}$$

where $||\cdot||$ denotes the L_2 norm and $\mathbf{M}$ is defined in eq. 3.1.

In view propagation, we check all pixels of the second view that are matched to our current pixel according to their plane, and assign the plane parameters to the current pixel if the transformed plane in the first view reduces the cost.

Let $\{\bar{\mathbf{r}}', \bar{\mathbf{s}}'\}$ be two possible matching points of $\bar{\mathbf{p}}$ in the other view. Let $\mathbf{r}$ and $\mathbf{s}$ be corresponding points of $\bar{\mathbf{r}}'$ and $\bar{\mathbf{s}}'$ in the disparity space lying on the planes $\boldsymbol{\pi}_\mathbf{r}$ and $\boldsymbol{\pi}_\mathbf{s}$ with disparity $d_\mathbf{r}$ and $d_\mathbf{s}$, and surface normals $\mathbf{n}_\mathbf{r}$ and $\mathbf{n}_\mathbf{s}$, respectively. Theoretically, matching points should have the same disparity magnitude with different unit normals, depending on which disparity space the normals are residing. We transfer the normals $\mathbf{n}_\mathbf{r}$ and $\mathbf{n}_\mathbf{s}$ to the other view by eq. 3.7.

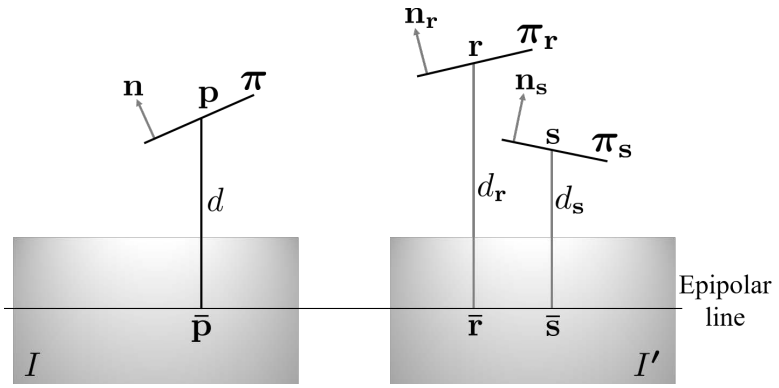


Figure 3.6: View propagation. Let $\{\bar{\mathbf{r}}', \bar{\mathbf{s}}'\} \in I'$ are two possible matching points of $\bar{\mathbf{p}} \in I$. Points $\mathbf{p}$, $\mathbf{r}$ and $\mathbf{s}$ are corresponding points of $\bar{\mathbf{p}}$, $\bar{\mathbf{r}}'$, $\bar{\mathbf{s}}'$ in the disparity space lying on the plane $\boldsymbol{\pi}$, $\boldsymbol{\pi}_\mathbf{r}$ and $\boldsymbol{\pi}_\mathbf{s}$ with disparity d, $d_\mathbf{r}$ and $d_\mathbf{s}$, and surface normal $\mathbf{n}$, $\mathbf{n}_\mathbf{r}$ and $\mathbf{n}_\mathbf{s}$, respectively. We transfer the normal and disparity of $\bar{\mathbf{r}}'$ and $\bar{\mathbf{s}}'$ to $\bar{\mathbf{p}}$ and find the new plane parameters. If the new plane parameters minimise the aggregated cost of the patch $\mathcal{W}(\bar{\mathbf{p}})$ centred at $\bar{\mathbf{p}}$, we update the plane parameters of $\bar{\mathbf{p}}$ with the new one.

Let $\mathbf{n}'_\mathbf{r}$ and $\mathbf{n}'_\mathbf{s}$ be the transformed unit normals of $\mathbf{n}_\mathbf{r}$ and $\mathbf{n}_\mathbf{s}$, and $\boldsymbol{\pi}'_\mathbf{r}$ and $\boldsymbol{\pi}'_\mathbf{s}$ be the their transformed planes respectively (Fig. 3.5). If the new plane parameters minimise the aggregated cost of $\mathcal{W}(\bar{\mathbf{p}})$ centred at $\bar{\mathbf{p}}$, then we update the plane parameters of $\bar{\mathbf{p}}$ with the new values (Fig. 3.6). Due to rectification error, corresponding points may be vertically displaced by one or two pixels. In our modified view propagation, we address this problem by assigning plane parameters to the immediate neighbours of the candidate pixels. We assign the disparity and the transformed plane normal of $\bar{\mathbf{r}}'$ and $\bar{\mathbf{s}}'$ to the four immediate neighbours (left, right, upper, lower) of $\bar{\mathbf{p}}$ and check whether the new plane parameters reduce the cost. If so, we also update the planes of the neighbours accordingly.

Finally, in *plane refinement*, we use a trust region[3] based, gradient-free non-linear optimiser BOBYQA [71] to further refine the plane parameters, as discussed in Section 3.4.

Post-processing

We follow the same post-processing scheme as mentioned in the *PMS* [9]. Additionally, we apply a weighted median filter to resulting disparity map to eliminate any isolated mismatches [72].

3.3 Constrained plane initialisation

The disparity range assumption states that the disparity of any pixel inside an image should lie between the maximum and minimum disparity. The disparity of a pixel within a patch is

[3]Trust region is a subset of the region defined by the objective function that is approximated using a model function.

computed with respect to the plane associated with the centre pixel of the patch. Hence it is important to assign the plane normals in a suitable way such that they comply with the disparity range assumption. Heise *et al.* [36] addressed the initialisation problem by constraining the first two components of the plane normal equally distributed over a unit circle in the disparity space. Galliani *et al.* [24] generated the unit normals evenly over the sphere in the disparity space. Both of these strategies can generate infeasible planes.

The core idea behind our constrained initialisation is to only assign geometrically feasible planes to every pixel of both images during initialisation. Furthermore, the plane normal bounds are maintained later during the plane refinement scheme. There can be no disadvantage with respect to *PMS*, because infeasible matches should never be propagated. The initialisation scheme is an integral part of the framework as the rate of convergence depends on the assignment of correct plane parameters during initialisation. The more viable the plane parameters are estimated, the faster the algorithm converges. The positive effects of the constrained initialisation are demonstrated during the experiments (Fig. 3.9).

3.3.1 Visibility constraint in the disparity space

When searching for the optimal plane containing a given scene point, we only consider those planes that are visible from the camera at the scene point [83]. We apply this constraint in the disparity space. The visibility constraint in the disparity space assumes that the plane in the disparity space is visible from the line of sight vector joining the image point and the corresponding point in the disparity space (Fig. 3.7). The constraint only considers whether the centre point of the patch and its corresponding point in the other view is visible from individual cameras. Due to various patch orientations, there may be cases where the patch is partially visible from the other camera. This issue is taken care by the truncation parameters in the cost function (eq. 3.3). This visibility constraint is only a necessary constraint that differs from the general visibility constraint that checks whether the surface points are truly visible in two views without occlusion.

If $\bar{\mathbf{p}} = (\xi, \eta) \in I$ with disparity δ, then the matching pixel in the other view is $\bar{\mathbf{p}}' = (\xi - \delta, \eta)^\mathsf{T} \in I'$ (Fig. 3.7). The corresponding points of $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$ in the disparity space are $\mathbf{p} = (\xi, \eta, \delta)^\mathsf{T} \in \mathcal{D}$ and $\mathbf{p}' = (\xi - \delta, \eta, \delta)^\mathsf{T} \in \mathcal{D}'$, respectively. The unit surface normal in I' is given by eq. 3.7.

In the left camera coordinate system, the line of sight vector of $\mathbf{n}$ at $\mathbf{p}$ with respect to $\bar{\mathbf{p}}$ is $\boldsymbol{\ell} = (0, 0, \delta)^\mathsf{T}$. Similarly, in the right camera coordinate system, the line of sight vector of $\mathbf{n}'$ at

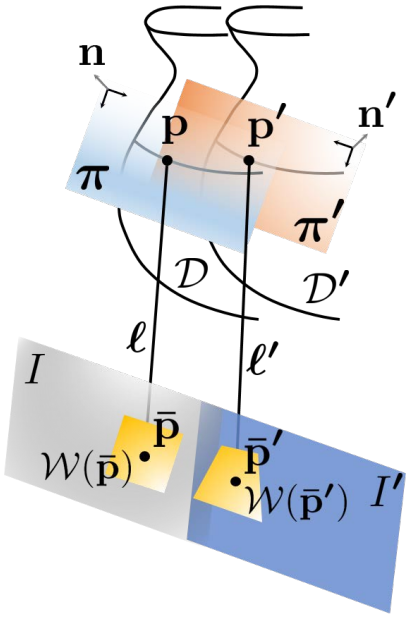


Figure 3.7: Surface normal constraints. Image points $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$ are matching pixels with disparity d lying on the reference image I and the search image I', respectively. The matching pixels $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$ correspond to $\mathbf{p} \in \mathcal{D}$ and $\mathbf{p}' \in \mathcal{D}'$, where $\mathcal{D}$ and $\mathcal{D}'$ represent the disparity spaces generated by I and I', respectively. Points $\mathbf{p}$ and $\mathbf{p}'$ lie on the plane $\boldsymbol{\pi}$ and $\boldsymbol{\pi}'$ with unit surface normal $\mathbf{n}$ and $\mathbf{n}'$ in the outward direction, respectively. A rectangular patch centred at $\bar{\mathbf{p}}$ is denoted by $\mathcal{W}(\bar{\mathbf{p}})$. The patch $\mathcal{W}(\bar{\mathbf{p}}')$ is the projection of $\mathcal{W}(\bar{\mathbf{p}})$ with respect to $\boldsymbol{\pi}$. The vector joining $\mathbf{p}$ and $\bar{\mathbf{p}}$ is defined as the line of sight vector $\boldsymbol{\ell}$ with respect to the left camera coordinate system. Similarly, $\boldsymbol{\ell}'$ is the line of sight vector joining $\mathbf{p}'$ and $\bar{\mathbf{p}}'$ with respect to the right camera coordinate system. The visibility constraint in the disparity space assumes $\boldsymbol{\pi}$ is visible from $\boldsymbol{\ell}$ and $\boldsymbol{\pi}'$ is visible from $\boldsymbol{\ell}'$. The disparity bound constraint on support window assumes the disparity of all the pixels inside $\mathcal{W}(\bar{\mathbf{p}})$ with respect to $\boldsymbol{\pi}$ is between $d_{\min}$ and $d_{\max}$.

$\mathbf{p}'$ with respect to $\bar{\mathbf{p}}'$ is $\boldsymbol{\ell}' = (0,0,\delta)^{\mathsf{T}}$. Planes $\boldsymbol{\pi}$ and $\boldsymbol{\pi}'$ are visible from $\boldsymbol{\ell}$ and $\boldsymbol{\ell}'$, respectively, if

$$\mathbf{n} \cdot \boldsymbol{\ell} > 0 \quad \text{and} \quad \mathbf{n}' \cdot \boldsymbol{\ell}' > 0 \,. \tag{3.8}$$

Substituting $\mathbf{n}$ and $\mathbf{n}'$ by their corresponding normal components u, v and w in inequality 3.8 we get

$$w > 0 \quad \text{and} \quad u > -w \,. \tag{3.9}$$

If $\bar{\mathbf{p}} \in I'$, then its matching pixel $\bar{\mathbf{p}}' = (\xi + \delta, \eta)^{\mathsf{T}} \in I$. The corresponding points of $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$ in the disparity space are $\mathbf{p} = (\xi, \eta, \delta)^{\mathsf{T}} \in \mathcal{D}'$ and $\mathbf{p}' = (\xi + \delta, \eta, \delta)^{\mathsf{T}} \in \mathcal{D}$, respectively. In both camera coordinate systems the line of sight direction remains unchanged except the plane normal

that can be retrieved using eq. 3.7. Again from inequality 3.8 we get:

$$w > 0 \quad \text{and} \quad u < w \cdot \tag{3.10}$$

Both inequality 3.9 and 3.10 imply that w is positive. We also get a bound (upper or lower) on u depending on which image $\bar{\mathbf{p}}$ is taken from. The visibility constraint does not give a complete bound (both upper and lower) for u. Moreover, the constraint does not provide any bounds for v. Therefore, we introduce our second constraint to obtain complete bounds on u and v.

3.3.2 Disparity bound constraint on support window

It is important to constrain the support window by disparity bounds as the *PMS* algorithm uses a large patch to measure the matching score. From the cost function (eq. 3.3), we know that the algorithm projects a support window in the other view (eq. 3.2) and compares the weighted pixel difference between the two support windows. To find the complete bound on the plane normals, our disparity bound constraint on the support window assumes that the disparity of all the pixels inside a support window with respect to the plane associated with the centre pixel of the patch must lie between the known maximum and the minimum allowed disparity, *i.e.,* the disparity of every pixel $\bar{\mathbf{q}} = (x, y) \in \mathcal{W}(\bar{\mathbf{p}})$ with respect to $\boldsymbol{\pi}$ should lie between $d_{\max}$ and $d_{\min}$ (Fig. 3.7). We should only consider planes that satisfy the following disparity bound condition:

$$d_{\min} \leq |\boldsymbol{\pi} \cdot \bar{\mathbf{q}}| \leq d_{\max} \quad \forall\, \bar{\mathbf{q}} \in \mathcal{W}(\bar{\mathbf{p}}),$$

where $|\boldsymbol{\pi} \cdot \bar{\mathbf{q}}|$ computes the perpendicular distance of $\bar{\mathbf{q}}$ to $\boldsymbol{\pi}$. This distance gives the disparity of $\bar{\mathbf{q}}$ with respect to $\boldsymbol{\pi}$ at $\mathbf{p}$. From eq. 3.2, the disparity d of $\bar{\mathbf{q}}$ is given by

$$d = \frac{u}{w}(\xi - x) + \frac{v}{w}(\eta - y) + \delta \ .$$

The disparity of $\bar{\mathbf{q}}$ should also satisfy the disparity bound condition:

$$d_{\min} \leq \frac{u}{w}(\xi - x) + \frac{v}{w}(\eta - y) + \delta \leq d_{\max} \ . \tag{3.11}$$

From the visibility constraint, we know that the value of w is always positive. Then inequality 3.11 can be simplified as:

$$-w(\delta - d_{\min}) \leq u(\xi - x) + v(\eta - y) \leq w(d_{\max} - \delta) \ . \tag{3.12}$$

Both $(\delta - d_{\min})$ and $(d_{\max} - \delta)$ are positive in the above expression. If $d^* = \min(\delta - d_{\min}, d_{\max} - \delta)$, we tighten the bounds of inequality 3.12 as:

$$-wd^* \leq u(\xi - x) + v(\eta - y) \leq wd^* \ . \tag{3.13}$$

Any solutions of inequality 3.13 will also satisfy inequality 3.12. The values of $(\xi - x)$ and $(\eta - y)$ depend on the patch size. For a patch of size $2r+1$, both their values ranges from $-r$ to r. A solution of inequality 3.13 for $\pm r$ is also valid for other values $< |r|$. Therefore, a solution of inequality 3.13 will also satisfy the following inequalities:

$$-wd^* \leq ur \pm vr \leq wd^* \ . \tag{3.14}$$

We can now solve inequality 3.14 to get the upper and lower bound of u.

$$-\frac{wd^*}{r} \leq u \leq \frac{wd^*}{r} \ . \tag{3.15}$$

A geometrical formulation on the bounds of u/w is shown in Fig. 3.8. Applying similar disparity

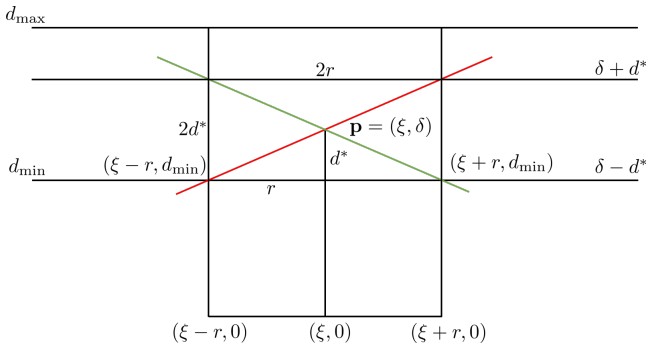


Figure 3.8: Bounds on u/w. Let $d^* = \delta - d_{\min}$. In the uw plane both the red and green lines (originated from $\mathbf{p}$) are extreme lines that follow the disparity bound constraint. Slopes of the red and the green line are d^*/r and $-d^*/r$, respectively. Any line whose slope is between $-d^*/r$ and d^*/r is a potential candidate plane.

bound constraint for the other view, we find the analogous representation of inequality 3.14 as:

$$-d^* \leq \frac{u}{w+u} r \pm \frac{v}{w+u} r \leq d^* \ . \tag{3.16}$$

If $\bar{\mathbf{p}} \in I'$, the equivalent representation of inequality 3.14 for the other view changes to:

$$-d^* \leq \frac{u}{w-u} r \pm \frac{v}{w-u} r \leq d^* \ . \tag{3.17}$$

Combining the bounds of u from inequality 3.15, 3.16 and 3.17 with the bounds from the visibility constraint (inequality 3.9 and 3.10), we get the bounds for u satisfying both constraints:

$$\begin{aligned} -\frac{wd^*}{r+d^*} < u \leq \frac{wd^*}{r} \quad \text{if } \bar{\mathbf{p}} \in I \ , \\ -\frac{wd^*}{r} \leq u < \frac{wd^*}{r+d^*} \quad \text{if } \bar{\mathbf{p}} \in I' \ , \end{aligned} \tag{3.18}$$

Finally, we get the bound of v from inequality 3.14, 3.16 and 3.17.

$$v \in \frac{s}{r} [-1, 1], \tag{3.19}$$

where

$$s = \begin{cases} \min\big(|wd^* \pm ur|, |(w+u)d^* \pm ur|\big) & \text{if } \bar{\mathbf{p}} \in I \ , \\ \min\big(|wd^* \pm ur|, |(w-u)d^* \pm ur|\big) & \text{if } \bar{\mathbf{p}} \in I' \ . \end{cases}$$

These are used as hard constraints during both the initialisation and the refinement process. The proposed scheme samples the unit normals multiple times until the constraints are satisfied. Using [60], we first generate random unit normals that are uniformly distributed over the visible hemisphere. Next, we only accept those unit normals that satisfy the constraints in inequality 3.9, 3.10, 3.18 and 3.19. The fronto-parallel windows can be enforced by setting $\mathbf{n} = (0, 0, 1)^\mathsf{T}$. Then we use d and $\mathbf{n}$ to find the required plane parameters at $\mathbf{p}$. These plane parameters are then used in Eq. 3.3 to find the disparity of all the other pixels inside a patch. During initialisation and plane refinement, while minimising the cost function over disparity, our constraints ensure that the centre pixel of the patch is visible from both views and the disparity of all the pixels inside a patch lie between the minimum and maximum disparity.

3.4 Constrained optimisation

We locally refine the plane parameters of $\boldsymbol{\pi}$ at a pixel $\bar{\mathbf{p}}$ to further reduce the matching cost. Here we change the disparity and the surface normal within bounds and seek an optimum disparity and unit normal. As our cost function is non-differentiable at some points due to the presence of discontinuous thresholds in the pixel dissimilarity function, we cannot use any standard gradient descent method to minimise it. It is also not convenient to mathematically compute the derivatives due to the threshold in the cost function. *PMS* uses a heuristic Luus-Jaakola type algorithm[4] [65]. This is not always effective, because it does not model the local structure of the cost function.

BOBYQA [71, 93] is a gradient-free non-linear trust region based algorithm for finding the minimum of an objective function $F(\mathbf{x})$, $\mathbf{x} \in \mathbb{R}^n$, subject to some constraints $\mathbf{a} \leq \mathbf{x} \leq \mathbf{b}$, without using the derivatives of $F(\mathbf{x})$. A trust region is a neighbourhood of the current iterate point, which is used in conjunction with a local quadratic approximation Q of F at that point. The quadratic surface Q is then minimised, and if the minimum of Q also reduces the value of F, then BOBYQA treats the minimum as a new iterate point. A larger trust region is generated around that point, and the process continues. If the minimum of Q does not reduce the value of F, then a smaller trust region is generated around the iterated point. The radius of the trust region depends on how well the quadratic model matches with the objective function and is updated after each iteration. BOBYQA consists of a very accurate and efficient system of updating the approximation models while maintaining a good set of interpolation points.

In more detail, BOBYQA starts with an initial vector $\mathbf{x}$ of dimension n, the constraints on $\mathbf{x}$ and a trust region radius. In each iteration, BOBYQA employs a local quadratic approximation Q of F such that $Q(\mathbf{x}_j) = F(\mathbf{x}_j)$, $j = 1, 2, \cdots, m$, where $m = 2n+1$. The initial interpolation points $\mathbf{x}_j$ are chosen and adjusted automatically. Let $\mathbf{x}_k$ be the point in the set $\{\mathbf{x}_j : j = 1, 2, \cdots, m\}$ that has the property $F(\mathbf{x}_k) = \min\{F(\mathbf{x}_j) : j = 1, 2, ..., m\}$ associated with the current trust region radius Δ_k. At each iteration, a new point $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{c}_k$, $\mathbf{c}_k \in \mathbb{R}^n$ is computed and one of the interpolation points, say $\mathbf{x}_j$, is replaced by the new iterate point $\mathbf{x}_{k+1}$ if $F(\mathbf{x}_{k+1}) < F(\mathbf{x}_k)$. The trust region step $\mathbf{c}_k$ is chosen by minimising $Q(\mathbf{x}_k + \mathbf{c})$, $\mathbf{c} \in \mathbb{R}^n$ subject to the prescribed bounds on variables $\mathbf{a} \leq \mathbf{x}_k + \mathbf{c} \leq \mathbf{b}$ under the condition $||\mathbf{c}|| \leq \Delta_k$. Further, a new trust region radius and quadratic approximation is generated using the new iterate point. In most cases the new trust

[4] Similar to bisection method.

region is computed as $\max\left(\frac{1}{2}\Delta_k, \|\mathbf{c}_k\|\right)$. Thus at each iteration, only one interpolation point is altered that minimises F among all the interpolation points from the minimising sequence $\mathbf{x}_k^\star$.

The disparity and the plane normals of a pixel obtained from the view propagation are used as initial inputs. The optimiser then minimises $\text{cost}(\bar{\mathbf{p}}, \mathscr{S})$ using the disparity and surface normal bounds as defined in Section 3.2.3 and 3.3. The disparity scale is very different from the unit normal scale, but BOBYQA compensates for unequal initial-step sizes in the different parameters by rescaling the parameters, in proportion to the initial trust region step [46]. As the input and the boundary conditions vary for each pixel, we let the BOBYQA subroutine compute the initial trust region radius heuristically from the bounds. The input vector in our case is the over-parametrised plane parameters of dimension $n = 4$ containing the disparity and the plane normals of a point in the disparity space. BOBYQA then heuristically interpolates $m = 9$ planes with the bounds and fits a quadratic approximation Q around the $\text{cost}(\bar{\mathbf{p}}, \mathscr{S})$. The quadratic model is later used to minimise the cost function along the trust region step $\mathbf{c}_k$.

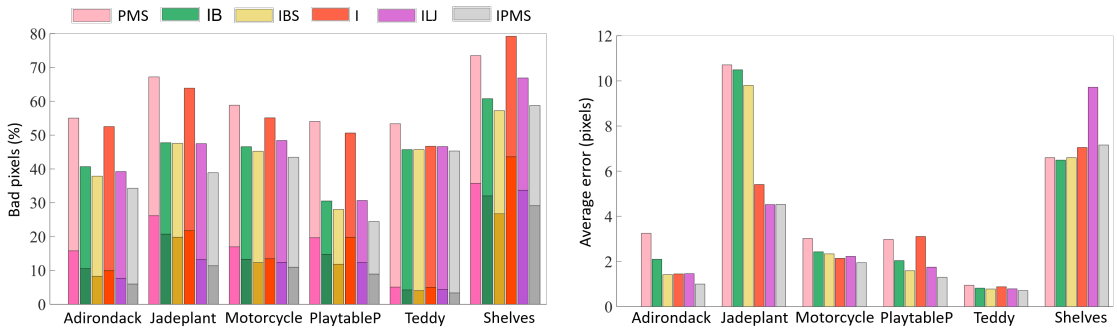
3.5 Results

3.5.1 Experimental set-up

We evaluate our proposed method, *IPMS*, using the Middlebury stereo benchmark, version 3 [76], which contains training and a test dataset, each with 15 pairs of stereo rectified images. The images contain a variety of challenges, such as radiometric changes and a large disparity range. Most images in the dataset have realistic *imperfect* rectification. The dataset comes in three resolutions (full, half and quarter). Due to the computational memory constraint, we use the half resolution dataset (up to 1500×1000 pixels) for our experiments. However, results are evaluated at full resolution to compare with other methods according to the benchmark.

There are six parameters in *IPMS*. We use the same parameters for all the stereo pairs. All the parameters are set empirically during training. We keep the patch size $\mathcal{W}(\bar{\mathbf{p}}) = 71 \times 71$ and use a 5×5 median filter for all the half resolution Middlebury images. The results are reported after the post processing step, unless otherwise stated. The values of γ_c and γ_g are the same as used in [91], α and τ_{grad} are the same as in [9] and τ_{col} is chosen empirically using the quarter resolution Middlebury training dataset:

$$\left(\gamma_c,\ \gamma_g,\ \alpha,\ \tau_{\text{col}},\ \tau_{\text{grad}}\right) := \left(5.0,\ \frac{\mathcal{W}(\bar{\mathbf{p}})}{2},\ 0.9,\ 0.01,\ 0.008\right).$$



(a) Dark and light shades represent percentage of non occluded bad pixels computed with error threshold of 2.0 and 0.5 pixels, respectively.

(b) Average disparity error for non occluded pixels.

Figure 3.9: Error comparison on a subset of the Middlebury training images (`Adirondack`, `Jadeplant`, `Motorcycle`, `PlaytableP`, `Teddy`, and `Vintage`). *PMS*: **P**atch**M**atch **S**tereo [9]; IB: PatchMatch Stereo with constrained **I**nitialisation and **B**OBYQA in plane refinement, cost function same as [9], no search image support weight; IBS: *PMS* with constrained **I**nitialisation and **B**OBYQA in plane refinement, cost function similar to [9], norm changed by TS**S**D, no search image support weight; I: **I**nitialised PatchMatch stereo with no plane refinement; ILJ : **I**nitialised PatchMatch stereo with a variant of **L**uus-**J**aakola optimisation in plane refinement; IPMS : **I**nitialised **P**atch**M**atch **S**tereo.

As a performance measure, we use the default metrics of the Middlebury stereo benchmark. The Middlebury error rate measures the *percentage of bad pixels*, *i.e.,* the percentage of pixels whose disparity errors are greater than a threshold with respect to the ground-truth disparity map. The default metric also measures the *average error* per pixel. The training dataset provides a mask for the occluded pixels allowing users to calculate the percentage of bad pixels and average error only on the non-occluded pixels. The percentage of bad pixels changes with the threshold, whereas the average error remains constant. We use the thresholds of 2.0 and 0.5 pixels.

3.5.2 Comparison with PMS

We first compare our results with *PMS* and other variants of *PMS* on a subset of the Middlebury training images. This subset was chosen because it exhibits different scene textures and disparity ranges. Results in Fig. 3.9 clearly show that the proposed modifications significantly decrease the percentage of bad pixels by $10-40\%$ and the average error by $25-50\%$ in most cases. It is also evident from Fig. 3.9 that the BOBYQA optimisation is more effective than the Luus-Jaakola method.

We compared both the initialisation schemes of *IPMS* and *PMS* and found that *IPMS* produces $35-45\%$ fewer bad planes than that of PMS during initialisation. We also compared the

Table 3.1: Percentage of bad pixels with 2.0 pixels error threshold on all pixels on a subset of the Middlebury half-resolution training dataset. The mean error is reported for 30 trials along with the standard deviation.

Method	weights		1	1	1	1	0.5	1
	mean	median	Adiron	Jadepl	Motor	PlaytP	Shelvs	Teddy
IPMS SP1	54.19	56.31	48.64 (1.55)	73.13 (0.81)	48.82 (1.47)	63.81 (2.17)	65.41 (0.79)	30.95 (1.88)
IPMS SP1 with init. as [24]	67.56	67.42	66.47 (1.63)	81.26 (0.78)	65.36 (1.24)	68.37 (1.69)	75.99 (1.15)	52.16 (1.87)
IPMS VP1	39.26	39.38	30.94 (1.19)	56.82 (0.58)	34.24 (0.56)	44.53 (1.48)	58.81 (0.37)	20.02 (0.59)
IPMS VP1 no neighbour	47.53	47.57	41.36 (1.08)	67.66 (1.08)	42.91 (0.99)	52.23 (0.76)	63.45 (0.73)	25.56 (0.55)
IPMS PR1	35.95	36.09	26.29 (1.09)	52.07 (0.62)	31.65 (0.49)	40.53 (1.35)	57.27 (0.34)	18.57 (0.47)
IMPS PR1 no vis. const. in opt.	39.85	39.01	33.61 (0.76)	56.28 (0.99)	35.27 (0.41)	42.73 (0.85)	60.84 (0.36)	20.84 (0.39)
IPMS	17.92	14.77	9.65 (0.25)	25.81 (0.37)	15.93 (0.48)	13.62 (0.42)	50.18 (0.35)	8.48 (0.27)
IPMS with init. as [24]	19.09	16.11	9.88 (0.28)	27.79 (0.43)	17.04 (0.50)	15.18 (0.36)	51.53 (0.56)	9.39 (0.47)
IPMS no neighbour in VP	18.63	15.57	9.81 (0.23)	27.41 (0.36)	16.86 (0.33)	14.28 (0.27)	51.09 (0.37)	8.56 (0.41)
IMPS no vis. const. in opt.	18.66	15.61	10.03 (0.19)	27.29 (0.36)	16.93 (0.44)	14.29 (0.48)	51.04 (0.32)	8.58 (0.35)

initialisation scheme of IPMS with that by Galliani *et al.* [24], which generates uniform normals on a sphere. Experimental results show that the proposed *IPMS* produces $30-40\%$ fewer bad planes compared to [24] during initialisation, $11-37\%$ fewer bad pixels during the first spatial propagation and $3-11\%$ fewer bad pixels on the final disparity map for error threshold of 2.0 on all pixels (Table 3.1).

Our modified view propagation allows more opportunities to improve the disparity by testing with neighbouring parameters. This approach is particularly useful where corresponding points are off by one or two pixels due to imperfect rectification. We performed new experiments on the Playtable stereo pair with both imperfect and perfect rectification [76] (Table 3.2). The results show that, in the case of imperfect rectification, there is a 20% improvement on the final disparity map. In comparison, for the perfectly rectified case, use of the full neighbourhood in view propagation did not change the final disparity results. Overall, the modified view propagation results in $20-35\%$ more plane propagations, reducing $17-28\%$ bad pixels during the first view propagation and $2-6\%$ fewer bad pixels on the final disparity map for error threshold of 2.0 on all pixels.

Table 3.2: Percentage of bad pixels with 2.0 error threshold on all pixels for the Playtable stereo pair with perfect and imperfect rectification in the first (FVP) and second (SVP) view propagation along with the final disparity disparity map. The mean error is reported for 30 trials along with the standard deviation.

Rectification	Disparity results	No neighbour	With neighbour
Imperfect	FVP	53.45 (2.01)	48.46 (1.87)
	SVP	40.12 (1.21)	31.34 (1.03)
	Final	**33.37** (**0.96**)	**26.73** (**0.76**)
Perfect	FVP	43.45 (1.42)	44.53 (1.48)
	SVP	19.33 (0.99)	19.12 (0.85)
	Final	**14.28** (**0.37**)	**13.62** (**0.42**)

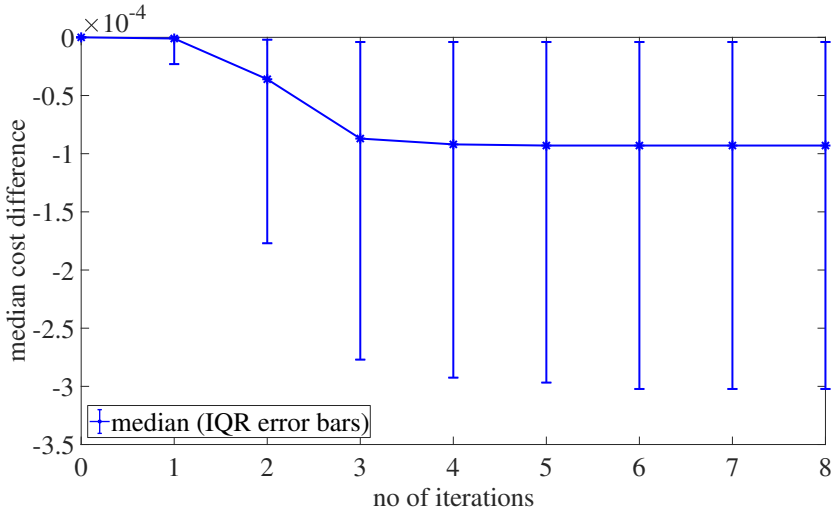
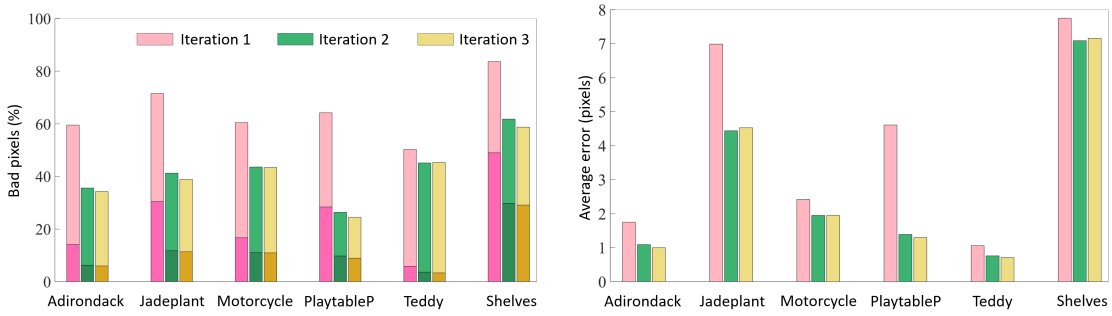


Figure 3.10: Cost difference of BOBYQA and Luus-Jaakola on the medium size `Jadeplant` dataset. We choose a grid of equally spaced points (131×99), 10 pixels apart and plot the median cost difference of BOBYQA and Luus-Jaakola along with the upper and lower quartile with respect to the no. of iterations. Both the optimisers start off from the same median cost but after the first iteration BOBYQA proves to be better than Luus-Jaakola.

We also experimented to find out the effectiveness of the constraints during optimisation. The performance of the BOBYQA optimiser depends on the specified bounds of the input variables. In the absence of visibility constraints, the trust region for the surface normal parameters will be unnecessarily large. This may result in a poor quadratic approximation to the empirical objective function, such that the minimum of the approximation does not offer any improvement (e.g. consider a U-shaped approximation around a W-shaped function). This interpretation is consistent with our experimental results in Table 3.1. In addition, it may be noted that although the disparity scale is very different from the unit normal scale, BOBYQA rescales the parameters in relation to the initial trust region step [46]. This process also depends on the specified bounds. Experimental results show $2-10\%$ fewer bad pixels during the first plane refinement and $2-6\%$ fewer bad pixels on the final disparity map for error threshold of 2.0 on all pixels.

We choose a sample of equally spaced pixels (131×99) on the half resolution `Jadeplant` stereo pair (10 pixels apart) and compared the median cost difference of BOBYQA and Luus-Jaakola along with the upper and lower quartile with respect to the number of iterations (Fig. 3.10). Both the optimisers start from the same median cost, but after the first iteration, BOBYQA proves to be better than Luus-Jaakola. The nature of the optimisation problem is significantly affected by the local image structure (and quality of the initial estimate), which is highly variable across the $131 \times 99 = 12969$ samples. The local performance is highly variable, due to differences in



(a) Dark and light shades represent percentage of non occluded bad pixels computed with error threshold of 2.0 and 0.5 pixels, respectively.

(b) Average disparity error for non occluded pixels.

Figure 3.11: Rate of convergence per iteration computed on a subset of the Middlebury training images (`Adirondack`, `Jadeplant`, `Motorcycle`, `PlaytableP`, `Teddy`, and `Vintage`). Note that two iterations are sufficient for convergence.

the available image structure (and initial estimates), across the visible scene. We also found out that BOBYQA does not increase the computational cost compared to the Luus-Jaakola method, as it also takes on average nine iterations to converge.

Fig. 3.11 shows that *IPMS* converges in only two iterations on the same subset of images. The main reason behind this faster convergence is the constrained initialisation of plane parameters, in contrast to the random initialisation used in *PMS*.

Our unoptimised C implementation of *PMS* takes around 150 minutes (standard intel core i7 desktop) to process an image of size 900×750 whereas *IPMS* is approximately five times more expensive, but still comparable to global methods such as [72]. The main reason for the additional computational time is the inclusion of the search image support weight in the cost function, which reduces false matching in occluded regions. Comparing *IPMS* to IBS clearly shows that the results are improved by adding the support image weights, especially for the `Jadeplant` stereo pair whose background is less textured with big occlusion. Importantly, if we do not include the support weight for the search image in the cost function, then *IPMS* is

Table 3.3: Results on the Middlebury training set: percentage of bad pixels and average error with 2.0 pixel error threshold on all pixels

		Method	Weights →	1	1	1	1	1	1	0.5	1	0.5	0.5	1	1	0.5	1	0.5
			Wt. avg.	Adiron	ArtL	Jadepl	Motor	MotorE	Piano	PianoL	Pipes	Playrm	Playt	PlaytP	Recyc	Shelvs	Teddy	Vintge
Bad pixel (%)	Global	MC-CNN-acrt [88]	19.9	10.0	25.6	32.8	12.7	12.6	18.8	24.4	**17.8**	25.6	23.3	21.6	**12.8**	34.4	14.3	30.5
		MC-CNN+RBS [4]	**19.5**	9.76	26.1	32.6	**12.2**	**12.4**	**17.9**	**22.7**	19.8	**25.1**	**22.8**	20.4	12.9	**32.4**	14.4	**27.3**
		MeshStereo [94]	21.1	10.1	20.1	36.6	14.1	14.6	19.9	28.9	24.0	32.0	26.7	16.6	16.5	41.0	13.9	27.5
		TMAP [72]	22.9	14.1	21.2	34.0	14.8	14.4	21.0	32.0	20.1	31.2	40.4	21.2	16.1	46.6	14.3	40.3
	Loc.	IPMS (proposed)	20.9	**9.65**	**15.46**	**25.81**	15.93	21.16	19.01	47.18	23.89	33.99	26.73	**13.62**	14.25	50.18	**8.48**	32.36
		IDR [52]	23.0	14.5	21.4	33.3	14.8	12.9	20.8	29.9	23.1	31.1	53.0	16.6	15.9	49.7	13.2	38.4
Average error	Global	MC-CNN-acrt [88]	11.8	4.24	18.7	34.1	7.21	7.22	6.00	9.35	13.5	18.3	9.71	9.37	4.64	6.62	9.35	21.6
		MC-CNN+RBS [4]	6.67	2.22	8.42	22.2	3.95	**3.87**	**2.34**	**4.74**	**13.9**	9.76	4.80	3.66	2.38	**4.63**	**5.90**	**5.13**
		MeshStereo [94]	7.59	2.39	6.44	36.4	5.40	5.71	3.25	5.45	11.6	6.34	4.92	2.73	2.25	11.1	1.90	5.62
		TMAP [72]	7.88	2.44	5.88	30.9	4.72	4.41	3.86	13.7	9.25	**6.12**	16.6	3.13	2.22	10.9	2.73	10.5
	Loc.	IPMS (proposed)	**5.79**	**1.79**	**4.16**	**17.9**	**3.56**	4.68	3.87	13.95	**8.42**	10.19	**3.11**	**2.00**	**2.42**	**8.4**	**1.35**	7.75
		IDR [52]	8.57	2.60	5.97	30.0	4.26	3.90	4.39	10.8	10.4	6.3	39.6	2.61	2.42	10.2	2.54	9.09

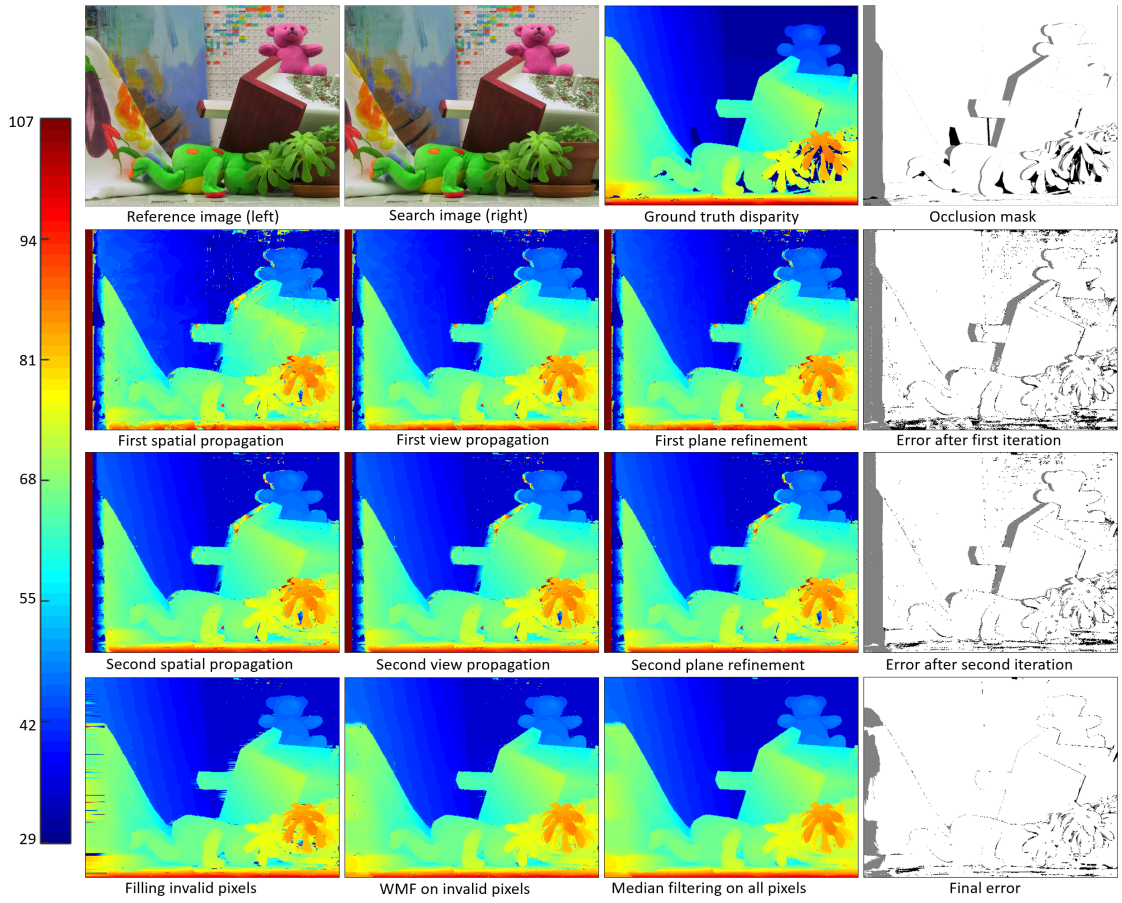


Figure 3.12: Teddy stereo pair results with default error (percentage of bad pixels) threshold of 2.0 pixels on all regions. Errors in occluded regions in the right-most column are shown in grey, while the errors in non-occluded areas are shown in black. The colour bar in the left-most column shows the disparity levels of the Teddy stereo pair.

one third faster than *PMS* due to BOBYQA, while still producing $15-35\%$ fewer bad pixels (Fig. 3.9).

3.5.3 Comparison with other methods

We now compare *IPMS* against state-of-the-art algorithms according to the Middlebury benchmark. Fig. 3.12 shows all the propagation results per iteration for the Teddy stereo pair reference image (900×750). Our method produces a high-quality disparity map even in the first iteration, because of the constrained initialisation.

To further illustrate the performance of our proposed method, we present the disparity map, the error map and the plane distribution map on a subset of the Middlebury training images in Fig. 3.13. Note that the proposed method successfully tackles the edge fattening problem, *e.g.,* `PlaytableP`. However, as with other local methods our algorithm has difficulties with sig-

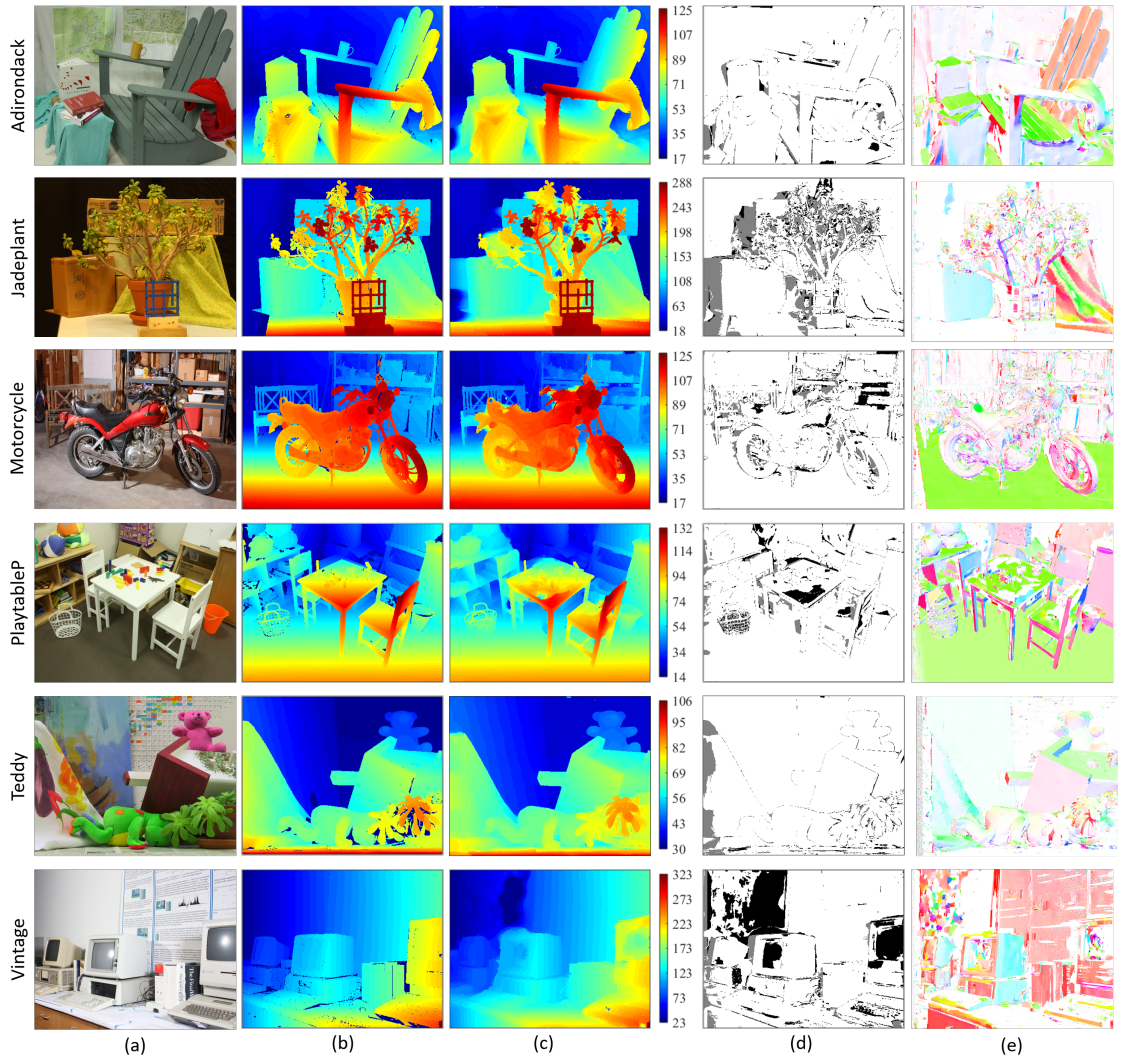


Figure 3.13: Results on a subset of the (a) Middlebury training images with (b) ground truth (`Adirondack`, `Jadeplant`, `Motorcycle`, `PlaytableP`, `Teddy`, and `Vintage`). (c) Disparity map. (d) Error map. Computed with default error threshold of 2.0 pixels on all regions. Errors in occluded regions are shown in grey, while the errors in non-occluded areas are shown in black. (e) Plane distribution. Hue and saturation represent surface tilt and slant, with white being fronto-parallel.

nificant radiometric changes and low textured regions, *e.g.,* `Vintage`. The percentage of bad pixels and average error on all pixels (without using a mask) for the Middlebury training dataset are shown in Table 3.3 along with other top performing published local and global methods. The results show that our method outperforms other local methods, and is among the top five overall (including global methods).

Table 3.4: Camera Specification

Camera	Resolution (MP)	Imaging sensor
PointGrey Flea3 USB 3.0 Digital Camera	3.2	Sony IMX036 CMOS, $1/2.8$", 2.5μ m, 60 FPS at 2080×1552
Nikon DSLR Camera	24.1	Nikon DX CMOS, 23.5 mm $\times$ 15.6 mm, Lens 18-55 mm. AF-S

3.6 Surface reconstruction

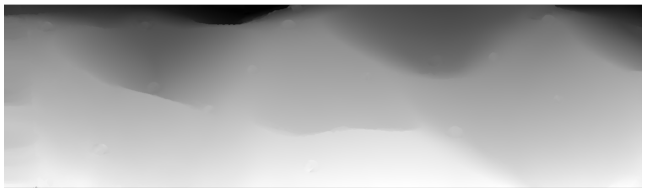
Apart from evaluating our algorithm on the Middlebury dataset, we also tested on our own sand-box dataset (Sec. 5) and also on crumpled paper dataset.

3.6.1 Imaging set-up

Initially, we used a pair of synchronised PointGrey cameras mounted over a stereo rig to capture the sandbox. For our application, we want the spatial resolution to be very high. Specifically, because the sand texture is so fine, we expect to see some colour-aliasing which will become very problematic for correlation-based stereo matching because two overlapping images will look different. During the experiment, we find out that the spatial resolution of the PointGrey cameras are lower than we would like for this application. To address this problem, we used a hand-held DSLR (Nikon D7100). All the camera specifications are given in Table 3.4. Because the scene is uncoloured, we use some coloured beads as markers to find the camera extrinsic (Sec 2.2.6). Finally the images were rectified (Sec 2.2.5).

(a)

(b)



(c)

Figure 3.14: Sand stereo pair. (a) Left image (b) Right image (c) Left disparity map

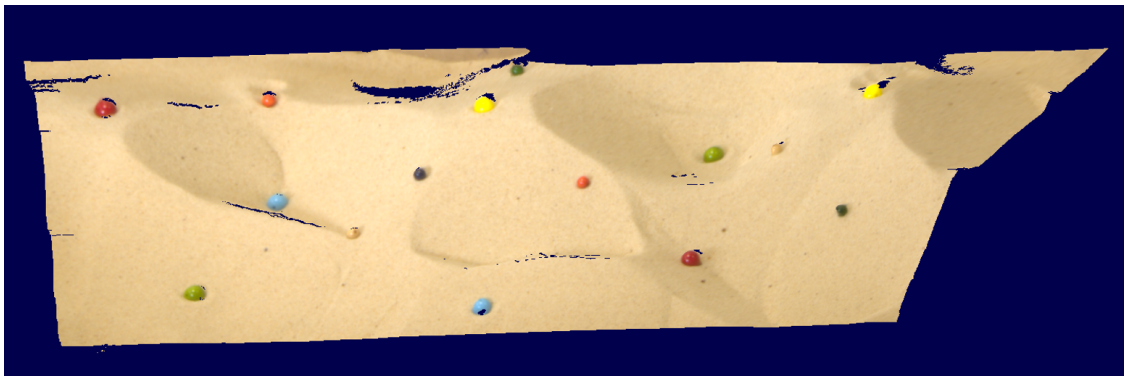
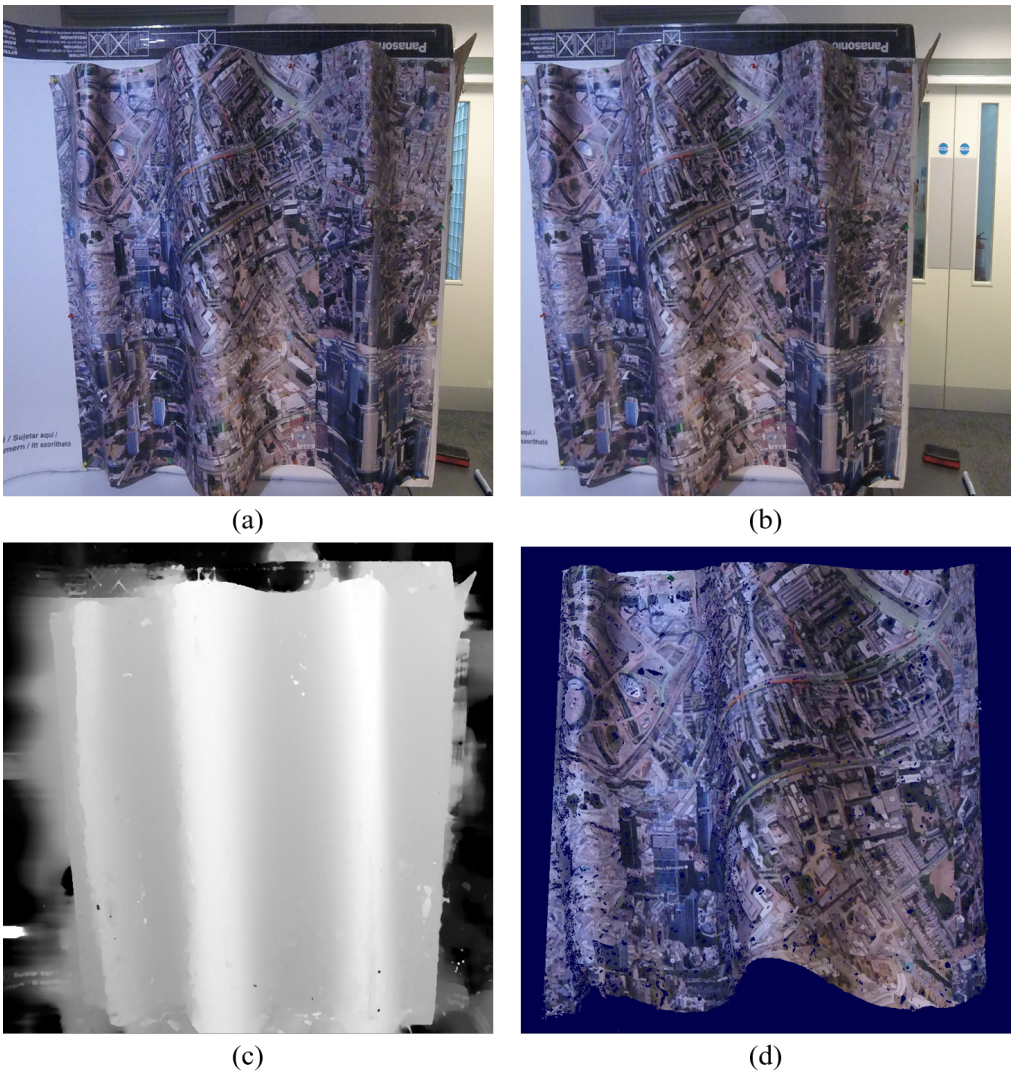


Figure 3.15: 3D visualisation of the left sand image



(a) (b)

(c) (d)

Figure 3.16: Crumpled paper dataset. (a) Left image (b) Right image (c) Left disparity map (d) Cropped 3D visualisation of the left crumpled paper.

3.6.2 Sand images

We created a sand stereo pair (1393×400) from a sandbox (75cm $\times$ 56cm) using a DSLR camera. We tested the proposed algorithm on the sand stereo pair (Fig. 3.14(a), (b)). By inspection, we found that the maximum and minimum disparity are 77 and 22 pixels, respectively. The disparity map (Fig. 3.14 (c)) was generated using a patch of size 51×51. A dense 3D reconstruction is shown in Fig. 3.14.

3.6.3 Crumpled paper surface reconstruction

To test whether the algorithm produces a smooth disparity map on curved surfaces, we captured a stereo pair (800×675) of a crumpled paper (Fig. 3.16(a), (b)). By inspection, we found the maximum and minimum disparities are 105 and 15 pixels. The disparity map (Fig. 3.16 (c)) was generated using a patch of size 51×51. A dense 3D reconstruction is shown in Fig. 3.16 (d).

3.7 Summary

We introduce a constrained initialisation scheme for the plane parameters in PatchMatch stereo, which ensures that only feasible planes are associated with each pixel. We also use the gradient-free non-linear optimiser BOBYQA, which we have shown to be more effective than Luus-Jaakola in refining the plane parameters. In addition, to tackle imperfectly rectified image pairs, we relax the view propagation. These modifications help our method to generate better disparity maps than state-of-the-art local methods and to converge in only two iterations. Moreover, the *IPMS* framework can be applied to any local stereo algorithm that can be cast in the PatchMatch stereo framework.

Chapter 4

Quadric surface model for PatchMatch stereo framework

4.1 Introduction

The *PMS* framework has six limitations. First, its initialisation process does not guarantee the association of a feasible plane at each pixel. Second, the plane refinement process uses a variant of the Luus-Jaakola optimisation [57] to minimise the cost function, which is inefficient to find a local minimum of the given cost function. Third, the framework assumes that the stereo images are perfectly rectified which is not the case for typical stereo pairs. Fourth, it generates false matches in low textured areas [32]. The model also fails to smoothly reconstruct curved surfaces in the disparity space as the framework is based on a planar model. Finally, the planar disparity model does not provide the curvature information of the associated surface, which is needed for a full understanding of the local surface structure.

The first four have been already addressed in [1]. In this chapter, we propose a quadric disparity model which successfully handles both curved and planar surfaces in the disparity space and, also estimates the curvature of the associated surface model for every pixel. We can get the curvature information by fitting a surface locally over a patch in the disparity space. The estimation depends on the patch size and the distribution of disparities over the patch and is highly affected by outliers. Moreover, such estimation does not use the surface normal information. In the proposed method, Quadric PatchMatch Stereo (*QPMS*), both spatial and normal information are used to approximate the surface locally by a quadric at each point in the disparity space.

Later, the quadrics are propagated within and across the stereo pair. We further address the false matching problem by introducing disparity guided spatial propagation, where a non-linear disparity dissimilarity function weights the aggregated cost. Disparity guided spatial propagation prevents false matches from growing and also fill them with the correct disparity value, provided there is at least one good surface approximation of the neighbours. Moreover, the proposed modifications will work with any stereo algorithm that can be cast in the PatchMatch Stereo framework.

The rest of the paper is organised as follows. The quadric disparity model is proposed in Section 4.2 and different quadric transformations are discussed in Section 4.3. The *QPMS* framework is introduced in Section 4.4. Different propagation schemes are explained in Section 4.5. The constrained optimisation is presented in Section 4.6. The curvature visualisation scheme is discussed in Section 4.7. Section 4.8 provides information on the experimental set-up, parameters used and results. Finally, conclusions are drawn in Section 4.9. In addition, some concepts of differential geometry are discussed in the Appendix B.

4.2 Disparity models

We define the disparity d as the horizontal shift between two matching pixels of a rectified stereo image pair (Sec. 2.3). A pixel $\bar{\mathbf{p}} = (x,y)^\mathsf{T}$ with disparity d is associated with a point $\mathbf{p} = (x,y,d)^\mathsf{T}$ in the disparity space $\mathcal{D}$ defined as the 3D space above the xy plane. As depth is inversely proportional to disparity, d is used to calculate the depth across the scene.

We start with a rectified colour stereo pair, I and I', where I is the reference image obtained by the left camera C and I' is the search image obtained by the right camera C'. Let $\mathcal{S}$ be the surface of the scene. Assume that we are trying to perform a pixel-wise matching of I with I'. Let $\mathbf{p}$ and $\mathbf{p}'$ be points in the disparity space that correspond to two matching points, $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$, projected from a scene point, $\mathbf{P} \in \mathcal{S}$, to the image plane. Let the surface $\tilde{\mathcal{S}}$ be a local approximation of $\mathcal{S}$ at $\mathbf{P}$ (Fig. 4.1). Our objective is to find the depth of the scene by estimating the local surface at each scene point.

The benefit of using local surface is two-fold. First, we can get the depth information. Second, we can estimate the geometric structure of the scene from the associated local surface. We first find the local surface approximations in the disparity space for every image point and later map them in the scene space to get the local surfaces of $\mathcal{S}$. Let $\mathscr{S}(x,y,d)$ be the approximated

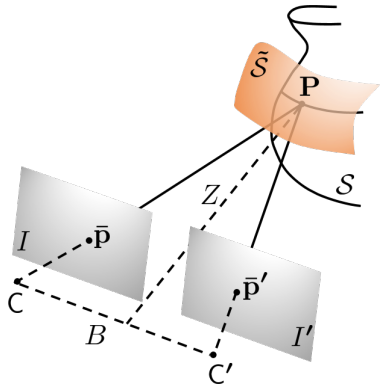


Figure 4.1: Geometric formulation. Image points $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$ are the projections of a scene point $\mathbf{P} \in \mathcal{S}$ on the reference image I and the search image I' from two different views obtained by the left camera C and the right camera C', where $\mathcal{S} \in \mathbb{R}^3$ is the surface of the scene. The baseline distance between C and C' is B. The scene point $\mathbf{P}$ is at a distance Z from B. The surface $\tilde{\mathcal{S}}$ is a local approximation of $\mathcal{S}$ at $\mathbf{P}$. Our objective is to find the depth of the scene by estimating the local surfaces at each scene point.

local surface of $\mathbf{p}$ in the disparity space $\mathcal{D}$. The local surface $\mathscr{S}$ is a differentiable expression (at least twice) that defines the disparity model.

4.2.1 Planar disparity model

The planar disparity model assumes $\mathscr{S}$ to be a linear function.

$$\mathscr{S} : ux + vy + wd + 1 = 0, \tag{4.1}$$

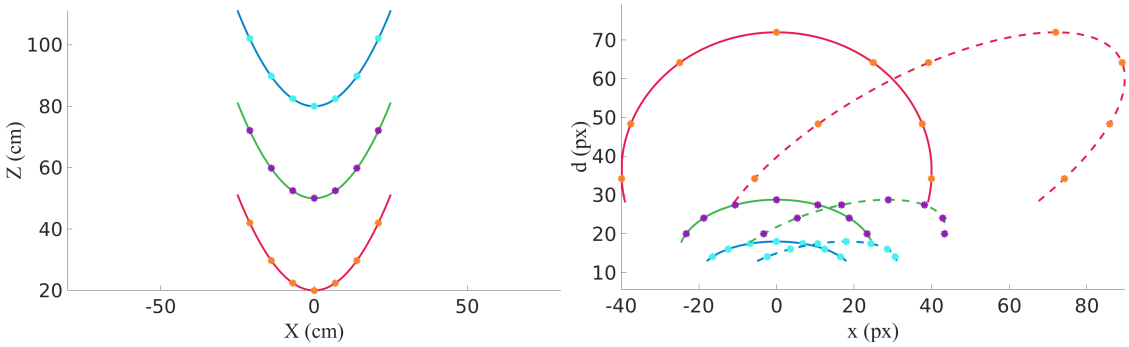
where $\boldsymbol{\pi} : (u, v, w)^\mathsf{T}$ are parameters of $\mathscr{S}$, known as the plane parameters. The *PMS* framework uses the planar model to associate planes for each pixel in both images.

4.2.2 Quadric disparity model

For the quadric disparity model, $\mathscr{S}$ is defined as the most general ten parameters implicit expression in x, y and d. The quadric disparity model is motivated by the transformation of a conic in between disparity spaces (left and right). Fig. 4.2b shows that the surfaces in disparity space can be double-valued in term of disparity. The quadric disparity model $\mathscr{S}$ is given by

$$\mathscr{S} : \mathbf{x}^\mathsf{T}\mathbf{Q}\mathbf{x} = 0, \quad \text{where } \mathbf{x} = (x, y, d, 1)^\mathsf{T} \in \mathcal{D} \text{ and } \mathbf{Q} \simeq \begin{pmatrix} a & f & g & u \\ f & b & h & v \\ g & h & c & w \\ u & v & w & k \end{pmatrix}. \tag{4.2}$$

Once the quadric is known, the disparity of an image point is given by the 'vertical' distance



(a) Parabolas in the scene space

(b) Disparity curves. Corresponding disparity curves and points have the same colour as in the scene curve. Solid curves corresponds to the left disparity space whereas the dotted curves corresponds to the right disparity space.

Figure 4.2: Quadric disparity models in 2D

of the quadric from the spatial position of the pixel. Computing the disparity is straightforward when the surface is a Monge patch (App. B.1) ($c = 0$). When $c \neq 0$, the disparity is double-valued. It is important to track the correct branch of the quadric out of the two choices to find out the true disparity. Solving eq. 4.2 for d, we get the disparity values (Fig. 4.3).

$$d = \begin{cases} -\dfrac{A}{2B}\,, & \text{if } c = 0\,, \\ \dfrac{-B \pm \sqrt{B^2 - cA}}{c}\,, & \text{otherwise}\,, \end{cases} \tag{4.3}$$

where $A = ax^2 + by^2 + 2fxy + 2ux + 2vy + k$ and $B = gx + hy + w$.

In this work, disparity is over parametrized by ten parameters, where $\mathscr{S} : (a, b, c, f, g, h, u,$

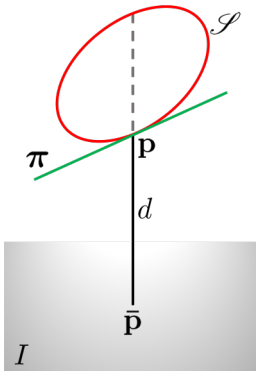


Figure 4.3: Disparity models in 1D. Pixel $\bar{\mathbf{p}} \in I$ with disparity d corresponds to $\mathbf{p} \in \mathcal{D}$. The green line represents the planar surface $\boldsymbol{\pi}$ and the red curve represents the quadric surface $\mathscr{S}$ at $\mathbf{p}$. For $\boldsymbol{\pi}$, disparity is single valued whereas we can get two disparity values for $\mathscr{S}$. In that case, it is important to track the correct branch of the quadric out of the two choices to find the correct disparity.

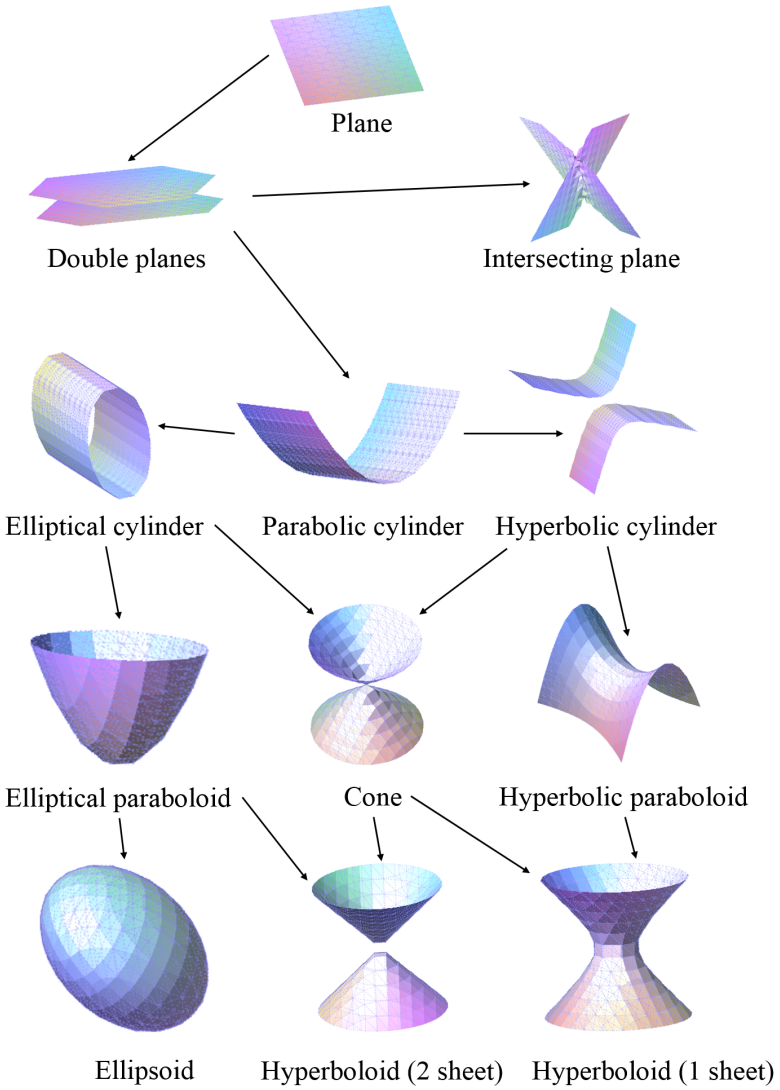


Figure 4.4: Quadric types (not including rotationally symmetric subtypes). Arrows indicate "is on the boundary of" relationships: One quadric type is on the boundary of another if any quadric of the first type can be perturbed by ε to create a quadric of the second type. These relationships are transitive, *e.g.,* planes lie on the boundary of all other quadric types. Figure reproduced from [2].

v, w, k) are called the quadric parameters. Apart from the surface normals, the quadric also gives us the curvature information, which provides full understanding of the surface structure.

A chart of quadric types (not including symmetric rotational subtypes) is shown in Fig. 4.4. Rotationally symmetric quadric types (circular cylinder, circular cones, spheroids, and circular hyperboloids) have the same equations, with an added constraint that any two squared terms have the same coefficient. We list the distinguishing properties of different quadric types in Tab. 4.1.

Table 4.1: Canonical form (centered at the origin and axis aligned with the Euclidean geometry) of common quadratic surface types. Table reproduced from [2].

Equation (Canonical Form)	Quadric Type (Canonical Position)	Distinguishing Property
$ax^2+by^2+cd^2+k=0$	Ellipsoid or Hyperboloid (Centered)	Ellipsoid if all squared terms have same sign
$ax^2+by^2+cd^2=0$	Cone (Centered)	The constant term is zero
$ax^2+by^2+wd+k=0$	Paraboloid (Aligned with z-axis)	One squared term is zero
$ax^2+by^2+k=0$	Cylinder (Aligned with z-axis)	Squared and corresponding linear term are both zero
$cd^2+k=0$	Double Plane (Plane normals aligned with z-axis)	Two pairs of squared and linear terms are zero
$ux+vy+wd+k=0$	Plane	Quadratic terms all zero

4.3 Quadric transformations

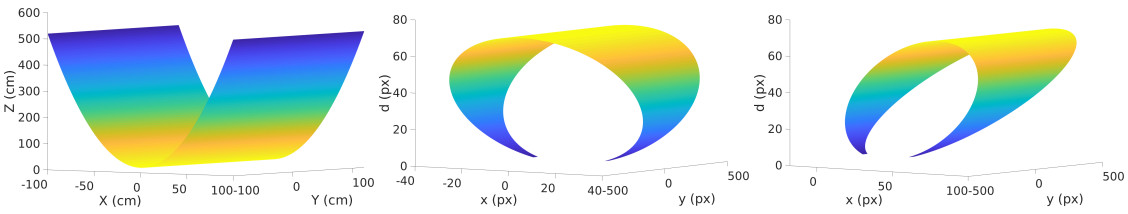
A quadric $\tilde{\mathbf{Q}}$ in the scene space is given by

$$\mathbf{X}^\mathsf{T}\tilde{\mathbf{Q}}\mathbf{X}=0, \quad \text{where } \mathbf{X}=(X,Y,Z,1)^\mathsf{T}\in\mathcal{S} \text{ and } \tilde{\mathbf{Q}}\simeq\begin{pmatrix} a & f & g & u \\ f & b & h & v \\ g & h & c & w \\ u & v & w & k \end{pmatrix}. \tag{4.4}$$

We start with the simplest quadric model, which is a parabolic cylinder (Fig. 4.5a). A parabolic cylinder in the scene space can be represented by a Monge patch as $aX^2+2wZ+k=0$. The quadric associated to the parabolic cylinder is given by :

$$\mathbf{X}^\mathsf{T}\tilde{\mathbf{Q}}\mathbf{X}=0, \quad \text{where } \tilde{\mathbf{Q}}\simeq\begin{pmatrix} a & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & w \\ 0 & 0 & w & k \end{pmatrix}. \tag{4.5}$$

Next, we investigate the transformation of parabolic cylinders and other quadrics in between disparity spaces.



(a) Parabolic cylinder in scene space. (b) Corresponding surface in the left disparity space. (c) Corresponding surface in the right disparity space.

Figure 4.5: Change of a quadric between scene space and disparity space.

4.3.1 Scene to image space conversion

For a scene point $\mathbf{X}$, let $\bar{\mathbf{x}}$ be the corresponding image point in homogeneous coordinates. Then, there exists a projective transformation (Sec. 2.2.3), *s.t.,*

$$\bar{\mathbf{x}} \simeq \mathbf{K} \left[\mathbf{I} \mid \mathbf{0} \right] \mathbf{X}, \tag{4.6}$$

for some camera matrix $\mathbf{K}$.

4.3.2 Scene to disparity space conversion

For a scene point $\mathbf{X} = (X, Y, Z, 1)^\mathsf{T} \in \mathcal{S}$, let $\mathbf{x} = (x, y, d, 1)^\mathsf{T} \in \mathcal{D}$ be the corresponding point in the disparity space $\mathcal{D}$. There exists a projective transformation $\mathbf{\Gamma}$, *s.t.,*

$$\mathbf{x} \simeq \mathbf{\Gamma X}, \quad \text{where } \mathbf{\Gamma} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \tag{4.7}$$

4.3.3 Transformation of a quadric from scene to disparity space

The corresponding quadric of eq. 4.4 in the disparity space is given by:

$$\mathbf{x}^\mathsf{T} \mathbf{Q} \mathbf{x} = 0, \quad \text{where } \mathbf{Q} = \mathbf{\Gamma}^{-\mathsf{T}} \tilde{\mathbf{Q}} \mathbf{\Gamma}^{-1} \simeq \begin{pmatrix} a & f & u & g \\ f & b & v & h \\ u & v & k & w \\ g & h & w & c \end{pmatrix} \tag{4.8}$$

It follows that a parabolic cylinder (Fig. 4.5a) in the scene space transforms to an elliptic or hyperbolic cylinder in the disparity space (Fig. 4.5b), and is represented by:

$$ax^2 + kd^2 + 2wd = 0 \tag{4.9}$$

Therefore, the transformed surface from scene space to disparity space is not a Monge patch, as it is no longer single-valued in d.

4.3.4 Transformation of a quadric among views in disparity space

Consider the following quadric $\mathbf{Q}$ in the left disparity space $\mathcal{D}_\mathsf{L}$.

$$\mathbf{x}^{\mathsf{T}}\mathbf{Q}\mathbf{x} = 0, \quad \text{where } \mathbf{Q} \simeq \begin{pmatrix} a & f & g & u \\ f & b & h & v \\ g & h & c & w \\ u & v & w & k \end{pmatrix}. \tag{4.10}$$

We are interested in finding the corresponding quadric in the right disparity space $\mathcal{D}_{\mathsf{R}}$. A point $\boldsymbol{\rho} = (\xi, \eta, \delta, 1) \in \mathcal{D}_{\mathsf{L}}$ is mapped to $\boldsymbol{\rho'} = (\xi - \delta, \eta, \delta, 1) \in \mathcal{D}_{\mathsf{R}}$.

$$\boldsymbol{\rho} = \mathbf{V}_{\mathsf{LR}}\,\boldsymbol{\rho'}, \text{ where } \mathbf{V}_{\mathsf{LR}} = \begin{pmatrix} 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{4.11}$$

The quadric in the left disparity space will change in the right disparity map as follows:

$$\mathbf{x}^{\mathsf{T}}\mathbf{Q}_{\mathsf{LR}}\mathbf{x} = 0, \text{ where } \mathbf{Q}_{\mathsf{LR}} = {\mathbf{V}_{\mathsf{LR}}}^{-\mathsf{T}}\mathbf{Q}{\mathbf{V}_{\mathsf{LR}}}^{-1}. \tag{4.12}$$

If the point $\mathbf{x} \in \mathcal{D}_{\mathsf{R}}$, then the corresponding quadric in the left disparity space is given by :

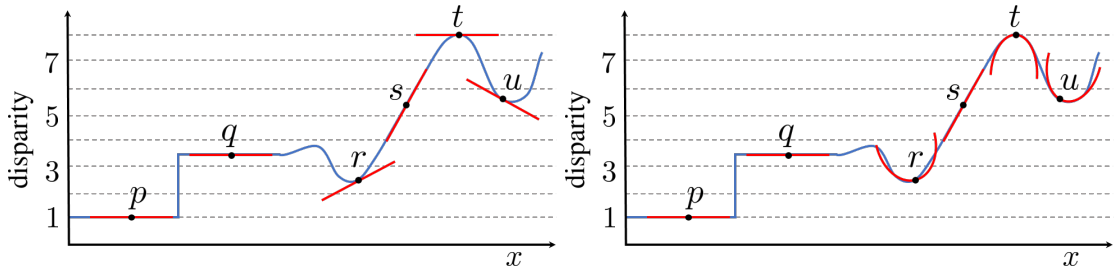
$$\mathbf{x}^{\mathsf{T}}\mathbf{Q}_{\mathsf{RL}}\mathbf{x} = 0, \quad \text{where } \mathbf{Q}_{\mathsf{RL}} = {\mathbf{V}_{\mathsf{RL}}}^{-\mathsf{T}}\mathbf{Q}{\mathbf{V}_{\mathsf{RL}}}^{-1} \text{ and } \mathbf{V}_{\mathsf{RL}} = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{4.13}$$

Eq. 4.13 and 4.12 shows that even if we start from a Monge patch in any one of the disparity space, the corresponding surface in the other disparity space may not be a Monge patch any-more due to the change of view-point. The corresponding quadric (Fig. 4.5c) of eq. 4.9 is given by:

$$ax^2 + (a+k)\,d^2 + 2axd + 2wd = 0.$$

4.4 Quadric PatchMatch Stereo (QPMS)

The planar model in *PMS* successfully tackles the sub-pixel disparity problem (Fig. 4.6a, point q) and reconstructs slanted planar surfaces (Fig. 4.6a, point s). However, one of the primary limitations of the planar model is that it fails to reconstruct curved surfaces smoothly (Fig. 4.6a, points r, t, u). The proposed Quadric PatchMatch Stereo (*QPMS*) method successfully handles curved surfaces and produces a smoother disparity map by fitting a quadric surface at each pixel in the disparity space onto which the support region is projected (Fig. 4.6b, points r, t, u). The key ideas of the *QPMS* framework are same as in *PMS*, that neighbouring pixels have coherent



(a) Planar support regions of PMS.

(b) Proposed quadric support regions.

Figure 4.6: Quadric surface models in 1D. The blue curve represents a 1D slice of the surface and the approximated surface models are shown in red. Figure partially reproduced from [9].

matches, and large numbers of random samples will yield some good initial estimates of the surface parameters. *QPMS* follows the same framework of *IPMS* with modifications highlighted in bold in Fig. 4.8. The principle idea of the method is to first fit tangent planes at every point in the disparity space and then propagate the plane parameters within and across the stereo pair. We skip the optimisation process in the first iteration and let the plane normal propagate in either direction of the images. The quadric disparity model is introduced after the second view propagation. The surface parameters are then propagated in further iterations (Fig. 4.7). We also start tracking the correct branch of the quadric disparity model from this process onwards. Similar initialisation and post-processing scheme are followed as mentioned in [1]. The *QPMS* framework also produces two separate disparity maps for the stereo pair.

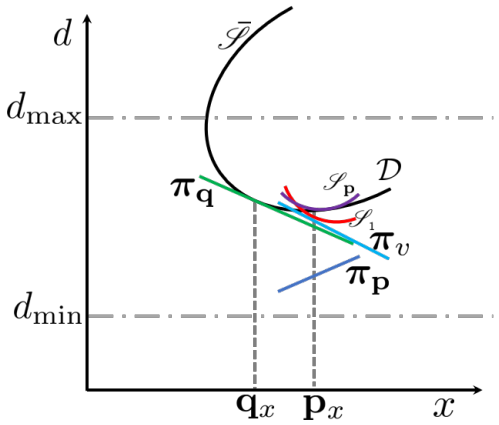


Figure 4.7: *QPMS* quadric initialisation in 1D. Let $\bar{\mathcal{S}} \in \mathcal{D}$ be a slice of a surface along the *xd* plane. The point $\mathbf{q}_x$ is a spatial neighbour of $\mathbf{p}_x$. Planes $\boldsymbol{\pi}_{\mathbf{p}}$ and $\boldsymbol{\pi}_{\mathbf{q}}$ are the initialised local surfaces for $\mathbf{p}_x$ and $\mathbf{q}_x$, respectively. The plane $\boldsymbol{\pi}_{\mathbf{q}}$ is propagated during the first spatial propagation as $\mathbf{p}_x$'s support region. The plane $\boldsymbol{\pi}_v$ is the local surface approximation of $\mathbf{p}_x$ after the second view propagation. The quadric disparity model is now introduced to approximate $\bar{\mathcal{S}}$ locally at $\mathbf{p}_x$ by $\mathcal{S}_1$ using the optimisation constraints. The local surface $\mathcal{S}_{\mathbf{p}}$ is the final quadric associated with $\mathbf{p}_x$ obtained after the second optimisation.

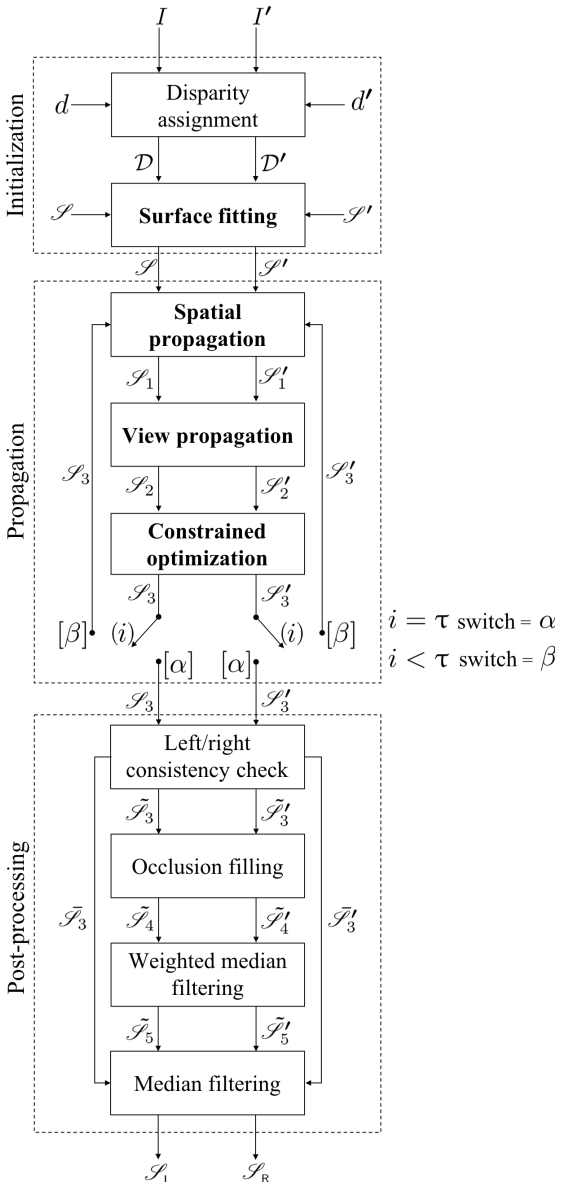


Figure 4.8: Quadric PatchMatch Stereo (*QPMS*) flowchart. Input I and I' are the rectified reference and search images, respectively. Left and right disparity spaces $\mathcal{D}$ and $\mathcal{D}'$ are generated by selecting disparities d and d' from the disparity constraints for every pixel in I and I', respectively. During initialisation, the surface normals $\mathbf{n}$ and $\mathbf{n}'$ are selected from the normal constraints in [1], which are used along with the point in the disparity space to generate the planes $\mathscr{S}$ and $\mathscr{S}'$ for every pixel in I and I', respectively. During the first two iterations, the planes are propagated via spatial, and view propagation. The quadrics are generated during the first constrained optimisation in the second iteration after view propagation using the BOBYQA optimiser. Note that, there is no optimisation in the first iteration. The whole process is then repeated using the quadrics. The total number of iterations is a user-defined parameter τ, while the iteration number is denoted by i. After each operation, the updated surfaces are represented by $\mathscr{S}_j$ and $\mathscr{S}'_j$. *QPMS* converges in three iterations. After the post-processing the final surface parameters for the left and right image are denoted by $\mathscr{S}_\mathsf{L}$ and $\mathscr{S}_\mathsf{R}$, respectively.

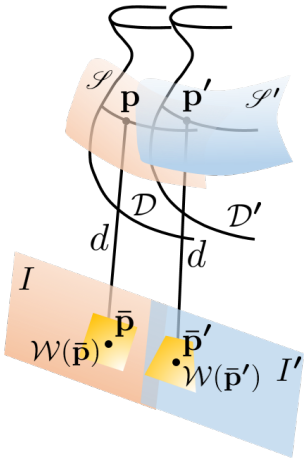


Figure 4.9: Quadric PatchMatch stereo. Image points $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$ are matching pixels with disparity d lying on the reference image I and the search image I', respectively. The matching pixels $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$ correspond to $\mathbf{p} \in \mathcal{D}$ and $\mathbf{p}' \in \mathcal{D}'$, where $\mathcal{D}$ and $\mathcal{D}'$ represent the disparity spaces generated by I and I', respectively. Points $\mathbf{p}$ and $\mathbf{p}'$ lie on the quadric $\mathscr{S}$ and $\mathscr{S}'$, respectively, where $\mathscr{S}'$ is the transformed surface of $\mathscr{S}$ due to change of view point. A rectangular patch centred at $\bar{\mathbf{p}}$ is denoted by $\mathcal{W}(\bar{\mathbf{p}})$. The patch $\mathcal{W}(\bar{\mathbf{p}}')$ is the projection of $\mathcal{W}(\bar{\mathbf{p}})$ with respect to $\mathscr{S}$.

To find corresponding pixels, *QPMS* starts with a rectified colour stereo pair, comprising I and I', where I is the reference image and I' is the search image (which will be exchanged during the course of the algorithm). Let $\bar{\mathbf{p}} = (x, y, 1)^\mathsf{T} \in I$ and $\bar{\mathbf{p}}' = (x', y, 1)^\mathsf{T} \in I'$ be corresponding pixels (Fig. 4.9). As I and I' are rectified and $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$ are matching pixels; $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$ have the same y coordinate and the same disparity magnitude d. Then $\mathbf{p} = (x, y, d, 1)^\mathsf{T} \in \mathcal{D}$ and $\mathbf{p}' = (x', y, d, 1)^\mathsf{T} \in \mathcal{D}'$ are two different projective transformations of $\mathbf{P}$ [36], where $\mathcal{D}$ and $\mathcal{D}'$ denote the disparity spaces generated by the reference and search image pixels, respectively. We have used different notations for the disparity spaces to highlight the projective transformations of $\mathbf{P}$ generating from two different views. As $\mathbf{p} \in \mathcal{D}$, $x' = x - d$, otherwise $x' = x + d$. The relation between $\mathbf{p}$ and $\mathbf{p}'$ in the disparity space is given by $\mathbf{V}_\star$ in eq. 4.11 or eq. 4.13, depending on which disparity space $\mathbf{p}$ resides. Disparity spaces $\mathcal{D}$ and $\mathcal{D}'$ will differ due to the visibility effects. *i.e.,* eq. 4.12 and eq. 4.13 are not true for all points in practice. Once the surface parameters are known for every pixel, the disparities are computed using eq. 4.3 and the correct branch.

To assign the disparity of I and I', we need to find an optimum quadric at each pixel that maximises the similarity of an image region across views. Let $\mathbf{p}$ be a point in the disparity space which corresponds to an image point $\bar{\mathbf{p}}$. Let $\mathbb{S}$ be the set of all candidate quadrics passing through

$\mathbf{p}$; we want to find a quadric $\bar{\mathscr{S}}$ that minimises the aggregated matching cost:

$$\bar{\mathscr{S}} = \underset{\mathscr{S}\in\mathbb{S}}{\arg\min}\ \text{cost}\big(\bar{\mathbf{p}}, \mathscr{S}\big)\ .$$

The aggregated cost of $\bar{\mathbf{p}}$ according to $\mathscr{S}$ is computed as

$$\text{cost}(\bar{\mathbf{p}}, \mathscr{S}) = \frac{\sum\limits_{\bar{\mathbf{q}}\in\mathcal{W}(\bar{\mathbf{p}})}\ \sum\limits_{\bar{\mathbf{q}}'\in\mathcal{W}(\bar{\mathbf{p}}')} A\big(\bar{\mathbf{p}}, \bar{\mathbf{q}}\big)\cdot A\big(\bar{\mathbf{p}}', \bar{\mathbf{q}}'\big)\cdot E\big(\bar{\mathbf{q}}, \bar{\mathbf{q}}'\big)}{\sum\limits_{\bar{\mathbf{q}}\in\mathcal{W}(\bar{\mathbf{p}})}\ \sum\limits_{\bar{\mathbf{q}}'\in\mathcal{W}(\bar{\mathbf{p}}')} A\big(\bar{\mathbf{p}}, \bar{\mathbf{q}}\big)\cdot A\big(\bar{\mathbf{p}}', \bar{\mathbf{q}}'\big)}\ , \tag{4.14}$$

where $\mathcal{W}(\bar{\mathbf{p}})$ denotes a square patch centered at $\bar{\mathbf{p}}$, $\bar{\mathbf{p}}'$ denotes the matching pixel of $\bar{\mathbf{p}}$ with respect to $\mathscr{S}$ and $\mathcal{W}(\bar{\mathbf{p}}')$ denotes the projection of $\mathcal{W}(\bar{\mathbf{p}})$ with respect to $\mathscr{S}$. $\bar{\mathbf{q}}'$ is the matching pixel of $\bar{\mathbf{q}}$ in the other view with respect to the quadric $\mathscr{S}$. The weight function $A(\bar{\mathbf{p}}, \bar{\mathbf{q}})$ and the error function $E(\bar{\mathbf{q}}, \bar{\mathbf{q}}')$ are the same as defined in eq. 3.4 and eq. 3.5, respectively.

4.5 Propagation

We use the *PMS* iterative scheme to propagate the surface parameters in two different directions considering both views. During even iterations, the algorithm starts with the top left pixel and traverse pixels in row-major order until the bottom right pixel is reached. The order is reversed during odd iterations, *i.e.,* starting with the right bottom pixel and stopping at the top left one. In every iteration, each pixel runs through three independent stages: spatial propagation, view propagation, and constrained optimisation, except the first one. We only allow spatial and view propagation during the first iteration. Our method converges in three iterations. An additional iteration is required for *QPMS* in comparison to *IPMS* as we start the optimisation process from the second iteration onwards. Next, we discuss the propagation schemes in detail.

4.5.1 Spatial propagation

The idea behind spatial propagation is that neighbouring pixels are likely to be associated with the same surfaces. Therefore, we pass the quadric parameters to its neighbours and check whether the new disparity of the neighbouring pixel is within the approved disparity range (in between maximum (d_{max}) and minimum (d_{min}) disparity) and improves the cost function (Fig. 4.10a).

For any pixel $\bar{\mathbf{p}}$ in the image, let $\mathbf{p}$ be its corresponding point in the disparity space lying on the quadric $\mathscr{S}$. Let $\bar{\mathbf{q}}$ be a spatial neighbour of $\bar{\mathbf{p}}$ and $\mathbf{q}$ be the projection of $\bar{\mathbf{q}}$ in the disparity space lying on the quadric $\mathscr{S}_{\mathbf{q}}$. We first check whether the new disparity of $\bar{\mathbf{p}}$ with respect to $\mathscr{S}_{\mathbf{q}}$

lies between $d_{\max}$ and $d_{\min}$. If so, we evaluate whether assigning $\mathscr{S}_{\mathbf{q}}$ to $\bar{\mathbf{p}}$ improves the aggregated cost, *i.e.,* we check the condition $\text{cost}(\bar{\mathbf{p}}, \mathscr{S}_{\mathbf{q}}) < \text{cost}(\bar{\mathbf{p}}, \mathscr{S})$. If this is the case, we accept and update $\mathscr{S}_{\mathbf{q}}$ as the new supporting quadric of $\bar{\mathbf{p}}$, *i.e.,* $\mathscr{S} := \mathscr{S}_{\mathbf{q}}$ (Fig. 4.10a). In even iterations we consider the left and upper neighbours as spatial neighbours, whereas in odd iterations the right and lower neighbours are verified (Fig. 4.10b and 4.10c).

Only in the second iteration, we guide the spatial propagation by introducing a weight on the cost aggregation based on the disparity difference of the immediate neighbours with the candidate pixel (Fig. 4.10b and 4.10c). The weight function between a candidate pixel $\bar{\mathbf{p}}$ and a neighbouring pixel $\bar{\mathbf{p}}_\ell$ is given by

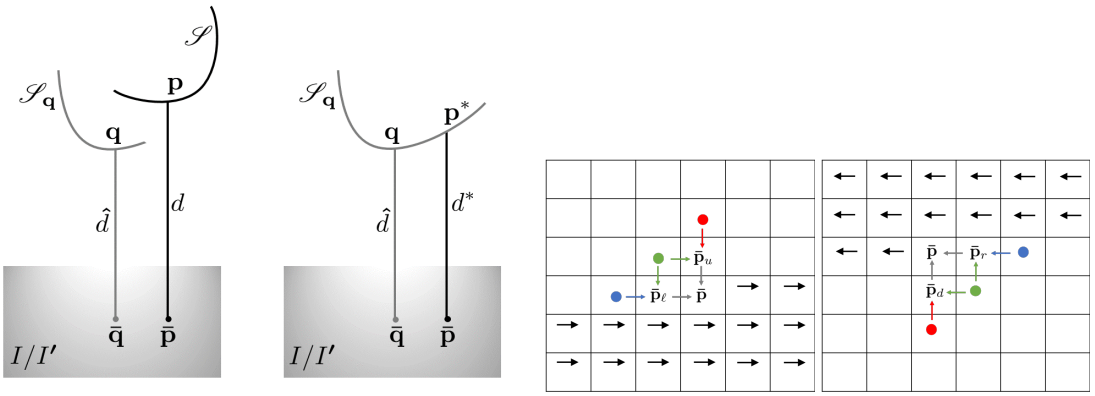
$$\mu(\bar{\mathbf{p}}, \bar{\mathbf{p}}_\ell) = \left(1 - \frac{\min\left(|d_{\bar{\mathbf{p}}_\ell}, d_{\bar{\mathbf{p}}_\ell^\dagger}|, |d_{\bar{\mathbf{p}}_\ell}, d_{\bar{\mathbf{p}}_\ell^\star}|\right)}{d_{\max} - d_{\min}}\right)^{-1}, \tag{4.15}$$

where $\bar{\mathbf{p}}_\ell^\dagger$ and $\bar{\mathbf{p}}_\ell^\star$ are neighbours of $\bar{\mathbf{p}}_\ell$. The weight function prefers frontal parallel planes than highly slanted ones, which causes false matching. Similar to the propagation scheme, in even iterations, we use the left and upper neighbour of $\bar{\mathbf{p}}_\ell$, whereas in odd iterations, only right and lower ones are considered. Disparity guided spatial propagation (DSP) prevents false matches from growing and also fill them with the correct disparity value, provided there is at least one good surface approximation of the neighbours.

4.5.2 View propagation

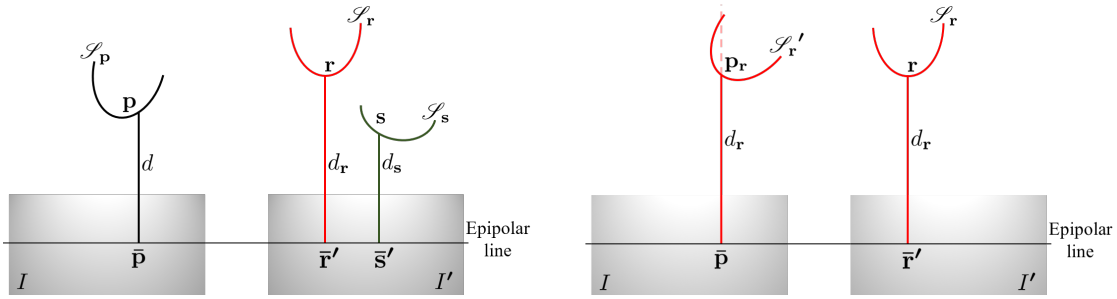
We exploit the strong coherency that exists between the left and right disparity maps so that a pixel and its matching pixel in the other view have the same disparity (Fig. 4.11a). However, the surfaces change across views due to different viewpoints in the disparity space. There is no guarantee that a Monge patch will remain a Monge patch in the other view (Fig. 4.11b). However, this is not a problem as we use the general quadric model and also track the branches.

Let $\{\bar{\mathbf{r}}', \bar{\mathbf{s}}'\} \in I'$ be two possible matching points of $\bar{\mathbf{p}} \in I$. Points $\mathbf{p}$, $\mathbf{r}$ and $\mathbf{s}$ are corresponding points of $\bar{\mathbf{p}}$, $\bar{\mathbf{r}}'$, $\bar{\mathbf{s}}'$ in the disparity space lying on the quadric $\mathscr{S}_{\mathbf{p}}$, $\mathscr{S}_{\mathbf{r}}$ and $\mathscr{S}_{\mathbf{s}}$ with disparity d, $d_{\mathbf{r}}$ and $d_{\mathbf{s}}$, respectively (Fig. 4.11a). We transfer the disparity and the support quadric of $\bar{\mathbf{r}}'$ and $\bar{\mathbf{s}}'$ to $\bar{\mathbf{p}}$ and find the new surface parameters (Fig. 4.11b). If the new quadric minimises the aggregated cost of the patch $\mathcal{W}(\bar{\mathbf{p}})$ centred at $\bar{\mathbf{p}}$, we update the surface parameters of $\bar{\mathbf{p}}$ with the new one. As corresponding pixels have equal disparities, we first find the corresponding quadric in the reference image using $\mathbf{V}_{\mathsf{RL}}$ (eq. 4.11) or $\mathbf{V}_{\mathsf{LR}}$ (eq. 4.13), depending on which disparity space the



(a) Disparity guided spatial propagation (b) Even iteration (c) Odd iteration

Figure 4.10: (a) Image point $\bar{\mathbf{q}}$ is a spatial neighbour of $\bar{\mathbf{p}}$. Points $\mathbf{p}$ and $\mathbf{q}$ are corresponding points of $\bar{\mathbf{p}}$ and $\bar{\mathbf{q}}$ in the disparity space lying on the quadric $\mathscr{S}$ and $\mathscr{S}_\mathbf{q}$, with disparity d and $\hat{d}$, respectively. Image point $\bar{\mathbf{p}}$ corresponds to a new point $\mathbf{p}^*$ in the disparity space with respect to $\mathscr{S}_\mathbf{q}$. In spatial propagation, we aggregate the cost of the patch $\mathcal{W}(\bar{\mathbf{p}})$ centered at $\bar{\mathbf{p}}$ with respect to $\mathscr{S}$ and $\mathscr{S}_\mathbf{q}$, if the new disparity d^* of $\bar{\mathbf{p}}$ with respect to $\mathscr{S}_\mathbf{q}$ is between d_{max} and d_{min}. We update the quadric of $\mathbf{p}$ to $\mathscr{S}_\mathbf{q}$ if the aggregated cost gets reduced by $\mathscr{S}_\mathbf{q}$. (b) and (c) show the direction of spatial propagation for odd and even iterations. The four immediate neighbours of $\bar{\mathbf{p}}$ are denoted by $\bar{\mathbf{p}}_\ell$, $\bar{\mathbf{p}}_r$, $\bar{\mathbf{p}}_u$, $\bar{\mathbf{p}}_d$ (left, right, upper and lower). In even iterations we consider the left and upper neighbours as spatial neighbours, whereas in odd iterations the right and lower neighbours are verified. During even iterations, the DSP weight for $\bar{\mathbf{p}}$ is chosen by comparing the disparity dissimilarity with $\bar{\mathbf{p}}_\ell$ and $\bar{\mathbf{p}}_u$. The DSP weight for $\bar{\mathbf{p}}_\ell$ is chosen by comparing with the blue and the green pixel, whereas for $\bar{\mathbf{p}}_u$, the green and the red pixels are compared. During odd iteratios, the DSP weight for $\bar{\mathbf{p}}$, $\bar{\mathbf{p}}_r$ and $\bar{\mathbf{p}}_d$ are chosen by comparing the disparity dissimilarity of $\bar{\mathbf{p}}_r$ and $\bar{\mathbf{p}}_d$, blue and green, green and red pixels, respectively.



(a) View propagation (b) Change of quadric across views

Figure 4.11: (a) Let $\{\bar{\mathbf{r}}', \bar{\mathbf{s}}'\} \in I'$ be two possible matching points of $\bar{\mathbf{p}} \in I$. Points $\mathbf{p}$, $\mathbf{r}$ and $\mathbf{s}$ are corresponding points of $\bar{\mathbf{p}}$, $\bar{\mathbf{r}}'$, $\bar{\mathbf{s}}'$ in the disparity space lying on the quadric $\mathscr{S}_\mathbf{p}$, $\mathscr{S}_\mathbf{r}$ and $\mathscr{S}_\mathbf{s}$ with disparity d, $d_\mathbf{r}$ and $d_\mathbf{s}$, respectively. We transfer the disparity and the support quadric of $\bar{\mathbf{r}}'$ and $\bar{\mathbf{s}}'$ to $\bar{\mathbf{p}}$ and find the new surface parameters. If the new quadric minimises the aggregated cost of the patch $\mathcal{W}(\bar{\mathbf{p}})$ centred at $\bar{\mathbf{p}}$, we update the surface parameters of $\bar{\mathbf{p}}$ with the new one. (b) As corresponding pixels have equal disparity, we transform the quadric accordingly using $\mathbf{V}_{\text{RL}}$ (eq. 4.13) or $\mathbf{V}_{\text{LR}}$ (eq. 4.12) and then match the disparities. In the figure, $\mathscr{S}_\mathbf{r}'$ is the transformed quadric of $\mathscr{S}_\mathbf{r}$ with respect to the reference image view point and, $\mathbf{p}_\mathbf{r}$ is the projection of $\bar{\mathbf{p}}$ in the disparity space with respect to $\mathscr{S}_\mathbf{r}'$. Note that, the supporting quadric is no longer a Monge patch.

quadric is residing, and later match the disparities.

4.6 Constrained optimisation

We locally refine the quadric at each pixel to further reduce the matching cost. Here we change the surface parameters within bounds and seek for an optimum surface. As our cost function is non-differentiable at some points due to the presence of discontinuous thresholds in the pixel dissimilarity function, we cannot use standard gradient descent methods to minimise it. It is also not convenient to mathematically compute the derivatives. We tackle this problem by using the bound optimisation by quadratic approximation (BOBYQA) (Sec. 3.4) algorithm to optimise the surface parameters similarly [1]. The quadric parameters of each pixel obtained from the view propagation are used as initial inputs. The optimiser then minimises cost($\bar{\mathbf{p}}, \mathscr{S}$) using the constraints mentioned later in this section.

The optimisation is performed over seven parameters; $\{ a, b, c, \alpha, \beta, \gamma, k \}$ where $\{ a, b, c \} \in \mathscr{S}$ and, α, β and γ are the rotation angles around the x, y and, d axis, respectively. The curvature of the surface is governed by $\{ a, b, c \}$. The surface normal is controlled by the angles α, β and γ. The constant term $k \in \mathscr{S}$ translates the quadric to a new point in the disparity space.

4.6.1 Quadric initialisation

We start the optimisation process from the second iteration onwards to make sure the plane parameters have propagated in either direction of the stereo pair. During quadric initialisation in the first optimisation process, We begin with a canonical quadric (centred at the origin and axis aligned with the Euclidean geometry). The canonical quadric is then aligned with the perturbed surface normal and translated back to the new disparity point. The principal curvature constraint (Sec. 4.6.2) and the principal direction constraint (Sec. 4.6.3) are used to generate the canonical quadric and the perturbed surface normal, respectively. The disparity bound constraint (Sec. 4.6.4) is used to translate the quadric to a new point in the disparity space. The surface normal constraint (Sec. 4.6.5) are used to check whether the perturbed normals are geometrically feasible.

In the later optimisation process, we already have a quadric attached to every point in the disparity space. In that case, we perturb the quadric parameters within bounds and seek for an optimal quadric.

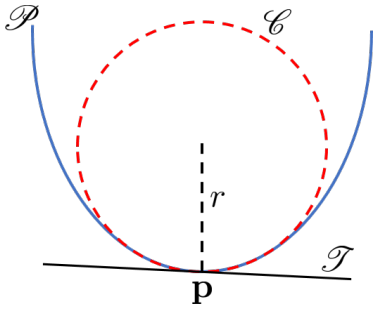


Figure 4.12: The circle $\mathscr{C}$ is the osculating circle of the curve $\mathscr{P}$ at the point $\mathbf{p}$. Both $\mathscr{C}$ and $\mathscr{P}$ have the same tangent $\mathscr{T}$ as well as the curvature at $\mathbf{p}$. The radius of curvature of $\mathscr{P}$ is given by $\frac{1}{r}$, where r is the radius of $\mathscr{C}$.

4.6.2 Principal curvature constraint

The osculating circle of a curve $\mathscr{P}$ at a point $\mathbf{p}$ is the circle $\mathscr{C}$ that has the same tangent as well as the curvature at $\mathbf{p}$ (Fig. 4.12). Similar to the tangent line $\mathscr{T}$ at $\mathbf{p}$, which is the best linear approximation of $\mathscr{P}$ at $\mathbf{p}$, $\mathscr{C}$ is the best circle that approximates $\mathscr{P}$ at $\mathbf{p}$. The radius of curvature of $\mathscr{P}$ is given by $\frac{1}{r}$, where r is the radius of $\mathscr{C}$. Therefore, the curvature of any curve at a point can be approximated by its osculating circle at the same point. We use this idea to find out the constraints on a, b and c of $\mathscr{S}$.

Let $\mathscr{P}_x$ be the curve, when the quadric $\mathscr{S}$ is sliced at $\mathbf{p} = (x, y, d)$ along the xd plane. We will find the bounds on 'a' by finding the possible osculating circles of $\mathscr{P}_x$ at $\mathbf{p}_x = (x, d)$, where $\mathbf{p}_x$ is the projection of $\mathbf{p}$ along the xd plane. A circle is uniquely defined by three points. We already have a point $\mathbf{p}_x$. The other two points are chosen based on the patch size and the disparity bound. For a patch of size $2r$, we get two sets of point pairs generating maximum and minimum curvature. Let $\kappa_1^\star$ be the radius of curvature of the circle $\mathscr{C}_1^\star$ passing through the three points

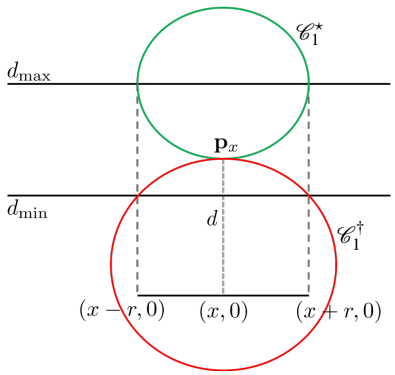


Figure 4.13: Curvature bounds by the osculating circles. The circle $\mathscr{C}_1^\star$ passes through the three points $\{(x-r, d_{\max}), (x, d), (x+r, d_{\max})\}$. The circle $\mathscr{C}_1^\dagger$ passes through the three points $\{(y-r, d_{\max}), (y, d), (y+r, d_{\max})\}$. The bound on '$a$' is obtained from the radius of curvature of $\mathscr{C}_1^\star$ and $\mathscr{C}_1^\dagger$.

$\{(x-r,d_{\max}),(x,d),(x+r,d_{\max})\}$ (Fig. 4.13). Similarly, let $\kappa_1^{\dagger}$ be the radius of curvature of the circle $\mathscr{C}_1^{\dagger}$ passing through the three points $\{(x-r,d_{\min}),(x,d),(x+r,d_{\min})\}$. Therefore, '$a$' is bounded by

$$-\kappa_1^{\dagger} \leq a \leq \kappa_1^{\star} . \tag{4.16}$$

We find the bound on 'b' in a similar fashion. Let $\mathscr{P}_y$ be the curve, when $\mathscr{S}$ is sliced at $\mathbf{p}$ along the yd plane. The bounds on 'b' of $\mathscr{P}_y$ can be approximated by the possible osculating circles passing through $\mathbf{p}_y = (y,d)$. Let $\kappa_2^{\star}$ be the radius of curvature of the circle $\mathscr{C}_2^{\star}$ passing through the three points $\{(y-r,d_{\max}),(y,d),(y+r,d_{\max})\}$. Similarly, let $\kappa_2^{\dagger}$ be the radius of curvature of the circle $\mathscr{C}_2^{\dagger}$ passing through the three points $\{(y-r,d_{\min}),(y,d),(y+r,d_{\min})\}$. Therefore, '$b$' is bounded by

$$-\kappa_2^{\dagger} \leq b \leq \kappa_2^{\star} . \tag{4.17}$$

Finally, the bound on c is given by:

$$-\max\{\kappa_1^{\dagger},\kappa_2^{\dagger}\} \leq c \leq \max\{\kappa_1^{\star},\kappa_2^{\star}\} . \tag{4.18}$$

4.6.3 Principal direction constraint

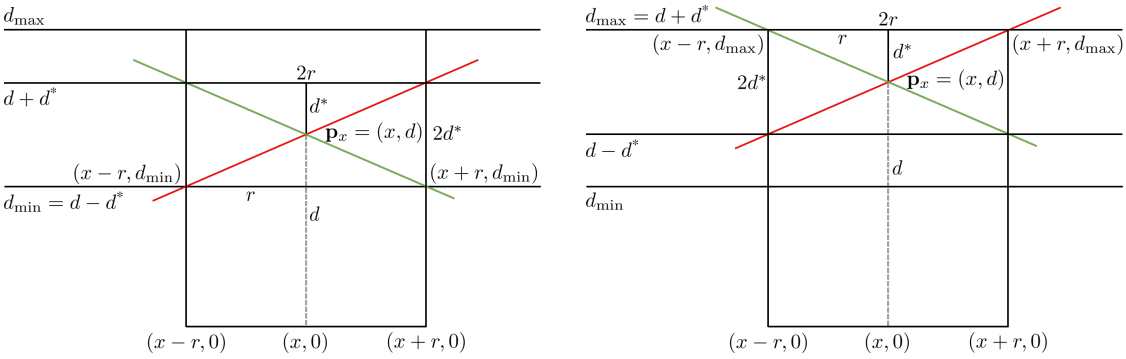
We use similar surface normal constraints as mentioned in [1] to find out the bounds on (α,β,γ), where α, β and γ are the rotation angle around the x, y and, d axis respectively. Slope of the tangent line at $\mathbf{p}_x = (x,d)$ in the xd plane is bounded by

$$-\frac{d^*}{r} \leq \tan(\alpha) \leq \frac{d^*}{r}, \tag{4.19}$$

where $d^* = \min(d-d_{\min}, d_{\max}-d)$ (Fig. 4.14). Similarly, slope of the tangent line at $\mathbf{p}_y = (y,d)$ in the yd plane is bounded by

$$-\frac{d^*}{r} \leq \tan(\beta) \leq \frac{d^*}{r}, \tag{4.20}$$

The rotation angle θ around the disparity axis is restricted from $-\frac{\pi}{2}$ to $\frac{\pi}{2}$.



(a) $d^* = d - d_{\min}$ (b) $d^* = d_{\max} - d$

Figure 4.14: Principal direction constraint. In the xd plane both the red and green lines (originated from $\mathbf{p}_x$) are extreme lines that follow the disparity bound constraint. Slopes of the red and the green line are d^*/r and $-d^*/r$, respectively, where $d^* = \min(d - d_{\min}, d_{\max} - d)$. Patch size is given by $2r$. Any line whose slope is between $-d^*/r$ and d^*/r is a potential candidate plane.

4.6.4 Disparity constraint

The bounds on $k \in \mathscr{S}$ are given by

$$-\{d_{\max}(cd_{\max} + 2B) + A\} \leq k \leq -\{d_{\min}(cd_{\min} + 2B) + A\} \tag{4.21}$$

where $\mathscr{S}$ is the support region of $\bar{\mathbf{p}} = (x, y)$, $A = ax^2 + by^2 + 2fxy + 2ux + 2vy$ and $B = gx + hy + w$.

4.6.5 Surface normal constraint

The unit surface normal $\mathbf{n}$ (App. B)of $\mathscr{S}$ at $\mathbf{x}$ is given by:

$$\mathbf{n} = (u, v, w) = \frac{1}{\sqrt{1 + \left(\frac{\partial d}{\partial x}\right)^2 + \left(\frac{\partial d}{\partial y}\right)^2}} \left(-\frac{\partial d}{\partial x}, -\frac{\partial d}{\partial y}, 1\right), \tag{4.22}$$

where,

$$\begin{aligned} \frac{\partial d}{\partial x} &= -\frac{ax + fy + gd + u}{cd + gx + hy + w}, \\ \frac{\partial d}{\partial y} &= -\frac{by + fx + hd + v}{cd + gx + hy + w}. \end{aligned} \tag{4.23}$$

We consider w positive, which follows from the visibility constraint of [1]. If $\mathscr{S} \in \mathcal{D}_{\mathrm{L}}$, u and v

are bounded by

$$-\frac{d^*}{r} \leq \frac{u}{w} \leq \frac{d^*}{r}, \quad -\frac{d^*}{r} \leq \frac{v}{w} \leq \frac{d^*}{r}, \tag{4.24}$$

where $d^* = \min(d - d_{\min}, d_{\max} - d)$ (Fig. 4.14). Applying similar disparity bound constraint for the other view, we find the analogous representation of inequality 4.24 as:

$$-\frac{d^*}{r} \leq \frac{u}{w+u} \leq \frac{d^*}{r}\,. \tag{4.25}$$

If $\mathscr{S} \in \mathcal{D}_{\mathrm{R}}$, the equivalent representation of inequality 4.24 for the other view changes to:

$$-\frac{d^*}{r} \leq \frac{u}{w-u} \leq \frac{d^*}{r}\,. \tag{4.26}$$

4.7 Local shape measures

The Gaussian and the mean curvature (App. B.1.5) at a point cannot fully characterise the local shape of a surface. However, the two principal curvatures (App. B.1.4) are far more informative. Koenderink *et al.* [51] proposed two novel measures of local shape, the 'curvedness' and the 'shape index'. The curvedness is a positive number that specifies the amount of curvature. The shape index is scale invariant and is in the range $[-1,+1]$. The shape index captures the intuitive notion of the local shape.

Let κ_1 and κ_2 be the principal curvatures of a surface at some point, where $\kappa_1 \geq \kappa_2$. The shape index s is defined as:

$$\mathrm{s} = \frac{2}{\pi}\arctan\left(\frac{\kappa_2+\kappa_1}{\kappa_2-\kappa_1}\right) \in [-1,+1], \quad \kappa_1 \geq \kappa_2\,. \tag{4.27}$$

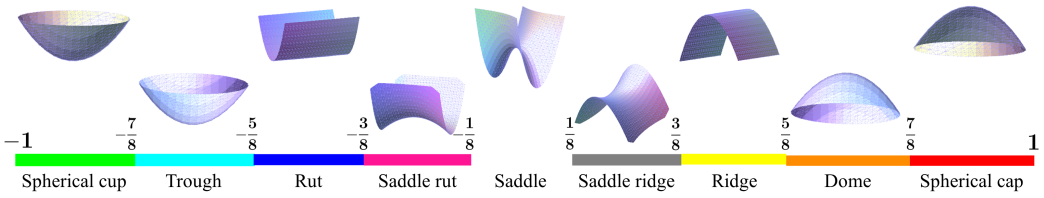


Figure 4.15: Shape categories based on shape index with outward normal pointing upwards. Figure reproduced from [51].

Table 4.2: Shape categories. Table reproduced from [51].

Mnemonic	Shape index range	Colour
Spherical cup	$\left[-1,-\frac{7}{8}\right)$	Green
Trough	$\left[-\frac{7}{8},-\frac{5}{8}\right)$	Cyan
Rut	$\left[-\frac{5}{8},-\frac{3}{8}\right)$	Blue
Saddle rut	$\left[-\frac{3}{8},-\frac{1}{8}\right)$	Pink
Saddle	$\left[-\frac{1}{8},\frac{1}{8}\right)$	White
Saddle ridge	$\left[\frac{1}{8},\frac{3}{8}\right)$	Grey
Ridge	$\left[\frac{3}{8},\frac{5}{8}\right)$	Yellow
Dome	$\left[\frac{5}{8},\frac{7}{8}\right)$	Orange
Spherical cap	$\left[\frac{7}{8},1\right]$	Red

The shape index scale is divided into nine categories: spherical cup, trough, rut, saddle rut, saddle, saddle ridge, ridge, dome and spherical cap. The categories are shown in Table 4.2 and the surfaces are shown in Fig. 4.15. It is clear from Fig. 4.15 that two shapes for which the shape index only differs by the sign signs represent complimentary pairs. Convexities and concavities find their places on opposite sides of the shape scale. These basic shapes are separated by saddle like objects which are neither convex nor concave.

The curvedness is defined as:

$$\mathrm{c} = \sqrt{\frac{{\kappa_1}^2 + {\kappa_2}^2}{2}} \in [0,\infty) \ . \tag{4.28}$$

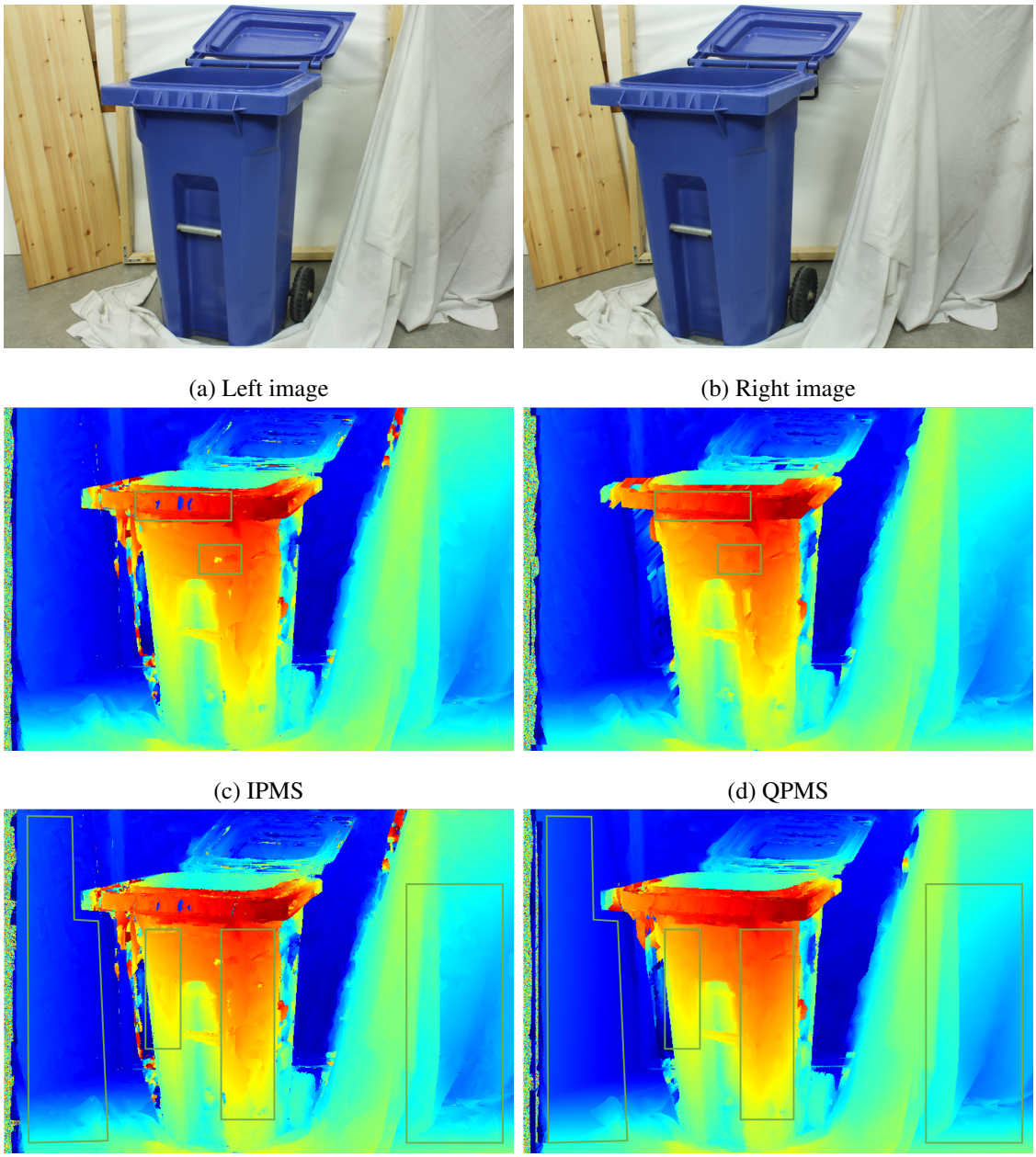
The curvedness measure is coordinate independent and scales inversely with the size. It vanishes only at planar point.

4.8 Results

4.8.1 Experimental set-up

We evaluate our proposed method, *QPMS*, using the quarter resolution (720×480) Recycle stereo pair from the Middlebury stereo benchmark, version 3 [76]. The minimum and maximum disparity of the Recycle stereo pair is 8 and 57 pixels, respectively. We also tested the proposed method on our own "poster" stereo pair (759×299). The poster stereo pair was created by reshaping a poster as a cylinder where the whole surface has constant curvature. The minimum and maximum disparity of the Poster stereo pair are 10 and 150 pixels, respectively.

There are six parameters in *QPMS*. We use the same parameters of *IPMS* for *QPMS*. The



(a) Left image (b) Right image

(c) IPMS (d) QPMS

(e) IPMS (f) QPMS

Figure 4.16: Disparity map comparison between *IPMS* and *QPMS* on the Recycle stereo pair. (c), (d): Disparity guided second spatial propagation. (e), (f): First constrained optimisation.

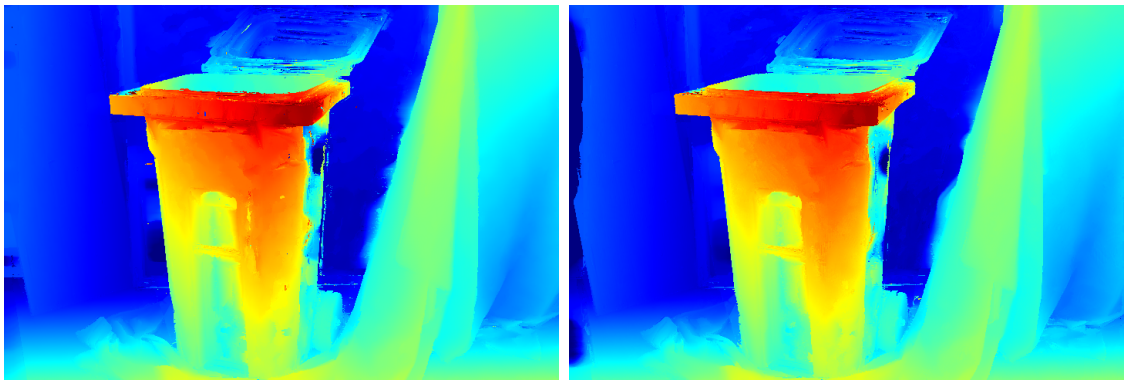
parameters were unchanged during the experiment for both stereo pairs. We keep the patch size $\mathcal{W}(\bar{\mathbf{p}}) = 23 \times 23$ and use a 5×5 median filter on the final disparity map during post-processing. The results are reported after the post-processing step unless otherwise stated. As a performance measure for the Recycle stereo pair, we use the default metrics of the Middlebury stereo benchmark and follow the same performance measure as *IPMS*. For the Poster stereo pair, we only present qualitative results. We use the jet colour-map to visualise the disparity maps where the

Table 4.3: Percentage of bad pixels with 2.0 pixels error on non occluded pixels for the Recycle stereo pair.

Iteration	Spatial propagation		View propagation		Refinement	
	IPMS	*QPMS*	*IPMS*	*QPMS*	*IPMS*	*QPMS*
1	8.29	8.29	6.04	6.04	-	-
2	6.29	**5.97**	6.28	**5.61**	6.16	**5.58**
3	6.08	**5.51**	6.04	**5.42**	6.02	**5.4**

Table 4.4: Average error with 2.0 pixels error on non occluded pixels for the Recycle stereo pair.

Iteration	Spatial propagation		View propagation		Refinement	
	IPMS	*QPMS*	*IPMS*	*QPMS*	*IPMS*	*QPMS*
1	1.05	1.05	0.75	0.75	-	-
2	0.67	**0.63**	0.66	**0.58**	0.64	**0.57**
3	0.63	**0.57**	0.62	**0.56**	0.62	**0.56**



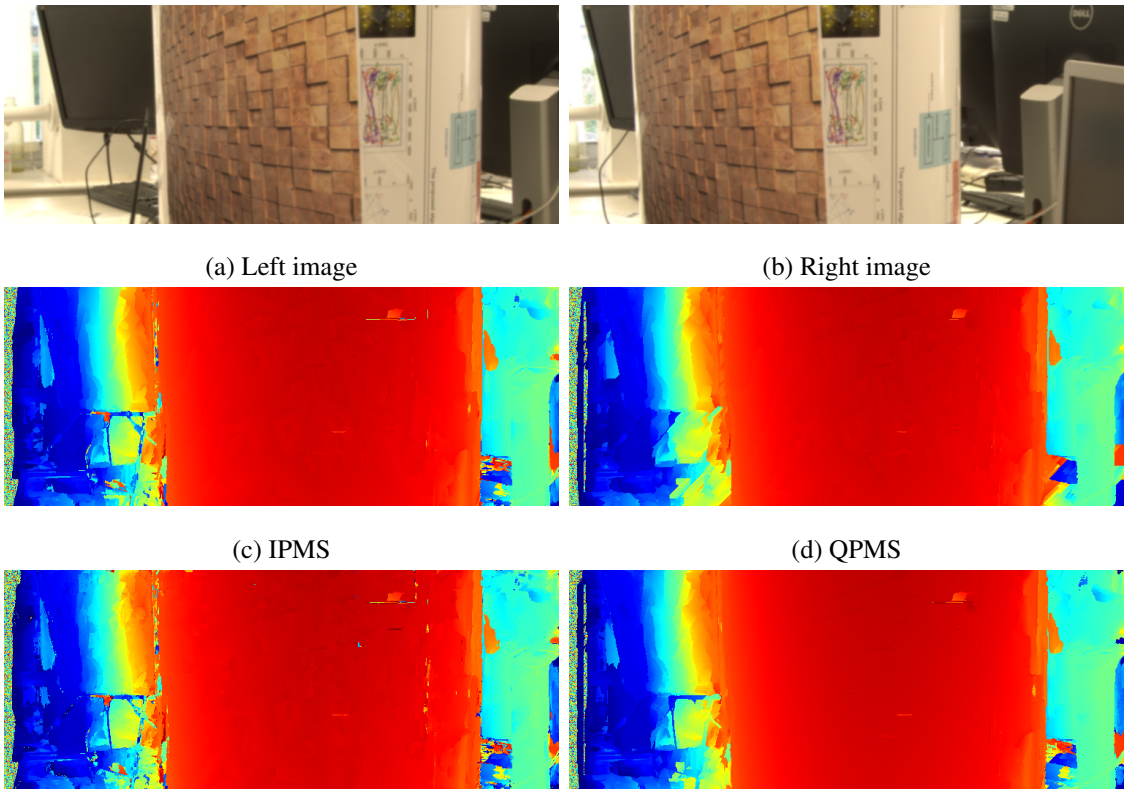
(a) IPMS (b) QPMS

Figure 4.17: Final disparity map of the Recycle stereo pair.

disparity scale changes from blue (min disparity) to red (max disparity).

4.8.2 Comparison with IPMS

We first compare our results with *IPMS*. For the Recycle stereo pair, Fig 4.16d shows the benefit of using the disparity guided spatial propagation, where the neighbours filled the bad matches with good planar support. One of the disadvantages of the spatial guided propagation is sometimes the depth discontinuities are not preserved. For this reason, we use it in the early stage of iteration and only use it once. The later iterations then fix the depth discontinuity. The advantage of the quadric disparity model is visible in the first constrained optimisation (Fig 4.16f). The disparity map is smooth along the areas with curved surfaces. Experimental results show that proposed *QPMS* produces 10% fewer bad pixels (Table 4.3) and 7% less average error (Table 4.4) on the final disparity map (Fig 4.17) for a error threshold of 2 pixels on non occluded pixels

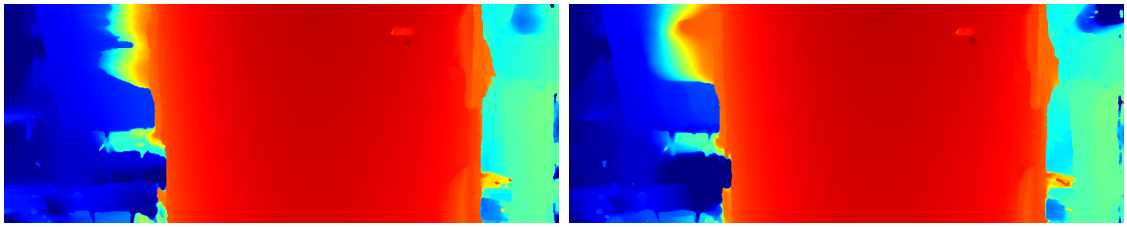


(a) Left image (b) Right image

(c) IPMS (d) QPMS

(e) IPMS (f) QPMS

Figure 4.18: Disparity map comparison between *IPMS* and *QPMS* on the poster stereo pair. (c), (d): Disparity guided second spatial propagation. (e), (f): First constrained optimisation.



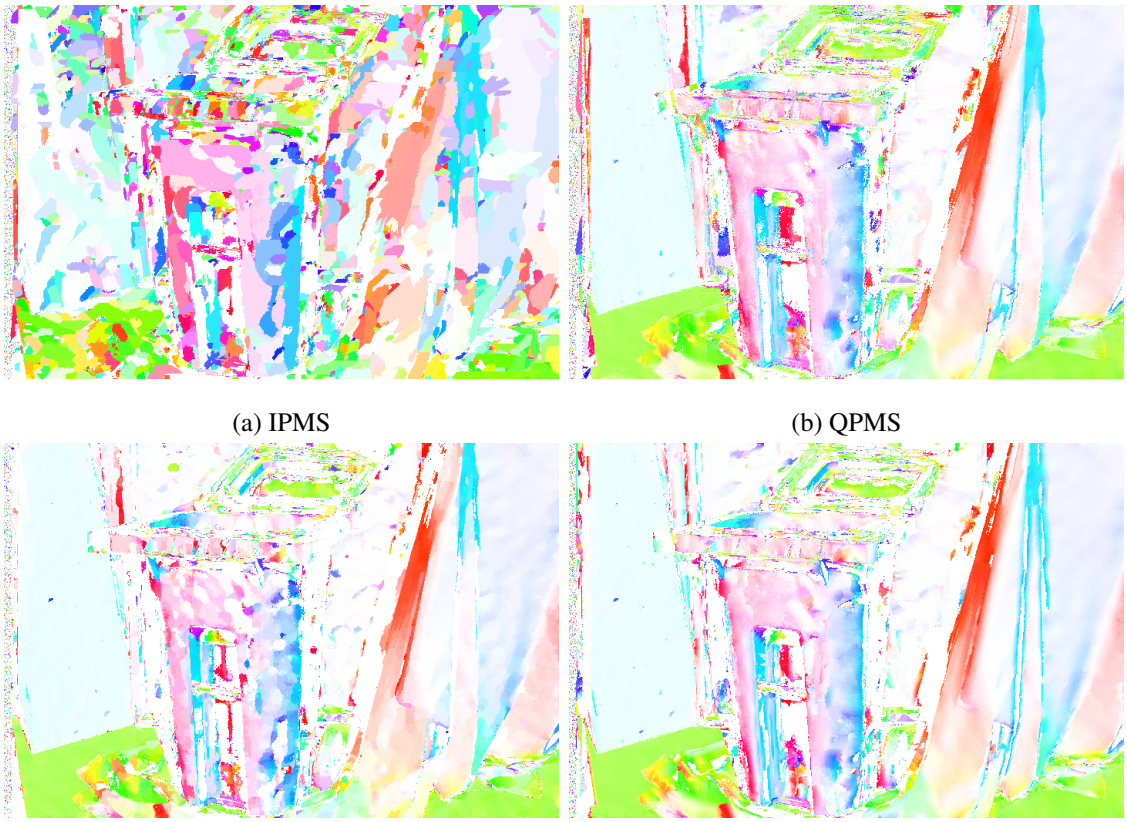
(a) IPMS (b) QPMS

Figure 4.19: Final disparity map of the poster stereo pair.

for the Recycle stereo pair.

For the Poster stereo pair the effect of the disparity guided spatial propagation (Fig. 4.18d) is not so clear as the surface is very smooth and there were no mismatches. However, the advantage of the quadric model is clear in the first constrained optimisation results (Fig. 4.18f). Fig. 4.19 show the final disparity map of the Poster stereo pair.

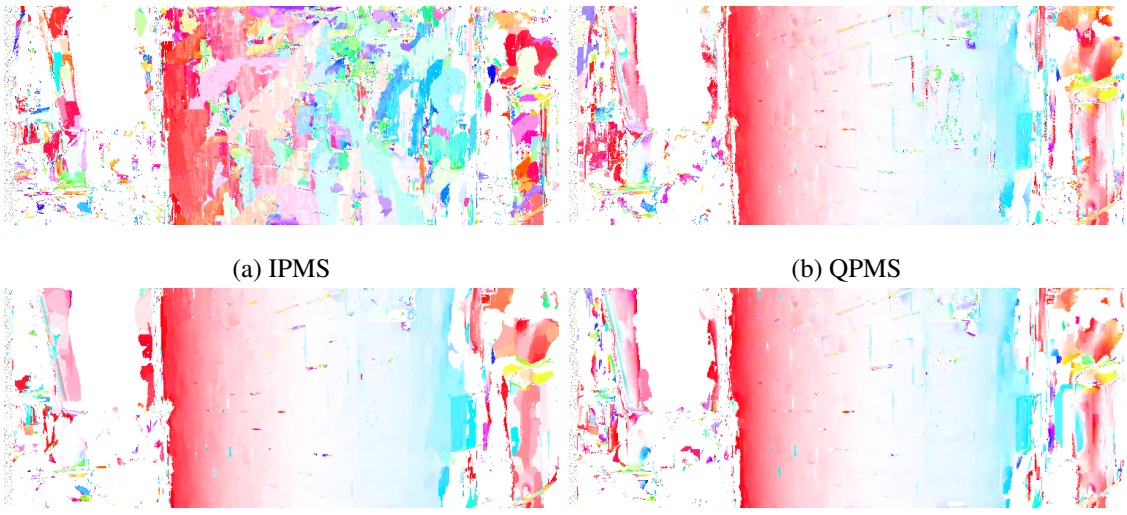
Fig. 4.20 and 4.21 shows the distribution of the surface normals for both algorithms. It is evident from the results that the distribution is more even for normal maps generated by *QPMS*.



(a) IPMS (b) QPMS

(c) IPMS (d) QPMS

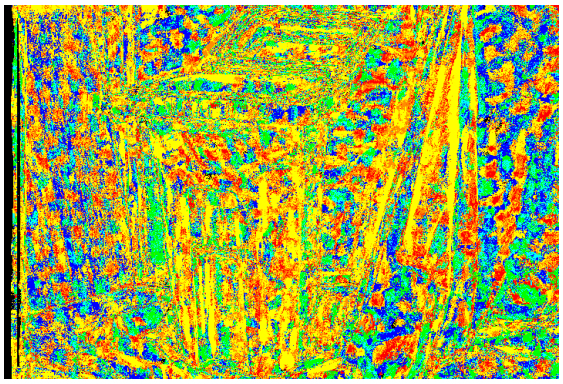
Figure 4.20: Distribution of surface normals generated by *IPMS* and *QPMS* on the Recycle stereo pair. (a), (b): First constrained optimisation. (c), (d): Second constrained optimisation.



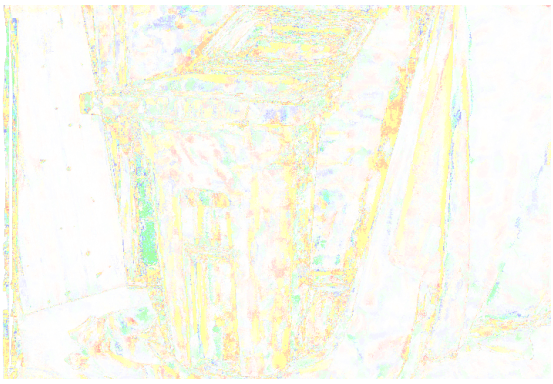
(a) IPMS (b) QPMS

(c) IPMS (d) QPMS

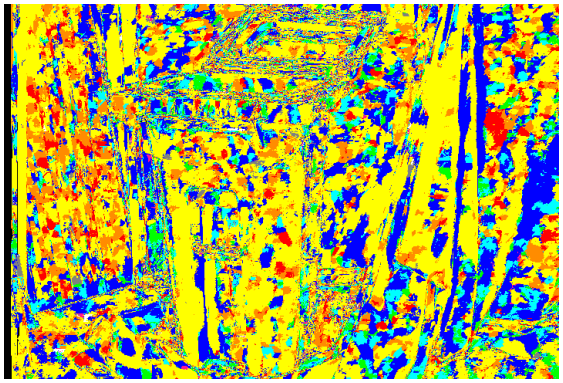
Figure 4.21: Distribution of surface normals generated by *IPMS* and *QPMS* on the Poster stereo pair. (a), (b): First constrained optimisation. (c), (d): Second constrained optimisation.



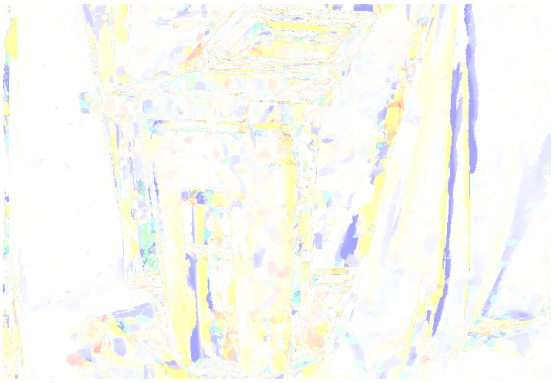
(a) First constrained optimisation



(b) First constrained optimisation

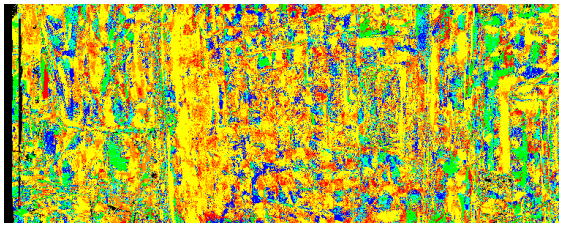


(c) Second constrained optimisation

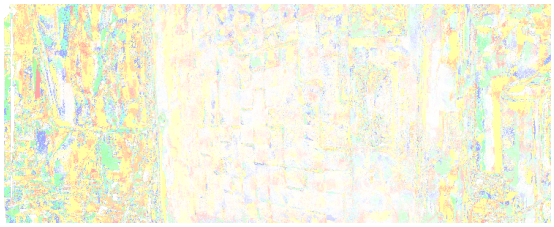


(d) Second constrained optimisation

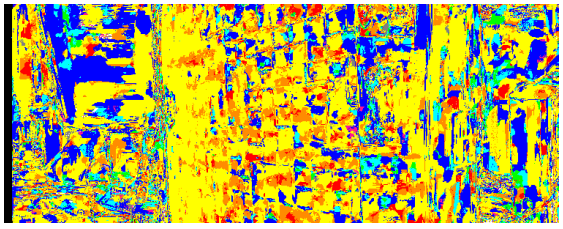
Figure 4.22: Shape index for the Recycle dataset. (a), (c): Shape index. (b), (d): Transparency set by curvedness measure for shape index.



(a) First constrained optimisation



(b) First constrained optimisation



(c) Second constrained optimisation

(d) Second constrained optimisation

Figure 4.23: Shape index for the Poster dataset. (a), (c): Shape index. (b), (d): Transparency set by curvedness measure for shape index.

4.8.3 Curvature analysis

The shape index for the stereo pairs are shown in Fig 4.22 and 4.23. We also used the curvedness measure to set the transparency of the shape index as the shape index does not specify the

amount of curvature. When the curvature is small, we can represent a surface locally by a shape with opposite curvature, *e.g.,* if we squash a spherical cap, it can represent a surface which is a spherical cup in nature. We see this effect in shape index results. However, the transparency is set by the curvedness measure; it is evident from the result that the algorithm can reconstruct the correct surfaces locally.

4.9 Summary

In addition to surface orientation, curvature information is needed for a full understanding of the local surface structure. The PatchMatch surface model is planar and does not directly estimate the local curvature. We proposed a new framework, which is based on a quadric disparity model, estimates local quadric at each point in the disparity space. We also introduce curvature constraint on the model, which ensures the association of feasible quadric. In addition, to tackle false matching, we propose a disparity guided spatial propagation, where a non-linear disparity dissimilarity function weights the aggregated cost. Disparity guided spatial propagation (DSP) prevents false matches from growing and also fill them with the correct disparity value, provided there is at least one good surface approximation of the neighbours. Initial results show that the modifications help our method to generate better disparity maps than *IPMS*. Moreover, the framework reduces to *IPMS* when the quadratic terms are set to be zero and, can be applied to any local stereo algorithm that can be cast in the PatchMatch stereo framework.

Chapter 5

Riverbed dataset

5.1 Introduction

The riverbed dataset was captured to encapsulate the evolution of a simulated riverbed. We also used it to validate the propositions in a relevant scenario. It contains stereo images along with ground truth depth information captured by a terrestrial laser scanner. The dataset is challenging for conventional stereo matching algorithms, because the visible surface consists of sand, which lacks large-scale image features. The primary objective was to create 3D topographic models at high resolution from the stereo pairs that can encapsulate the topographic changes in riverbed morphology over time and link the topographic changes to topological changes in the braided channel network.

5.2 Riverbed set-up

We used a river simulator to simulate the changes of a riverbed over time. The simulator was 2.15 m long and 0.88 m wide at the height of 0.13 m. We flow the water over the riverbed for the first 20 min and leave it to dry for the following 10 min. Stereo pairs of the dry riverbed are then captured from an overhead camera. Disparity maps are then generated using the proposed algorithms. Later, we register and interpolate the disparity maps to a common reference frame to find the change of depth over time in a continuous domain.

The slope of the flume is an important parameter which ensures enough energy to support high sediment transportation. In our experiment, the slope was kept at 1.15° (2%). The amplitude

of the topography is likely to remain mostly at millimetre scale, though the greatest topographic difference in the active area we are interested in from a geomorphic perspective may stray just into cm scales, but is very likely to remain less than $2-3$ cm.

The initial set-up time to go from a straight channel to a braided river platform took 4 hours. From this point, we allowed the flume to dry down overnight. From next day onwards, we worked directly on the pre-formed morphology which saved a lot of time on set-up.

5.2.1 Water flow speed

The speed of water in the flume is hard to measure, owing to the very shallow water depth. We initialised the discharge around 8 l/min and kept it likely in the region of 10 l/min. To simulate sporadic floods, we increased the release to 12 l/min.

5.2.2 Lights

The wet sand produces reflection and glaring effects while traditional light builds are used to capture the stereo pairs. To reduce these problems, a professional continuous softbox lighting were used for the experiment (Fig. 5.1), which mimics a cloudy day in a real outdoor scene. Three lighting stands were fixed at the three corners throughout the experiment. The stands were 2 m high and five 38 W pure daylight bulbs were attached with every stand. A softbox of size 50×70 cm^2 was attached around the bulbs, to create a natural diffused light with the three lighting kit.

Figure 5.1: Riverbed set-up

Figure 5.2: camera set-up

5.2.3 Stereo pair

We captured 39 stereo pairs using a Nikon *D*7100 camera with a $18-55$ mm lenses (Fig. 5.2) at an interval of 30 mins. The camera was mounted on top of the riverbed in a moving rail. We move the camera horizontally along the rail to capture the stereo pair with an approximated baseline distance of 12 cm.

A regular chessboard method was used to find the camera intrinsics using an OpenCV routine (Sec. 2.2.4). The chessboard had 9×7 inner corners and each side of the small squares measures 36 mm. Calibrations were performed thrice a day; morning, noon and afternoon. We found a little deviation among the camera intrinsics as expected.

The camera images are saved in Nikon's raw NEF format. NEF (Nikon Electronic Format) images are converted to png using "dcraw". The NEF image format stores only one value per pixel (either R, G or B) whereas the png images interpolate each pixel values and assigns RGB values to each of them. The size of a NEF image is around 25 MB, but after converting to png, the image size becomes 32 MB. The dimension of each image is 6036×3388 px. Due to memory limitation, we down-sample the images to 1509×847 px for stereo matching.

5.2.4 Ground truth depth

The ground truth depth of the riverbed were collected using a factory calibrated terrestrial laser scanner [28] (Fig. 5.3) from three different positions. The three scans were then merged into one by considering a common reference frame obtained by a total station [29] and fixed markers attached to the walls. Due to time limitations, we captured 15 sets of laser scans, where each set

Figure 5.3: TLS set-up. The markers on the wall are used to merge three scans taken from different view points.

contains scans from three different position.

The position accuracy of the TLS is the combination of range and angular accuracy. It forms a sphere of possible error for the measured point. Given the scanner is calibrated, each location within 5m range from the scanner will be correct within a 5 mm sphere of the coordinate registered.

The laser scans are stored in the Zettabyte File System (ZFS) format. We convert the ZFS data to a flat ASCII (x, y, z, intensity) format using 'Cyclone' [27] software. TLS intensity is the backscattered optical power received by the sensor detector which normally has a non-linear response curve. The intensity values are in the range $[-2047, +2048]$ which is later normalized to $[0, 255]$. The PCL viewer follows the colour scheme based on the normalised intensity values, where 0 is red, and 255 is blue.

5.2.5 Amount of sand added after each capture

Volumetric quantification is hard for wet sediment as we recycle the sediment washed out of the flume. We used a volumetric beaker for a crude estimation of the wet sediment. At every 10 min, we added a predefined amount of wet sediment (~ 730 g) without considering the flume speed.

Table 5.1: Hydraulic parameters

Slope	Base-level (cm)	Discharge (l/min)		Deposition (g / 20 min)	Feed (cm^3 / 10 min)
1.15°(2%)	1.5	Baseflow	10	$\sim$ 1186	$\sim$ 730
		Flood	12	$\sim$ 1405	

5.2.6 Flume set-up

The experiment began from a straight channel of relatively uniform dimensions cut into the levelled river-bed. The hydraulic parameters used in our experiment are listed in Table 5.1. After setting flume slope and base-level, the flume was initially run with a discharge of 8 l/min. Sediments exported from the flume were collected and recirculated. Initially, around 200 cm^3 (measured using a volumetric beaker) of the exported sediment was added evenly across the water inlet. To monitor the development of the braided platform, total (B_t) and active (B_a) braiding indices were surveyed at every 20 min time interval (time step). The intensity of braiding is measured by B_t, calculated as an average of the number of active channels counted at ten cross-sections spaced evenly along the area of interest in the flume. The sediment transport activity is measured by B_a, calculated as an average of the number of flowing channels also transporting sediment at each cross-section. The area of interest in the present study began at 0.5 m from the flume inlet and ended at 1.5 m. This area was chosen to minimise the impact of the entrance and exit effects that result from the flow of water from a pump and pooling of water at the flume outlet. The flume was run until consecutive braiding index measurements stabilised, which took five-time steps.

The base flow and the sediment feed control the amplitude of the morphological units in the flume. Increasing both the amount of water (and thus energy) and sediment feed improve the amplitude of morphological units and aid surveying. Following the initial assessment from the TLS depth map, we set the base flow to 10 l/min. We fix the sediment feed volumetrically such that both the feed and the deposit are relatively the same. A rough quantification of the amount of deposit being collected every 10 min was found to be $\sim$ 730 cm^3 using the predefined base flow (Table 5.1). We used this volume as our sediment feed which kept the deposit relatively constant. To mimic a flood situation, we need to increase the base flow. It is important to set the flood discharge properly otherwise a higher flow increases the energy conditions in the flume to such proportions that sediment transport vastly outstrips the rate of sediment input, causing the river to entrench into one or two large main channels. Following some initial experiments,

the discharge was set to 12 l/min. A continuous increase of deposit during flood situation was noticed, indicating the rate of transport was higher than the sediment feed rate (Table 5.1).

Five sediment samples were collected at both base-flow and flood discharges to give a rough indication of the amount of material being transported out of the flume. Sediment washed from the flume were collected over 30 min time interval after the flume had been turned off and the water had been drained down. The samples were weighed to measure the amount of deposition once they were fully dry (Table 5.1).

The experiment was conducted for five consecutive days with the flume stopped and drained every 20 min to capture the stereo pair. Up to four TLS scans were taken each day, concurrently with stereo pairs, with a relatively even distribution of scans between flood and base-flow morphologies. A local coordinates system was set up, using a total station to record the position of the three targets around the flume, which were visible in all scans. These targets were used to register the point clouds obtained from the TLS and the stereo pair.

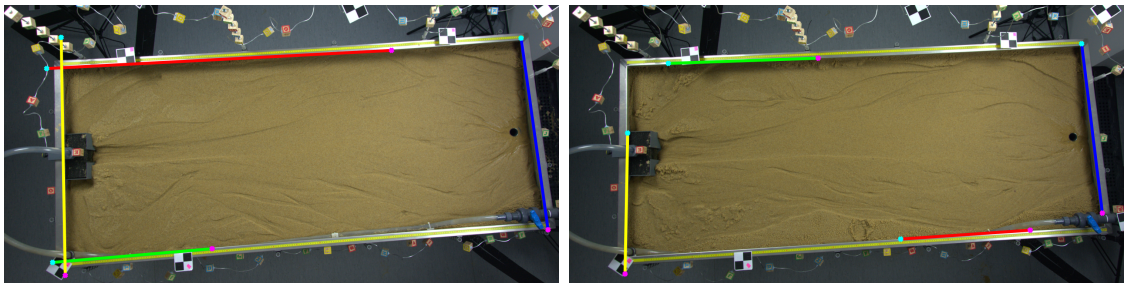
5.3 Post-processing

Some of the images were of a different dimension. We cropped those images to create a database of the same size. Later, we rescaled (25%) and rectified (Sec. 2.2.5) the images for stereo matching.

To merge the three TLS scans, we manually marked three markers in one scan and their corresponding markers in the other two scans. Using these point correspondences and the reference frame obtained from the total station, we merge all three scans into one.

5.3.1 Image registration

To find the change of depth across surfaces, we need to register the disparity maps with respect to a reference frame. We use the Hough transform [20] to find the four inner sides of the river bed table in both views. Later corresponding lines are matched manually, and a homography (Sec. 2.2.1) is computed across images from the virtual planes generated by those lines (Fig. 5.4). We refer the plane as the tabletop plane. The homography is then used to register the disparity maps. After the disparity maps are registered with respect to a fixed reference frame, we subtract each disparity map to find the change of surface over time.



(a) Reference image (b) Search image

Figure 5.4: Homography between the image pair using Hough transform. The lines along the inner-side of the table are manually matched to get the homography between the virtual planes. Same colour lines do not correspond to matching lines.

5.3.2 Disparity map visualisation

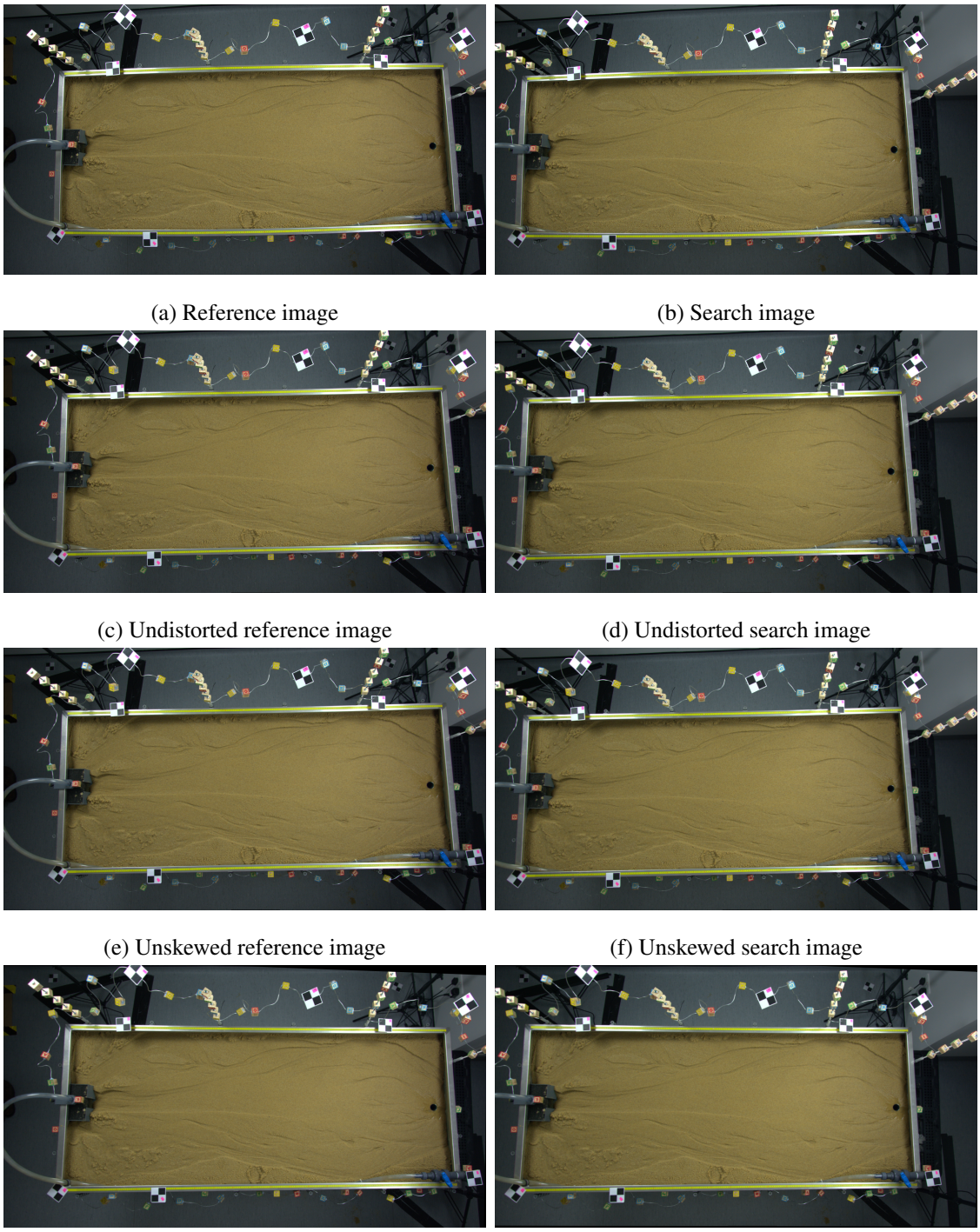
The change of depth in the region of interest is very subtle. Also, the depth varies over time. To be consistent with the disparity map visualisation, we first calculate the virtual tabletop plane using the inner side table edges generated by the Hough transform. As the virtual tabletop is fixed across images, we compute the difference between the virtual plane and the disparity surface and visualise the difference to be consistent over the change of surfaces.

5.3.3 Matching TLS and camera coordinates

The camera is mounted overhead whereas the TLS is placed on the ground over a tripod. Also, the image resolution greatly differs from the laser scans. Using the camera intrinsic and the disparity map, we first reconstruct the scene in 3D. Next, we manually match the scale of the point clouds generated by the TLS and the 3D scene by using the 'CloudCompare' [81] software. The relationship between the camera model and the TLS is inversely solved by fitting the two resultant points clouds using a rigid body transformation. ICP (iterative closest point) [78] algorithm is used to collapse one point cloud on another. We first manually match corresponding markers in the 3D model and the laser scan, which is later used to initialise the ICP algorithm. Manual matching and ICP are all done using CloudCompare software.

5.4 Results

We have generated the disparity maps of all the stereo pairs using IPMS. Also, each set of laser scans are merged. The disparity maps are registered with a reference frame to compute the change of surface over time. We have also matched the TLS and the camera coordinates. The



(a) Reference image (b) Search image

(c) Undistorted reference image (d) Undistorted search image

(e) Unskewed reference image (f) Unskewed search image

(g) Rectified reference image (h) Rectified search image

Figure 5.5: Stero pair generation

root mean square error (RMSE) between the two clouds is under one millimetre.

5.4.1 Stereo pair generation

The intrinsic camera matrix is used to project the disparity map in 3D. While finding the camera intrinsic from the chessboard images using a Matlab routine, we found that the camera axis is skewed. We also noticed that the OpenCV camera calibration routine does not consider any skew parameter in the camera model, which is later used to rectify the image pair. We solve the skewing problem by first undistorting the input image pair (Fig. 5.5) and later applying the inverse skew matrix on the undistorted images. Finally, the unskewed image pair is rectified by an OpenCV routine (Sec. 2.2.6), which is used as input to the stereo matching algorithm.

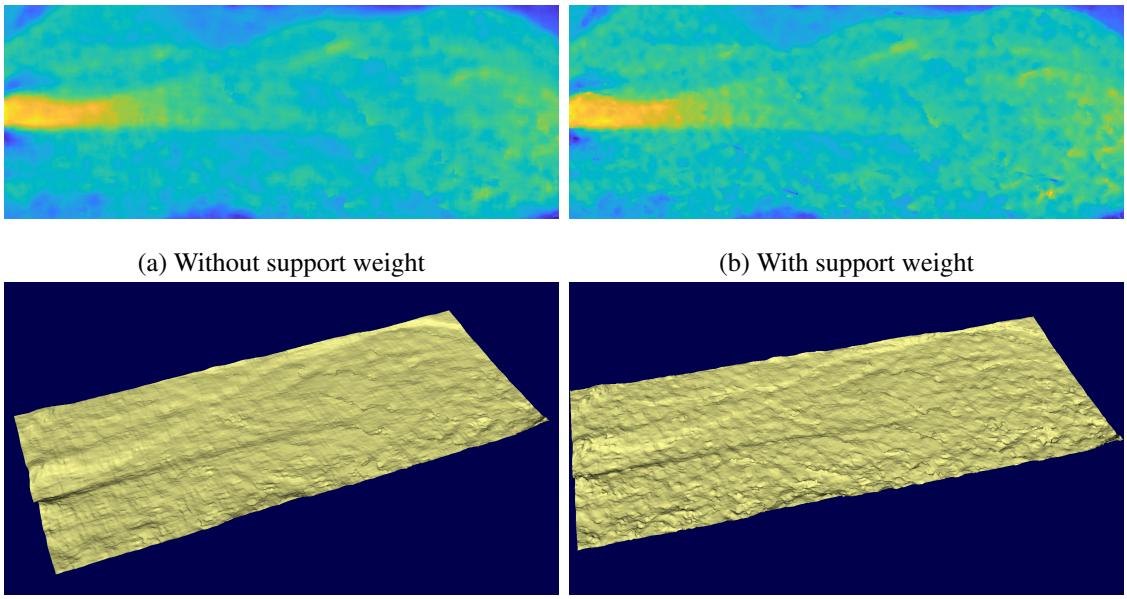
5.4.2 Disparity maps with different patch size with/without support weight

Fig. 5.7, 5.8, 5.9, 5.10, 5.11 represent the distance of the disparity map with respect to a fixed plane roughly passing through the table top surface of the cropped left image (Fig. 5.6) with/without support weight for different patch sizes, along with the 3D reconstruction. During this experiment, we observed that the disparity map is over-smoothed and can not preserve edge boundaries in the sand texture while support weight is not used. Comparing all the disparity maps with different patch size with/without support weight, we found that the optimal patch size is 71×71 px^2 and the support weight preserves the fine structures and edge boundaries in the sand images. We also observed the behaviour of the cost function on an arbitrary pixel with/without support weight using different patch size (Fig. 5.12).

5.4.3 Comparing laser scan with stereo depth map

We compared the disparity map of patch size 71×71 px^2 with a laser scan (Fig. 5.13) using the "Cloudcompare" software. The laser depth cloud is considered as the reference scan, and

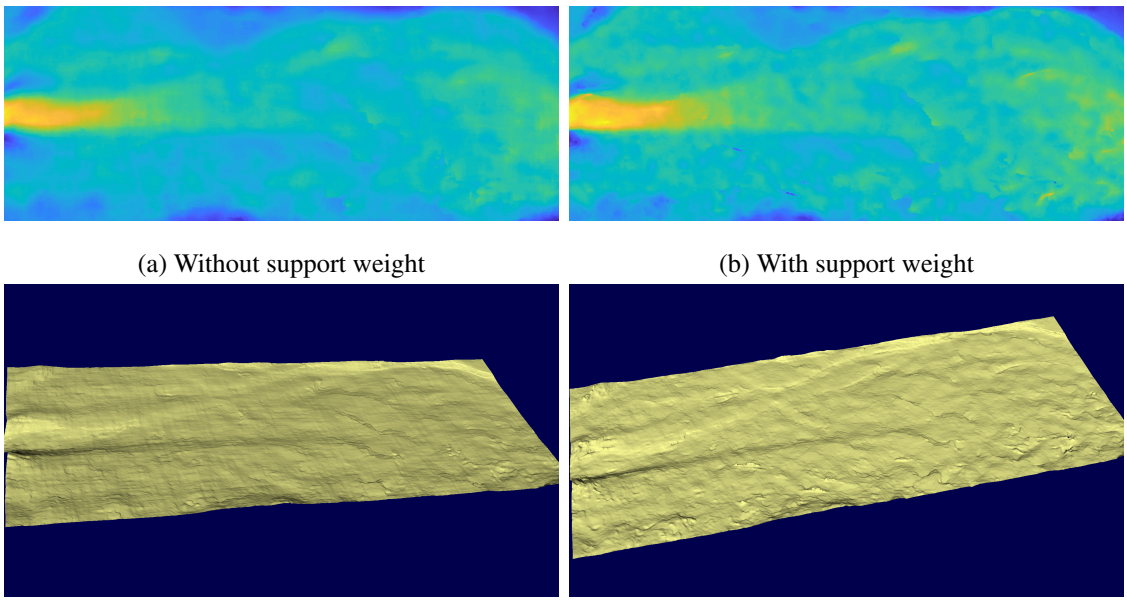
Figure 5.6: Cropped sand image with different patch sizes (in px unit) shown as squares; 21×21 (red), 31×31 (green), 41×41 (blue), 51×51 (black), 61×61 (magenta), 71×71 (yellow).



(a) Without support weight (b) With support weight

(c) 3D Reconstruction without support weight (d) 3D Reconstruction with support weight

Figure 5.7: Disparity map obtained from patch size 21×21 px^2. (a) & (b) Difference between the virtual tabletop plane and the disparity surface. (c) & (d) 3D reconstruction from the disparity map.

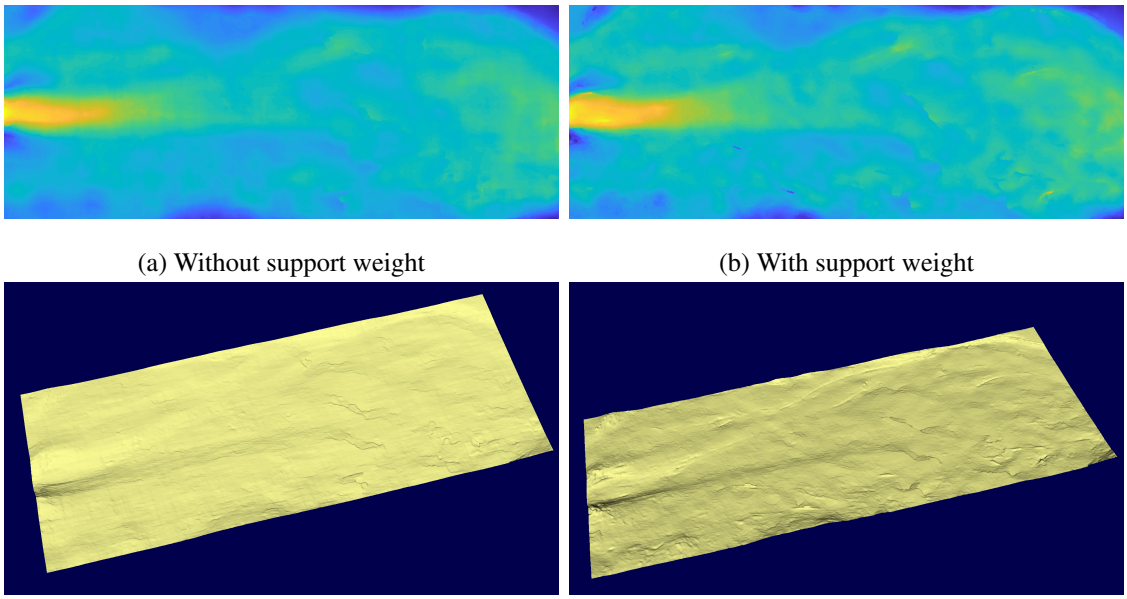


(a) Without support weight (b) With support weight

(c) Without support weight (d) With support weight

Figure 5.8: Disparity map obtained from patch size 31×31 px^2. (a) & (b) Difference between the virtual tabletop plane and the disparity surface. (c) & (d) 3D reconstruction from the disparity map.

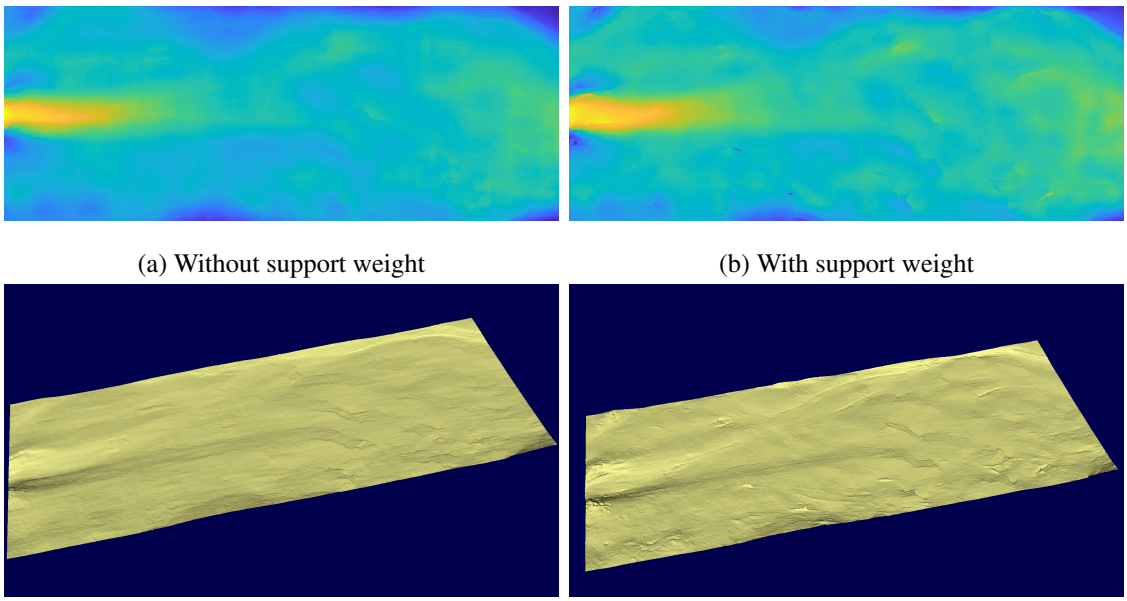
only the stereo depth cloud was changed to match with the laser depth cloud. The comparison process comprises of four steps. First, we match the scales of both clouds by comparing the physical dimension of an object in the image with the stereo depth cloud. In the second step,



(a) Without support weight (b) With support weight

(c) Without support weight (d) With support weight

Figure 5.9: Disparity map obtained from patch size 41×41 px^2. (a) & (b) Difference between the virtual tabletop plane and the disparity surface. (c) & (d) 3D reconstruction from the disparity map.

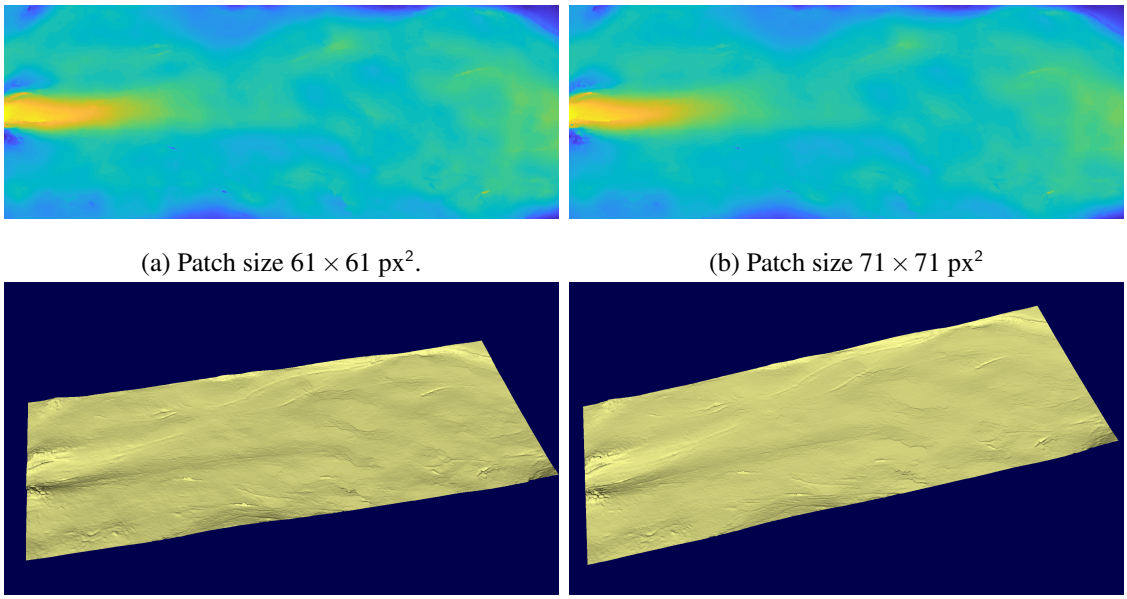


(a) Without support weight (b) With support weight

(c) Without support weight (d) With support weight

Figure 5.10: Disparity map obtained from patch size 51×51 px^2. (a) & (b) Difference between the virtual tabletop plane and the disparity surface. (c) & (d) 3D reconstruction from the disparity map.

we roughly align both clouds by manually matching the markers around the table. Then we use the iterative closest point (ICP) algorithm to minimise the difference between the two point clouds. Finally, the error map was computed based on the least-square best fitting plane that



(a) Patch size 61×61 px^2. (b) Patch size 71×71 px^2

(c) Patch size 61×61 px^2. (d) Patch size 71×71 px^2.

Figure 5.11: With support weight. (a) & (b) Difference between the virtual tabletop plane and the disparity surface. (c) & (d) 3D reconstruction from the disparity map.

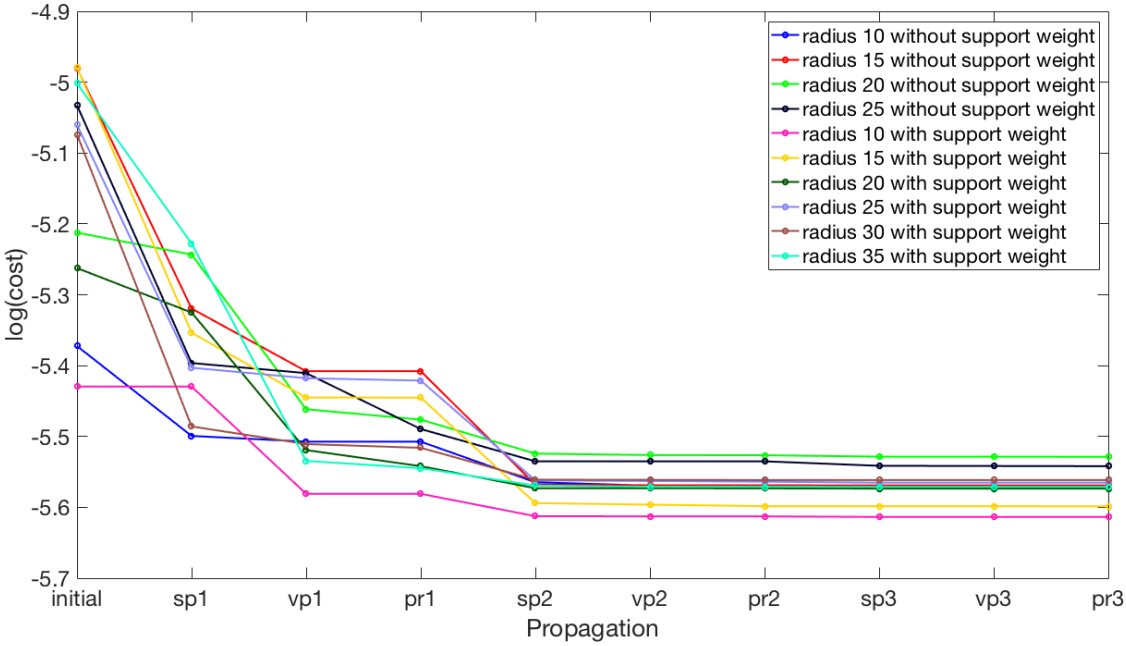


Figure 5.12: Cost function behaviour over patch radius and support weight. sp :spatial propagation, vp: view propagation, pr: plane refinement.

goes through the nearest point and its neighbours (Table 5.2). We also sliced the disparity map into seven segments and compared each segment with the corresponding laser scan segment (Fig. 5.13, 5.14, 5.15, 5.16, 5.17, 5.18, 5.19, 5.20). Observing the results, we conclude that the maximum error (red) occurs in occluded regions only.

Table 5.2: Error between the laser and stereo depth cloud.

slice	mean (mm)	std. dev. (mm)
1	0.975291	0.872799
2	0.850085	0.669202
3	0.969599	0.836569
4	0.978881	0.866514
5	0.872279	0.663666
6	0.934217	0.709914
7	0.996348	0.769356
whole	0.938999	0.778193

5.4.4 Comparison of 1D laser and stereo point cloud slices

We slice both the laser and stereo point cloud at eight places (Fig. 5.21) and compare each of the slices individually. It is evident from Fig. 5.22 and Fig. 5.23 that the stereo point cloud model matches with the laser point cloud except for areas with small channels and occlusions.

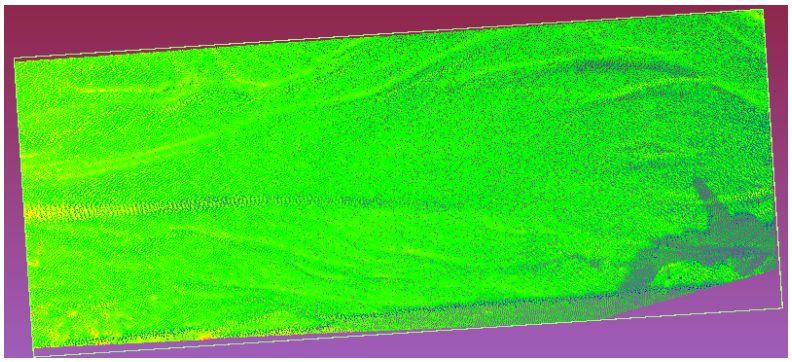
Algorithm 1 Curvature evaluation from scattered point cloud depth map

```
1: procedure CURVATURE
2: input: R                                   ▷ depth map obtained from TLS
3: output: s                                  ▷ shape index (transparency set by curvedness)
4:     pcdenoise ← R                          ▷ de-noise the point cloud data
5:     griddata ← pcdenoise                   ▷ organise the scattered data in a regular grid array
6:     gaussian ← griddata                    ▷ apply Gaussian filter on the depth map
7:     derivative ← gaussian                  ▷ compute first and second order partials of the depth map
8:     s ← derivative                         ▷ compute shape index and curvedness
9: end procedure
```

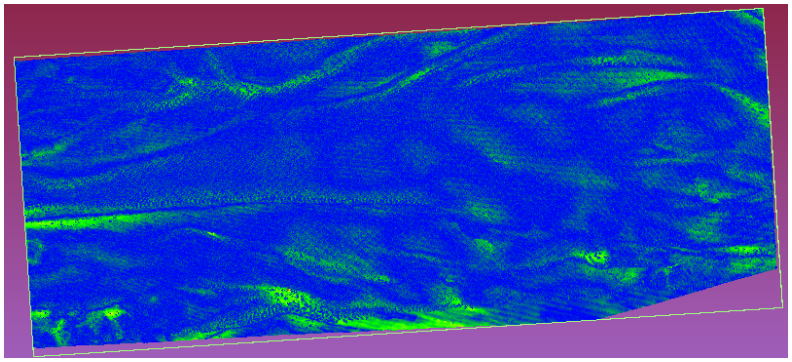
5.5 Surface curvature analysis

We further analyse the curvature of the depth map obtained from the laser scanner following the local shape measures (Sec. 4.7) (Alg. 1). Fig. 5.24 shows the curvature of the depth map. The active channels of the riverbed network can be retrieved from the curvature map.

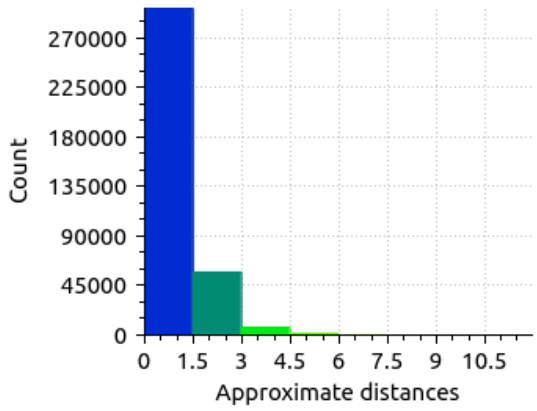
(a) Depth map



(b) Laser scan



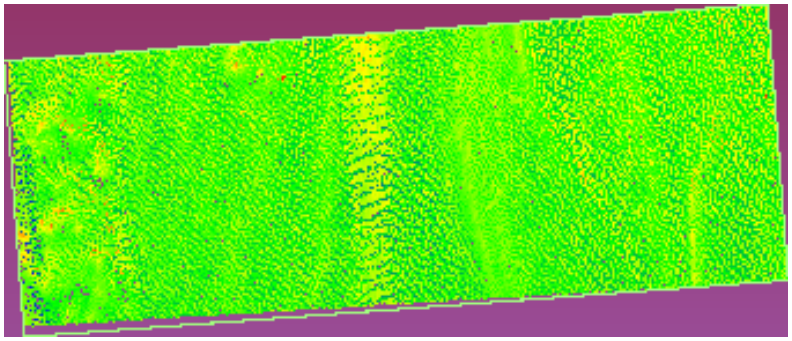
(c) Error



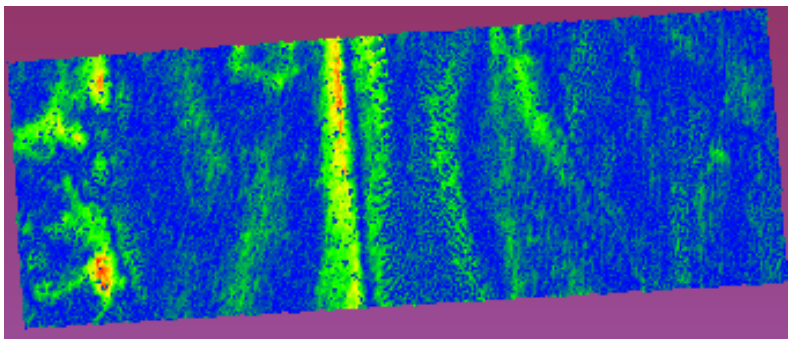
(d) Error histogram

Figure 5.13: Depth map comparison with laser scan, patch size 71×71 px^2

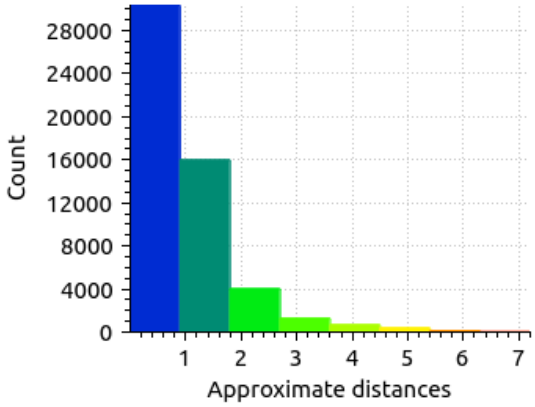
(a) Depth map



(b) Laser scan

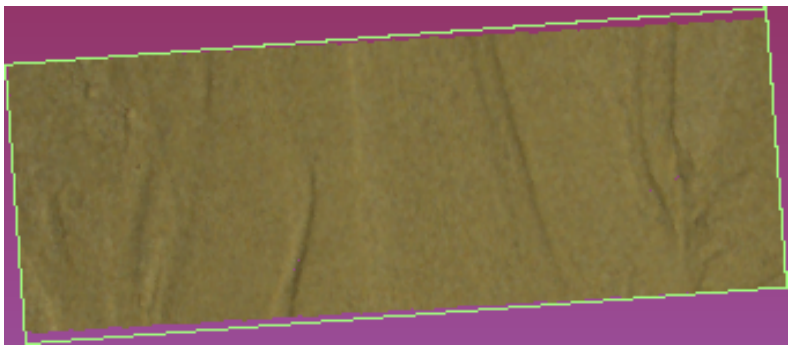


(c) Error

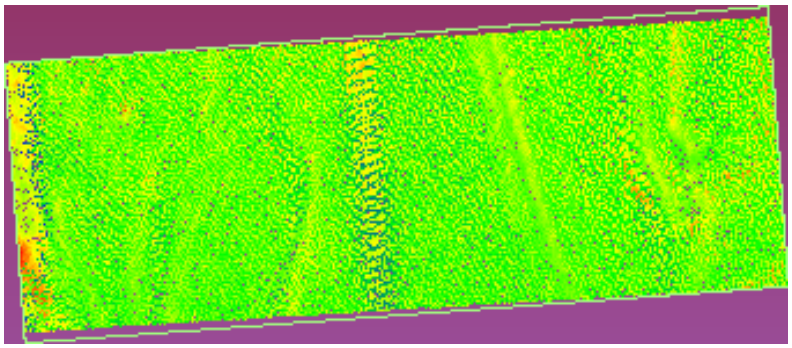


(d) Error histogram

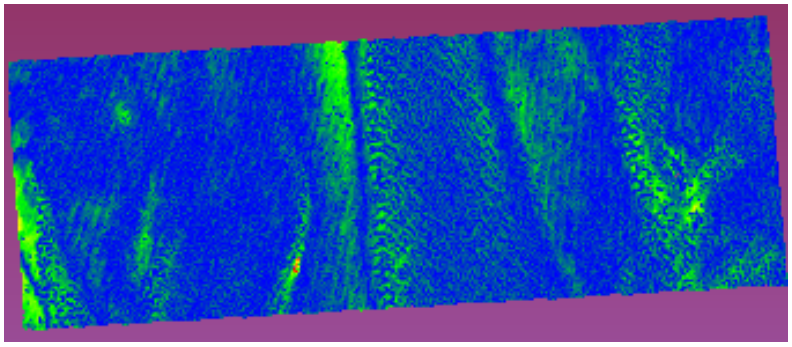
Figure 5.14: Depth map comparison with laser scan on slice 1, patch size 71×71 px^2



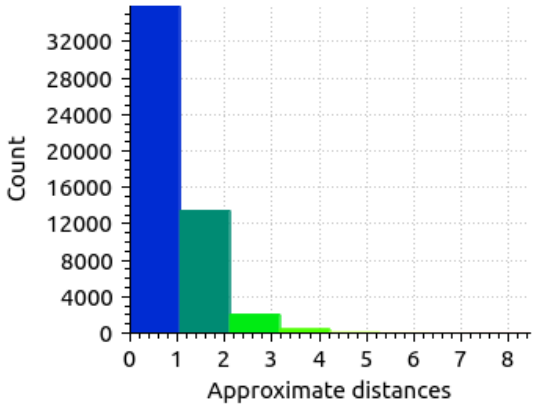
(a) Depth map



(b) Laser scan



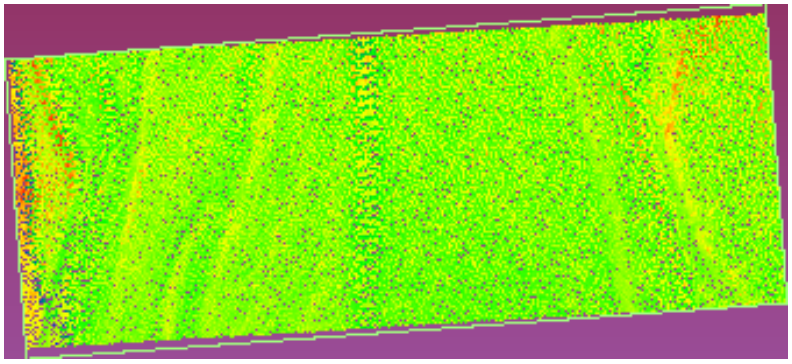
(c) Error



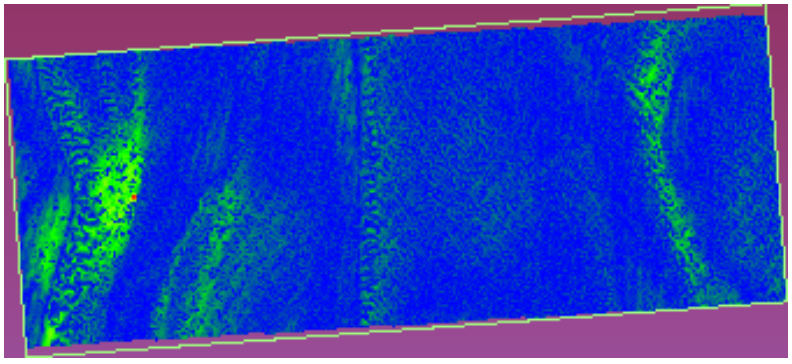
(d) Error histogram

Figure 5.15: Depth map comparison with laser scan on slice 2, patch size 71×71 px^2

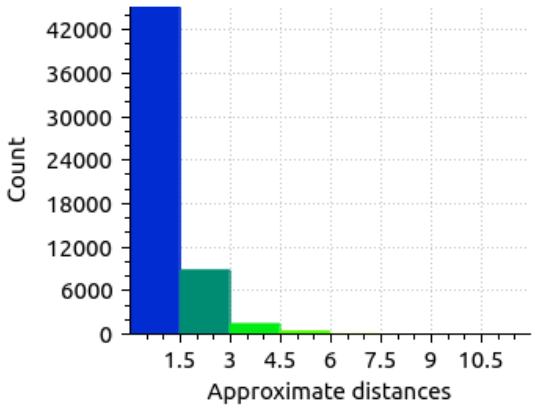
(a) Depth map



(b) Laser scan



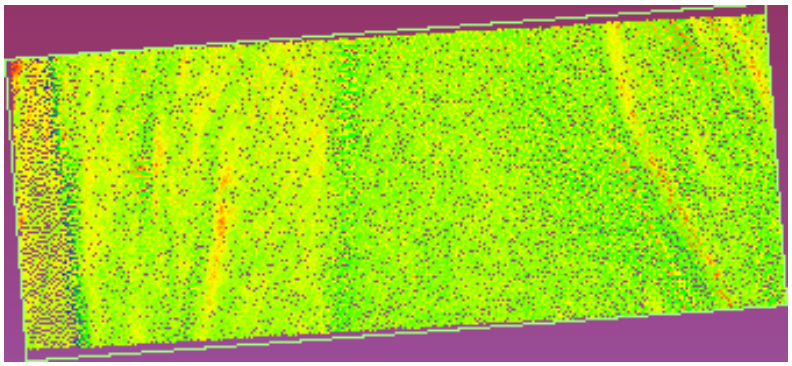
(c) Error



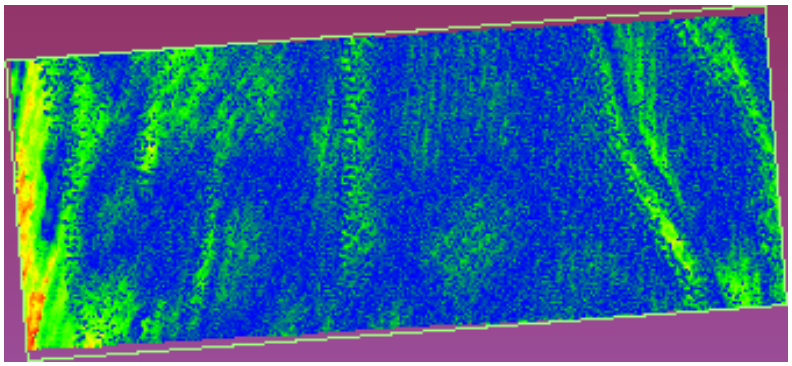
(d) Error histogram

Figure 5.16: Depth map comparison with laser scan on slice 3, patch size 71×71 px^2

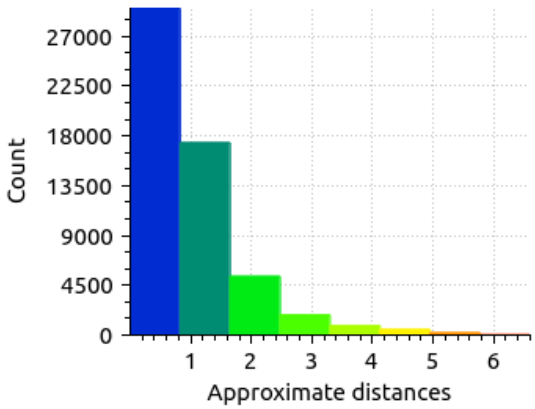
(a) Depth map



(b) Laser scan



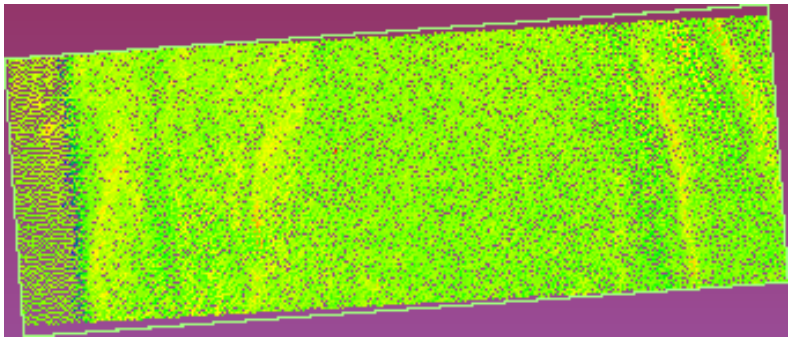
(c) Error



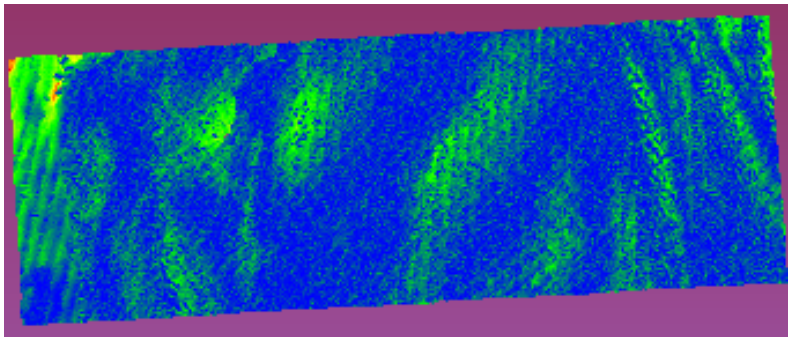
(d) Error histogram

Figure 5.17: Depth map comparison with laser scan on slice 4, patch size 71×71 px^2

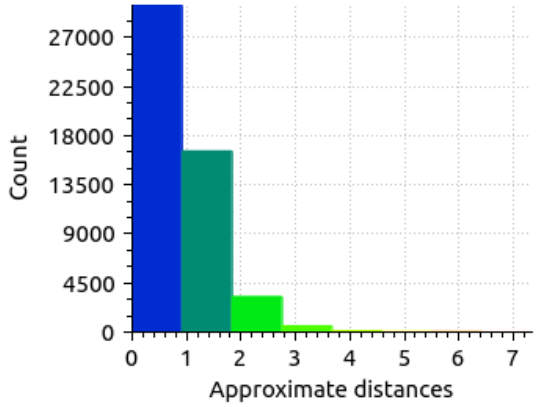
(a) Depth map



(b) Laser scan



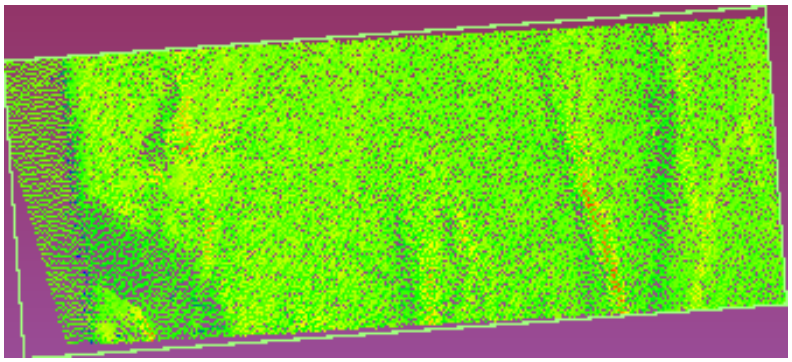
(c) Error



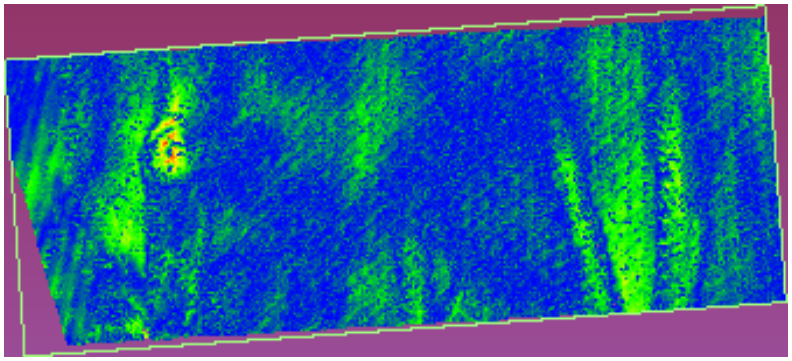
(d) Error histogram

Figure 5.18: Depth map comparison with laser scan on slice 5, patch size 71×71 px^2

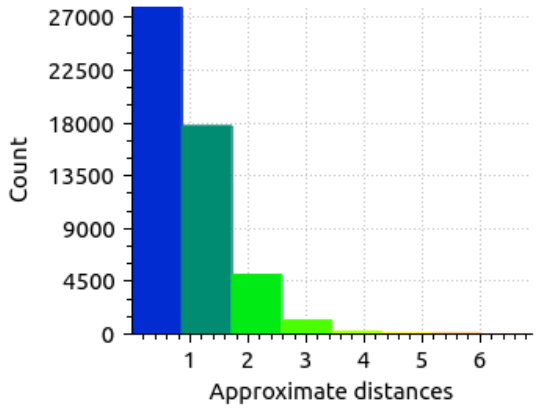
(a) Depth map



(b) Laser scan



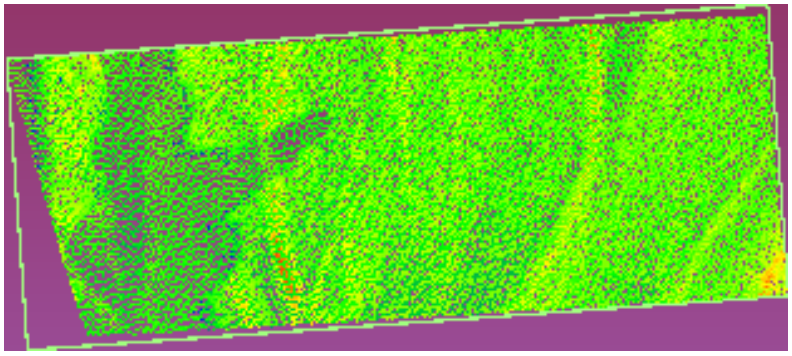
(c) Error



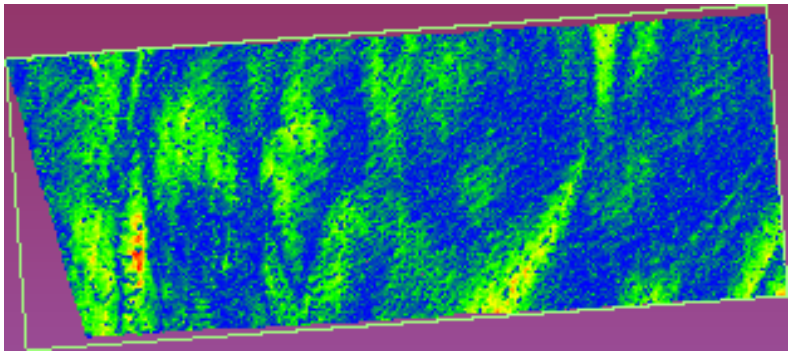
(d) Error histogram

Figure 5.19: Depth map comparison with laser scan on slice 6, patch size 71×71 px^2

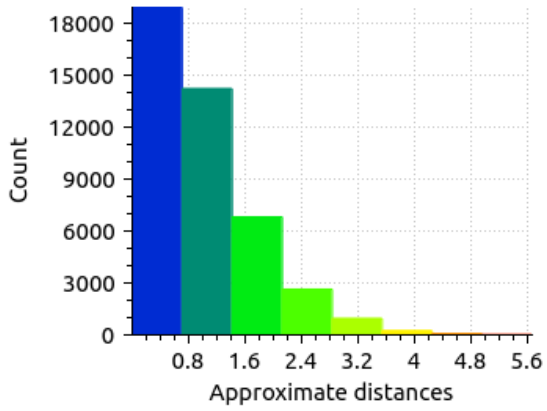
(a) Depth map



(b) Laser scan

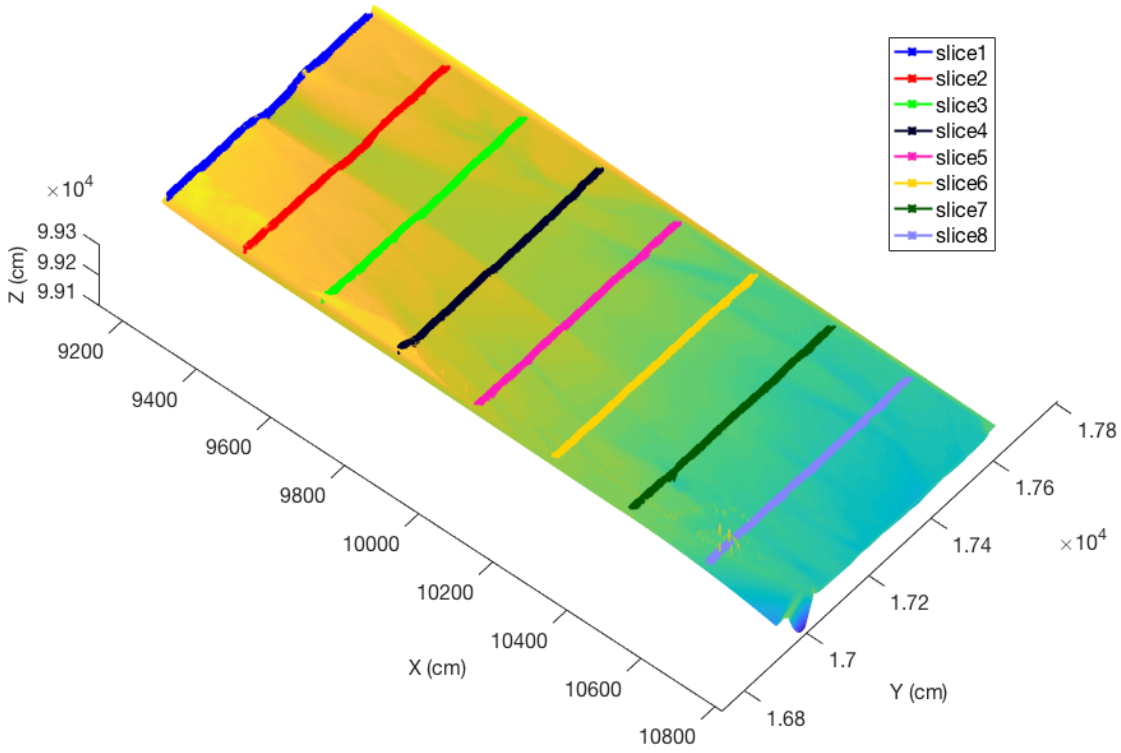


(c) Error

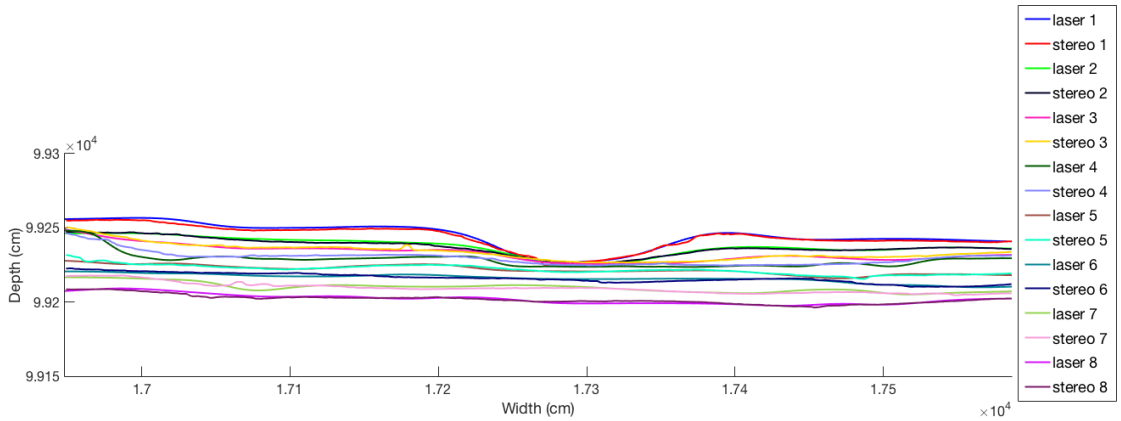


(d) Error histogram

Figure 5.20: Depth map comparison with laser scan on slice 7, patch size 71×71 px^2

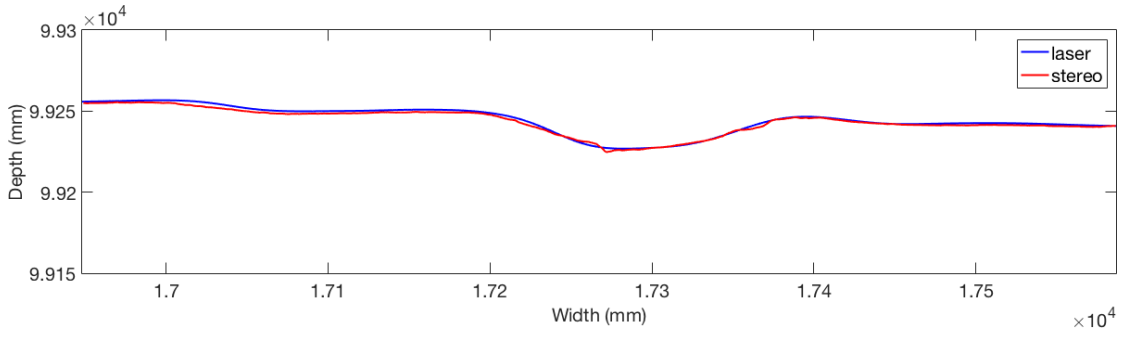


(a) Position of each slice.

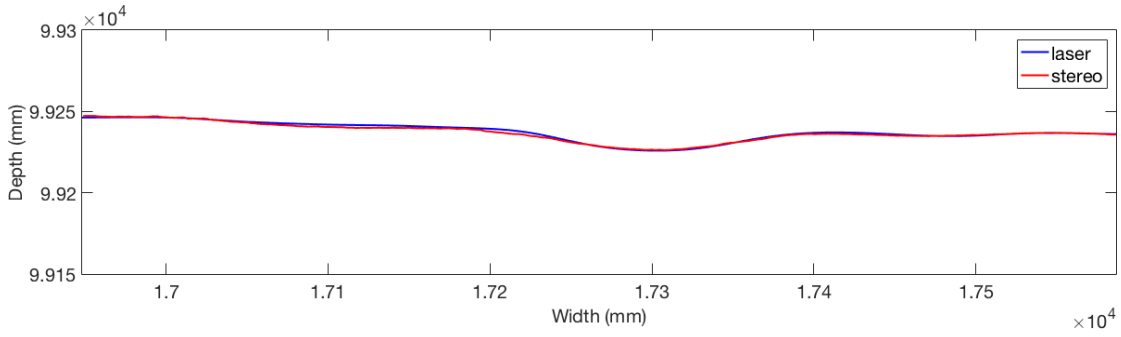


(b) Slice

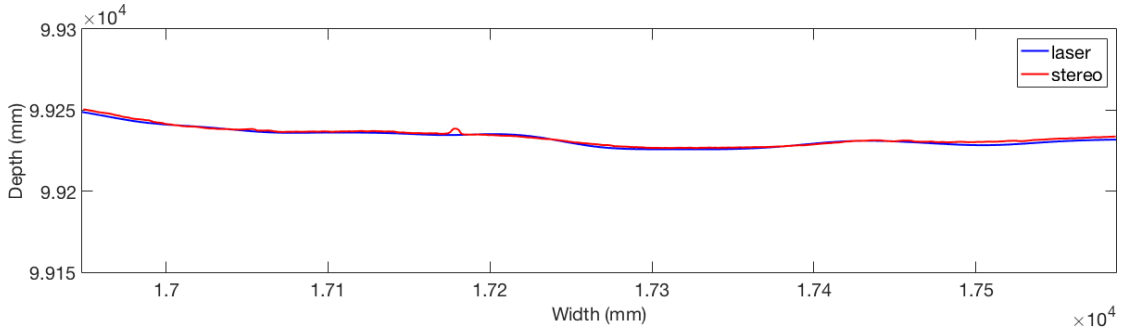
Figure 5.21: 1D slice comparison between laser and stereo point clouds.



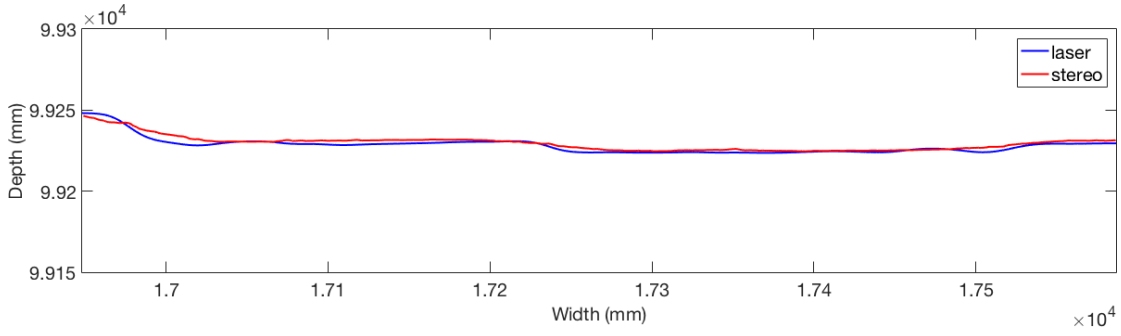
(a) Slice 1



(b) Slice 2

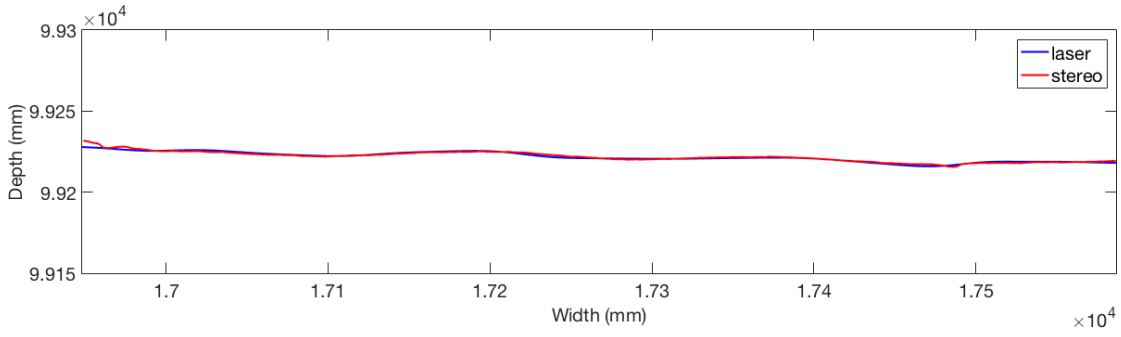


(c) Slice 3

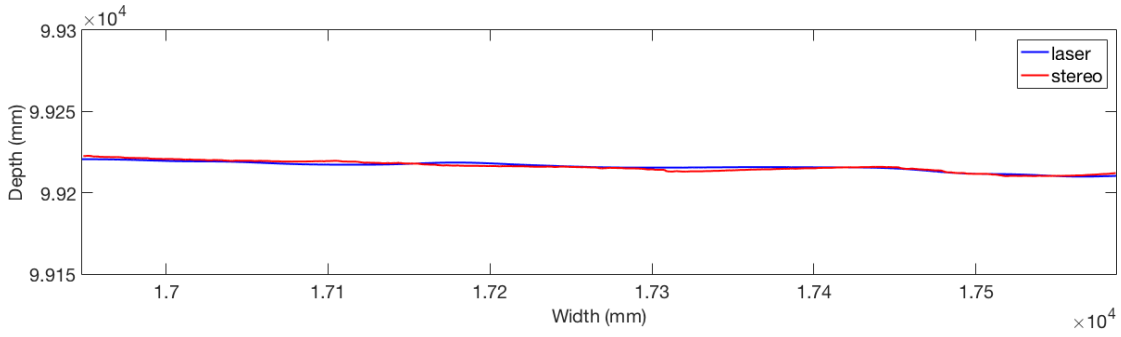


(d) Slice 4

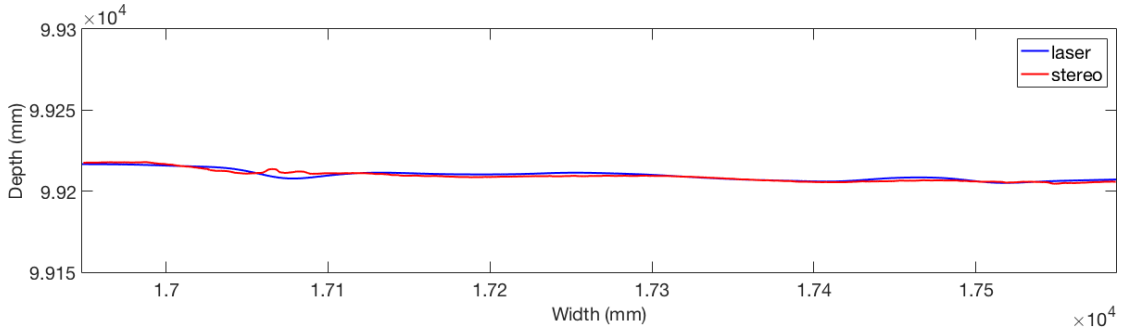
Figure 5.22: 1D slice comparison between laser and stereo point clouds.



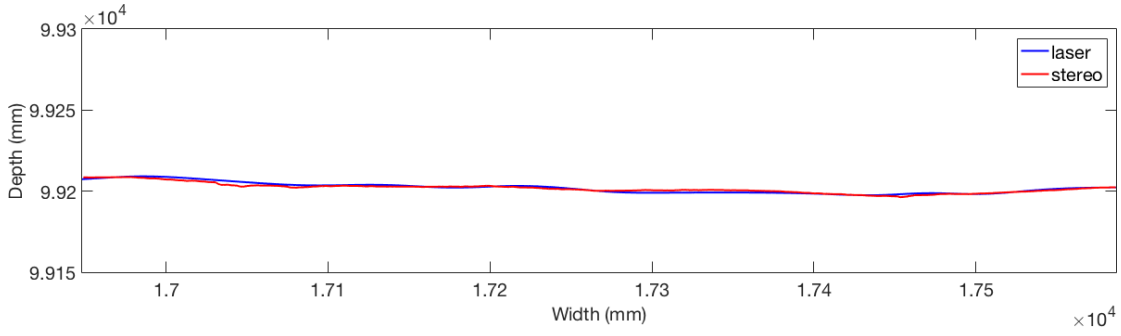
(a) Slice 5



(b) Slice 6

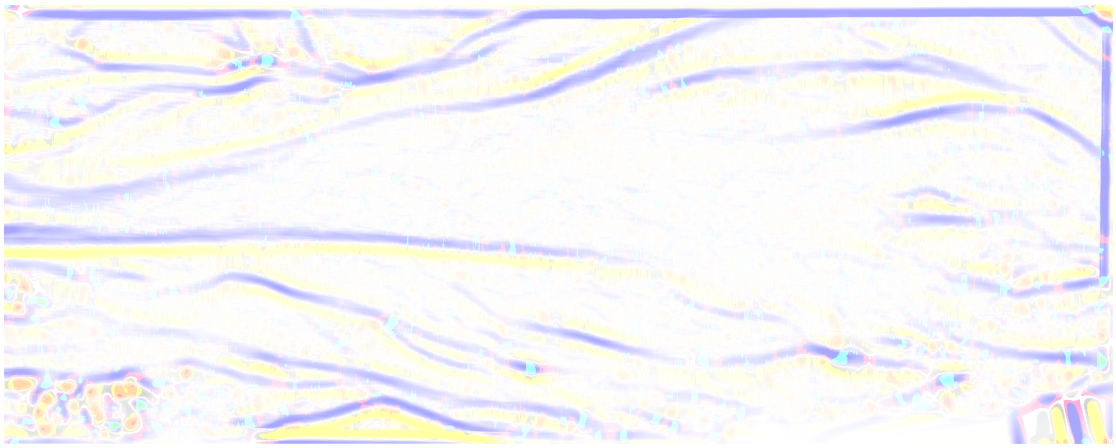


(c) Slice 7



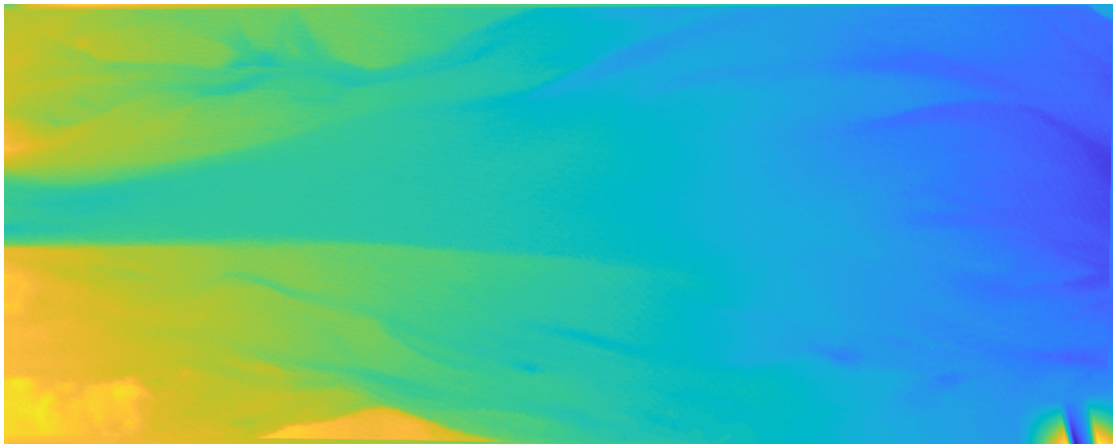
(d) Slice 8

Figure 5.23: 1D slice comparison between laser and stereo point clouds



(a) Shape index (transparency set by curvedness) of the depth map obtained from the laser scanner.

(b) RGB image by DSLR.



(c) Depth map obtained from the TLS scan

Figure 5.24: Depth map obtained from the TLS scan

5.6 Summary

In this chapter, we present the riverbed dataset which is a unique and challenging dataset consisting of 39 stereo pairs along with 15 ground truth captures by a laser scanner. We discuss the

experimental set-up and later compared the disparity map obtained by *IPMS* with the ground-truth depth map. Even though the images have fine structures, we found that *IPMS* can find the fine structures with the help of support weight and large patch size. Experimental results show that the root mean square error between the depth clouds is under one millimetre. We also sliced both clouds at several places and plotted the surface in one dimension to analyse the result quantitatively. Finally, we analyse the surface curvature of the depth map obtained from the laser scan to identify the active channels of the riverbed network.

Chapter 6

Conclusion

6.1 Summary

Stereo matching algorithms have been extensively studied over the last few decades in the computer vision field to estimate the depth of a scene from calibrates stereo images. It is one of the oldest problems in computer vision but remains an active area of research and studied with great importance in the field of machine vision, computer vision, virtual reality, robot navigation, depth measurements and environment reconstruction as well as in many other aspects of production, security, defence, exploration and entertainment [66]. However, there are still various challenges (Sec. 2.4).

Apart from depth, surface orientation is also crucial in understanding 3D scene geometry. In this thesis, we focused on the state of the art PatchMatch stereo framework. The PMS framework is a dense local stereo method that estimates both depth and surface normals for each input image. However, the framework has certain limitations as mentioned in Section 2.7, which we have primarily addressed in this thesis by introducing two new frameworks. The proposed frameworks work with any algorithm that can be cast in the PMS framework.

One of the main drawbacks of the framework is the initialisation procedure, which is done in an unconstrained manner, where the algorithm randomly selects the plane parameters for every point. As a result, we end up with a proportion of geometrically impossible planes. In Chapter 3, we proposed the Initiased PatchMatch Stereo (*IPMS*) framework. The framework uses visibility and disparity bound constraints for large slanted patches in the scene. These constraints can be

used to associate geometrically feasible planes with each point in the disparity space. The new constraints are validated in the PatchMatch Stereo framework. We use these new constraints not only for initialisation but also in the local plane refinement step of this iterative algorithm. The proposed constraints increase the probability of estimating correct plane parameters and lead to an improved 3D reconstruction of the scene. Furthermore, the proposed constrained initialisation reduces the number of iterations before convergence to the optimal plane parameters. To update the plane parameters in the plane refinement step, we used the gradient-free non-linear optimiser BOBYQA, which we have shown to be more effective than the original Luus-Jaakola scheme [65] in refining the plane parameters. In addition, as most stereo image pairs are not perfectly rectified, we relaxed the view propagation by assigning the plane parameters to the neighbours of the candidate pixel. These modifications help our method to generate better disparity maps than state-of-the-art local methods and to converge in only two iterations. We also investigated the role of parameters for the framework in detail.

Besides surface orientation, curvature information is also important to understand the geometric structure of the scene. One of the other major limitations of the PMS framework is the planar disparity model that it uses to project the patches in another view. As the surface model is planar, *PMS* does not directly estimate the local curvature. In Chapter 4, we propose the Quadric PatchMatch Stereo (*QPMS*) framework. The *QPMS* framework is built on top of the *IPMS* framework and uses a quadric disparity model which successfully handles both curved and planar surfaces in the disparity space and, also provides curvature estimates from the associated surfaces for every pixel using both spatial and surface normal information. We also proposed principal curvature and direction constraints which associate geometrically feasible quadrics with each point in the disparity space. We further address the false matching problem by introducing disparity guided spatial propagation, where a non-linear disparity dissimilarity function weights the aggregated cost. Disparity guided spatial propagation prevents false matches from growing and also fill them with the correct disparity value, provided there is at least one good surface approximation of the neighbours.

Moreover, we captured and processed a unique and challenging riverbed dataset in Chapter 5. The dataset shows the evolution of a simulated riverbed, which we have also used to validate the propositions in a relevant scenario. The dataset deals with fine textures (sand) which makes it hard to use conventional stereo matching algorithms. The dataset contains stereo images along

with ground truth depth information captured by a terrestrial laser scanner. We created 3D topographic models at high resolution from the stereo pairs and compared with the laser scan to find out the topographic changes in river-bed morphology over time. Experimental results show that the root mean square error between the depth clouds is under one millimetre. We further analysed the curvature of the depth map captured by the laser scanner to identify the active channels of the riverbed network, which can be useful to study their evolution.

6.2 Future work

The two frameworks mentioned in this thesis work only with static scenes. Future work can expand this limitation by modelling disparity changes in the dynamic scene by estimating the d-motion (Sec. 2.3.1). It would be possible to add a temporal propagation phase in the existing framework. However, it may be hard to match individual patches as the photo-consistency assumption will be broken.

Another future challenge can be extending the quadric model to a more general one. The *IPMS* framework fits the surface by tangent planes whereas the *QPMS* fits the surface by a quadric. Fitting a quadric is challenging as there are many classifications. However, fitting a more general surface is even more challenging as the geometry becomes complicated to match among views.

A possible alternate extension could be fusing *QPMS* with active sensors. The sensor can generate a rough curvature estimate of the structure of the scene. The rough estimate can then be used by *QPMS* along with other geometric constraints to obtain a detailed depth map.

The smoothness constraints in a cost function hold the key to determine the quality of matches in any stereo algorithm. For the *IPMS* and *QPMS* framework, it would be possible to add the surface normal or the curvature information to create a robust cost function, which could possibly help to move towards general real-world scenario. However, there might be problems with wider baseline and photometrically inconsistent stereo pairs.

It would be possible to extend both frameworks to multi-view stereo. Using the camera parameters, we can estimate the homography between each support region, in order to project them to other views.

The majority of the stereo pairs in the Middlebury stereo dataset deal with planar objects. There is a growing need for a dataset comprising curved surfaces. Such a dataset would further

facilitate the improvement of state of the art.

Finally, an immediate extension for the riverbed dataset could be analysing the evolution of the channel bed from the depth and curvature map. The d-motion can be used further to study the temporal changes.

Appendix A

Two view geometry

In this appendix, we present some basic geometry of two views based on [33, 15, 50, 58, 84].

A.1 Coordinate systems

Our world can be seen as an Euclidean space, and the position of any object can be represented by the Euclidean coordinate system $[U,V,W]$ with origin at $\mathbf{O}_\mathsf{W}$. This is also known as the world coordinate system. Any 3D point $\mathbf{P}$, known as the scene point, is represented by $\mathbf{P} = (U,V,W)^\intercal \in \mathcal{S}$, where $\mathcal{S}$ denotes the scene space (Fig. A.1).

Every camera, similar to the world coordinate system, has its coordinate system $[X,Y,Z]$ with origin at $\mathbf{O}_\mathsf{C}$, called the camera coordinate system. This coordinate system depends on the position and orientation of the camera. Any scene point $\mathbf{P}$ is represented by $(X,Y,Z)^\intercal$ in the camera

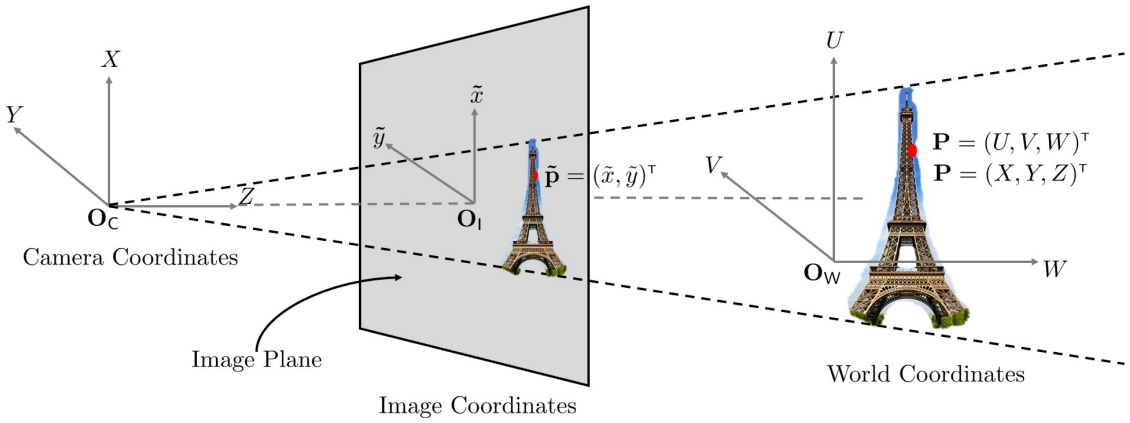


Figure A.1: Different coordinate systems. $\mathbf{O}_\mathsf{W}$, $\mathbf{O}_\mathsf{C}$ and $\mathbf{O}_\mathsf{I}$ are the origins of the world, camera and image coordinates, respectively.

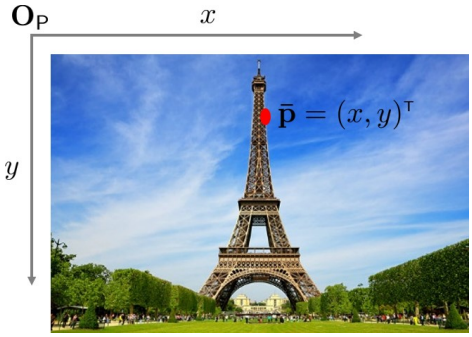


Figure A.2: Pixel coordinate system with origin $\mathbf{O}_\mathsf{P}$ at the top left corner.

coordinate system. For simplicity, we assume the camera coordinate system is aligned with the world coordinate system, since a rotation and translation can merge the coordinate frames. Under this assumption a 3D point $\mathbf{P} = (U,V,W)^\mathsf{T}$ can also be written as $\mathbf{P} = (X,Y,Z)^\mathsf{T} \in \mathcal{S}$ in the camera coordinate system (Fig. A.1).

The image plane is a two-dimensional Euclidean coordinate system $[\tilde{x},\tilde{y}]$. The image plane lies at a distance f (measured in physical unit, *e.g.,* millimetre), known as the focal length, from the camera optical centre $\mathbf{O}_\mathsf{C}$. The origin of this coordinate system, known as the principal point, is the point where the optical axis (Z-axis) intersects the image plane and is denoted by $\mathbf{O}_\mathsf{I}$. Let $\tilde{\mathbf{p}}$ be the projection of $\mathbf{P}$ on the image plane, known as an image point and is denoted by the image coordinates $\tilde{\mathbf{p}} = (\tilde{x},\tilde{y})^\mathsf{T}$ (Fig. A.1).

Every digital image has a coordinate system known as the pixel coordinates $[x,y]$. In an image coordinate system the coordinates are measured in physical units, whereas in pixels coordinate system, they are changed to pixel unit by multiplying the physical units by a scale factor. Also in pixel coordinate system, the origin $\mathbf{O}_\mathsf{P}$ lies on the top left corner of the image (Fig. A.2). An image point $\tilde{\mathbf{p}}$ transforms to $\bar{\mathbf{p}}$, known as the pixel point, and is denoted by $\bar{\mathbf{p}} = (x,y)^\mathsf{T}$ in pixel coordinates.

A.1.1 Homogeneous coordinates

The 2D projective space can be thought of as the 2D Euclidean space $(\mathbb{R}^2)$ with additional points added (points at infinity) which are considered to lie on a new line (line at infinity). Parallel lines in the Euclidean space intersect at a point at infinity corresponding to their common direction. Given a point $(x,y)^\mathsf{T} \in \mathbb{R}^2$, the corresponding homogeneous coordinate is defined as the ordered tuple $(xz,yz,z)^\mathsf{T}$, $z \neq 0$. By this definition, multiplying the homogeneous coordinates by a common, non-zero factor gives a new set of homogeneous coordinates for the same point. Formally,

we say that two vectors $\{\mathbf{x},\mathbf{y}\} \in \mathbb{P}^2$ are equivalent if $\mathbf{x} = \lambda\mathbf{y}$ for some non-zero scalar λ and write $\mathbf{x} \simeq \mathbf{y}$, where $\simeq$ denotes equality up to a scale factor. In particular, $(x,y,1)^\mathsf{T}$ is such a homogeneous coordinate system for the point $(x,y)^\mathsf{T}$. The original Cartesian coordinates are recovered by dividing the first two components by the third one. The triple $(0,0,0)^\mathsf{T}$ is omitted from $\mathbb{P}^2$ and does not represent any point.

To interpret $(0,1,0)^\mathsf{T}$ geometrically, we look at $(0,1,\varepsilon)^\mathsf{T}$, where ε is a small positive number. This point has a non-zero third coordinate and is equivalent to $(0,\frac{1}{\varepsilon},1)^\mathsf{T}$, *i.e.,* it is a point with x-coordinate zero and a very large y-coordinate. Making ε smaller we see that $(0,1,0)^\mathsf{T}$ can be interpreted as a point infinitely far away in the direction $(0,1)^\mathsf{T}$. We call this type of point a vanishing point or a point at infinity.

Homogeneous representation of lines

A line in the plane is defined as the set of points $(x,y)^\mathsf{T}$ that satisfies $ax+by+c=0$, where a,b, and c are real numbers. A point $\mathbf{x} = (x,y,1)^\mathsf{T}$ in homogeneous coordinates lies on a line $\mathbf{l} = (a,b,c)^\mathsf{T}$ if and only if $\mathbf{x}^\mathsf{T}\mathbf{l} = 0$.

Line joining two points

The line $\mathbf{l}$ passing through two points $\mathbf{x}$ and $\mathbf{y}$ in homogeneous coordinates is given by $\mathbf{l} = \mathbf{x} \times \mathbf{y}$, where $\times$ is the cross-product between two vectors.

Transformation of lines

Under the point transformation $\mathbf{x'} = \mathbf{H} = \mathbf{x}$, a line $\mathbf{l}$ passing through $\mathbf{x}$ transforms as:

$$\mathbf{l'} = \mathbf{H}^{-\mathsf{T}}\mathbf{l} \tag{A.1}$$

Transformation of conics

Under the point transformation $\mathbf{x'} = \mathbf{H} = \mathbf{x}$, a conic $\mathcal{C}$ passing through $\mathbf{x}$ ($\mathbf{x}^\mathsf{T}\mathcal{C}\mathbf{x} = 0$) transforms to:

$$\mathcal{C'} = \mathbf{H}^{-\mathsf{T}}\mathcal{C}\mathbf{H}^{-1} \tag{A.2}$$

A.2 Lens distortion

In practice, no lens is perfect due to manufacturing errors. There are primarily two types of distortions; radial distortions arise as a result of the shape of a lens, whereas tangential distortions

arise from the assembly process of the camera as a whole.

During radial distortion, the lens often noticeably distorts the location of pixels near the edges of the image. This bulging phenomenon is the source of the "barrel" or "fish-eye" effect (Fig. A.3). The radial distortion is zero at the optical centre of the image and increases towards the edges. Let $(x,y)^\mathsf{T}$ denotes the image location of a 3D point $\mathbf{P}$ under the pin-hole camera model (Sec. 2.2.2) and $(x_\mathsf{r},y_\mathsf{r})^\mathsf{T}$ denotes the image location of $\mathbf{P}$ with radial distortion. The relation between $(x,y)^\mathsf{T}$ and $(x_\mathsf{r},y_\mathsf{r})^\mathsf{T}$ is described by the following equations [14]:

$$\begin{pmatrix} x_\mathsf{r} \\ y_\mathsf{r} \end{pmatrix} = \begin{pmatrix} o_x \\ o_y \end{pmatrix} + L(r) \begin{pmatrix} x - o_x \\ y - o_y \end{pmatrix},$$

where $(o_x,o_y)^\mathsf{T}$ is the principal point, and r denotes the Euclidean distance of $(x,y)^\mathsf{T}$ from the principal point, *i.e.,* $r^2 = (x-o_x)^2 + (y-o_y)^2$; and $L(r)$ is typically defined as $L(r) = 1 + \kappa_1 r + \kappa_2 r^2 + \kappa_3 r^3$.

The second largest common distortion is the tangential distortion, which results from the lens not being parallel to the image plane (Fig. A.4). Tangential distortion is minimally characterized by two additional parameters [13], ρ_1 and ρ_2, such that:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} 2y & r^2 + 2x^2 \\ r^2 + 2y^2 & 2x \end{pmatrix} \begin{pmatrix} \rho_1 \\ \rho_2 \end{pmatrix},$$

where $(x',y')^\mathsf{T}$ denotes the image location of $\mathbf{P}$ with tangential distortion. In total, there are five distortion coefficients; three for radial distortion $(\kappa_1,\kappa_2,\kappa_3)$, and two for tangential distortion

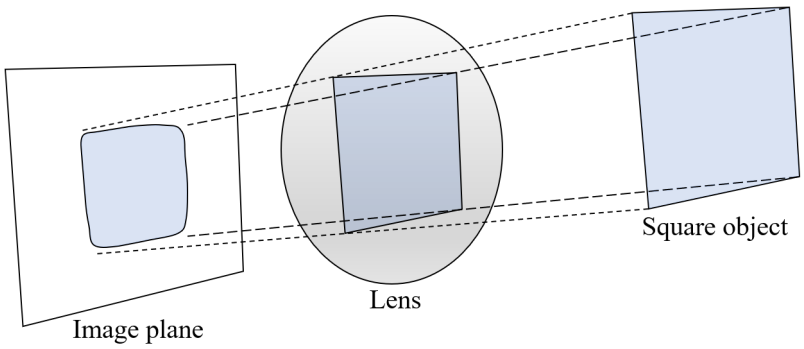


Figure A.3: Radial distortion. Rays far away from the centre bend too much compared to the rays closer to the center; thus the sides of a square appear to bow out on the image plane. Image reproduced from [11].

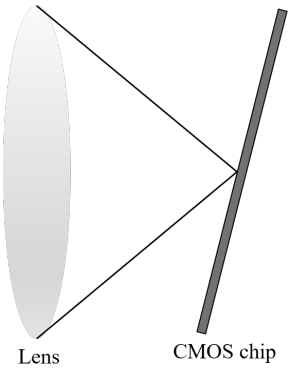


Figure A.4: Tangential distortion. The lens is not fully parallel to the image plane. Image reproduced from [11].

(ρ_1, ρ_2).

A.2.1 Epipolar geometry

We now formally define the epipolar geometry [33, 84] between a pair of images. Let C and C' be a pair of pinhole cameras in 3D space. Let $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$ be the projections through the left camera C and the right camera C' of a 3D scene point $\mathbf{P}$ in the left image I and the right image I' respectively. The associated geometry is shown in Fig. A.5.

Epipole. Point of intersection of the line joining the camera centres with the image plane.

Epipolar plane. The plane passing through the centres of projection and a point in the scene.

Epipolar line. The intersection of the epipolar plane with the image plane.

The epipolar constraint is defined as $\bar{\mathbf{p}}'^{\mathsf{T}}\mathbf{F}\bar{\mathbf{p}} = 0$, for all pairs of point correspondences $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$, where $\mathbf{F}$ is called the fundamental matrix. The fundamental matrix $\mathbf{F}$ is a 3×3 rank-2 homogeneous matrix that maps points in I to lines in I', *i.e.,* for a point $\bar{\mathbf{p}} \in I$, $\mathbf{l}' = \mathbf{F}\bar{\mathbf{p}}$ is an epipolar line in I' and $\bar{\mathbf{p}}'\mathbf{l}' = 0$. In fact, any point $\bar{\mathbf{p}}'$ that corresponds with $\bar{\mathbf{p}}$ must lie on the epipolar line $\mathbf{F}\bar{\mathbf{p}}$. For a fundamental matrix $\mathbf{F}$, there exists a pair of unique points $\mathbf{e} \in I$ and $\mathbf{e}' \in I'$ such that $\mathbf{F}\mathbf{e} = \mathbf{0} = \mathbf{F}^{\mathsf{T}}\mathbf{e}'$, where $\mathbf{0} = (\ 0\ 0\ 0\ )^{\mathsf{T}}$ is the zero vector. The points $\mathbf{e}$ and $\mathbf{e}'$ are known as the epipoles of images planes I and I' respectively. The epipoles have the property that all epipolar lines in I pass through $\mathbf{e}$, similarly all epipolar lines in I' pass through $\mathbf{e}'$.

In 3D space, $\mathbf{e}$ and $\mathbf{e}'$ are the intersections of the line CC' with the planes containing image I and I'. The set of planes containing the line CC' are called epipolar planes. Any 3D point $\mathbf{P}$ not lying on the line CC' will define an epipolar plane and the intersection of this epipolar plane with

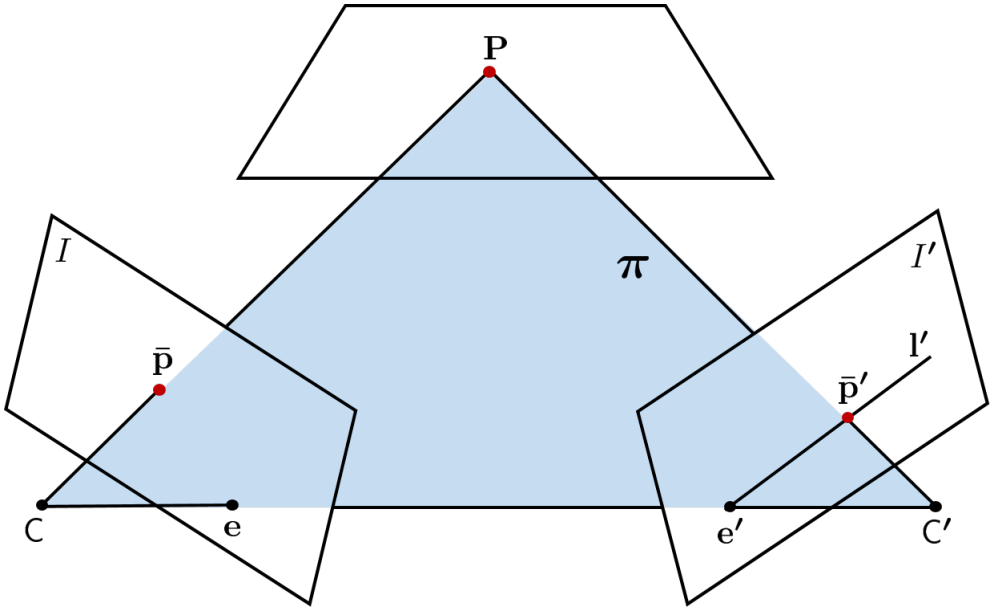


Figure A.5: **Epipolar geometry between a pair of images.** A point $\bar{\mathbf{p}}$ in the left image I is transferred via the plane π to a matching point $\bar{\mathbf{p}}'$ in the right image I'. The epipolar line $\mathbf{l}'$ is obtained by joining $\bar{\mathbf{p}}'$ to the epipole $\mathbf{e}'$. In symbols one may write $\bar{\mathbf{p}}' = \mathbf{H}_\pi \bar{\mathbf{p}}$ and $\mathbf{l}' = [\mathbf{e}']_\times \bar{\mathbf{p}}' = [\mathbf{e}']_\times \mathbf{H}_\pi \bar{\mathbf{p}} = \mathbf{F}\bar{\mathbf{p}}$, where $\mathbf{F} = [\mathbf{e}']_\times \mathbf{H}_\pi$ is defined as the fundamental matrix [33].

the plane containing I or I' will result in an epipolar line (Fig. A.5). Hence each epipole is the projection of the 'other' camera centre.

Appendix B

Differential geometry

In this appendix, we present some basic concepts of differential geometry based on [69, 86, 30, 19].

B.1 Differential geometry on a Monge patch

Let $U \in \mathbb{R}^2$ denote a rectangular patch in the plane. A Monge patch is a regular (differentiable) smooth height map where there exists a bijective mapping from $\mathbb{R}^2$ to $\mathbb{R}^3$ and vice versa. We will use a Monge patch $\mathscr{M} \in \mathbb{R}^3$ to denote the image of U. Let σ denote a differentiable function from U to $\mathscr{M}$, such that

$$\sigma : U \to \mathscr{M} \qquad \sigma(x,y) = (\sigma_1(x,y), \sigma_2(x,y), \sigma_3(x,y)) = (x,y,d(x,y)), \tag{B.1}$$

where $d(x,y) : U \to \mathbb{R}$ is a real valued differentiable function. The graph of σ is the set of all points in $\mathbb{R}^3$ whose coordinates satisfy the equation

$$z = d(x,y). \tag{B.2}$$

B.1.1 Surface differential properties

Partial differentiation can be performed on σ by fixing one variable at a time. Let $\bar{\mathbf{p}} = (x_0, y_0) \in U$ is mapped to $\mathbf{p} = (x_0, y_0, , d(x_0, y_0))$ via σ, *i.e.,* $\sigma(\bar{\mathbf{p}}) = \mathbf{p}$. We fix y at y_0 and vary x. Then $\sigma(x, y_0)$ depends on only one parameter and represents a curve. It is called an x parameter curve.

Similarly, we can get a y-parameter curve $\sigma(x_0, y)$ by fixing $x = x_0$. Note that, both the x and y parameter curves pass through $\sigma(x_0, y_0)$ in $\mathscr{M}$. Tangent vectors for the x parameter and the y parameter curves are given by differentiating the component functions of σ with respect to x and y respectively.

$$\begin{gathered}\sigma_x = (1, 0, d_x), \quad \sigma_y = (0, 1, d_y), \\ \sigma_{xx} = (0, 0, d_{xx}), \quad \sigma_{xy} = (0, 0, d_{xy}) = (0, 0, d_{yx}) = \sigma_{yx}, \quad \sigma_{yy} = (0, 0, d_{yy}).\end{gathered} \tag{B.3}$$

A crucial result is that the order does not matter here, *i.e.,* mixed partials commute. The tangent (velocity) vectors of the parameter curves at the point $\mathbf{p}$ are given by $\sigma_x(x_0, y_0)$ and $\sigma_y(x_0, y_0)$. The above partials can be written in a matrix form by the following expression:

$$d\sigma(\bar{\mathbf{p}}) = \mathbf{A}(\bar{\mathbf{p}}), \text{ where } \mathbf{A} = \begin{pmatrix} \frac{\partial \sigma_1}{\partial x}(\bar{\mathbf{p}}) & \frac{\partial \sigma_1}{\partial y}(\bar{\mathbf{p}}) \\ \frac{\partial \sigma_2}{\partial x}(\bar{\mathbf{p}}) & \frac{\partial \sigma_2}{\partial y}(\bar{\mathbf{p}}) \\ \frac{\partial \sigma_3}{\partial x}(\bar{\mathbf{p}}) & \frac{\partial \sigma_3}{\partial y}(\bar{\mathbf{p}}) \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ d_x(\bar{\mathbf{p}}) & d_y(\bar{\mathbf{p}}) \end{pmatrix}. \tag{B.4}$$

The matrix $\mathbf{A}$ is called the Jacobian matrix of σ at $\bar{\mathbf{p}}$. The rows of $\mathbf{A}$ are the gradients of the components of σ. As the rank of $\mathbf{A}$ is always 2, the Monge Patch $\mathscr{M}$ is always a regular surface. The tangent plane of $\mathscr{M}$ at $\mathbf{p}$ is defined as:

$$T_{\mathbf{p}}(\mathscr{M}) = \{\mathbf{v} \mid \mathbf{v} \text{ is tangent to } \mathscr{M} \text{ at } \mathbf{p}\}. \tag{B.5}$$

Immediately we know two curves which pass through $\mathbf{p}$: the x and y parameter curves with velocity vectors σ_x and σ_y. It can be shown that $\{\sigma_x, \sigma_y\}$ form a basis for the vector space $T_{\mathbf{p}}(\mathscr{M})$. Note that, the basis is not orthogonal in general.

Let $\mathbf{v} \in T_{\mathbf{p}}(\mathscr{M})$. We are interested in finding the change of each component function of $\sigma = (\sigma_1, \sigma_2, \sigma_3)$ on $\mathscr{M}$ along $\mathbf{v}$. The directional derivative of σ_i in the direction of $\mathbf{v}$ at $\bar{\mathbf{p}}$ is defined as:

$$\mathbf{v}[\sigma_i](\bar{\mathbf{p}}) = \nabla \sigma_i \cdot \mathbf{v}, \tag{B.6}$$

where ∇ is the gradient operator. The directional derivative of σ_1, σ_2 and σ_3 along $\mathbf{v}$ can be computed similarly. Let $f : \mathscr{M} \to \mathbb{R}$, then the composition with a patch σ can be formed as

$f(\sigma(x,y)) = f \circ \sigma$. Then we have

$$\sigma_x[f] = \frac{\partial f}{\partial x}\,|_{x=x_0} \quad \text{and} \quad \sigma_y[f] = \frac{\partial f}{\partial y}\,|_{y=y_0}. \tag{B.7}$$

The unit surface normal $\mathbf{N}$ for $T_{\mathbf{p}}(\mathscr{M})$ is given by,

$$\mathbf{N} = \frac{\sigma_x \times \sigma_y}{\|\sigma_x \times \sigma_y\|}(x_0,y_0) = \frac{(-d_x,-d_y,1)}{\sqrt{1+{d_x}^2+{d_y}^2}}. \tag{B.8}$$

We can also write $\mathbf{N} = (n_1,n_2,n_3)$ in terms of the standard basis $\mathbf{e}_1 = (1,0,0)$, $\mathbf{e}_2 = (0,1,0)$, $\mathbf{e}_3 = (0,0,1)$ for $\mathbb{R}^3$.

$$\mathbf{N} = n_1\mathbf{e}_1 + n_2\mathbf{e}_2 + n_3\mathbf{e}_3 = \sum_{i=1}^{3} n_i\mathbf{e}_i, \tag{B.9}$$

where n_1, n_2, n_3 are functions from $\mathscr{M}$ to $\mathbb{R}$. The change in $\mathbf{N}$ in a direction $\mathbf{v} \in T_{\mathbf{p}}(\mathscr{M})$ can be described by observing the changes of n_1, n_2 and n_3 in the $\mathbf{v}$ direction. This initial rate of change of $\mathbf{N}$ in the $\mathbf{v}$ direction is defined as the covariant derivative and is given by

$$\nabla_{\mathbf{v}}\mathbf{N} = (\mathbf{v}[n_1], \mathbf{v}[n_2], \mathbf{v}[n_3]) = \sum_{i=1}^{3} \mathbf{v}[n_i]\,\mathbf{e}_i. \tag{B.10}$$

The covariant derivative tells us how $\mathscr{M}$ curves in the $\mathbf{v}$ direction, which gives us an idea about the shape of the surface.

The Gauss map is a mapping $G : \mathscr{M} \to S^2$ from the surface $\mathscr{M}$ to the unit sphere S^2 is given by

$$G(\mathbf{p}) = \mathbf{N}(\mathbf{p}), \tag{B.11}$$

where $\mathbf{N}(\mathbf{p})$ is the unit normal to $\mathscr{M}$ at $\mathbf{p}$. The Gauss map is alternatively used to find the shape operator, discussed later.

B.1.2 Orthonormal basis in tangent space

We already know that $\{\sigma_x, \sigma_y\}$ form a basis for the vector space $T_{\mathbf{p}}(\mathscr{M})$. However, the basis is not orthogonal in general. Our objective is to find an orthonormal basis $\{\mathbf{u}_x, \mathbf{u}_y\}$ of $T_{\mathbf{p}}(\mathscr{M})$ to create a local 3D coordinate frame.

Now, following the Gram-Schmidt procedure, we make the first vector of unit length. Let

$$\mathbf{u}_x = \frac{\sigma_x}{\|\sigma_x\|} = \frac{1}{\sqrt{1+{d_x}^2}}(1,0,d_x) \tag{B.12}$$

Next we project σ_y on $\mathbf{u}_x$:

$$\text{Proj}_{\mathbf{u}_x}\sigma_y = \langle \sigma_y, \mathbf{u}_x \rangle \mathbf{u}_x = \frac{d_x d_y}{1+{d_x}^2}(1,0,d_x) \tag{B.13}$$

We create a vector perpendicular to $\mathbf{u}_x$ by subtracting $\text{Proj}_{\mathbf{u}_x}\sigma_y$ from σ_y.

$$\sigma_y - \text{Proj}_{\mathbf{u}_x}\sigma_y = \frac{1}{1+{d_x}^2}\left(-d_x d_y, 1+{d_x}^2, d_y\right) \tag{B.14}$$

Finally, we define $\mathbf{u}_y$ as

$$\mathbf{u}_y = \frac{\sigma_y - \text{Proj}_{\mathbf{u}_x}\sigma_y}{\left\|\sigma_y - \text{Proj}_{\mathbf{u}_x}\sigma_y\right\|} = \frac{1}{\sqrt{\left(1+{d_x}^2\right)\left(1+{d_x}^2+{d_y}^2\right)}}\left(-d_x d_y, 1+{d_x}^2, d_y\right) \tag{B.15}$$

Let $\mathbf{N} = (a,b,c)$ be the unit normal of the tangent plane $T_{\mathbf{p}}(\mathscr{M})$ at $\mathbf{p}$ on $\mathscr{M} \in \mathbb{R}^3$. Then the orthonormal basis $\{\mathbf{u}_x, \mathbf{u}_y\}$ of $T_{\mathbf{p}}(\mathscr{M})$ is given by

$$\begin{aligned} \mathbf{u}_x &= \frac{1}{\sqrt{a^2+c^2}}(c,0,-a) \\ \mathbf{u}_y &= \frac{1}{\sqrt{a^2+c^2}}\left(-ab, a^2+c^2, -bc\right) \end{aligned} \tag{B.16}$$

B.1.3 Shape operator

The shape operator (or Weingarten map) of $\mathscr{M}$ at $\mathbf{p}$ is defined as

$$S_{\mathbf{p}}(\mathbf{v}) = -\nabla_{\mathbf{v}}\mathbf{N}. \tag{B.17}$$

It can be shown that the shape operator is a linear transformation from $T_{\mathbf{p}}(\mathscr{M})$ to itself. The following are some properties of the shape operator.

1. The shape operator is a symmetric linear transformation, *i.e.,* for $\mathbf{u}, \mathbf{w} \in T_{\mathbf{p}}(\mathscr{M})$

$$S_{\mathbf{p}}(\mathbf{u}) \cdot \mathbf{w} = S_{\mathbf{p}}(\mathbf{w}) \cdot \mathbf{u}. \tag{B.18}$$

2. If $S_{\mathbf{p}} = 0$, for every $\mathbf{p} \in \mathscr{M}$, then $\mathscr{M}$ is contained in a plane.

3. The shape operator has real eigenvalues.

4. $S_{\mathbf{p}}(\sigma_x) = -\mathbf{N}_x$ and $S_{\mathbf{p}}(\sigma_y) = -\mathbf{N}_y$.

5. If $\boldsymbol{\alpha}$ is a curve passing through $\mathbf{p} \in \mathscr{M}$, then

$$S_{\mathbf{p}}(\boldsymbol{\alpha}') \cdot \boldsymbol{\alpha}' = \boldsymbol{\alpha}'' \cdot \mathbf{N}. \tag{B.19}$$

   Further, $S_{\mathbf{p}}(\sigma_x) \cdot \sigma_x = \sigma_{xx} \cdot \mathbf{N}$, $S_{\mathbf{p}}(\sigma_x) \cdot \sigma_y = \sigma_{xy} \cdot \mathbf{N} = \sigma_{yx} \cdot \mathbf{N} = S_{\mathbf{p}}(\sigma_y) \cdot \sigma_x$ and $S_{\mathbf{p}}(\sigma_y) \cdot \sigma_y = \sigma_{yy} \cdot \mathbf{N}$.

6. The Gauss map has an induced derivative map $G_\star : T_{\mathbf{p}}(\mathscr{M}) \to T_{G(\mathbf{p})}S^2$. Then, by definition, $G_\star(\mathbf{v}) = -S_{\mathbf{p}}(\mathbf{v})$.

We already know that $\{\sigma_x, \sigma_y\}$ form a basis for the vector space $T_{\mathbf{p}}(\mathscr{M})$ at $\mathbf{p} \in \mathscr{M}$. Furthermore, any vector in $T_{\mathbf{p}}(\mathscr{M})$ can be written as a linear combination of $\{\sigma_x, \sigma_y\}$. Hence, we can write the effects of the shape operator as

$$S_{\mathbf{p}}(\sigma_x) = a\,\sigma_x + b\,\sigma_y \quad \text{and} \quad S_{\mathbf{p}}(\sigma_y) = c\,\sigma_x + d\,\sigma_y. \tag{B.20}$$

The associated matrix of $S_{\mathbf{p}}$ with respect to the basis $\{\sigma_x, \sigma_y\}$ is given by

$$S_{\mathbf{p}} = \begin{pmatrix} a & c \\ b & d \end{pmatrix}. \tag{B.21}$$

B.1.4 Curvatures

Consider a curve lying on a two-dimensional surface, which is embedded in $\mathbb{R}^3$. The curvature of a surface is defined by the rate at which the unit surface normal is changing along the projection of the curve in $\mathbb{R}^3$.

The Gaussian curvature of a surface $\mathscr{M}$ at $\mathbf{p} \in \mathscr{M}$ is defined to be $K(\mathbf{p}) = \det(S_{\mathbf{p}})$.

The mean curvature of a surface $\mathscr{M}$ at $\mathbf{p} \in \mathscr{M}$ is defined to be $H(\mathbf{p}) = \frac{1}{2}\operatorname{trace}(S_{\mathbf{p}})$.

Note that, the shape operator $S_{\mathbf{p}}$ and the mean curvature H depend on the choice of the unit nor-

mal $\mathbf{N}$. However, the Gaussian curvature K is independent of that choice.

For a unit vector $\mathbf{u} \in \mathbf{T}_{\mathbf{p}}(\mathscr{M})$, the normal curvature of $\mathscr{M}$ in the $\mathbf{u}$ direction is given by

$$\kappa(\mathbf{u}) = S_{\mathbf{p}}(\mathbf{u}) \cdot \mathbf{u}. \tag{B.22}$$

The sign of the normal curvature tells us about the bending of the surface towards or away from its normal in a given direction. If $\kappa(\mathbf{u}) > 0$, then $\mathscr{M}$ bends towards $\mathbf{N}$. If $\kappa(\mathbf{u}) < 0$, then $\mathscr{M}$ bends away from $\mathbf{N}$. If $\kappa(\mathbf{u}) = 0$, we can only say that the rate of bending of $\mathscr{M}$ near $\mathbf{p}$ is small.

There are unit vectors $\mathbf{u}_1$ and $\mathbf{u}_2$ such that

$$\kappa(\mathbf{u}_1) = \kappa_1 = \max_{\mathbf{u}} \kappa(\mathbf{u}), \quad \kappa(\mathbf{u}_2) = \kappa_2 = \min_{\mathbf{u}} \kappa(\mathbf{u}). \tag{B.23}$$

The unit vectors $\mathbf{u}_1$ and $\mathbf{u}_2$ are called principal vectors, and κ_1 and κ_2 are called principal curvatures.

The eigenvalues of the shape operator $S_{\mathbf{p}}$ of a regular surface $\mathscr{M} \in \mathbb{R}^3$ at $\mathbf{p} \in \mathscr{M}$ are precisely the principal curvature κ_1 and κ_2 of $\mathscr{M}$ at $\mathbf{p}$. The corresponding unit eigenvectors are unit principal vectors, and vice versa. If $\kappa_1 = \kappa_2$, then $S_{\mathbf{p}}$ is a scalar multiplication of their common value. Otherwise, the unit eigenvectors $\mathbf{e}_1$ and $\mathbf{e}_2$ of $S_{\mathbf{p}}$ are perpendicular, and $S_{\mathbf{p}}$ is given by

$$S_{\mathbf{p}}\mathbf{e}_1 = \kappa_1\mathbf{e}_1 \quad \text{and } S_{\mathbf{p}}\mathbf{e}_2 = \kappa_2\mathbf{e}_2. \tag{B.24}$$

Let θ denote the oriented angle from $\mathbf{e}_1$ to $\mathbf{u} \in T_{\mathbf{p}}(\mathscr{M})$, so that $\mathbf{u} = \mathbf{e}_1\cos\theta + \mathbf{e}_2\sin\theta$. Then the normal curvature of $\kappa(\mathbf{u})$ is given by

$$\kappa(\mathbf{u}) = \kappa_1\cos^2\theta + \kappa_2\sin^2\theta. \tag{B.25}$$

The Gaussian curvature and mean curvature of $\mathscr{M}$ are related to the principal curvature by $K = \kappa_1\kappa_2$ and $H = \frac{1}{2}(\kappa_1 + \kappa_2)$. Hence, the principal curvatures κ_1 and κ_2 are the roots of the quadratic

equation $x^2 - 2Hx + K = 0$. Therefore,

$$\kappa_1 = H + \sqrt{H^2 - K} \quad \text{and} \quad \kappa_2 = H - \sqrt{H^2 - K}. \tag{B.26}$$

For a point $\mathbf{p}$ in a regular surface $\mathscr{M} \in \mathbb{R}^3$, we say that

- $\mathscr{M}$ is **elliptic** at $\mathbf{p}$ if $K(\mathbf{p}) > 0$ (equivalently, κ_1 and κ_2 have the same sign)
- $\mathscr{M}$ is **hyperbolic** at $\mathbf{p}$ if $K(\mathbf{p}) > 0$ (equivalently, κ_1 and κ_2 have opposite sign)
- $\mathscr{M}$ is **parabolic** at $\mathbf{p}$ if $K(\mathbf{p}) = 0$ but $S_\mathbf{p} \neq 0$ (equivalently, exactly one of κ_1 and κ_2 is zero)
- $\mathscr{M}$ is **plane** at $\mathbf{p}$ if $K(\mathbf{p}) = 0$ and $S_\mathbf{p} = 0$ (equivalently, $\kappa_1 = \kappa_2 = 0$)

B.1.5 Fundamental forms

Let $\mathscr{M}$ be a regular surface in $\mathbb{R}^3$ and $\mathbf{u}, \mathbf{v} \in T_\mathbf{p}(\mathscr{M})$. The first fundamental form I gives us the inner product between the tangent vectors:

$$I(\mathbf{u}, \mathbf{v}) = \mathbf{u} \cdot \mathbf{v} \tag{B.27}$$

The second fundamental form is the symmetric bilinear form II on a tangent space $T_\mathbf{p}(\mathscr{M})$ given by

$$II(\mathbf{u}, \mathbf{v}) = S_\mathbf{p}(\mathbf{u}) \cdot \mathbf{v} = S_\mathbf{p}(\mathbf{v}) \cdot \mathbf{u} \tag{B.28}$$

For any non zero tangent vector $\mathbf{w} \in T_\mathbf{p}(\mathscr{M})$ the normal curvature is given by

$$\kappa(\mathbf{w}) = \frac{II(\mathbf{w}, \mathbf{w})}{I(\mathbf{w}, \mathbf{w})} \tag{B.29}$$

Finally, The third fundamental form III is given by

$$III(\mathbf{u}, \mathbf{v}) = S_\mathbf{p}(\mathbf{u}) \cdot S_\mathbf{p}(\mathbf{v}). \tag{B.30}$$

Note that, III, in contrast to II, does not depend on the choice of the surface normal $\mathbf{N}$. The third fundamental theorem does not contain any new information, since it is expressible in terms of I

and II.

$$III - 2HII + KI = 0, \tag{B.31}$$

where H and K denote the mean curvature and the Gaussian curvature of $\mathscr{M}$.

The matrix of the first fundamental form I and the second fundamental form II is given by:

$$I : \begin{pmatrix} E & F \\ F & G \end{pmatrix} = \begin{pmatrix} \sigma_x \cdot \sigma_x & \sigma_x \cdot \sigma_y \\ \sigma_y \cdot \sigma_x & \sigma_y \cdot \sigma_y \end{pmatrix} = \begin{pmatrix} 1 + d_x^2 & d_x d_y \\ d_x d_y & 1 + d_y^2 \end{pmatrix} \tag{B.32}$$

$$II : \begin{pmatrix} l & m \\ m & n \end{pmatrix} = \begin{pmatrix} S_{\mathbf{p}}(\sigma_x) \cdot \sigma_x & S_{\mathbf{p}}(\sigma_x) \cdot \sigma_y \\ S_{\mathbf{p}}(\sigma_y) \cdot \sigma_x & S_{\mathbf{p}}(\sigma_y) \cdot \sigma_y \end{pmatrix} = \begin{pmatrix} \sigma_{xx} \cdot \mathbf{N} & \sigma_{xy} \cdot \mathbf{N} \\ \sigma_{yx} \cdot \mathbf{N} & \sigma_{yy} \cdot \mathbf{N} \end{pmatrix} = \frac{1}{\sqrt{1 + d_x^2 + d_y^2}} \begin{pmatrix} d_{xx} & d_{xy} \\ d_{xy} & d_{yy} \end{pmatrix} \tag{B.33}$$

Note that, although $S_{\mathbf{p}}$ is a symmetric bilinear operator, its matrix (a_{ij}) relative to $\{\sigma_x, \sigma_y\}$ need not be symmetric, because σ_x and σ_y are not in general perpendicular to one another.

With respect to the basis $\{\sigma_x, \sigma_y\}$, $\mathrm{d}\mathbf{N} = (\mathbf{N}_x, \mathbf{N}_y)^\top$ is given by the matrix $-I^{-1}II$ and $S_{\mathbf{p}}$ is given by $I^{-1}II$.

$$S_{\mathbf{p}} = I^{-1}II = \begin{pmatrix} a & c \\ b & d \end{pmatrix} = \frac{1}{EG - F^2} \begin{pmatrix} Gl - Fm & Gm - Fn \\ -Fl + Em & -Fm + En \end{pmatrix}$$

$$= \frac{1}{\left(1 + {d_x}^2 + {d_y}^2\right)^{\frac{3}{2}}} \begin{pmatrix} 1 + {d_y}^2 & -d_x d_y \\ -d_x d_y & 1 + {d_x}^2 \end{pmatrix} \begin{pmatrix} d_{xx} & d_{xy} \\ d_{xy} & d_{yy} \end{pmatrix}$$

$$= \frac{1}{\left(1 + {d_x}^2 + {d_y}^2\right)^{\frac{3}{2}}} \begin{pmatrix} \left(1 + {d_y}^2\right) d_{xx} - d_x d_y d_{xy} & \left(1 + {d_y}^2\right) d_{xy} - d_x d_y d_{yy} \\ \left(1 + {d_x}^2\right) d_{xy} - d_x d_y d_{xx} & \left(1 + {d_x}^2\right) d_{yy} - d_x d_y d_{xy} \end{pmatrix}. \tag{B.34}$$

Note that, the matrix $S_{\mathbf{p}}$ is not necessarily symmetric, unless we use the orthonormal basis $\{\mathbf{u}_x, \mathbf{u}_y\}$.

The Gaussian curvature is given by

$$K = \frac{ln - m^2}{EG - F^2} = \frac{d_{xx}\, d_{yy} - {d_{xy}}^2}{\left(1 + {d_x}^2 + {d_y}^2\right)^2}. \tag{B.35}$$

The mean curvature is given by

$$H = \frac{Gl + En - 2Fm}{2\left(EG - F^2\right)} = \frac{d_{xx}\left(1 + {d_y}^2\right) - 2d_{xy}\, d_x\, d_y + d_{yy}\left(1 + {d_x}^2\right)}{2\left(1 + {d_x}^2 + {d_y}^2\right)^{3/2}}. \tag{B.36}$$

B.1.6 Alternative formulas

Let $\sigma : U \to \mathscr{M} \in \mathbb{R}^3$ be a regular patch. Then

$$l = \frac{[\sigma_{xx}\ \sigma_x\ \sigma_y]}{\sqrt{EG - F^2}},\ m = \frac{[\sigma_{xy}\ \sigma_x\ \sigma_y]}{\sqrt{EG - F^2}},\ n = \frac{[\sigma_{yy}\ \sigma_x\ \sigma_y]}{\sqrt{EG - F^2}}, \tag{B.37}$$

where $[\cdot]$ denotes the scalar triple product.

Similarly, we can get K and H.

$$\begin{aligned} K &= \frac{[\sigma_{xx}\ \sigma_x\ \sigma_y]\ [\sigma_{yy}\ \sigma_x\ \sigma_y] - [\sigma_{xy}\ \sigma_x\ \sigma_y]}{\left(\|\sigma_x\|^2 \|\sigma_y\|^2 - (\sigma_x \cdot \sigma_y)^2\right)^2} \\ K &= \frac{[\sigma_{xx}\ \sigma_x\ \sigma_y]\, \|\sigma_y\|^2 - 2\,[\sigma_{xy}\ \sigma_x\ \sigma_y]\,(\sigma_x \cdot \sigma_y) + [\sigma_{yy}\ \sigma_x\ \sigma_y]\, \|\sigma_x\|^2}{2\left(\|\sigma_x\|^2 \|\sigma_y\|^2 - (\sigma_x \cdot \sigma_y)^2\right)^{\frac{3}{2}}} \end{aligned} \tag{B.38}$$

Bibliography

[1] S. Ahmed, M. Hansard, and A. Cavallaro. Constrained optimization for plane-based stereo. *IEEE Trans. on Image Processing*, 27(8):3870–3882, Aug 2018.

[2] J. Andrews and J. C. Séquin. Type-constrained direct fitting of quadric surfaces. *Computer Aided Design and Applications*, 2013.

[3] C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. PatchMatch: A randomized correspondence algorithm for structural image editing. *ACM Trans. Graph.*, 28(3):24, 2009.

[4] J. T. Barron and B. Poole. The fast bilateral solver. In *Proc. ECCV*, Oct 2016.

[5] F. Besse, C. Rother, A. Fitzgibbon, and J. Kautz. PMBP: PatchMatch belief propagation for correspondence field estimation. *Int. J. Comput Vision*, 110(1):2–13, 2014.

[6] M. J. Black. *Robust Incremental Optical Flow*. PhD thesis, Department of Computer Science, Yale University, New Haven, CT, USA, 1992.

[7] M. J. Black and P. Anandan. The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. *Comput. Vis. Image Underst.*, 63(1):75–104, 1996.

[8] M. Bleyer and C. Breiteneder. Stereo matching-state-of-the-art and research challenges. In *Advanced Topics in Computer Vision*, pages 143–179. Springer London, 2013.

[9] M. Bleyer, C. Rhemann, and C. Rother. PatchMatch stereo - stereo matching with slanted support windows. In *Proc. BMVC*, Sept 2011.

[10] J. Y. Bouguet. *The calibration toolbox for Matlab, example 5: Stereo rectification algorithm*. [Last accessed Aug.-2018] `http://www.vision.caltech.edu/bouguetj/calib_doc/htmls/example5.html`.

[11] G. Bradski and A. Kaehler. *Learning OpenCV: Computer Vision in C++ with the OpenCV Library*. O'Reilly Media, Inc., 2nd edition, 2013.

[12] S. Brahmbhatt. *Practical OpenCV*. Apress, 1st edition, 2013.

[13] D. C. Brown. Decentering distortion of lenses. *Photogrammetric Engineering*, 32(3):444–462, 1966.

[14] D. C. Brown. Close-range camera calibration. *Photogrammetric Engineering*, 37(8):855–866, 1971.

[15] W. Burger and M. J. Burge. *Principles of Digital Image Processing: Fundamental Techniques*. Springer Publishing Company, Incorporated, 1 edition, 2009.

[16] C. Castejón, B. L. Boada, D. Blanco, and L. Moreno. Traversable region modeling for outdoor navigation. *Journal of Intelligent and Robotic Systems*, 43(2):175–216, 2005.

[17] L. De-Maeztu, A. Villanueva, and R. Cabeza. Stereo matching using gradient similarity and locally adaptive support-weight. *Pattern Recognition Letters*, 32(13):1643 – 1651, 2011.

[18] D. Demirdjian and T. Darrell. Using multiple-hypothesis disparity maps and image velocity for 3-D motion estimation. *Int. J. Comput Vision*, 47(1-3):219–228, 2002.

[19] M. P. do Carmo. *Differential geometry of curves and surfaces*. Prentice Hall, 1976.

[20] R. O. Duda and P. E. Hart. Use of the hough transformation to detect lines and curves in pictures. *Commun. ACM*, 15(1):11–15, January 1972.

[21] O. Faugeras, T. Viéville, E. Theron, J. Vuillemin, B. Hotz, Z. Zhang, L. Moll, P. Bertin, H. Mathieu, P. Fua, G. Berry, and C. Proy. Real-time correlation-based stereo : algorithm, implementations and applications. Research Report RR-2013, INRIA, 1993.

[22] S. Foix, G. Alenya, and C. Torras. Lock-in time-of-flight (ToF) cameras: A survey. *IEEE Sensors J.*, 11(9):1917–1926, Sept 2011.

[23] A. Fusiello, V. Roberto, and E. Trucco. Efficient stereo with multiple windowing. In *Proc. CVPR*, pages 858–863, June 1997.

[24] S. Galliani, K. Lasinger, and K. Schindler. Massively parallel multiview stereopsis by surface normal diffusion. In *Proc. ICCV*, pages 873–881, Dec 2015.

[25] D. Gallup, J. M. Frahm, P. Mordohai, Y. Qingxiong, and M. Pollefeys. Real-time plane-sweeping stereo with multiple sweeping directions. In *Proc. CVPR*, pages 1–8, June 2007.

[26] S. K. Gehrig and U. Franke. Improving stereo sub-pixel accuracy for long range stereo. In *Proc. ICCV*, pages 1–7, Oct 2007.

[27] Leica Geosystems. *Cyclone*. `https://hds.leica-geosystems.com/en/Leica-Cyclone_6515.htm`.

[28] Leica Geosystems. *Leica HDS6200*. `https://hds.leica-geosystems.com/en/Leica-HDS6200_64228.htm`.

[29] Leica Geosystems. *Leica TS06 Plus*. https://leica-geosystems.com/en-GB/products/total-stations/manual-total-stations/leica-flexline-ts06plus.

[30] A. Gray, E. Abbena, and S. Salamon. *Modern Differential Geometry of Curves and Surfaces with Mathematica, Third Edition (Studies in Advanced Mathematics)*. Chapman & Hall/CRC, 2006.

[31] Z. Gu, X. Su, Y. Liu, and Q. Zhang. Local stereo matching with adaptive support-weight, rank transform and disparity calibration. *Pattern Recognition Letters*, 29(9):1230 – 1235, 2008.

[32] Y. HaCohen, E. Shechtman, D. B. Goldman, and D. Lischinski. Non-rigid dense correspondence with applications for image enhancement. *ACM Trans. Graph.*, 30(4), July 2011.

[33] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004.

[34] H. Hattori and A. Maki. Stereo matching with direct surface orientation recovery. In *Proc. BMVC*, pages 356–366, Sept 1998.

[35] M. Hebert. Outdoor scene analysis using range data. In *Proc. ICRA*, volume 3, pages 1426–1432, Apr 1986.

[36] P. Heise, S. Klose, B. Jensen, and A. Knoll. PM-Huber: PatchMatch with Huber regularization for stereo matching. In *Proc. ICCV*, pages 2360–2367, Dec 2013.

[37] H. Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Trans. PAMI*, 30(2):328–341, Feb 2008.

[38] H. Hirschmuller, P. R. Innocent, and J. Garibaldi. Real-time correlation-based stereo vision with reduced border errors. *Int. J. Comput. Vision*, 47(1-3):229–246, apr 2002.

[39] H. Hirschmuller and D. Scharstein. Evaluation of stereo matching costs on images with radiometric differences. *IEEE Trans. PAMI*, 31(9):1582–1599, Sept 2009.

[40] R. Hoffman and A. K. Jain. Segmentation and classification of range images. *IEEE Trans. PAMI*, 9(5):608–620, May 1987.

[41] A. Hosni, M. Bleyer, and M. Gelautz. Secrets of adaptive support weight techniques for local stereo matching. *Comput. Vis. Image Underst.*, 117(6):620 – 632, 2013.

[42] A. Hosni, M. Bleyer, C. Rhemann, M. Gelautz, and C. Rother. Real-time local stereo matching using guided image filtering. In *Proc. ICME*, pages 1–6, July 2011.

[43] Y. Huang, M. K. Ng, and Y. W. Wen. A fast total variation minimization method for image restoration. *Multiscale Modeling & Simulation*, 7(2):774–795, 2008.

[44] S. Huq, A. Koschan, and M. Abidi. Occlusion filling in stereo: Theory and experiments. *Comput. Vis. Image Underst.*, 117(6):688 – 704, 2013.

[45] Itseez. *The OpenCV Reference Manual*, 3.4.1 edition, Feb. 2018.

[46] S. G. Johnson. *The NLopt nonlinear-optimization package*. [Last accessed Dec.-2017] `http://ab-initio.mit.edu/nlopt`.

[47] T. Kanade and M. Okutomi. A stereo matching algorithm with an adaptive window: theory and experiment. In *Proc. ICRA*, pages 1088–1095, April 1991.

[48] T. Kanade and M. Okutomi. A stereo matching algorithm with an adaptive window: Theory and experiment. *IEEE Trans. PAMI*, 16(9):920–932, September 1994.

[49] A. Klaus, M. Sormann, and K. Karner. Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure. In *Proc. ICPR*, volume 3, pages 15–18, Aug 2006.

[50] R. Klette. *Concise Computer Vision: An Introduction into Theory and Algorithms*. Springer Publishing Company, Incorporated, 2014.

[51] J. J. Koenderink and A. J. V. Doorn. Surface shape and curvature scales. *Image and Vision Computing*, 10(8):557–565, Oct 1992.

[52] J. Kowalczuk, E. T. Psota, and L. C. Perez. Real-time stereo matching on CUDA using an iterative refinement method for adaptive support weight correspondences. *IEEE Trans. CSVT*, 23(1):94–104, 2013.

[53] G. Li and S. W. Zucker. Differential geometric inference in surface stereo. *IEEE Trans. PAMI*, 32(1):72–86, Jan 2010.

[54] L. Li, S. Zhang, X. Yu, and L. Zhang. Pmsc: Patchmatch-based superpixel cut for accurate stereo matching. *IEEE Trans. on Circuits and Systems for Video Technology*, 28(3):679–692, March 2018.

[55] Z. N. Li and G. Hu. Analysis of disparity gradient based cooperative stereo. *IEEE Trans. on Image Processing*, 5(11):1493–1506, Nov 1996.

[56] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. Comput Vision*, 60(2):91–110, November 2004.

[57] R. Luus and T. H. I. Jaakola. Optimization by direct search and systematic reduction of the size of search region. *AIChE Journal*, 19(4):760–766, 1973.

[58] Y. Ma, S. Soatto, J. Kosecka, and S. S. Sastry. *An Invitation to 3-D Vision: From Images to Geometric Models*. SpringerVerlag, 2003.

[59] K. D. Mankoff and T. A. Russo. The Kinect: a low-cost, high-resolution, short-range 3D camera. *Earth Surface Processes and Landforms*, 38(9):926–936, July 2013.

[60] G. Marsaglia. Choosing a point from the surface of a sphere. *Ann. Math. Statist.*, 43(2):645–646, 1972.

[61] X. Mei, X. Sun, M. Zhou, S. Jiao, H. Wang, and X. Zhang. On building an accurate stereo matching system on graphics hardware. In *Proc. ICCV Workshop*, pages 467–474, Nov 2011.

[62] D. Min, J. Lu, and M. N. Do. Joint histogram-based cost aggregation for stereo matching. *IEEE Trans. PAMI*, 35(10):2539–2545, Oct 2013.

[63] P. Mordohai and G. Medioni. Stereo using monocular cues within the tensor voting framework. *IEEE Trans. PAMI*, 28(6):968–982, June 2006.

[64] K. Mühlmann, D. Maier, J. Hesser, and R. Männer. Calculating dense disparity maps from color stereo images, an efficient implementation. *Int. J. Comput. Vision*, 47(1-3):79–88, April 2002.

[65] G. G. Nair. On the convergence of the LJ search method. *Journal of Optimization Theory and Applications*, 28(3):429–434, Jul 1979.

[66] L. Nalpantidis, C. S. Georgios, and Antonios. G. Review of stereo vision algorithms: From software to hardware. *Int. J. Optomechatronics*, 2(4):435–462, 2008.

[67] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon. KinectFusion: Real-time dense surface mapping and tracking. In *Proc. ISMAR*, pages 127–136, Oct 2011.

[68] M. Nitsche, J. M. Turowski, A. Badoux, D. Rickenmann, T. K. Kohoutek, M. Pauli, and J. W. Kirchner. Range imaging: a new method for high-resolution topographic measurements in small- and medium-scale field sites. *Earth Surface Processes and Landforms*, 38(8):810–825, 2013.

[69] J. Oprea. *Differential Geometry and its Applications*. Prentice Hall, 1997.

[70] R. B. Potts. Some generalized order-disorder transformations. *Mathematical Proc. of the Cambridge Philosophical Society*, 48(1):106–109, 1952.

[71] M. J. D. Powell. The BOBYQA algorithm for bound constrained optimization without derivatives. *(Report No. DAMTP 2009/NA06). Centre for Mathematical Sciences, University of Cambridge, UK.*, Aug 2009.

[72] E. T. Psota, J. Kowalczuk, M. Mittek, and L. C. Perez. MAP disparity estimation using hidden markov trees. In *Proc. ICCV*, pages 2219–2227, Dec 2015.

[73] N. Qian. Binocular disparity and the perception of depth. *Neuron*, 18(3):359–368, Mar 1997.

[74] C. Rhemann, A. Hosni, M. Bleyer, C. Rother, and M. Gelautz. Fast cost-volume filtering for visual correspondence and beyond. In *Proc. CVPR*, pages 3017–3024, June 2011.

[75] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Int. J. Comput Vision*, 47(1-3):7–42, 2002.

[76] D. Scharstein, R. Szeliski, and H. Hirschmuller. *Middlebury Stereo Vision*. [Last accessed Aug.-2018] `http://vision.middlebury.edu/stereo/`.

[77] D. Scharstein, T. Taniai, and S. N. Sinha. Semi-global stereo matching with surface orientation priors. *CoRR*, abs/1712.00818, 2017.

[78] A. V. Segal, D. Haehnel, and S. Thrun. Generalized-ICP. In *Robotics: Science and Systems*. The MIT Press, 2009.

[79] H. Sekkati and S. Negahdaripour. *3-D Motion Estimation for Positioning from 2-D Acoustic Video Imagery*. Springer Berlin Heidelberg, 2007.

[80] S. N. Sinha, D. Scharstein, and R. Szeliski. Efficient high-resolution stereo matching using local plane sweeps. In *Proc. CVPR*, pages 1582–1589, June 2014.

[81] GPL software. *CloudCompare (version 2.5)*. `https://www.danielgm.net/cc/`.

[82] R. Staiger. Terrestrial laser scanning - technology, systems and applications. In *Proc. 2nd FIG Regional Conference*, Dec 2003.

[83] C. V. Stewart, R. Y. Flatland, and K. Bubna. Geometric constraints and stereo disparity computation. *Int. J. Comput Vision*, 20(3):143–168, 1996.

[84] R. Szeliski. *Computer Vision: Algorithms and Applications*. Springer, 2010.

[85] R. Szeliski and D. Scharstein. Sampling the disparity space image. *IEEE Trans. PAMI*, 26(3):419–425, March 2004.

[86] K. Tapp. *Differential Geometry of Curves and Surfaces*. Springer International Publishing, 2016.

[87] O. Veksler. Stereo correspondence with compact windows via minimum ratio cycle. *IEEE Trans. PAMI*, 24(12):1654–1660, Dec 2002.

[88] J. Žbontar and Y. LeCun. Stereo matching by training a convolutional neural network to compare image patches. *Journal of Machine Learning Research*, 17(65):1–32, 2016.

[89] M. J. Westoby, J. Brasington, N. F. Glasser, M. J. Hambrey, and J. M. Reynolds. Structure-from-Motion photogrammetry: A low-cost, effective tool for geoscience applications. *Geomorphology*, 179(0):300 – 314, 2012.

[90] O. Woodford, P. Torr, I. Reid, and A. Fitzgibbon. Global stereo reconstruction under second-order smoothness priors. *IEEE Trans. PAMI*, 31(12):2115–2128, Dec 2009.

[91] K. J. Yoon and I. S. Kweon. Adaptive support-weight approach for correspondence search. *IEEE Trans. PAMI*, 28(4), 2006.

[92] S. Yoon, S. K. Park, S. Kang, and Y. K. Kwak. Fast correlation-based stereo matching with the reduction of systematic errors. *Pattern Recognition Letters*, 26(14):2221–2231, 2005.

[93] Y. X. Yuan. Recent advances in trust region algorithms. *Mathematical Programming*, 151(1):249–281, 2015.

[94] C. Zhang, Z. Li, Y. Cheng, R. Cai, H. Chao, and Y. Rui. MeshStereo: A global stereo model with mesh alignment regularization for view interpolation. In *Proc. ICCV*, June 2015.

[95] Z. Zhang. A flexible new technique for camera calibration. *IEEE Trans. PAMI*, 22(11):1330–1334, Nov 2000.

[96] Z Zhang. Microsoft kinect sensor and its effect. *IEEE MultiMedia*, 19:4–12, April 2012.