



THE UNIVERSITY
of ADELAIDE

**As-Projective-As-Possible Image Stitching with
Moving DLT**

by

Julio César Hernández Zaragoza

A thesis submitted for the degree of Doctor of Philosophy

in the

Faculty of Engineering, Computer and Mathematical Sciences
School of Computer Science

June 2014

Declaration of Authorship

I certify that this work contains no material which has been accepted for the award of any other degree or diploma in any university or other tertiary institution and, to the best of my knowledge and belief, contains no material previously published or written by another person, except where due reference has been made in the text. In addition, I certify that no part of this work will, in the future, be used in a submission for any other degree or diploma in any university or other tertiary institution without the prior approval of the University of Adelaide and where applicable, any partner institution responsible for the joint-award of this degree. I give consent to this copy of my thesis when deposited in the University Library, being made available for loan and photocopying, subject to the provisions of the Copyright Act 1968. The author acknowledges that copyright of published works contained within this thesis resides with the copyright holder(s) of those works. I also give permission for the digital version of my thesis to be made available on the web, via the University's digital research repository, the Library catalogue and also through web search engines, unless permission has been granted by the University to restrict access for a period of time.

In carrying out the research that underlies this thesis the following papers were published or are currently under review:

1. Julio Zaragoza, Tat-Jun Chin, Michael Brown and David Suter, "As-Projective-As-Possible Image Stitching with Moving DLT", in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, Portland, Oregon, USA, June, 2013.
2. Julio Zaragoza, Tat-Jun Chin, Quoc-Huy Tran, Michael Brown and David Suter, "As-Projective-As-Possible Image Stitching with Moving DLT", in *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, November, 2013.
3. Quoc-Huy Tran, Tat-Jun Chin, Julio Zaragoza, Michael Brown and David Suter, "Outlier Rejection in Deformable Registration with Moving Least Squares", in *Transactions on Image Processing (TIP)*, manuscript submitted for review.

Julio César Hernández Zaragoza Signed: _____ Date: _____

“Anybody who has been seriously engaged in scientific work of any kind realises that over the entrance to the gates of the temple of science are written the words: ‘Ye must have faith.’”

Max Planck

THE UNIVERSITY OF ADELAIDE

Abstract

Faculty of Engineering, Computer and Mathematical Sciences

School of Computer Science

Doctor of Philosophy

by Julio César Hernández Zaragoza

The last ten years have witnessed important advances in image stitching algorithms. Such advances have allowed the development of several commercial tools that are based on or incorporate image stitching. Amongst these tools there are well known image editing suites like Adobe Photoshop, Microsoft's Image Composite Editor which is part of the web-based photo organization tool Photosynth, "dedicated" stitching software like Autostitch and its commercial counterparts AutoPano and AutoPano Giga, the image stitching functionality of the iOS from Apple, as well as the built-in stitching functionality of several off-the-shelf digital cameras.

The widespread availability of stitching tools often leads to the impression that image stitching is a solved problem. The reality is: many of these tools often fail to produce convincing results when given *non ideal data*, i.e., images that deviate from fairly *restrictive assumptions* of image stitching; the main two being that the photos correspond to views that differ purely by rotation, or that the imaged scene is effectively planar. Such assumptions underpin the usage of 2D projective transforms or homographies to align the photos. In the hands of the casual user, these conditions are often violated, yielding misalignment artifacts or "ghosting" in the results. Accordingly, many existing image stitching tools depend critically on post-processing routines to conceal ghosting.

This thesis proposes a novel estimation technique called Moving Direct Linear Transformation (*Moving DLT*) that is able to "tweak" or fine-tune the projective warp to accommodate the deviations of the input data from the idealised conditions. This produces "*as-projective-as-possible*" image alignments that significantly reduce ghosting without compromising the geometric realism of perspective image stitching. The Moving DLT technique lessens the dependency on potentially expensive post-processing algorithms.

In addition, this thesis also describes how Moving DLT can be performed in a “bundled” manner to simultaneously align multiple images in order to generate “long” panoramas while reducing the error propagation of the incremental stitching techniques. It is important to note that such a bundle adjustment formulation, which we call *Bundled Moving DLT*, is the first of its kind. There is no other bundle adjustment formulation that is able to simultaneously refine multiple non-rigid warps for image stitching.

The experimental results show that Moving DLT (and Bundled Moving DLT) can produce much better results than current state-of-the-art image stitching software and other recent methods for image stitching.

Acknowledgements

There is a lot of people that I would like to thank for accompanying me on the amazing adventure that this PhD has represented. First of all I would like to thank my principal supervisor: Dr. Tat-Jun Chin (TJ). It is no exaggeration to say that this research could not have been possible without the excellent guidance and support from TJ. Thank you so much for the countless hours of discussions, thank you so much for the ideas, corrections, suggestions, modifications and changes, but above all, thank you so much TJ for showing me how *great research* is done. I would not change such experience for the world.

I would also like to thank my co-supervisor Prof. David Suter for offering this amazing and life-changing opportunity, for taking me under his wing, for providing novel insights and giving fantastic advice on some of the initial and unpolished ideas that were presented over the course of three years.

Besides TJ and David, I would also like to thank Dr. Qinfeng (Javen) Shi for all of the help, support and advice he offered me during my candidature, specially during the early days. But in particular, thanks a lot for all of the “non-research-related” comments and suggestions. Thanks a lot for having an open window when all of the other doors were closed.

During my time as a PhD student I performed some teaching activities as well. Such activities improved several aspects of my professional life like self confidence and communication skills, and performing these activities also made me realise how much I enjoy the “teaching experience”. I owe this great opportunity to Dr. Claudia Szabo who has given me this chance for the last 2.5 years. Oh and thanks a lot for the weekly chocs of course!

Also, I would like to thank my lab-mates, Sim, Xue, Huy, Trung, Guosheng and Alvaro for sharing the experience. Thank you guys for the useful discussions, help, tips and advice. Thanks a lot for the amazing and funny trips and meals, but most importantly, thanks a lot for the support and friendship.

Lastly, the most important person in the world that I would like to thank to, is my mom. Thank you so much for all of the love and support you have offered me from day one of my life. Thanks a lot for being my number one fan. Thanks a lot for always giving your best for me, for encouraging me, for helping me, for giving me advice, for always pushing me forward, but most of all, thank you so much for being my mom. *Te quiero hoy y siempre hasta el “delfinito” y de regreso y mil veces más.*

Contents

Declaration of Authorship	iii
Abstract	vii
Acknowledgements	ix
Contents	x
List of Figures	xv
List of Tables	xix
List of Algorithms	xxi
Abbreviations	xxii
1 Introduction	1
1.1 Background and Motivation	1
1.2 Contributions of the Thesis	7
1.3 Thesis Structure	9
1.3.1 Chapter 2	9
1.3.2 Chapter 3	9
1.3.3 Chapter 4	9
1.3.4 Chapter 5	10
1.3.5 Chapter 6	10
1.3.6 Chapter 7	10
2 Background	11
2.1 Introduction	11
2.2 The Homography Matrix	13
2.2.1 Plane Induced Homography	13
2.2.2 Rotation Induced Homography	16
2.3 Direct Linear Transformation (DLT)	18
2.4 Normalised Direct Linear Transformation	20
2.5 Image Stitching with Projective Transformations	22

2.6	Summary	25
3	Related Work	27
3.1	Introduction	27
3.2	Pairwise Stitching	28
3.3	Bundle Adjustment and Local Refinements	33
3.4	3D Reconstruction and Plane-Plus-Parallax	38
3.5	Direct Estimation of Flexible Warps	42
3.6	Arbitrary Camera Motions	47
3.7	Summary	48
4	As-Projective-As-Possible Warps	51
4.1	Introduction	51
4.2	Moving Direct Linear Transformation (Moving DLT)	52
4.3	Efficient Computation for Image Stitching	55
4.3.1	Partitioning into cells	55
4.3.2	Updating weighted SVDs	57
4.4	Comparing Moving DLT against (affine) Moving Least Squares for Image Stitching	58
4.5	Summary	62
5	Simultaneous Refinement of Multiple As-Projective-As-Possible Warps	63
5.1	Introduction	63
5.2	Selecting the Reference Frame	63
5.3	Bundled Moving Direct Linear Transformation (Bundled Moving DLT)	64
5.4	Comparing Bundled Moving DLT and Bundle Adjustment	67
5.5	Summary	69
6	Experiments and Results	73
6.1	Introduction	73
6.2	Comparisons with Flexible Warp Methods	74
6.2.1	Preprocessing and Parameter Settings	74
6.2.2	Qualitative Comparisons	75
6.2.3	Runtime Information	100
6.2.4	Quantitative Benchmarking	100
6.3	Comparisons with Bundle Adjustment	103
6.3.1	Stitching Full Panoramas without post-processing	103
6.3.2	Stitching Full Panoramas with post-processing	107
6.4	Summary	114
7	Conclusions	115
7.1	Future Work	116
7.1.1	Non-Rigid Image Registration	116
7.1.2	Non-Rigid Structure from Motion	118
7.1.3	Video Stabilisation	121
7.2	Drawbacks and Limitations	123

A	Image Sets Used in the Experiments	125
A.1	Images Used in Pairwise Stitching	125
A.2	Images Used For Stitching Full Panoramas	128
B	The Image Formation Process	131
B.1	The Camera Matrix	132
B.1.1	External Parameters	132
B.1.2	Camera Intrinsic	133
B.2	Relating Points in Two Views through the Camera Matrix	134
	Bibliography	137

List of Figures

1.1	An example of a high resolution panorama generated with Autostitch.	2
1.2	One of the panoramas generated by Google’s street views.	3
1.3	Illustrating the stitching problem when the images to align do not follow the prescribed conditions of the projective model.	4
1.4	A magnified (and cropped) view of Fig. 1.2 that shows some of the artifact errors caused by failed post-processing routines in image stitching.	6
2.1	A point in a 3D plane being projected into two image planes.	14
2.2	A homography matrix relating points from a planar view.	16
2.3	A 3D point projected into two views undergoing a pure (3D) rotation motion.	18
2.4	Demonstrating image stitching with the Direct Linear Transformation method. The input images correspond to views that only differ by rotation.	23
2.5	1D analogy of image stitching with projective warps.	24
2.6	Demonstrating image stitching with the Direct Linear Transformation method. The input images correspond to views that differ by rotation <i>and</i> translation.	26
3.1	Stitching long panoramas through incremental (pairwise) approaches.	29
3.2	An example of a panorama for video compression.	31
3.3	The gap closing technique for image stitching.	32
3.4	An example of a panorama generated by means of Brown and Lowe’s Bundle Adjustment method [12].	34
3.5	Example of the panorama recognition process of Brown and Lowe [12].	34
3.6	The bundle adjustment process of Marzotto [58] for panorama creation.	36
3.7	A “joiner” example from [94].	38
3.8	An example of a multi-view panorama.	39
3.9	The 3D reconstruction approach for image stitching of [99].	40
3.10	A video summary created with the plane+parallax approach of [36].	41
3.11	An overview of the Smoothly Varying Affine Image Stitching method from [51].	43
3.12	The hierarchical non-rigid image alignment process of [52].	44
3.13	An example of the content preserving warps for video stabilisation.	45
3.14	The dual homography warps for image stitching from [23].	46
3.15	Two panorama examples generated with the adaptive manifold mosaicking technique of Peleg <i>et al.</i> [65]. This example appears in [65].	47
3.16	The imaging process of the pushbroom camera modeled by the multi-perspective projection process of [62, 64, 66].	48
3.17	An anaglyph image constructed from a stereo panorama generated with the panoramic stereo imaging method of [66].	48
4.1	1D analogy of image stitching with as-projective-as-possible warps.	54

4.2	Demonstrating image stitching with the proposed Moving DLT method.	56
4.3	Histogram of number of weights $\neq \gamma$ for the cells in Fig. 2.6(b).	57
4.4	An affine-as-possible warp obtained by means of Moving Least Squares.	59
4.5	Image stitching results obtained with as-affine-as-possible warps generated with MLS.	60
4.6	Image stitching results obtained with APAP warps generated with Moving DLT.	61
4.7	Feature points extracted from the <i>temple</i> image pair.	61
5.1	Selecting the reference frame for Bundled Moving DLT.	64
5.2	Stitching results obtained with the proposed Bundled Moving DLT approach for panorama creation.	66
5.3	(Two page figure) Comparing the “raw” image alignment results of different techniques for panorama creation.	70
6.1	(Two page figure) Qualitative comparisons on the <i>railtracks</i> image pair.	78
6.2	(Two page figure) Qualitative comparisons on the <i>temple</i> image pair.	80
6.3	(Two page figure) Qualitative comparisons on the <i>bikes</i> image pair.	82
6.4	(Two page figure) Qualitative comparisons on the <i>construction site</i> image pair.	84
6.5	(Two page figure) Qualitative comparisons on the <i>train</i> image pair.	86
6.6	(Two page figure) Qualitative comparisons on the <i>garden</i> image pair.	88
6.7	(Two page figure) Qualitative comparisons on the <i>carpark</i> image pair.	90
6.8	(Two page figure) Qualitative comparisons on the <i>apartments</i> image pair.	92
6.9	(Two page figure) Qualitative comparisons on the <i>chess/girl</i> image pair.	94
6.10	(Two page figure) Qualitative comparisons on the <i>rooftops</i> image pair.	96
6.11	(Two page figure) Qualitative comparisons on the <i>couch</i> image pair.	98
6.12	Point cloud and average RMSE on the training set and the testing set as a function of inter-camera translational distance.	102
6.13	Panorama results without post-processing on the <i>construction site</i> image set.	104
6.14	Panorama results without post-processing on the <i>garden</i> image set.	105
6.15	Panorama results without post-processing on the <i>train</i> image set.	106
6.16	(Two page figure) Panorama results with post-processing on the <i>construction site</i> image set.	108
6.17	(Two page figure) Panorama results with post-processing on the <i>garden</i> image set.	110
6.18	(Two page figure) Panorama results with post-processing on the <i>train</i> image set.	112
7.1	Typical input data for Non-Rigid Image Registration tasks.	117
7.2	Examples of Non-Rigid Image Registration results obtained with Moving DLT	117
7.3	Non-Rigid Structure from Motion results obtained with Moving DLT	120
7.4	An example of the dense 3D reconstruction results of [25]	121
7.5	Comparing single and bundled camera paths for video stabilisations.	122
7.6	The image stabilisation process of Bundled Camera Paths.	123
A.1	<i>railtracks</i> image pair. Size of images: 2000×1500 pixels. Number of inliers after RANSAC is: 2753.	125
A.2	<i>temple</i> image pair [23]. Size of images: 730×487 pixels. Number of inliers after RANSAC is: 415.	126
A.3	<i>carpark</i> image pair [23]. Size of images: 653×490 pixels. Number of inliers after RANSAC is: 359.	126

A.4	<i>apartment</i> image pair [23]. Size of images: 1632×1224 pixels. Number of inliers after RANSAC is: 2634.	126
A.5	<i>chess/girl</i> image pair [51]. Size of images: 1824×1368 pixels. Number of inliers after RANSAC is: 1381.	127
A.6	<i>couch</i> image pair [51]. Size of images: 1824×1368 pixels. Number of inliers after RANSAC is: 1329.	127
A.7	<i>rooftops</i> image from [51]. Size of images: 320×240 pixels. Number of inliers after RANSAC is: 161.	127
A.8	<i>bikes</i> image pair. Size of images: 1632×1224 pixels. Number of inliers after RANSAC is: 2561.	128
A.9	<i>construction site</i> dataset. Size of images: 2000×1329 pixels. Number of inliers between the pair of images used for the pairwise experiments in Section 6.2: 5068. Number of inliers between the images used for the panorama experiments in Section 6.3: 12616.	128
A.10	<i>garden</i> dataset. Size of images: 2000×1329 pixels. Number of inliers between the pair of images used for the pairwise experiments in Section 6.2: 4567. Number of inliers between the images used for the panorama experiments in Section 6.3: 10693.	129
A.11	<i>train</i> dataset. Size of images: 2456×1632 pixels. Number of inliers between the pair of images used for the pairwise experiments in Section 6.2: 4231. Number of inliers between the images used for the panorama experiments in Section 6.3: 13380.	129
B.1	The image formation process through the pinhole camera model.	131
B.2	Example of the simplified camera intrinsic parameters.	133
B.3	Relationship between the projections or pictures of a 3D point in two different images.	135

List of Tables

2.1	Hierarchy of 2D linear transformations in projective space.	12
6.1	Average RMSE (in pixels) and % outliers over 20 repetitions for 5 methods on 11 image pairs. See Figs. 6.1 to 6.11 to view the qualitative stitching results. . .	101

List of Algorithms

1	Direct Linear Transform algorithm for the estimation of a 2D projective warp (homography).	21
2	Normalised Direct Linear Transform algorithm for the estimation of a 2D projective warp (homography).	22
3	Moving Direct Linear Transform algorithm for the estimation of a 2D as-projective-as-possible warp.	54
4	Simultaneous refinement of multiple as-projective-as-possible warps for panorama creation.	67

Abbreviations

2D	Two-Dimensional
3D	Three-Dimensional
APAP	As-Projective-As-Possible
Bundled Moving DLT	Bundled Moving Direct Linear Transformation
CPW	Content Preserving Warps
DHW	Dual-Homography warps
DLT	Direct Linear Transformation
MRF	Markov Random Field
Moving DLT	Moving Direct Linear Transformation
MLS	Moving Least Squares
SLAM	Simultaneous Localisation and Mapping
SVA	Smoothly Varying Affine
SVD	Singular Value Decomposition

To my mom, for a lifetime of love and support.

Chapter 1

Introduction

1.1 Background and Motivation

Image stitching is one of the earliest and long-standing problems in the computer vision and photogrammetry communities. The task of stitching consists of *aligning multiple overlapping images* of a scene into a common reference frame or canvas in order to generate seamless high resolution mosaics.

During the last decade, people has been able to witness impressive advances in image stitching research. Such progress has probably been possible thanks to the growth and widespread availability of computer-based technology like digital cameras, tablets and smart-phones, which generate massive amounts of rich and diverse visual data. It is thanks to the combination of this new technology and new research that users are currently able to organise, store, analyse, share, discover, interact and “get immersed” in huge amounts of images in new and exciting ways.

For example, currently, it is possible to make use of photo organisation tools like Microsoft’s *Photosynth*¹ or *Autostitch*², which allow the creation of ultra high resolution panoramas from unordered collections of images that casual users obtain during their holidays and trips (Fig. 1.1 shows an example of a high resolution panorama). In fact, Microsoft is currently pushing the boundaries of the image stitching technologies in Photosynth, in order to give users the ability to generate 3D panoramas, which will add another level of realism to the panoramic mosaics that people can generate (see <http://photosynth.net/preview> for more details).

¹www.photosynth.net

²www.cs.bath.ac.uk/brown/autostitch/autostitch.html



Figure 1.1: (First row) An example of a high resolution panorama generated with the commercial software Autostitch. The panorama was generated by stitching a sequence of 18 overlapping images at Adelaide's University campus. The size of the panorama is 12700×2100 pixels. (Second Row) 5 of the images used for generating the panorama in the first row.

As another example, *Google* allows users to “virtually visit” and contemplate thousands of cities around the world thanks to their *street view* technology embedded in *Google maps*³. A street view is a seamless panorama generated by stitching the frames of videos captured by fleets of specially adapted vehicles such as cars, tricycles, snowmobiles and boats. It is these panoramas what gives the user the ability to navigate along the streets of remote and exciting places of the world (Fig. 1.2 shows an example of a panorama generated by Google’s street view).



Figure 1.2: One of the panoramas generated by Google’s street views. The panoramas are generated by stitching frames of videos taken with specially adapted vehicles like cars, tricycles, boats or snowmobiles.

It is also image stitching-related research what allows the stabilisation of shaky videos taken with hand held cameras or smartphones [53, 55] so that they look as taken by professionals, or taken by use of special stabilisation equipment; and it is also image stitching research what led to the development of some the most commonly used, motion compensated, video compression algorithms [37, 47]⁴ so that it is possible to easily store and share videos in websites like *YouTube*⁵.

Interestingly, most of the current image stitching solutions (including Autostitch, Photosynth and Google’s street view) make use of 2D *projective transformations* or *homographies* (refined by means of bundle adjustment [11, 12, 74]) in order to bring the images into alignment.

³maps.google.com

⁴The work of [47], constitutes Microsoft’s proposal to the MPEG [46] standardization process.

⁵www.youtube.com



Figure 1.3: Illustrating the stitching problem when the images to align do not follow the prescribed conditions of the projective model. Row 1: *Raw alignment result* from Autositch [12] with significant ghosting due to the inability of the projective warp at characterising the required warp. Rows 2 and 3: Final results (with advanced pixel compositing) from Autositch and Photosynth, where glaring artifacts exist. Row 4: *Raw alignment result* using as-projective-as-possible (APAP) warps obtained with the proposed Moving DLT method. The alignment results exhibit little noticeable ghosting.

However, (as detailed in the following Chapter) projective transformations are justified only if the images to align come from views that *differ purely by rotation, or if the imaged scene is effectively planar*, e.g., when the scene is sufficiently far away [76]. In fact, many commercial image stitching tools specify the previous input conditions, at least implicitly; see for example the FAQ pages on Autostitch⁶ and Photosynth⁷.

Realistically, these *imaging restrictions* are difficult for casual users to satisfy. Casual users are, in general, not familiar with image stitching fundamentals or even if they are familiar with image stitching, achieving a pure rotational or *parallax free* motion with a hand-held camera or smart-phone is a difficult task. Secondly, the desire to stitch a collection of images may be an afterthought, i.e., when it is not possible to revisit the scene to reshoot under the required imaging conditions. Lastly, for some particular applications that make use of image stitching, capturing the images under the required imaging conditions is an impractical (or even impossible) task to do.

Unfortunately, if the images to align deviate from the aforementioned imaging restrictions, a projective model will not be able to adequately characterise the required warp, thus, producing misalignments or ghosting effects in the final stitched results. Fig. 1.3 (row 1) shows a “raw” stitching result (i.e., the mosaic is composited only with simple intensity averaging) from Autostitch that exhibits significant parallax errors. Note that this problem is due primarily to the *inadequacy of the projective model in characterising the required warp*, and not inaccuracies in the warp estimation process.

In order to solve this issue, most of the currently available stitching tools often make use of post-processing or de-ghosting techniques like seam cutting [2, 17], Laplacian pyramid blending [12, 13], Poisson image blending [67], multiband blending [11, 12] or patch synthesis [15] that try to conceal the alignment errors. The problem is, if the image alignments are bad to begin with, many state-of-the-art techniques cannot produce convincing results, *even with advanced pixel compositing or post-processing*. Fig. 1.3 (rows 2 and 3) depicts the final postprocessed results from Autostitch and Photosynth where unwanted artifacts evidently still exist.

Another example of a failed post-processing algorithm for image stitching appears in Fig. 1.2 (Fig. 1.4 offers a magnified and cropped view of this issue). By taking a close look to the original figure it is possible to notice how the stitched contents of the image, in particular, the first level of the Eiffel tower, was not aligned in a “satisfactory” way and, as consequence, the post-processing routines introduced artifacts in the final composited mosaic. There are several

⁶www.cs.bath.ac.uk/brown/autostitch/autostitch.html#FAQ

⁷photosynth.net/faq.aspx

examples like this in Google maps; the most probable cause of this issue is the parallax induced by the inherent *translation motion* of the vehicles that capture the videos. Google's street view is one of the applications of image stitching where capturing the image data under controlled conditions becomes an infeasible task.



Figure 1.4: A magnified (and cropped) view of Fig. 1.2 that shows some of the artifact errors (circled in red) caused by failed post-processing routines in image stitching. This problem is probably because most of the images used in Google's street view are obtained with cameras mounted on vehicles, undergoing translational motions.

Instead of focusing on concealing the image alignment errors, other work on image stitching has been oriented towards improving the image alignment functions. Amongst the proposed solutions there are methods that perform some form of 3D reconstruction in order to infer the 3D structure and camera matrices of the imaged scene [3, 99] and stitch the images by making use of this information; methods that make use of two-step approaches for locally improving over initial image alignments [72, 74]; methods that make use of plane+parallax approaches that attempt at modelling the parallax motion caused by the image translations and varying scene depths [16, 36, 89] and methods that aim at obtaining a “stitching order” for minimising the visible image alignment errors and artifacts [40, 58]. However, most of these methods rely on overly complicated, multi-stage, heuristic or computationally expensive solutions for image alignment tasks. This is probably the reason why current stitching tools still prefer to make use of the basic (and simple) projective transformation for performing the alignments.

Only in more recent years it is that the work on image alignment has focused on the generation of methods for the direct estimation of *flexible warps*. Flexible warps constitute one of the most recent advances in image stitching research. The goal of these methods is to directly adapt the (2D) image warping function in order to stitch the uncooperative images, thus, discarding the need for 3D reconstructions or other elaborate steps. Unfortunately, some of the current solutions for flexible warps (e.g., [53]) either constrain the flexibility of the image alignments,

or cannot be applied to general scenes (e.g., [23]), or are flexible and can be applied to (more) general scenes but make use of affine regularisations [51, 52].

Affine regularisations may be suboptimal for image stitching since an affinity does not contain sufficient degrees of freedom to achieve a projective warp [76] but, more importantly, affine regularisations tend to introduce distortions in the areas where the images do not overlap. This occurs because, in such regions, there is no data to guide the local deformation and the warp reverts to global affinity (this point will be further discussed and demonstrated in Chapters 4 and 6 of this thesis).

The above problem raises a strong motivation for improving current methods for “flexible image stitching”. In particular, the research in this thesis relies on the premise that a much better approach for image stitching is to “give *homographies*” the ability to align the data that deviates from the assumptions of the projective warp, thus, reducing the dependency on expensive post-processing routines without compromising the geometric realism of the resulting mosaics. Imposing *projective regularisations* for image stitching allows handling of more general scene types over affine priors and (as shown in Chapter 4), extrapolations of homographic nature are more amenable and, what is more important, are theoretically justified for image stitching. As detailed in the following section, giving homographies such ability, is exactly the purpose of this work.

1.2 Contributions of the Thesis

This thesis investigates projective estimation under model inadequacies. More specifically, this work considers situations where the enabling assumptions for the projective model are not fully satisfied by the data; thus, fundamentally limiting the achievable accuracy of the homography fit.

The main argument underpinning this work is that, instead of relying on a projective model which is often inadequate, and then trying to fix the resulting errors in subsequent post-processing or compositing stages, homography-based alignments must flexibly account for images that deviate from the expected projective trend. To this end, this thesis makes the following contributions:

- This work proposes a new projective estimation method called *Moving Direct Linear Transformation (Moving DLT)*. Moving DLT is able to produce *as-projective-as-possible (APAP)* warps, i.e., warps that aim to be globally projective, yet allow local non-projective

deviations to account for violations to the assumed imaging conditions of image stitching. Fig. 1.3 row 4 shows a *raw* (i.e., without post-processing) image alignment result obtained with Moving DLT. As can be seen in this figure, very few (if any) visible alignment mistakes remain.

Moving DLT is based on the theory of Moving Least Squares (MLS) [45] but unlike MLS and other state-of-the-art approaches for image stitching, Moving DLT makes use of projective regularisation which gives the method the ability to flexibly adapt to the uncooperative data while gracefully going back to a projective trend in the extrapolation areas. Such characteristic gives Moving DLT the ability to reduce image alignment error mistakes while maintaining the consistency of the scene.

- Two different techniques that allow fast and efficient implementations of Moving DLT are also explored in this work. The current MATLAB/C implementation of Moving DLT is able to estimate the parameters of an APAP warp for stitching a pair of high resolution images with 5068 keypoint matches in ~ 6 seconds, using approximately 1 extra MegaByte of RAM memory when compared against the memory used by the basic projective warp. Other competing, state-of-the-art methods require large amounts of memory and minutes or even hours of processing time (e.g., [51]).
- The bundle adjustment version of Moving DLT, which is called *Bundled Moving DLT*, is also proposed in this work. Bundled Moving DLT allows the simultaneous refinement of multiple APAP warps in order to accurately align multiple images for large panorama creation.

It is worth noting that, for most of the current flexible warp estimation methods, no bundle adjustment version has been proposed in the literature. Bundled Moving DLT is the first method for the simultaneous estimation of multiple non-rigid⁸ warps for image stitching.

- Moving DLT and Bundled Moving DLT are compared against several state-of-the-art methods and commercial software for image stitching. These experiments show the advantages of the proposed methods with respect to other, current solutions.

Even though, the focus of this work is on image stitching, the methods proposed in this thesis are envisioned to be more widely applicable, for example, in video stabilisation, deformable surface registration and reconstruction and non-rigid structure from motion. A discussion of these and other possible applications of Moving DLT is presented in the final Chapter of this thesis.

⁸In the context of this work, “non-rigid” is employed as a synonym of “flexible” and not as a reference to Euclidean or “rigid” transformations (unless stated otherwise).

1.3 Thesis Structure

This thesis is organized into 7 chapters which are briefly described next.

1.3.1 Chapter 2

The goal of this Chapter is to describe the imaging assumptions under which projective transformations are justified as image alignment functions for image stitching.

After explaining the characteristics of the projective warp, this Chapter explains the projective transformation estimation process, which for the purposes of this thesis, is carried out by means of one of the most popular estimation algorithms: the Direct Linear Transformation.

This Chapter also explains and illustrates how a projective warp is effectively able to fit “ideal” image stitching data and how it fails when the data deviates from the projective warp assumptions, thus, generating image alignment errors or ghosting in the final stitched results.

1.3.2 Chapter 3

In Chapter 3 a review of the image stitching methods that are relevant to this thesis is presented. This thesis is about stitching uncooperative data through the generation of “better” image alignment functions rather than through the use of post-processing or de-ghosting algorithms. Thus, the review concentrates on methods for improving the image alignments in image stitching. Such methods include 3D reconstruction and plane+parallax approaches, direct estimation of flexible warps and methods for stitching image data under arbitrary camera motions. The purpose of this review is to provide an analysis of the advantages and disadvantages of these methods, which will help to situate and contrast the proposed approach for image stitching (which is given in Chapter 4) with respect to the other approaches.

1.3.3 Chapter 4

In Chapter 4 the proposed Moving DLT approach for image stitching is presented. Similar to Chapter 2, this Chapter describes and illustrates the advantages of the novel APAP warps (obtained through Moving DLT) as well as the main differences between projective and affine regularisation for image stitching.

This Chapter also provides two different techniques for speeding up the Moving DLT estimation process. These techniques allow the previously mentioned fast generation times (in the order of few seconds) of APAP warps for image stitching.

1.3.4 Chapter 5

Since (similar to a projective warp), the proposed APAP warps are only able to stitch pairs of images, Chapter 5 presents the bundle adjustment version of Moving DLT which is able to simultaneously refine multiple as-projective-as-possible warps for large panorama creation.

The purpose of Bundled Moving DLT is to minimise the dreaded amplification and propagation of alignment errors, which is present in most of the pairwise or incremental stitching methods, applied to more than two images.

1.3.5 Chapter 6

Comprehensive experiments and comparisons are carried out in Chapter 6. In particular, two types of evaluations are performed: qualitative and quantitative.

The *qualitative experiments* show that Moving DLT and Bundled Moving DLT provide significant improvements in image stitching quality and efficiency over other state-of-the-art methods and software tools for image stitching. These results are supported by *quantitative experiments*, which show how the proposed approaches are able to generate stitching results with much less visible alignment error mistakes and less artifacts.

1.3.6 Chapter 7

Chapter 7 ends this thesis by discussing the limitations of the proposed projective estimation methods for image stitching, provides a summary of this work and motivates possible, future lines of work.

Chapter 2

Background

2.1 Introduction

One of the fundamental tasks in computer vision is to analyse and discover the relationships between different images obtained from the same scene. These relationships enable the development of computer vision algorithms that perform different tasks like 3D reconstruction, augmented reality and, the main concern of this thesis: image stitching.

As mentioned during the introductory Chapter of this thesis, for the case of image stitching, projective transformations have become the de-facto standard solution for modelling the relationship between images taken from different viewpoints. But, what exactly makes a projective transformation the “weapon of choice” for the task of image stitching? A projective transformation is the most general of the 2D linear transformations in projective space that are available for image alignment; they include translations, Euclidean transformations, similarities and affine transformations (Table. 2.1 shows this hierarchy of 2D linear transformations). Thus, in principle, since projective transformations have more degrees of freedom than any of the other transformations (a projective transform has 8 degrees of freedom while an affine transformation has 6, similarities 4, Euclidean transforms have 3 and translations 2), they are able to achieve better image alignments than the rest of the planar transformations, which is the ultimate goal in image stitching.

However, besides the number of degrees of freedom, a much fundamental justification for a homography as an image-to-image mapping function for stitching lies in the fact that projective transformations are able to maintain the geometric consistency in the aligned images, which, as opposed to the other planar transformations, provides perspective realism to the resulting





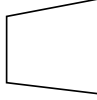
Transformation	Matrix	Invariant properties	Distortion
<i>Translation</i>	$\mathbf{T} = \begin{bmatrix} \mathbf{I} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$	orientations +	
<i>Euclidean</i>	$\mathbf{E} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta & t_x \\ \sin \theta & \cos \theta & t_y \\ 0 & 0 & 1 \end{bmatrix}$	lengths +	
<i>Similarity</i>	$\mathbf{S} = \begin{bmatrix} s\mathbf{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} = \begin{bmatrix} s \cos \theta & -s \sin \theta & t_x \\ s \sin \theta & s \cos \theta & t_y \\ 0 & 0 & 1 \end{bmatrix}$	angles +	
<i>Affine</i>	$\mathbf{F} = \begin{bmatrix} \mathbf{L} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} = \begin{bmatrix} l_{11} & l_{12} & t_x \\ l_{21} & l_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$	parallelism +	
<i>Projective</i>	$\mathbf{H} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$	straight lines	

Table 2.1: Hierarchy of 2D linear transformations in projective space. The distortion column shows the typical effects of the transformations on a square. The matrix \mathbf{I} is a 2×2 identity matrix, $\mathbf{t} = [t_x \ t_y]^T$ is a 2D translation vector, $\mathbf{0} = [0 \ 0]^T$ is 2×1 vector of zeros, \mathbf{R} is a 2D rotation matrix (for a Euclidean transformation \mathbf{R} should be orthonormal i.e., $\mathbf{R}\mathbf{R}^T = \mathbf{I}$ and $\|\mathbf{R}\| = 1$), s is a scale factor and \mathbf{L} is a 2×2 invertible matrix. Each one of the transformations in this table preserves the properties of the transformations in the rows below. The projective transformation is the most general of the 2D planar transformations, it includes all of the other transformations above it. In this case, except for the projective transformation, all of the matrices are augmented with a third row $[0 \ 0 \ 1]$ to form a 3×3 matrix, which allows homogeneous coordinate transformations. This table is based on the ‘‘Hierarchy of 2D coordinate transformations’’ table presented in [77, Pp. 35].

image alignments. What is even more important about homographies is the fact that, in order to achieve this goal, it is not necessary to obtain information about the 3D structure of the imaged scene (therefore, the ‘‘image-to-image’’ mapping denomination), which is information that is often unavailable for most of the digital cameras and pictures. On top of that, current methods for estimating the parameters of a homography are very fast and reliable. Thus, making homographies a very practical option for image stitching.

The current Chapter begins by describing the (two different) image formation processes which justify the use of projective transformations as image aligning function. One of the main purposes of this Chapter is to explain why, even though they are widely used by several state-of-the-art technologies and offer several advantages over other 2D transformations and other methods, homographic warps are a fundamentally limited choice for the task of image stitching: they

only “succeed” when the images to align are obtained under very specific conditions and, unfortunately, such conditions are hard to satisfy in practice. However, it is actually because of the properties underlying these limitations that it is possible to disregard the use of 3D data in perspective image alignments.

This Chapter also describes the Direct Linear Transformation (DLT) method which is one of the most popular (if not the most popular) projective warp estimation approach. Explaining “the basics” of DLT is the second goal of this Chapter. Such a estimation method is fundamental for understanding the underlying principles of the proposed Moving DLT approach, which aims at relaxing the restrictions of the basic projective warp for image stitching.

2.2 The Homography Matrix

If two images I and I' are known to correspond to a *planar scene* or if the images are taken with a camera undergoing a *pure rotational* motion, there is a geometric constraint that relates the corresponding homogeneous points in the two views of the scene (for a description of homogeneous points, the reader is referred to Appendix B). Such a geometric constraint is known as a *projective transformation* or *homography* and it is algebraically represented by an invertible matrix, which is called the *homography matrix*.

Mathematically, this relationship is expressed via the equation

$$\tilde{\mathbf{x}}' \sim \mathbf{H}\tilde{\mathbf{x}}, \quad (2.1)$$

where \mathbf{H} is the 3×3 homography matrix, $\tilde{\mathbf{x}} = [x \ y \ 1]^T$ and $\tilde{\mathbf{x}}' = [x' \ y' \ 1]^T$ are corresponding homogeneous points from I and I' , respectively, and \sim indicates equality up to scale. Homography matrices differing by a non-zero scalar factor are considered to correspond to the same projective transformation.

2.2.1 Plane Induced Homography

Suppose a scene plane Π is imaged from two different viewpoints. Let $\mathbf{P} = \mathbf{K}[\mathbf{R} \ -\mathbf{R}\mathbf{t}]$ and $\mathbf{P}' = \mathbf{K}'[\mathbf{R}' \ -\mathbf{R}'\mathbf{t}']$ be the 3×4 *camera matrices* for the two views I and I' (the camera matrix is described in Section B.1 of Appendix B). If none of the centres of projection \mathbf{t} or \mathbf{t}' are lying on Π , it is then possible to show that Eq. 2.1 holds true for any two corresponding image points $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{x}}'$.

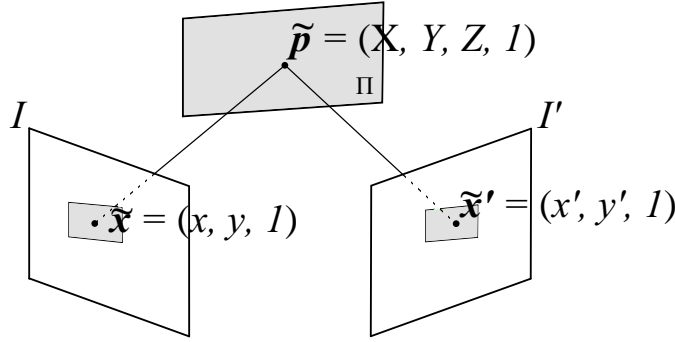


Figure 2.1: A point in a 3D plane being projected into two image planes. \mathbf{p} is the 3D point in the plane and \mathbf{x} and \mathbf{x}' are its corresponding “pictures” or projections in images I and I' .

In order to show this, let $\tilde{\mathbf{p}}$ be the homogeneous representation of a 3D point $\mathbf{p} = [X \ Y \ Z]^T$ in Π . The projections of this point into images I and I' are given by

$$\tilde{\mathbf{x}} \sim \mathbf{P}\tilde{\mathbf{p}} \quad \text{and} \quad \tilde{\mathbf{x}}' \sim \mathbf{P}'\tilde{\mathbf{p}}, \quad (2.2)$$

(Fig. 2.1 illustrates this this scenario). Eq. 2.2 is equivalent to the following equations

$$\lambda\tilde{\mathbf{x}} = \mathbf{P}\tilde{\mathbf{p}} \quad \text{and} \quad \lambda'\tilde{\mathbf{x}}' = \mathbf{P}'\tilde{\mathbf{p}}, \quad (2.3)$$

where λ and λ' are scale factors.

Substituting the camera matrices \mathbf{P} and \mathbf{P}' in (2.3) for their corresponding external and intrinsic matrices produces

$$\lambda\tilde{\mathbf{x}} = \mathbf{K}[\mathbf{R} \quad -\mathbf{R}\mathbf{t}]\tilde{\mathbf{p}} \quad \text{and} \quad \lambda'\tilde{\mathbf{x}}' = \mathbf{K}'[\mathbf{R}' \quad -\mathbf{R}'\mathbf{t}']\tilde{\mathbf{p}}, \quad (2.4)$$

where \mathbf{K} , \mathbf{K}' , \mathbf{R} , \mathbf{R}' and \mathbf{t} , \mathbf{t}' are the calibration and rotation matrices and the centres of projection of the camera matrices $\tilde{\mathbf{P}}$ and $\tilde{\mathbf{P}}'$, respectively.

The left-hand side (LHS) of Eq. 2.4 can be written as

$$\lambda\tilde{\mathbf{x}} = \mathbf{K}\mathbf{R}[\mathbf{I} \quad -\mathbf{t}]\tilde{\mathbf{p}}, \quad (2.5)$$

(where \mathbf{I} is a 3×3 identity matrix) and since $\tilde{\mathbf{p}} = [\mathbf{p}^T \ 1]^T$, (2.5) implies

$$\mathbf{p} = \lambda\mathbf{R}^{-1}\mathbf{K}^{-1}\tilde{\mathbf{x}} + \mathbf{t}. \quad (2.6)$$

If the imaged scene plane Π has a unit outward normal $\mathbf{n} = [n_x \ n_y \ n_z]^T$ and it is situated at a distance $-c$ from the origin of the world coordinate system, then the point \mathbf{p} should satisfy the

equation

$$\mathbf{n}^T \mathbf{p} + c = 0, \quad (2.7)$$

(this is because the initial assumption was that the point \mathbf{p} lies on the plane Π).

Substituting the value of \mathbf{p} from Eq. 2.6 into (2.7) yields the following result

$$\mathbf{n}^T (\lambda \mathbf{R}^{-1} \mathbf{K}^{-1} \tilde{\mathbf{x}} + \mathbf{t}) + c = 0, \quad (2.8)$$

which allows to obtain the following value of λ :

$$\lambda = (-\mathbf{n}^T \mathbf{t} - c) (\mathbf{n}^T \mathbf{R}^{-1} \mathbf{K}^{-1} \tilde{\mathbf{x}})^{-1}. \quad (2.9)$$

Having obtained \mathbf{p} and λ , it is possible to substitute their values into the right-hand side (RHS) of Eq. 2.4, which allows to obtain the homography matrix \mathbf{H} relating $\tilde{\mathbf{x}}$ with $\tilde{\mathbf{x}}'$. In order to do so, it is necessary to write the RHS of Eq. 2.4 in the following form:

$$\begin{aligned} \lambda' \tilde{\mathbf{x}}' &= \mathbf{K}' [\mathbf{R}' \quad -\mathbf{R}' \mathbf{t}'] \tilde{\mathbf{p}}, \\ \lambda' \tilde{\mathbf{x}}' &= \mathbf{K}' \mathbf{R}' [\mathbf{I} \quad -\mathbf{t}'] \tilde{\mathbf{p}}, \\ \lambda' \tilde{\mathbf{x}}' &= \mathbf{K}' \mathbf{R}' (\mathbf{p} - \mathbf{t}'). \end{aligned} \quad (2.10)$$

Substituting \mathbf{p} from Eq. 2.6 into (2.10) gives the following relation

$$\begin{aligned} \lambda' \tilde{\mathbf{x}}' &= \mathbf{K}' \mathbf{R}' (\lambda \mathbf{R}^{-1} \mathbf{K}^{-1} \tilde{\mathbf{x}} + \mathbf{t} - \mathbf{t}') \\ &= \lambda \mathbf{K}' \mathbf{R}' \mathbf{R}^{-1} \mathbf{K}^{-1} \tilde{\mathbf{x}} + \mathbf{K}' \mathbf{R}' \mathbf{t} - \mathbf{K}' \mathbf{R}' \mathbf{t}' \\ &= \lambda \mathbf{K}' \mathbf{R}' \mathbf{R}^{-1} \mathbf{K}^{-1} \tilde{\mathbf{x}} + \mathbf{K}' \mathbf{R}' (\mathbf{t} - \mathbf{t}'). \end{aligned} \quad (2.11)$$

The term in the LHS of (2.11) is a linear combination of the vectors on the right, thus, such relation will not change if the RHS is multiplied by the scalar λ^{-1} as follows:

$$\lambda' \tilde{\mathbf{x}}' = \mathbf{K}' \mathbf{R}' \mathbf{R}^{-1} \mathbf{K}^{-1} \tilde{\mathbf{x}} + \mathbf{K}' \mathbf{R}' (\mathbf{t} - \mathbf{t}') \lambda^{-1}. \quad (2.12)$$

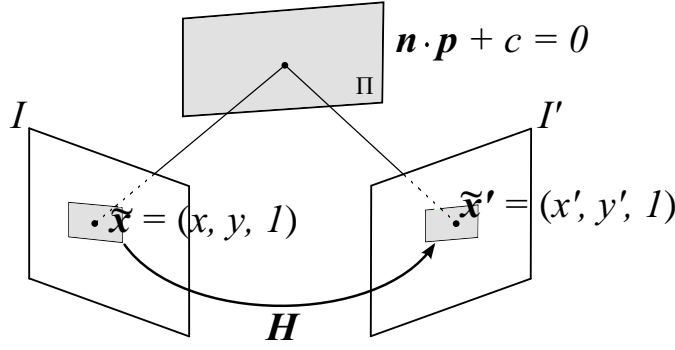


Figure 2.2: A homography matrix relating points from a planar view. Since it is not possible to obtain the depth from the images, a planar homography (induced by the points lying on the plane $\hat{\mathbf{n}} \cdot \mathbf{p} + c = 0$) is able to relate the views.

Then, after replacing the value of λ from Eq. 2.9 into Eq. (2.12), it is possible to obtain the following equation:

$$\begin{aligned} \lambda' \tilde{\mathbf{x}}' &= \mathbf{K}' \mathbf{R}' \mathbf{R}^{-1} \mathbf{K}^{-1} \tilde{\mathbf{x}} + [\mathbf{K}' \mathbf{R}' (\mathbf{t} - \mathbf{t}')] [(-\mathbf{n}^T \mathbf{t} - c) (\mathbf{n}^T \mathbf{R}^{-1} \mathbf{K}^{-1} \tilde{\mathbf{x}})^{-1}]^{-1}, \\ &= \mathbf{K}' \mathbf{R}' \mathbf{R}^{-1} \mathbf{K}^{-1} \tilde{\mathbf{x}} - (\mathbf{n}^T \mathbf{t} + c)^{-1} \mathbf{K}' \mathbf{R}' (\mathbf{t} - \mathbf{t}') \mathbf{n}^T \mathbf{R}^{-1} \mathbf{K}^{-1} \tilde{\mathbf{x}}, \\ &= \mathbf{K}' \mathbf{R}' [\mathbf{I} - (\mathbf{n}^T \mathbf{t} + c)^{-1} (\mathbf{t} - \mathbf{t}') \mathbf{n}^T] \mathbf{R}^{-1} \mathbf{K}^{-1} \tilde{\mathbf{x}}, \end{aligned} \quad (2.13)$$

$$= \mathbf{H} \tilde{\mathbf{x}}, \quad (2.14)$$

$$\tilde{\mathbf{x}}' \sim \mathbf{H} \tilde{\mathbf{x}}.$$

where the homography matrix induced by a planar scene (2.13) is given by the generalisation of the formula derived by Faugeras and Lustman [18].

Thus, the matrix \mathbf{H} in Eq. 2.14, corresponds to the 3×3 homography matrix that relates a point $\tilde{\mathbf{x}}$ from image I with a point $\tilde{\mathbf{x}}'$ in the second image I' , as long as I and I' correspond to images of a planar scene. Fig. 2.2 illustrates this process.

2.2.2 Rotation Induced Homography

A homography of a simpler form is applicable in the situations where both of the cameras have a common centre of projection. Such configuration appears when one of the cameras rotates, thus, the two camera matrices represent the (same) rotating camera in two different instants. If this is the case then both of the camera matrices have the form $\mathbf{P} = \mathbf{K}[\mathbf{R} \quad -\mathbf{R}\hat{\mathbf{t}}]$ and $\mathbf{P}' = \mathbf{K}'[\mathbf{R}' \quad -\mathbf{R}'\hat{\mathbf{t}}]$. Note how the centres of projection are the same in both camera matrices ($\mathbf{t} = \mathbf{t}' = \hat{\mathbf{t}}$). Similar to the case of the plane induced homography, the camera

matrices \mathbf{P} and \mathbf{P}' are written as

$$\mathbf{P} = \mathbf{KR}[\mathbf{I} \quad -\hat{\mathbf{t}}] \quad \text{and} \quad \mathbf{P}' = \mathbf{K}'\mathbf{R}'[\mathbf{I} \quad -\hat{\mathbf{t}}], \quad (2.15)$$

where \mathbf{I} is a 3×3 identity matrix. This allows to write the LHS of (2.15) in the following form

$$[\mathbf{I} \quad -\hat{\mathbf{t}}] = (\mathbf{KR})^{-1}\mathbf{P} = (\mathbf{KR})^{-1}\mathbf{KR}[\mathbf{I} \quad -\hat{\mathbf{t}}]. \quad (2.16)$$

Thus, substituting (2.16) in the right hand side term of (2.15) gives the following equation

$$\mathbf{P}' = \mathbf{K}'\mathbf{R}'(\mathbf{KR})^{-1}\mathbf{KR}[\mathbf{I} \quad -\hat{\mathbf{t}}] = \mathbf{K}'\mathbf{R}'\mathbf{R}^{-1}\mathbf{K}^{-1}\mathbf{P}, \quad (2.17)$$

Eq. 2.17 is equivalent to

$$\mathbf{P}' = \mathbf{HP}, \quad (2.18)$$

where

$$\mathbf{H} = \mathbf{K}'\mathbf{R}'\mathbf{R}^{-1}\mathbf{K}^{-1}, \quad (2.19)$$

is a 3×3 matrix. If a point $\tilde{\mathbf{p}}$ in the 3D scene produces two image points $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{x}}'$ (as in Eq. 2.4), then

$$\tilde{\mathbf{x}}' \sim \mathbf{P}'\tilde{\mathbf{p}} = \mathbf{HP}\tilde{\mathbf{p}} = \mathbf{H}\tilde{\mathbf{x}}, \quad (2.20)$$

which shows that $\tilde{\mathbf{x}}'$ is the image of $\tilde{\mathbf{x}}$ via the homography associated with \mathbf{H} . It is important to note that, in this case, \mathbf{H} is the limit of the plane induced homography matrix as c tends to $-\infty$ with \mathbf{n}^T , \mathbf{t} and \mathbf{t}' kept fixed. Thus, the rotation induced homography described by \mathbf{H} in Eq. 2.19 coincides with the homography induced by the plane at infinity Π_∞ (Fig. 2.3 illustrates).

The rotation induced homography is sometimes called the *infinite homography*. The infinity or rotation induced homography is the second and final imaging process that justifies the use of a projective transformation for relating points in two overlapping views.

If, in Eq. 2.19, the calibration matrices \mathbf{K} and \mathbf{K}' have known aspect ratios and centres of projection, the homography matrix can be parameterised by the rotation and the unknown focal lengths of the two cameras. This particular formulation is commonly used in bundle adjustment approaches for image stitching [11, 12, 74] (which are described in Section 3.3 of the following Chapter).

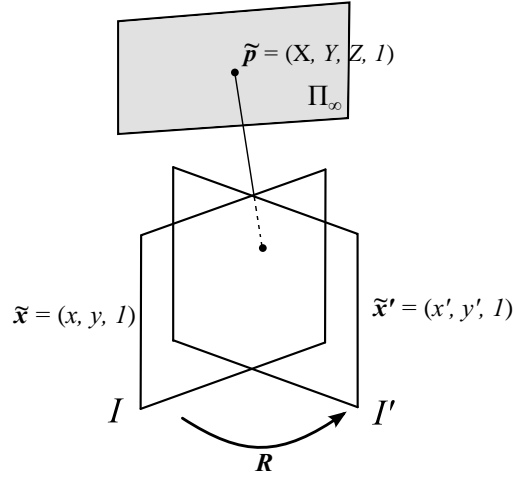


Figure 2.3: A 3D point projected into two views undergoing a pure (3D) rotation motion.

In order to estimate the parameters of a homography matrix, several methods have been proposed in the literature ([31, Chapter 4] presents some of these methods). In the following section one of the most popular approaches for homography estimation is reviewed, this method is called the Direct Linear Transformation.

2.3 Direct Linear Transformation (DLT)

The typical way of estimating the parameters of a homography matrix through the Direct Linear Transformation [96] is by first establishing corresponding points (e.g., SIFT matches [56]) between the two images. In order to remove wrong matches or *outliers*, robust methods like RANSAC [20] are usually employed. The homography matrix is then estimated by making use of the remaining inliers.

Let $\mathbf{x} = [x \ y]^T$ and $\mathbf{x}' = [x' \ y']^T$ be (inlier) matching points across overlapping images I and I' . Recalling from Section 2.2, a projective warp transforms $\tilde{\mathbf{x}}$ to $\tilde{\mathbf{x}}'$ following the relation from Eq. 2.1 (which is copied below):

$$\tilde{\mathbf{x}}' \sim \mathbf{H}\tilde{\mathbf{x}},$$

where (again) $\tilde{\mathbf{x}} = [\mathbf{x}^T \ 1]^T$ and $\tilde{\mathbf{x}}' = [\mathbf{x}'^T \ 1]^T$ are, respectively, points \mathbf{x} and \mathbf{x}' in homogeneous coordinates and \mathbf{H} is the 3×3 homography matrix.

In inhomogeneous coordinates,

$$x' = \frac{\mathbf{r}_1[x \ y \ 1]^T}{\mathbf{r}_3[x \ y \ 1]^T} \quad \text{and} \quad y' = \frac{\mathbf{r}_2[x \ y \ 1]^T}{\mathbf{r}_3[x \ y \ 1]^T}, \quad (2.21)$$

where \mathbf{r}_j is the j -th row of \mathbf{H} .

The divisions in Eq. 2.21 cause the homography function to be non-linear (in affine coordinates), which is crucial to allow a fully perspective warp (Fig. 2.5(a) in page 24 shows a 1D analogy of a projective warp, as can be seen in the figure, a projective warp is a non-linear model). Unfortunately, estimating the parameters of non-linear functions is, in general, a complicated task that requires the use of complex and usually slow algorithms like Gauss-Newton [7] or Levenberg-Marquardt [7, 48, 57].

However, the Direct Linear Transformation is a method that enables a simple *linear solution* for \mathbf{H} to be derived. Being able to estimate the parameters of a non-linear homography warp through linear methods is an impressive property that makes DLT a very popular, fast and reliable method for projective warp estimation (and, as it will be seen in subsequent Chapters, this property is exactly what the proposed Moving DLT method exploits).

DLT estimates \mathbf{H} from a set of N noisy (inlier) point matches $\{\mathbf{x}_i, \mathbf{x}'_i\}_{i=1}^N$ across I and I' . In DLT, Eq. 2.1 is rewritten as the implicit condition

$$\mathbf{0}_{3 \times 1} = \tilde{\mathbf{x}}' \times \mathbf{H}\tilde{\mathbf{x}}, \quad (2.22)$$

(writing Eq. 2.1 in the form of (2.22) is what gives DLT the ability to estimate \mathbf{H} through linear methods). In order to solve Eq. 2.22, and obtain the parameters of \mathbf{H} , it is necessary to rewrite the right hand side of Eq. 2.1 as

$$\mathbf{H}\tilde{\mathbf{x}} = \begin{bmatrix} \mathbf{r}_1^T \tilde{\mathbf{x}} \\ \mathbf{r}_2^T \tilde{\mathbf{x}} \\ \mathbf{r}_3^T \tilde{\mathbf{x}} \end{bmatrix}. \quad (2.23)$$

By setting $\tilde{\mathbf{x}}' = [x' y' 1]^T$, the cross product in the right hand side term in Eq. 2.22, can be derived explicitly as

$$\tilde{\mathbf{x}}' \times \mathbf{H}\tilde{\mathbf{x}} = \begin{bmatrix} y' \mathbf{r}_3^T \tilde{\mathbf{x}} - \mathbf{r}_2^T \tilde{\mathbf{x}} \\ \mathbf{r}_1^T \tilde{\mathbf{x}} - x' \mathbf{r}_3^T \tilde{\mathbf{x}} \\ x' \mathbf{r}_2^T \tilde{\mathbf{x}} - y' \mathbf{r}_1^T \tilde{\mathbf{x}} \end{bmatrix}, \quad (2.24)$$

and since $\mathbf{r}_j^T \tilde{\mathbf{x}} = \tilde{\mathbf{x}}^T \mathbf{r}_j$, for $j = 1, \dots, 3$, this produces a set of three equations in the entries of \mathbf{H} which can be written in the form

$$\mathbf{0}_{3 \times 1} = \begin{bmatrix} \mathbf{0}_{1 \times 3} & -\tilde{\mathbf{x}}^T & y' \tilde{\mathbf{x}}^T \\ \tilde{\mathbf{x}}^T & \mathbf{0}_{1 \times 3} & -x' \tilde{\mathbf{x}}^T \\ -y' \tilde{\mathbf{x}}^T & x' \tilde{\mathbf{x}}^T & \mathbf{0}_{1 \times 3} \end{bmatrix} \mathbf{h}, \quad \mathbf{h} = \begin{bmatrix} \mathbf{r}_1^T \\ \mathbf{r}_2^T \\ \mathbf{r}_3^T \end{bmatrix}, \quad (2.25)$$

where \mathbf{h} is a 9×1 matrix (or more specifically, a vector) that is obtained by vertically stacking each one of the rows from \mathbf{H} (as shown in the right hand side of (2.25)).

Only two of the rows in (2.25) are linearly independent, thus, removing the last row from Eq. 2.25 one can let

$$\mathbf{A}_i = \begin{bmatrix} \mathbf{0}_{1 \times 3} & -\tilde{\mathbf{x}}_i^T & y'_i \tilde{\mathbf{x}}_i^T \\ \tilde{\mathbf{x}}_i^T & \mathbf{0}_{1 \times 3} & -x'_i \tilde{\mathbf{x}}_i^T \end{bmatrix} \in \mathbb{R}^{2 \times 9} \quad (2.26)$$

be the first two rows of the left-hand-side matrix in Eq. 2.25 computed for the i -th point match $\{\mathbf{x}_i, \mathbf{x}'_i\}$. Given an estimate \mathbf{h} , the quantity $\|\mathbf{A}_i \mathbf{h}\|$ is the *algebraic error* of the i -th datum.

DLT minimises the sum of squared algebraic errors

$$\hat{\mathbf{h}} = \underset{\mathbf{h}}{\operatorname{argmin}} \sum_{i=1}^N \|\mathbf{A}_i \mathbf{h}\|^2 \quad \text{s.t.} \quad \|\mathbf{h}\| = 1, \quad (2.27)$$

where the norm constraint prevents the trivial solution. DLT is, thus, also referred to as *algebraic least squares* [96].

Stacking vertically \mathbf{A}_i for all i into matrix $\mathbf{A} \in \mathbb{R}^{2N \times 9}$, the problem in Eq. 2.27 can be rewritten as

$$\hat{\mathbf{h}} = \underset{\mathbf{h}}{\operatorname{argmin}} \|\mathbf{A} \mathbf{h}\|^2 \quad \text{s.t.} \quad \|\mathbf{h}\| = 1. \quad (2.28)$$

The solution is just the right singular vector corresponding to the singular values of \mathbf{A} . Algorithm 1 summarises the DLT method for the estimation of a 2D homography.

2.4 Normalised Direct Linear Transformation

The results generated by the DLT algorithm depend on the origin and scale of the coordinate system of the images. Since the matrix \mathbf{A} in Eq. 2.28 consists of products of image coordinates which can have very different scales, the algorithm as described so far, is unstable and sensitive

Algorithm 1 Direct Linear Transform algorithm for the estimation of a 2D projective warp (homography).

Require: A set of keypoint matches $\{\mathbf{x}_i, \mathbf{x}'_i\}_{i=1}^N$ (with $N \geq 4$) across images I and I' .

- 1: Initialise matrix \mathbf{A} : $\mathbf{A} = []$.
 - 2: **for** $i = 1, \dots, N$ **do**
 - 3: Generate matrix \mathbf{A}_i (Eq. 2.26) from keypoint match $\{\mathbf{x}_i, \mathbf{x}'_i\}$.
 - 4: Stack vertically \mathbf{A}_i into \mathbf{A} : $\mathbf{A} = [\mathbf{A}; \mathbf{A}_i]$.
 - 5: **end for**
 - 6: Perform SVD on \mathbf{A} : $[\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}] = \text{svd}(\mathbf{A})$.
 - 7: The solution $\hat{\mathbf{h}}$ is the last column of \mathbf{V} .
 - 8: Obtain \mathbf{H} reshaped from $\hat{\mathbf{h}}$.
-

to noise [30]. In order to reduce noise sensitivity and avoid issues with numerical precision, prior to DLT, the data can be normalised.

The data normalisation process usually involves transforming the coordinates of the keypoint matches of each image so that their centroid is at the origin and scale the keypoint coordinates so that the average distance from the origin is equal to $\sqrt{2}$ [31]. Thus, instead of applying DLT to $\{\mathbf{x}_i, \mathbf{x}'_i\}_{i=1}^N$, DLT is applied to the normalised data $\{\mathbf{z}_i, \mathbf{z}'_i\}_{i=1}^N$ which are obtained from multiplying

$$\tilde{\mathbf{z}} = \mathbf{N}\tilde{\mathbf{x}} \quad \text{and} \quad \tilde{\mathbf{z}}' = \mathbf{N}'\tilde{\mathbf{x}}', \quad (2.29)$$

with the matrix \mathbf{N} generated from the keypoint coordinates $\mathbf{x} = [x \ y]^T$ in image I and defined as

$$\mathbf{N} = g \begin{bmatrix} 1 & 0 & -\bar{x} \\ 0 & 1 & -\bar{y} \\ 0 & 0 & 1/g \end{bmatrix}. \quad (2.30)$$

From Eq. 2.30, $\bar{x} = \frac{\sum_{i=1}^N x_i}{N}$ and $\bar{y} = \frac{\sum_{i=1}^N y_i}{N}$ are the mean values of x_i and y_i and serve for translating the coordinates of the keypoint matches to the origin, and g scales the keypoint coordinates and is written as

$$g = \frac{\sqrt{2}N}{\sum_{i=1}^N \sqrt{(x_i - \bar{x})^2 + (y_i - \bar{y})^2}}. \quad (2.31)$$

The matrix \mathbf{N}' is obtained in a similar way, making use of the keypoint matches from image I' .

Applying DLT to the normalised data $\{\mathbf{z}_i, \mathbf{z}'_i\}_{i=1}^N$ will produce a “normalised” homography $\tilde{\mathbf{H}}$ which maps $\tilde{\mathbf{z}}$ to $\tilde{\mathbf{z}}'$. In order to recover the homography \mathbf{H} from $\tilde{\mathbf{H}}$ (thus, recovering the projective warp that transforms $\tilde{\mathbf{x}}$ to $\tilde{\mathbf{x}}'$), it is necessary to de-normalise $\tilde{\mathbf{H}}$.

Substituting the unnormalised data \mathbf{x} and \mathbf{x}' in Eq. 2.1 with the normalised \mathbf{z} and \mathbf{z}' from Eq. 2.29, one can derive the following equation

$$\tilde{\mathbf{z}}' \sim \mathbf{N}'\mathbf{H}\mathbf{N}^{-1}\tilde{\mathbf{z}}. \quad (2.32)$$

Eq. 2.32 implies $\check{\mathbf{H}} = \mathbf{N}'\mathbf{H}\mathbf{N}^{-1}$, thus, the homography de-normalisation process can be achieved via the following equation:

$$\mathbf{H} = \mathbf{N}'^{-1}\check{\mathbf{H}}\mathbf{N}. \quad (2.33)$$

Algorithm 2 presents the normalised DLT algorithm for 2D homography estimation.

Algorithm 2 Normalised Direct Linear Transform algorithm for the estimation of a 2D projective warp (homography).

Require: A set of keypoint matches $\{\mathbf{x}_i, \mathbf{x}'_i\}_{i=1}^N$ (with $N \geq 4$) across images I and I' .

- 1: Generate normalisation matrices \mathbf{N} and \mathbf{N}' (Eq. 2.30) with \mathbf{x} and \mathbf{x}' respectively.
 - 2: Normalise data: $\tilde{\mathbf{z}} = \mathbf{N}\tilde{\mathbf{x}}$ and $\tilde{\mathbf{z}}' = \mathbf{N}'\tilde{\mathbf{x}}'$.
 - 3: Initialise matrix \mathbf{A} : $\mathbf{A} = []$.
 - 4: **for** $i = 1, \dots, N$ **do**
 - 5: Generate matrix \mathbf{A}_i (Eq. 2.26) from normalised keypoint match $\{\mathbf{z}_i, \mathbf{z}'_i\}$.
 - 6: Stack vertically \mathbf{A}_i into \mathbf{A} : $\mathbf{A} = [\mathbf{A}; \mathbf{A}_i]$.
 - 7: **end for**
 - 8: Perform SVD on \mathbf{A} : $[\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}] = \text{svd}(\mathbf{A})$.
 - 9: The solution $\hat{\mathbf{h}}$ is the last column of \mathbf{V} .
 - 10: Obtain (normalised) $\check{\mathbf{H}}$ reshaped from $\hat{\mathbf{h}}$.
 - 11: Obtain (de-normalised) \mathbf{H} from $\check{\mathbf{H}}$: $\mathbf{H} = \mathbf{N}'^{-1}\check{\mathbf{H}}\mathbf{N}$.
-

2.5 Image Stitching with Projective Transformations

Once the estimated homography matrix \mathbf{H} has been obtained (either with Alg. 1 or Alg. 2), in order to align the images I and I' , an arbitrary pixel $\tilde{\mathbf{x}}_*$ in the source image I can be warped to the position $\tilde{\mathbf{x}}'_*$ in the target image I' via the equation

$$\tilde{\mathbf{x}}'_* \sim \mathbf{H}\tilde{\mathbf{x}}_*, \quad (2.34)$$

then, in order to recover the inhomogeneous image or pixel coordinates it is necessary to apply Eq. 2.21 to the contents of vector $\tilde{\mathbf{x}}'_*$, thus, aligning image I with I' .

Fig. 2.4 presents a raw stitching result obtained with this process. In this case the images were obtained with a *rotating camera* mounted on a tripod. The image alignments were performed by

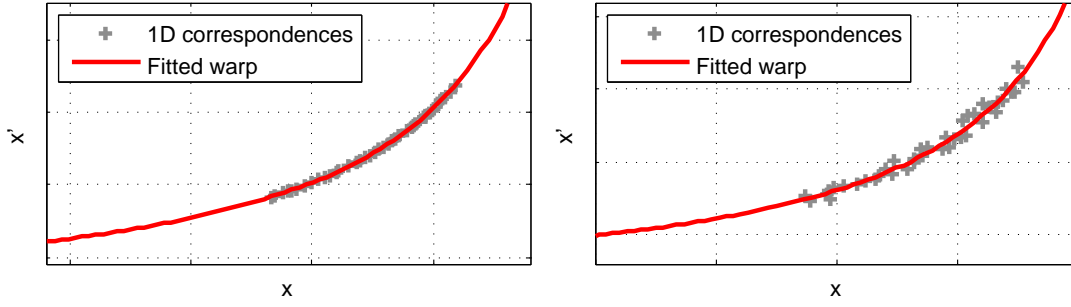
(a) Target image I' .(b) Source image I .

(c) Image alignment results obtained by means of a basic homography warp. Since the images are obtained with a rotating camera, no visible alignment mistakes are visible.

Figure 2.4: Demonstrating image stitching with the Direct Linear Transformation method. The input images correspond to views that differ *only* by rotation. The images are both of size 1280×690 pixels. The number of SIFT matches $\{\mathbf{x}_i, \mathbf{x}'_i\}_{i=1}^N$ (not shown) after RANSAC is 1007.

means of a 3×3 homography matrix \mathbf{H} estimated with the Direct Linear Transformation method from Alg. 2. As can be seen in the figure, under these *controlled conditions*, the homography warp is able to align the images with no evident geometric alignment mistakes, thus, (without accounting for image exposure differences) solving the image stitching problem.

Fig. 2.5(a) shows a 1D analogy of this image stitching process. To generate this 1D analogy of image stitching, a set of 1D correspondences $\{\mathbf{x}_i, \mathbf{x}'_i\}_{i=1}^N$ are generated by projecting a 2D



(a) A 1D analogy of a projective warp stitching data from two views that differ only by rotation. (b) A 1D analogy of a projective warp stitching data from two views that differ by rotation *and* translation.

Figure 2.5: A 1D analogy of image stitching with projective warps. To generate a 1D analogy of image stitching, a set of 1D correspondences $\{\mathbf{x}_i, \mathbf{x}'_i\}_{i=1}^N$ are generated by projecting a 2D point cloud onto two 1D image planes. (a) The two views differ only by rotation, which justifies the use of projective warps. The red line corresponds to the estimated 1D projective warp, parametrised by a 2×2 homography. (b) The views differ by rotation *and* translation. The 1D homography warp is unable to model the local deviations of the data. Note that these deviations are caused purely by model inadequacy since there is no noise in the data.

point cloud $\{\mathbf{p}_i\}_{i=1}^N$ onto two 1D image planes. This projection process is performed via Eq. 2.2

by means of two 2×3 camera matrices \mathbf{P} and \mathbf{P}' . For the camera matrix \mathbf{P} : $\mathbf{K} = \begin{bmatrix} f & c_x \\ 0 & 1 \end{bmatrix}$,

$\mathbf{R} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}$ and $\mathbf{t} = \begin{bmatrix} t_x \\ t_y \end{bmatrix}$, where f is the focal length of \mathbf{P} , c_x is the position of the principal point of the 1D image \bar{I} , θ is the camera rotation angle and t_x and t_y are the camera translations (and similarly for \mathbf{P}').

Similar to the stitching example from Fig. 2.4, the image data or correspondences are obtained with a camera undergoing *pure rotation* (i.e., in both of these cameras matrices the 2×1 translation vectors are set to $\mathbf{t} = \mathbf{t}' = [0 \ 0]^T$) which fulfils the prescribed imaging conditions of the projective warp for successful image stitching. In order to obtain the 1D (2×2) homography matrix \mathbf{H} for this 1D analogy, Alg. 2 was used but, in this case, each matrix \mathbf{A}_i is defined as:

$$\mathbf{A}_i = \begin{bmatrix} -\tilde{\mathbf{x}}_i^T & x'_i \tilde{\mathbf{x}}_i^T \end{bmatrix} \in \mathbb{R}^{1 \times 4} \quad (2.35)$$

As can be seen in Fig. 2.5(a), under a pure rotational motion, the 1D homography warp is able to successfully fit (i.e., align) the image data, which is the reason why, for the 2D case in Fig. 2.4, such process generates “successful” image alignments for image stitching.

Unfortunately, given images that deviate from the assumptions of the projective warp, a simple homography will not be able to model the local deviations of the data, thus, producing alignment errors or ghosting in the stitched results. These alignment errors are caused either by unmodelled camera motion or by multiple depths in the imaged scenes, which (as seen in Sections 2.2.1

and 2.2.2) are elements that a homographic warp is not able to characterise. Fig. 2.5(b) illustrates this condition. In this case the 1D image data was generated with a camera undergoing *rotation and translation* (i.e., the translation vectors from the camera matrices a set to non-zero values, e.g., $\mathbf{t} = [t_x \ 0]^T$ and $\mathbf{t}' = [t'_x \ 0]^T$ with t_x or t'_x or both $\neq 0$), which clearly violates the assumptions of projective estimation. In particular, note how, because of the camera translations or parallax, the image data deviates from the homographic fit (please note that, in this synthetic example, there is no noise in the data or errors in the warp estimation process).

For the 2D case, Fig. 2.6 provides an example of a mosaic stitched from the data in Figs. 2.6(a) and 2.6(b). As can be seen in this image, alignment mistakes are evident since the images also differ by rotation and translation (Chapter 6 of this thesis provides more examples of non-ideal images that were stitched with a basic projective warp, as expected, in all of those examples alignment mistakes remain obvious). The, just discussed, problem motivated a vast amount of novel research in the areas of image alignment, image stitching, image processing and 3D reconstruction during the last decade. The purpose of the following Chapter is to provide an analysis of the advantages and disadvantages of the most relevant work that attempts to produce better image alignments for image stitching when given non-ideal data. This analysis will help exhibit the advantages and improvements of the proposed approach for image stitching over other, similar intentioned research.

2.6 Summary

The main contribution of this Chapter was to explain under which set of conditions a homography warp is justified as a image alignment function (for image stitching) and what happens to the image data if such conditions are not fully satisfied.

This Chapter also presented the Direct Linear Transformation. DLT has the ability to estimate the parameters of a (non-linear) projective transformation from a set of noisy corresponding keypoint matches across two images. But the most important aspect of DLT is the fact that the method is able to achieve this task by making use of simple linear methods, specifically, the singular value decomposition.

As a final note from this Chapter, it is important to mention that DLT is a *feature based* method (i.e., it makes use of keypoint matches in order to perform). Other (non feature based) formulations estimate the parameters of the homography matrix by minimising the pixel intensity differences between the images to align. Such methods (which were developed in the early days of image alignment research) are often called *direct* or *pixel based* methods. However, direct

(a) Target image I' .(b) Source image I .

(c) Image alignment results obtained by means of a basic homography warp. As can be seen in this image, due to the images deviating from the idealised conditions of the homography warp, several alignment mistakes (circled in red) can be observed.

Figure 2.6: Demonstrating image stitching with the Direct Linear Transformation method. The input images correspond to views that differ by rotation *and* translation. The images are both of size 2000×1500 pixels. The number of SIFT matches $\{\mathbf{x}_i, \mathbf{x}'_i\}_{i=1}^N$ (not shown) after RANSAC is 2753.

methods generally involve Non-Linear Least Squares formulations which are usually solved by means of the aforementioned Gauss-Newton or Levenberg-Marquardt algorithms, which operate on all of the pixel data of the images to align. In general, such algorithms are slow and require “good” initial estimates of the homography matrix. Nowadays this type of approaches are commonly used as an extra step for the refinement of feature based solutions. Direct approaches are not the focus of this thesis, however, [74], [5] and [76, Section 3] are excellent sources for direct homography estimation methods.

Chapter 3

Related Work

3.1 Introduction

While the fundamentals of image stitching are well studied (see [76] for an excellent survey), how to produce good results when the data is noisy or uncooperative is still an open problem.

In this thesis, previous works on image stitching are categorised into two groups: (1) methods that reduce alignment errors or (parallax induced) ghosting by constructing better alignment functions, and (2) methods that reduce ghosting after alignment using advanced methods in compositing, pixel selection or blending.

Chief amongst the second group are the seam cutting methods [2, 17, 44] that optimize pixel selection across the overlapping section of the images in order to minimise visible seams; patch-based methods [6, 15, 32, 91] which synthesize the transition region to smoothly transform from one image to another; advanced pixel blending techniques such as Laplacian pyramid blending [12, 13] and Poisson image blending [67, 79] that minimise blurring due to misalignments or exposure differences. Though vital to produce visually acceptable results, such post-processing routines are nevertheless imperfect and may not work all the time (see [39] for examples). It is thus strategic to attempt to reduce errors as much as possible during the image alignment step which is the goal of this research. Thus, this review concentrates on methods that attempt to produce better alignments for image stitching. Ideally, methods from both groups should be jointly used for best results.

Amongst the works that perform better image alignments for image stitching, four main categories are distinguished: (1) two-step methods that perform an initial “rigid” image alignment,

which is obtained by means of a projective transformation (or any of the other 2D planar transformation from Table 2.1), and then make use of a further refinement process for improving the alignment results; (2) methods that make use of 3D reconstruction techniques for inferring the structure of the scene and the location of the cameras in order to stitch the images; (3) methods for arbitrary camera motion, which are based on the manifold mosaicing work or [65], and (4) methods that are able to directly generate flexible warps for image stitching. Moving DLT belongs to this last category.

The following sections present the most relevant works on image stitching from these four categories. However, this review begins by providing a glimpse of some of the “classical” approaches that make use of basic projective transformations for image stitching. It is worth mentioning that most of these methods assume “ideal” imaging conditions or are used as initialisation for further image aligning refinement methods.

3.2 Pairwise Stitching

The previous Chapter explained how, by means of a projective transformation, it is possible to align a pair of overlapping images. In order to align a long sequence of images, earlier work on image stitching made use of pairwise homographies which are “chained” or multiplied in order to incrementally generate long panoramas.

Fig. 3.1(a) shows a set of four overlapping images. The first step of the incremental stitching process consists of obtaining a homography between each pair of overlapping images. In this case, \mathbf{H}^{kl} is a 3×3 homography matrix that maps points from image I^l to image I^k . These homographies can be estimated e.g., by means of the DLT method from Chapter 2 or by means of Gauss-Newton or Levenberg-Marquardt algorithms. Then, each image that overlaps with the reference frame I^R , which, in this example, is selected as image I^3 , can be stitched or warped by making use of the estimated pairwise homographies, as shown in Fig. 3.1(b). Since, for all of the other images, there is no homography that aligns them with the reference frame, a homography chaining process is performed. The chaining process consists on multiplying the homographies of all of the images that “lead” to the reference frame. In Fig. 3.1(a), image I^1 is not overlapping with the reference frame, however, image I^2 overlaps with I^1 and it also overlaps with the reference frame I^3 , thus, in order to align I^1 with I^3 , a new homography $\mathbf{H}^{31} = \mathbf{H}^{21}\mathbf{H}^{32}$ is generated. This homography brings I^1 into alignment with image I^3 (Fig. 3.1(c) shows an example). For all of the other images, a similar chaining process is performed.

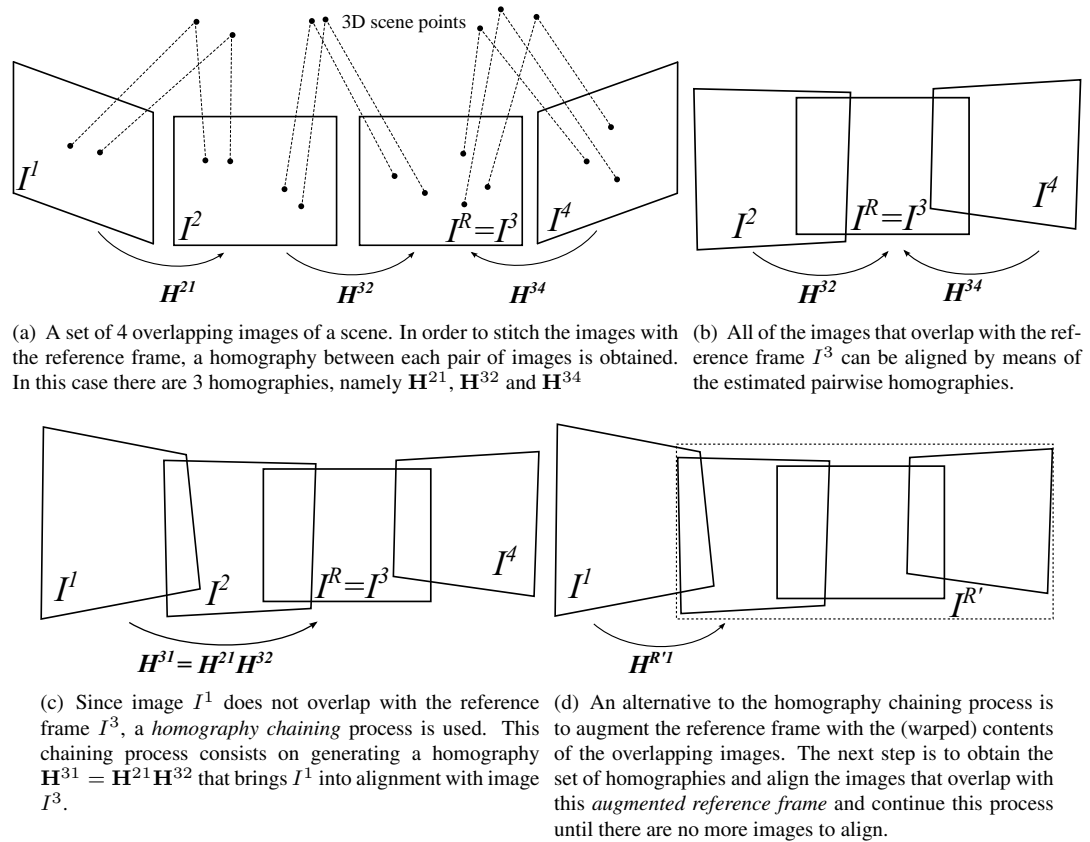


Figure 3.1: Stitching long panoramas through incremental, i.e., pairwise approaches. I^3 is selected as the reference frame.

A variant of the previous approach consists on only obtaining the homographies of the images that overlap with the reference frame. Then, after performing the alignment of these images with the reference frame (like in Fig. 3.1(b)), the resulting stitched mosaic is labelled as the new reference frame $I^{R'}$. The next step is to identify the new set of images that overlap with this augmented reference frame, and repeat this process until the reference frame contains all of the images to stitch (Fig. 3.1(d) illustrates). This form of incremental stitching is sometimes referred to as “threading”.

A notable work that makes use of incremental stitching approaches is that of Irani *et al.* [37]. The goal of such work is to perform video compression and the idea was to generate a panorama by registering the frames that compose the video so that, instead of storing all of the contents of each video frame, regions of this panorama can be used for recovering the contents of several of the original video frames.

In that particular work the pairwise homographies are estimated via the equation

$$E(\mathbf{H}) = \sum_{\mathbf{x}} (I(\mathbf{x}) - I'(f(\mathbf{x}, \mathbf{H})))^2. \quad (3.1)$$

Eq. 3.1 is a (pixel based) cost function that aims at obtaining the homography \mathbf{H} that minimises the pixel error between two images I and I' . \mathbf{x} is a pixel coordinate defined over the area of I that overlaps with I' . f is a function that warps point \mathbf{x} to image I' using the homography \mathbf{H} .

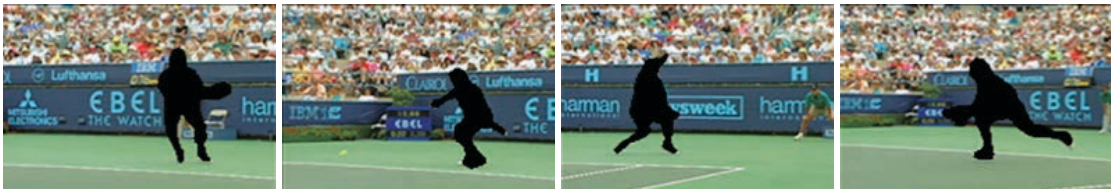
Using Eq. 3.1 (which is solved by means of the iterative Gauss-Newton algorithm), [37] incrementally generates panoramas with two video compression applications in mind: video storage and low bit-rate video transmission. For a video storage application, a scene change detection algorithm (such as the one in [95]) is first applied to the video. Then, an independent panorama is generated by aligning all of the images in each one of the automatically detected scenes. Once the images have been aligned a compositing stage removes ghosting and blurring artifacts so that the contents of the video frames can be recovered with the lowest possible loss. The data that corresponds to moving objects that were not stitched to the panorama are coded separately. For low-bit rate video transmission applications, the panorama cannot be generated in a batch, i.e., the panorama must be constantly updated with the contents of the current image frame; thus, ensuring that the panorama contains only the most recent information. In order to do this, each new frame is aligned with the current contents of the panorama and composited in order to reduce alignment errors and quality loss.

Another approach that makes use of incrementally generated panoramas is that of [47]. In such work, the authors proposed a video encoding system that is based on a layered representation: a set of layers are responsible for encoding the pixel-data of the dynamic objects in the video while another layer encodes the information that corresponds to the static objects, e.g., the video background. For the encoding of static objects in the video, “sprites” are generated by means of projective transformations. Similar to the previously mentioned work of [37], a sprite is a mosaic obtained by incrementally stitching the pixels of the static objects that appear in several frames in the video. By making use of these sprites each static object can be recovered in each video frame. Fig. 3.2(a) shows a panorama generated by incrementally stitching the frames of a video sequence of a tennis match. Fig. 3.2(b) shows the recovered background on four video frames (the moving objects do not form part of the panorama, thus, their pixels are not recovered by this approach).

In order to encode the data of the dynamic objects, affine transformations [76] (refer to Table 2.1) are used. The goal of the affine transformation is to warp the pixels of the dynamic objects from one frame to another, thus, reducing the amount of pixel-data that needs to be encoded per video. However, instead of using one single affine transformation per moving object (which is often insufficient) the authors make use of a set of affine transformations. To generate this set of affine transformations, a dense feature matching process is applied to each moving object so that dense motion vectors are obtained between frames. The motion vectors are then clustered in patches



(a) An example of a panorama generated from the video sequence of a tennis match.



(b) 4 video frames recovered by making use of the contents of the sprite from Fig. 3.2(a)

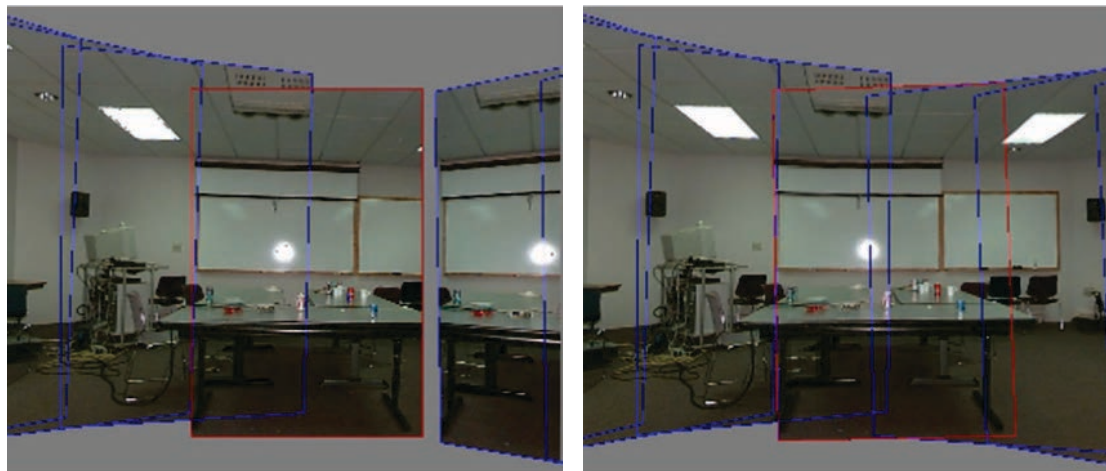
Figure 3.2: An example of a (stitched) sprite and 4 recovered video frames. The “missing regions” in the video frames correspond to dynamic objects. This example appears in [76].

or “blocks” of 32×32 pixels and, by making use of the dense motion vectors in each block, an affine transformation is obtained. The output from this process is a set of *piecewise* affine transformations that describe the motion of a moving object between consecutive frames. It is important to note that this approach does not consider smoothness between neighbour blocks.

Other similar incremental stitching approaches led to the development of important applications in the areas of video compression and encoding [46], video indexing [36, 42], video stabilisation [29], medical image registration [14, 38, 72, 82] and even the generation of storyboards from videos [27]. However, an important drawback of most of the incremental image stitching frameworks is the fact that small image alignment mistakes are amplified and propagated to subsequent images [76], thus, “drifting” the images that compose the panoramas. Such drifting problem becomes more evident when generating 360° panoramas or every time a part of a scene is revisited when stitching video frames. In order to solve this problem, Szeliski and Shum [78] propose what they call “gap closing” techniques. In [78], the homography matrix \mathbf{H} that relates two views I and I' is written as:

$$\mathbf{H} = \mathbf{K}'\mathbf{R}'\mathbf{R}^{-1}\mathbf{K}^{-1} = \mathbf{K}'\mathbf{R}'(\Omega)\mathbf{R}'\mathbf{R}^{-1}\mathbf{K}^{-1}, \quad (3.2)$$

where the left-hand side of Eq. 3.2 corresponds to the rotation induced homography matrix from Section 2.2.2. In such work, the process of estimating the homography matrix \mathbf{H} begins by providing an initial set of values to the matrices \mathbf{R} and \mathbf{R}' and, assuming the values of f and



(a) A section of a 360° panorama generated with pairwise alignments. Note how, because of the chained alignment errors, the ends of the panorama do not match. (b) The ends of the panorama after gap closing. No gap is visible after the correct focal length estimation.

Figure 3.3: An example of the panorama gap closing technique from [78]. This image appears in [78].

f' in $\mathbf{K} = \text{diag}([f, f, 1])$ and $\mathbf{K}' = \text{diag}([f', f', 1])$ are given¹, estimating the final parameters of the homography \mathbf{H} is achieved by performing incremental updates to the values of \mathbf{R} by means of the rotation matrix $\mathbf{R}(\Omega)$. Such incremental updates are performed until convergence or until some image alignment error criteria is matched, and this same process is performed with each pair of images in the panorama. Fig. 3.3(a) shows a circular panorama generated with these pairwise projective alignments, note how the ends of the panorama are not properly aligned. The authors solve this problem by matching the first image in the sequence with the last image. The difference in the rotation matrices (or more specifically, the quotient) associated with these two images indicates the amount of registration error. The gap closing technique aims to make use of the registration error between the first and last image of the panorama in order to perform incremental updates to all of the rotation matrices and focal lengths of the homography estimates, thus, distributing the registration error evenly across the panorama. This is done by converting the difference in the rotation matrices of the first and last image into a quaternion, and dividing this quaternion error by the number of images in the panorama (which is equivalent to assuming relatively constant inter-image rotations).

In a similar way, the focal length of each camera matrix can also be updated. In order to do so, the error quaternion is converted into a gap angle θ_g and then the focal length is updated through the equation $f_{new} = f_{old}(360^\circ - \theta_g/360^\circ)$. Fig. 3.3(b) shows the panorama from Fig. 3.3(a), after this gap closing process of [78].

¹ $\text{diag}(\cdot)$ is a function that creates a diagonal matrix with the elements given in the input vector.

Two major issues of the gap closing approach are that the method needs to estimate the focal lengths of the cameras and that the approach only works for panoramas where the camera is panning in one direction. The issues behind most of the pairwise image stitching approaches and the inability of the gap closing techniques for working under more general camera motions is what motivated one of the most important developments in image alignment research: the use of *bundle adjustment* techniques for image stitching.

3.3 Bundle Adjustment and Local Refinements

Given a set of overlapping images, state-of-the-art methods and tools for image stitching (like Autostitch [12], Photosynth and Street View) perform bundle adjustment [84] in order to optimise the focal lengths and camera poses, which then give rise to inter-image homographies to perform the alignments. In particular, [12] proposes minimising the following energy function:

$$E(\Theta) = \sum_{k=1}^K \sum_{l \in N(k)} \sum_{i \in F(k,l)} h(\mathbf{r}_i^{kl}) \quad (3.3)$$

where $\Theta = [\mathbf{K}^1, \dots, \mathbf{K}^K, \mathbf{R}^1, \dots, \mathbf{R}^K]$ is the set of parameters to optimise, K is the number of images, $N(k)$ is the number of images that overlap with image I^k and $F(k, l)$ is the set of keypoint matches between image I^k and I^l . $\mathbf{r}_i^{kl} = \mathbf{x}_i^k - \tilde{\mathbf{x}}_i^{kl}$ is the residual function that measures the error between the i -th point \mathbf{x}_i^k in image I^k and the (inhomogeneous) projection of its corresponding match from image I^l ; such point projection is defined as:

$$\tilde{\mathbf{x}}_i^{kl} = \mathbf{K}^k \mathbf{R}^k \mathbf{R}^{lT} \mathbf{K}^{l-1} \tilde{\mathbf{x}}_j^l \quad (3.4)$$

with $\mathbf{K} = \text{diag}([f, f, 1])$ and \mathbf{R} being, respectively, the camera intrinsics and the rotation matrices.

In Eq. 3.3, $h(\cdot)$ is a Huber robust function [33]. The purpose of this function is to make the process robust to any remaining outliers after RANSAC. The Huber robust function is defined as:

$$h(r) = \begin{cases} |r|^2, & \text{if } |r| < \sigma \\ 2\sigma|r| - \sigma^2 & \text{if } |r| \geq \sigma \end{cases} \quad (3.5)$$



Figure 3.4: An example of a panorama generated by means of Brown and Lowe's Bundle Adjustment method [12]. The panorama is composited with simple pixel average blending. This panorama was taken from [12].

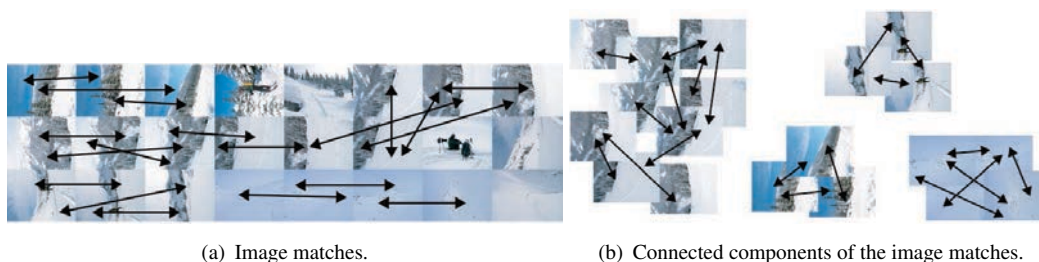


Figure 3.5: Identifying sets of connected images with the panorama recognition process of Brown and Lowe [12].

It is important to note that the multiplications of \mathbf{K} and \mathbf{R} in the right hand side of Eq. 3.4, can be re-written in the following form

$$\mathbf{H}^{kl} = \mathbf{K}^k \mathbf{R}^k \mathbf{R}^{lT} \mathbf{K}^{l-1}. \quad (3.6)$$

This homography formulation, which is expressed in terms of camera intrinsics and rotation matrices, was previously defined in Eq. 2.19 from Chapter 2 of this thesis.

Bundle adjustment avoids the error propagation issue of the incremental approaches since it distributes the errors evenly across all the images. While Brown and Lowe [12] use SIFT keypoint correspondences [56] in order to define the error term, Shum and Szeliski [74] propose a similar bundle adjustment stitching approach, that use the difference of pixel values at regularly sampled patch positions instead of keypoint matches. Fig. 3.4 shows an example of a panorama generated with the feature based bundle adjustment approach of Brown and Lowe [12], in this case, the panorama has been composited with simple pixel average blending.

A second refinement stage is also conducted in [74] to account for local misalignments (i.e., regions that deviate from the projective model) in the mosaic. For each patch position, the average of the backprojected rays from each view is taken, which is subsequently projected again onto each view to yield the desired patch position in 2D. The differences between the

original and desired patch positions are then interpolated, e.g., using Thin-Plate-Splines [8], to form a correction field for parallax error removal. However, such two step approaches are cumbersome and raise questions regarding the optimality of the overall process, e.g., how to regularize the correction field from overly distorting the original projective warps. It is also worth noting that similar two step methods have also been used for pairwise image registration, e.g., [72]. By directly estimating as-projective-as-possible warps, the method proposed in this thesis avoids a separate refinement step as it will be described in Chapter 4 of this thesis.

Brown and Lowe [12] also introduce a *panorama recognition* step based on SIFT matching that is able to determine the subsets of images that belong to the same panorama, given an unordered collection of photos. The process consists on identifying overlapping images by making use of the number of inlier matches (after RANSAC) between each image pair. For a set of images found to overlap with each other, a panorama can then be constructed with them. Isolated images are ignored from the panorama creation process. Fig. 3.5 shows four different sets of overlapping images that have been identified as belonging to four different panoramas.

Instead of estimating relative image rotations and camera matrices, other works directly estimate inter-image homographies, then thread the homographies to stitch multiple images onto a common reference frame [40]. But, contrary to the methods in Sec. 3.2, the focus of these works is on finding the optimal order of threading such that the errors are not propagated and amplified excessively.

Under this framework [58] proposes an energy term where, for each patch position, the overall registration error is distributed amongst all possible triplets of images that contain (or match) such patch (Fig. 3.6 illustrates this process).

In particular, the work of [58] aims at minimising the function

$$E(\mathbf{H}^1, \dots, \mathbf{H}^K) = \sum_{i=1}^N \frac{1}{|L_i|} \sum_{(k,l) \in L_i} \|\mathbf{x}_i - f(\mathbf{x}_i, \mathbf{H}^k \mathbf{H}^{kl} \mathbf{H}^{l-1})\|^2, \quad (3.7)$$

where N is the total number of points in the reference frame or canvas I^R , \mathbf{x}_i is a point in I^R , L_i is the set of images where point \mathbf{x}_i appears, \mathbf{H}^k is a homography that maps a point from image I^k to the reference frame and \mathbf{H}^{kl} is a homography that maps points from image I^l to image I^k . Similar to (3.1), $f(\mathbf{x}, \mathbf{H})$ is a function that returns the inhomogeneous coordinates of \mathbf{x} after being warped with the homography \mathbf{H} . Eq. 3.7 is solved by means of the Levenberg-Marquardt algorithm.

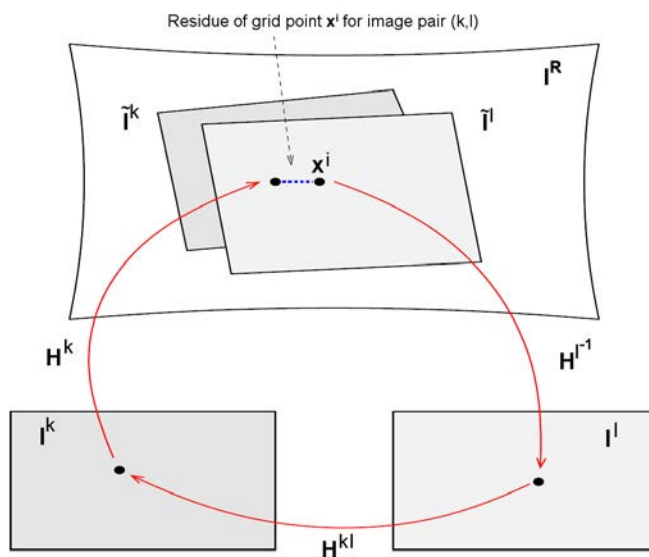


Figure 3.6: The bundle adjustment process of Marzotto [58] for panorama creation. This process aims at estimating the parameters of the inter-image homographies that minimise the alignment error between all triplets of images. \tilde{I}^k is image I^k after being warped to the reference frame I^R . This image appears in [58].

The process in (3.7) ensures a uniform distribution of the misalignment error in the mosaic and since it computes the alignment errors in a reference frame (instead of the individual image frames) it minimises a quantity which is more closely related to the perceived misalignment in the mosaic.

Unfortunately, the dependence on projective alignments means that none of the previous bundle adjustment or threading methods can handle non-ideal data. Thus, other, more recent approaches attempt at solving this problem either by recovering the 3D structure of the scene and stitching the images by making use of this 3D information, or by directly estimating flexible image warps that smoothly adapt to the data that deviates from the projective model. Such methods are reviewed in the next sections but before doing that, it is important to analyse a different application of image alignment which is the generation of collages [60] or “joiners” [94].

Collages and joiners are basically mosaics where the seams of the images are purposely left visible, and the images to align should not undergo any type of distortion caused by the aligning warps (a projective warp introduces “perspective distortion” in the images, that is why projective warps are not used for the generation of collages and joiners). The main difference between a joiner and a collage is the fact that the images that compose a joiner are usually taken from multiple viewpoints and multiple time instances, thus, increasing the number of incorrect matches (outliers) and, in general, making the image alignment process a much harder task.

In order to generate a joiner [94] aims to simultaneously align all of the images with a (common) reference frame by minimising the following *weighted* sum of projection errors:

$$E = \sum_{k=1}^K \sum_{l \in N(k)} \sum_{i \in F(k,l)} w_{kl}^i h(\tilde{\mathbf{x}}_i^k - \mathbf{S}^{kl} \tilde{\mathbf{x}}_i^l) \quad (3.8)$$

where $N(k)$ is the set of images that overlap (have keypoint matches) with image I^k and $F(k, l)$ is the set of feature matches between image I^k and image I^l . $\tilde{\mathbf{x}}_i^k - \mathbf{S}^{kl} \tilde{\mathbf{x}}_i^l$ is the residual between the position of keypoint i in image I^k and the projection of its corresponding match from image I^l . In this case, \mathbf{S}^{kl} is a *similarity transformation* [86] that brings image I^l into alignment with image I^k (refer to Table 2.1).

Similarities are used in order to avoid the introduction of “deformations” to the aligned images which can be caused by the more general affine or projective transformations. w_{kl}^i is the weight assigned to the i -th keypoint match between images I^k and I^l . In order to make the image alignment robust to outliers, Eq. 3.8 also makes use of a robust error function $h(\cdot)$ [33] which is similar to the robust error function in Eq. 3.5.

Eq. 3.8 is a non-linear least squares problem which is solved iteratively by means of the Levenberg-Marquardt algorithm.

A very important aspect of the image alignment process presented in Eq. 3.8 is the fact that different keypoint matches can have different weights w_{kl}^i . The goal of these weights is to give more (or less) importance to some keypoint matches, thus, “guiding” the image alignment results. For example, the left column in Fig. 3.7 shows examples of joiners obtained by giving equal weights to all of the keypoint matches. The right column from Fig. 3.7 shows the results obtained by assigning higher weights to the keypoints near the image boundaries. As can be seen in these figures, better image alignment results can be obtained by giving more importance to the keypoint matches that lie around the overlapping regions of the images.

The purpose of presenting this method for collage generation is to show another approach for improving the image alignment processes. The Moving DLT approach for image stitching presented in this thesis also makes use of weights that give more importance to the keypoint matches around particular areas of the images to align. It is probably worth saying that Moving DLT does not simply replace the similarity transformations in Eq. 3.8 by homographies nor is based in any way on Eq. 3.8. Moving DLT is more fundamental; it does not use the weights only for guiding the 2D parametric warp in order to improve the (still rigid) image alignments. Thus, instead of using the weights for “tilting” the image plane, Moving DLT’s weights are used for locally

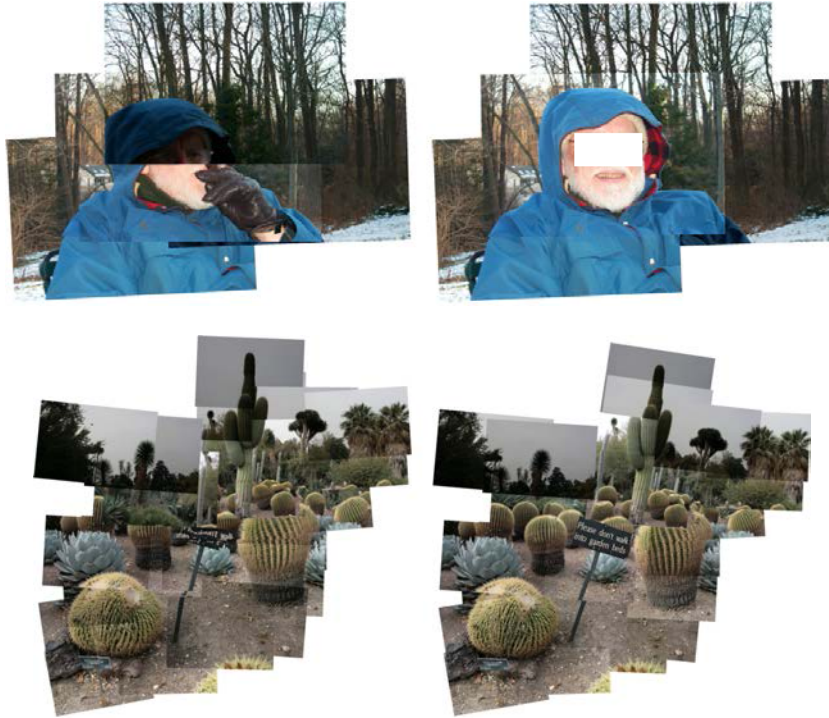


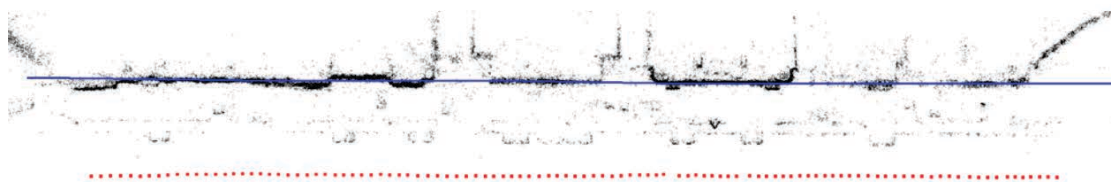
Figure 3.7: Joiner examples from [94] obtained by means of Eq. 3.8. The left column shows joiners obtained by assigning equal weights to all of the keypoint matches from Eq. 3.8. The right column shows the alignment results obtained by giving higher weights to the keypoint matches that lie around the image boundaries. As can be seen, better alignment results can be obtained by giving higher importance to the keypoint matches on the image boundaries.

deforming the projective plane in order to align or adapt the warp to the non-projective data. Other, similar, non-rigid approaches are reported in Section 3.5.

3.4 3D Reconstruction and Plane-Plus-Parallax

Theoretically, it is possible to first recover the 3D structure of the scene from overlapping views in arbitrary poses (e.g., via structure from motion and dense stereo), and then re-project each scene point onto a larger reference image or plane in order to yield the mosaic (this process is described in Section B.2 of Appendix B). This is the approach followed, e.g., by the multi-viewpoint panoramas of [3].

In such work the authors make use of structure from motion algorithms to recover the 3D structure of the scene and the camera matrices for each image. Then, by making use of these camera matrices, all of the images are aligned (projected) into a common surface or reference frame which is defined by the user on the reconstructed 3D scene, as Fig. 3.8(a) illustrates.



(a) A top view of the reconstructed scene from the images that compose the panorama in Fig. 3.8(c). The red dots denote the recovered camera positions. The blue line corresponds to a user defined reference frame/plane.



(b) A panorama generated by projecting a set of images into a user defined reference plane. This panorama is composited with simple image average blending.



(c) The final postprocessed panorama from Fig. 3.8(b).

Figure 3.8: An example of the multi-view panoramas presented in [3].

Once all of the images are projected, a panorama where the images are composited by simple average blending is generated, see Fig. 3.8(b). Then, in order to obtain a final smooth and seamless panorama a Markov Random Field (MRF) optimisation process (which is based on seam cutting [2]) for eliminating ghosting and other artifacts is used. This MRF also aims at rendering each one of the objects in the panorama from a viewpoint roughly in front of it. According to the authors, such characteristic reduces perspective distortion. A final panorama, generated with this approach, can be seen in Fig. 3.8(c).

The most interesting functionalities of [3] include allowing the user to select the reference frame and to interact with the pixel compositing process.

Another approach that makes use of 3D reconstruction techniques for image stitching is that of [99]. The goal of that work is to generate a thin strip panorama (reviewed in Section 3.6) from a sparse set of images, which is a hard task to do since such processes often require a large amount of data (e.g., a video stream) for building the panorama. The method begins by obtaining the camera intrinsic and extrinsic parameters of the available images (through Zhang's method [97]) and the camera poses by means of Structure from Motion algorithms. In order to generate the required dense sampling that the strip panoramas require, virtual cameras are generated between the estimated camera positions. Each observed image is then projected into these virtual cameras. The output from this process is a set of images which is then used for generating the thin strip panorama. Fig. 3.9 provides an example of this method.

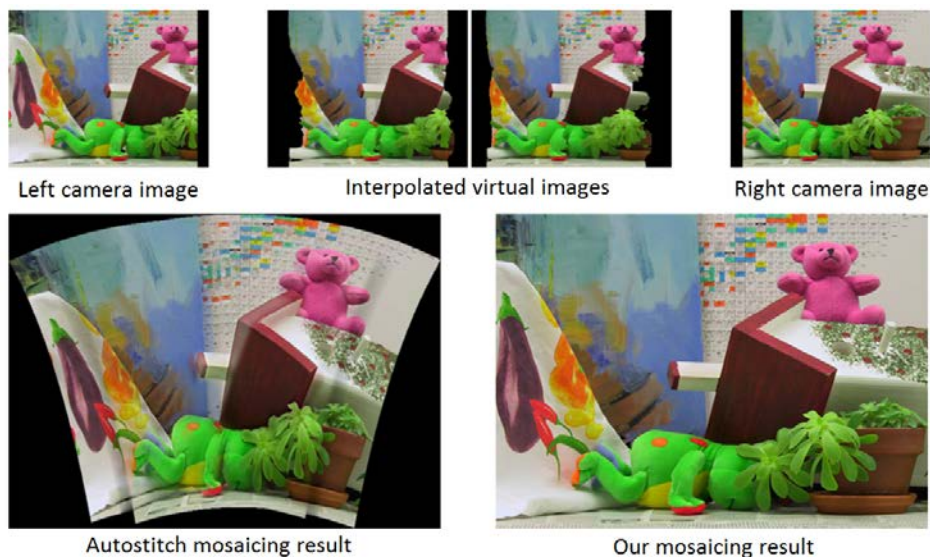


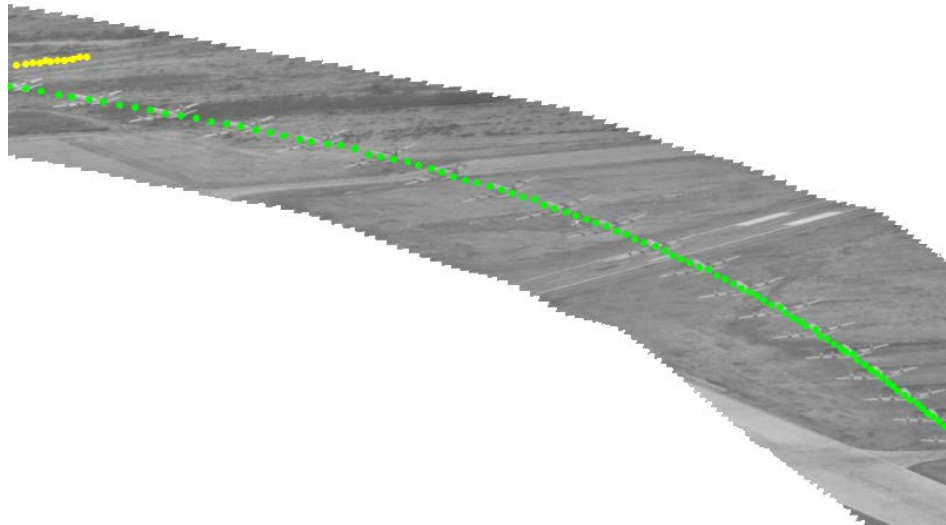
Figure 3.9: The 3D reconstruction approach for image stitching of [99]. The method generates synthetic images in order to produce dense data that is used for generating a thin strip panorama. This image is taken from [99]. In this image the authors also compare their stitching results against the results obtained by Autostitch.

Even though methods based on 3D reconstruction are useful for the generation of mosaics, a full 3D approach can be “overkill” if the goal is just to stitch images; in fact, many advanced compositing methods [2, 67] simply focus on creating perceptually good mosaics with little regard to 3D structure. Also, 3D reconstruction only works for scene points in the overlapping regions. Further, structure from motion algorithms may be brittle in views with small (but not exactly zero) baselines, which represents many of the image stitching cases in practice.

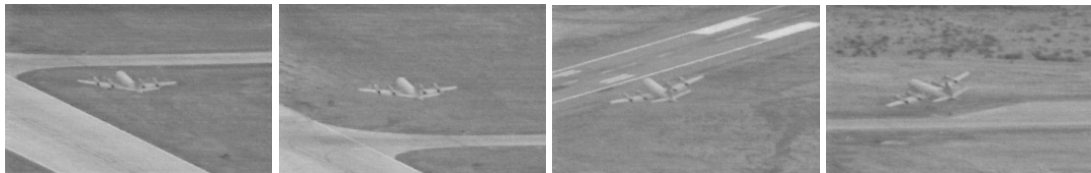
Other methods that attempt to stitch images when the camera undergoes more general camera motions are those of Irani and Anandan [36] and Dornaika and Chung [16]. They propose to account for the camera translations and multiple scene depths (which are, often, the main cause of misalignments) by making use of the *plane+parallax* approach [35, 41, 70, 73].

Since a homography transformation is able to account for the effects of camera rotation, zoom and calibration (without explicitly modelling each one of these elements), the remaining image motion is due to unmodelled translation motions of the camera and deviations of the scene structure from the planar surface. The idea behind *plane+parallax* is to model this “remaining” translation motion. So, in order to align points $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{x}}'$ in two images, the methods make use of the following equation:

$$\tilde{\mathbf{x}}' = \mathbf{K}\mathbf{R}\mathbf{K}^{-1}\tilde{\mathbf{x}} + z\mathbf{K}\mathbf{t} \quad (3.9)$$



(a) A video summary created by stitching representative frames of a video sequence. The video summary also shows the trajectories of the two moving objects: a car and airplane, in yellow and green colours respectively.



(b) Selected representative frames of a video sequence of a flying airplane. In these images, only the airplane is visible, although, the whole sequence also contains a moving car.

Figure 3.10: A video summary created with the plane+parallax approach of [36].

where \mathbf{R} and \mathbf{t} describe the rotation and translation from the first to the second camera frame, \mathbf{K} contains the camera intrinsics and $z = \frac{1}{d}$ is the *parallax component* to be estimated where d is the projective depth of point \mathbf{x} (as described in Section B.2).

In Eq. 3.9, the term $\mathbf{H} = \mathbf{K}\mathbf{R}\mathbf{K}^{-1}$ defines a homography matrix and $\mathbf{K}\mathbf{t}$ is the epipole in the second image, thus, the plane+parallax equation can be rewritten as:

$$\tilde{\mathbf{x}}' = \mathbf{H}\tilde{\mathbf{x}} + z\mathbf{e}_2 \quad (3.10)$$

where \mathbf{e}_2 is the epipole in the second image.

Wang *et al.* [89] make use of the plane+parallax approach for incrementally stitching sequences of x-ray images while Irani and Anandan [36] use plane+parallax in order to generate panoramas for video indexing. The goal in video indexing is to generate an image that summarises a whole video sequence. Fig. 3.10(a) shows an example of a video summary obtained with a one-minute-long video sequence of a flying airplane. Fig. 3.10(b) shows some representative frames from the video.

Unfortunately, plane+parallax approaches either need to estimate the focal length for each one of the cameras of the images that compose the scene, or they need dense matches so that a dense parallax map can be obtained, or they need to obtain the epipolar geometry of the scene (for obtaining the 3D position of each point) in order to recover the parallax component z for all pixels in the images to be aligned. Failing to estimate the focal lengths will lead to incorrect parallax estimates, which will lead to alignment errors.

Aiming at simplifying the parameter estimation task of the plane+parallax approaches, [16] proposes two heuristics. For each pair of images to stitch, the first heuristic makes use of an extra “intermediate” frame. In order to build a mosaic from two images with parallax, all pixels of the image to be warped are mapped into this third frame using a homography plus an approximated parallax. Then, the original pixels are transferred into the reference frame using 3D projective reconstruction followed by a projection. In the second heuristic, the entire scene viewed by the image to be registered is segmented into several planar patches in the projective space. The geometrical transfer of the original pixels is performed using projective mappings between the projective structure of the scene and the image plane, thus, estimating the 3D projective coordinates of the target image pixels without having to compute their 2D location in the intermediate image frame. Even though the authors managed to generate satisfactory stitching results on several image sets, heuristic approaches are often sub-optimal and hard to generalise.

3.5 Direct Estimation of Flexible Warps

Allied to the Moving DLT method proposed in this thesis, there are several recently proposed approaches that directly improve or adapt the 2D warping function in order to reduce misalignment errors. Before this section begins describing them, it is crucial to note that the bundle adjustment version of these approaches has not yet been proposed in the literature. Thus, the methods reported in this Section are limited to performing sequential or incremental warp estimation in order to stitch a set of images, which may cause error propagation issues. As a consequence, when creating large panoramas, the quality of the results of those methods is highly dependent on the accuracy of pairwise stitching and the chaining order of alignment functions.

A smoothly varying affine warp was proposed by [51] for image stitching. Starting from the motion coherence-based point-set registration method of [59], in order to stitch two images, a global affine initialization is introduced by [51]. The next step of the process involves performing local deformations to the affine model, which minimises registration errors while maintaining global affinity (Fig. 3.11 illustrates).

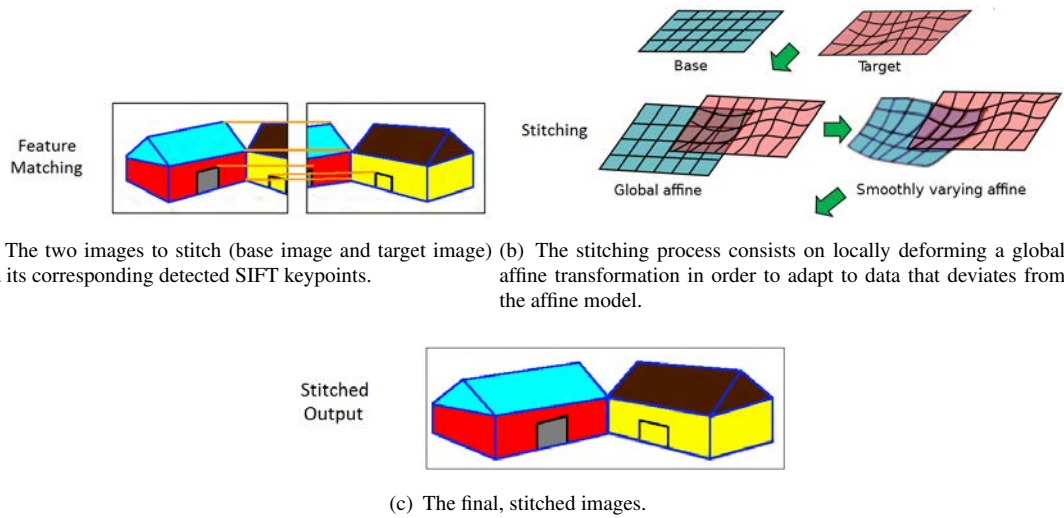


Figure 3.11: An overview of the Smoothly Varying Affine Image Stitching method from [51]. This figure appears in [51].

Mathematically, under the smoothly varying affine stitching framework, the affine transformation that relates point $\tilde{\mathbf{x}}_i$ in image I with point $\tilde{\mathbf{x}}'_i$ in I' is expressed as:

$$\mathbf{F}_i = \mathbf{F} + \Delta\mathbf{F}_i \quad (3.11)$$

where $\Delta\mathbf{F}_i$ is the deviation of the i -th feature's affine transformation from the global affine model \mathbf{F} . The local deviations $\Delta\mathbf{F}_i$ are represented by a continuous affine stitching field. It is important to note that, in this framework, \mathbf{x} and \mathbf{x}' are formed with the concatenation of the 2D vector with the coordinates of the points in the images along with the SIFT descriptor vector.

Since the framework of [51] can make use of other types of information besides the coordinates of the keypoint matches, [52] also incorporated the orientation of the SIFT descriptors into the same optimisation framework. The authors of [52] showed that by refining the smoothly affine warps by making use of the orientation of the SIFT descriptors, the method is able to cope with wider baselines for image registration and even align images with several illuminations changes. Fig. 3.12 illustrates this refinement process which is built on top of the “original” smoothly varying affine warps.

Contrary to most of the image aligning and stitching methods that make use of keypoint matches, one important aspect of the optimisation framework presented in [51, 52] is the ability to disregard outliers, thus, eliminating the need of other “external” processes like RANSAC [20] for outlier removal.

Even though such a characteristic is very important since it makes the method resilient to matching errors, it also makes the overall optimisation process computationally expensive and slow;

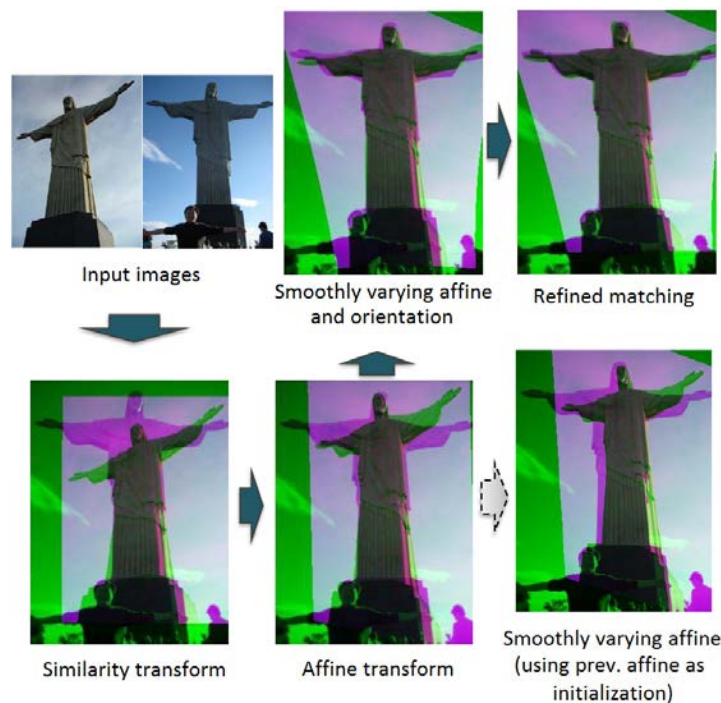


Figure 3.12: The hierarchical non-rigid image alignment process of [52]. This process extends the Smoothly Varying Affine Image Stitching method from [51] by adding an extra step which refines the affine warp by making use of the orientation information of the SIFT descriptors. This image was taken from [52].

[51] reports a total time of ~ 8 minutes for stitching a pair of images with 1200 SIFT keypoint matches, which is in practice, a very long time for image stitching since the process usually involves stitching more than two images. On top of that, affinities are inadequate to achieve a perspective warp [76], e.g., an affine warp may counterproductively preserve parallelism in the areas where the images do not overlap. Therefore, while this stitching method can, in principle, interpolate flexibly and accurately due to local adaptations, the warps will (incorrectly) maintain global affinity when there is no data to guide the local deformation, thus, introducing severe distortions to the aligned results. This problem will be further explained in Section 4.4 from Chapter 4 of the present thesis and demonstrated in the experiments in Chapter 6.

In the context of video stabilization, Liu et al. [53] proposed content preserving warps. Given matching points between the original and stabilized image frames, the novel view is synthesized by warping the original image using an as-similar-as-possible warp [34] that jointly minimises the registration error and preserves the rigidity of the scene. The content preserving warp is obtained by minimising the following equation:

$$E = E_d + \alpha E_s \quad (3.12)$$

where E_d is the data term which aims at minimising registration errors and E_s is the similarity

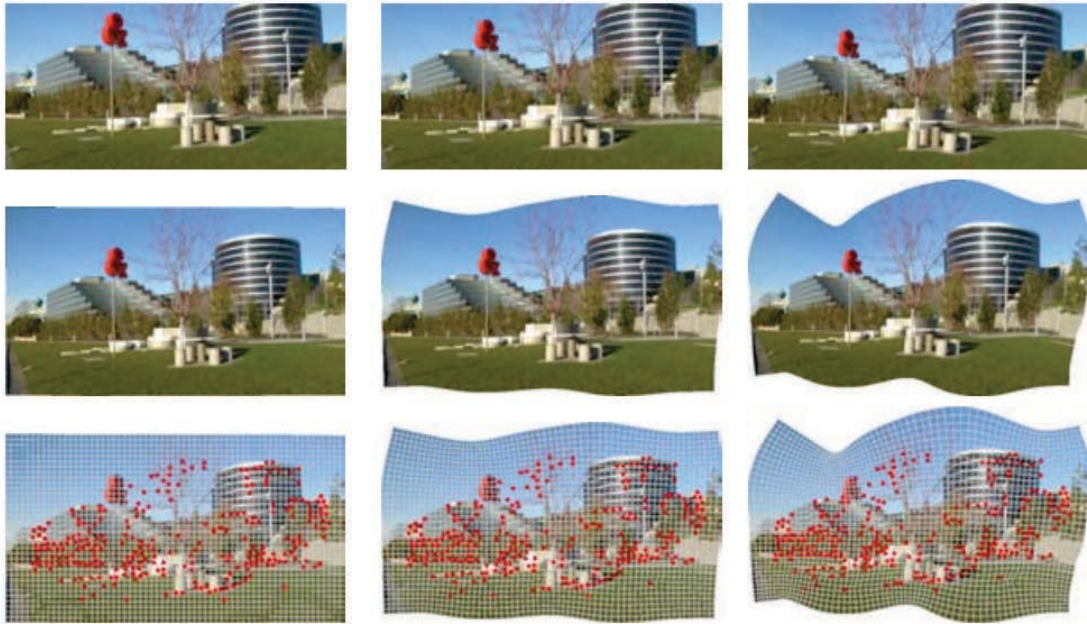


Figure 3.13: An example of the content preserving warps for video stabilisation. The top row shows the final cropped result. The second row shows the warped results. The bottom row shows the corresponding grid and the points that guide the warp. For each one of the cells in this grid a similarity transformation that aligns the cell in the original image with the stabilised image, is generated. As the camera displacements increase (right hand side of the image) the warps start to become unstable. This example image appears in [53].

transformation term. α is a free parameter which balances the influence of the data and similarity terms.

Imposing scene rigidity as in [53] minimises the dreaded “wobbling” effect in video stabilization. However, in image stitching where there can be large rotational and translational difference between views, their method does not interpolate flexibly enough due to the rigidity constraints or the warps become visibly distorted for large image displacements (Fig. 3.13 illustrates). This may not be an issue in [53] since the original and smoothed camera paths are close ([53, Sec. 4]), i.e., the motion between the views to align is small.

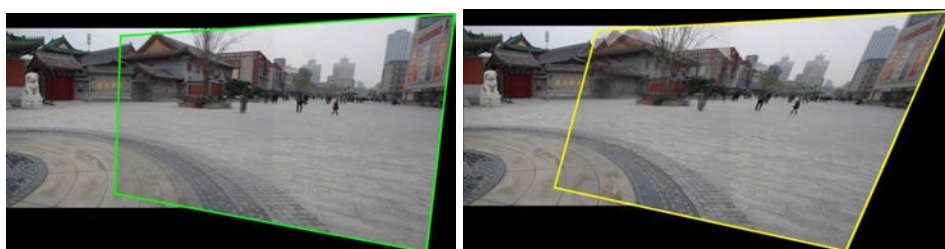
Before applying the content preserving warp, the method pre-warps the original image with a homography, thus effectively yielding a locally adapted homography. However, the local adaptation of content preserving warps is performed by means of similarity transformations and, with Moving DLT, this process is performed with locally adapted homographies which gives the method more flexibility and accuracy as it will be shown in the experiments in Chapter 6.

By assuming that the scene contains a ground plane and a distant plane, Gao et al. [23] proposed dual homography warps for image stitching. In such work, the idea is to perform a clustering process which generates two sets of keypoint matches: keypoints that belong to the ground

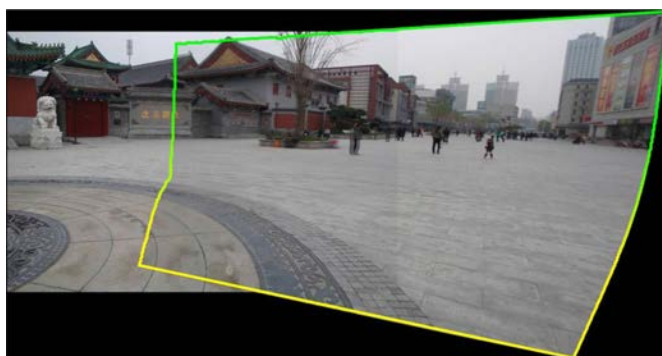
plane and keypoints that belong to the background or distant plane. Making use of these two sets of keypoints, two basic homography warps are estimated independently (by means of the DLT algorithm). The two homography warps are then combined in order to stitch the images. Fig. 3.14 illustrates this process.



(a) The images to stitch and the clustered *ground* and *distant* plane SIFT matches (in yellow and green colour, respectively).



(b) The homography warp obtained with the key- (c) The homography warp obtained with the keypoints points from the distant plane. from the ground plane.



(d) The dual homography warp obtained by combining the ground and distant plane homographies.

Figure 3.14: The dual homography warps for image stitching from [23].

Basically a dual homography warp is a special case of a piece-wise homographic warp, which is more flexible than using a single homography. While the method performs well if the required setting is true, it may be difficult to extend this approach for an arbitrary scene. Even with recent research advances like the ones from [80, 81, 92, 93], how to estimate the appropriate number models (homographies) and their parameters remains a challenging and open problem.

3.6 Arbitrary Camera Motions

There exist methods that consider image stitching under arbitrary camera motions. A notable example is *manifold mosaicing* [65] that is based on *slit* or *pushbroom* cameras.

The imaging process of the pushbroom camera can be modeled by a multi-perspective projection: for each strip the projection is perspective, while different strips may be acquired from different viewpoints. Thus, in the direction of the strips, the projection is perspective, while in the scanning direction the projection is parallel. Since under parallel projection there is no parallax, the strips in the resulting mosaic are aligned at the seams. Using a standard perspective camera, a pushbroom camera can be approximated by continuously “sweeping” the scene in video. This is done by extracting and aligning thin strips from the images of the video onto a manifold such that the optical flow becomes approximately uniform. The width of each one of these strips is proportional to the camera motion. Fig. 3.15 shows two panoramas generated with this stitching technique.

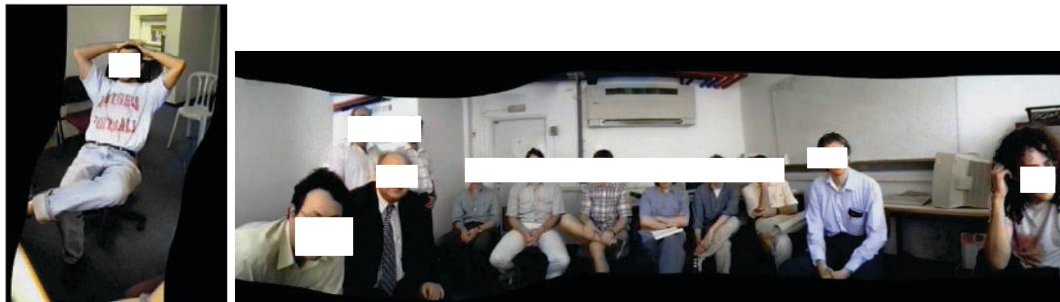


Figure 3.15: Two panorama examples generated with the adaptive manifold mosaicking technique of Peleg *et al.* [65]. This example appears in [65].

This method has also been used for the generation of stereo mosaics [62, 64, 66]. A stereo mosaic consists of two panoramas where one of the panoramas is for the left eye and the other panorama is for the right eye. These panoramas are used for creating an anaglyph image, which when viewed with anaglyph glasses, generates a visual 3D effect.

Since, in order to build the anaglyph image, there must be some disparity between the two panoramas, instead of taking the central strip of each one of the video frames (which is the process for building an “ordinary” panorama), the method takes strips from each side of each frame. By taking strips from the left side of the images, the viewing direction is tilted counter-clockwise from the image surface, generating the panorama for the right eye. When the strips are taken from the right side of the images, the panorama for the left eye is obtained (Fig. 3.16 illustrates this process).

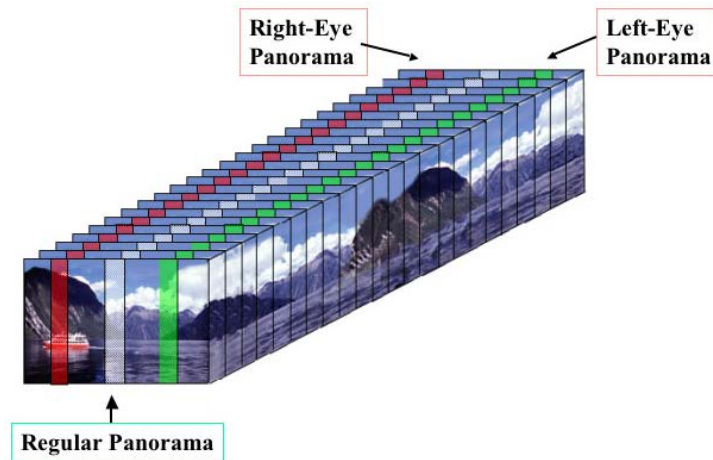


Figure 3.16: The imaging process of the pushbroom camera modeled by the multi-perspective projection process of [62, 64, 66]. When the strips are taken from the center of the images an ordinary panorama is obtained. When the strips are taken from the left side of each image, the viewing direction is tilted counterclockwise from the image surface, obtaining the right-eye panorama. When the strips are taken from the right side of each image, the left-eye panorama is obtained. This image is taken from [63].

Fig. 3.17 shows an anaglyph image obtained from a stereo panorama. This example appears in [63]. If this figure is seen with anaglyph glasses, a 3D panorama effect can be observed.



Figure 3.17: An anaglyph image constructed from a stereo panorama generated with the panoramic stereo imaging method of [66]. This example appears in [63].

Unfortunately, even though manifold mosaicking has been successfully used for the generation of panoramas under general (but smooth) camera motions, the method requires streams of images i.e., videos for approximating the pushbroom camera model, otherwise, the images must be “hallucinated” or synthesised like in the method of [99], reviewed in Section 3.4. Thus, this method has not shown to be applicable to stitch still images in a “discrete” photo collection, such as those handled by Autostitch and Photosynth.

3.7 Summary

This Chapter presented the most relevant work oriented towards the generation of better image alignments for stitching uncooperative data. Some of these works proposed the use of two-step approaches where, after an initial “rigid” image alignment process, a refinement stage attempts

at locally aligning the regions of the images that deviate from the 2D parametric model. Unfortunately, such two-step approaches are often cumbersome and raise questions about the optimality of the image alignments.

Other approaches tackle the parallax problem in image stitching by reconstructing the scene and estimating the parameters of the camera matrices. Once the 3D structure of the scene has been recovered, all of the scene points can be projected into a common reference frame by means of the estimated camera matrices. Unfortunately, 3D reconstruction processes overcomplicate the image stitching task if the goal is “only” to align images and what is worse, Structure from Motion algorithms are not particularly suitable for scenes with small baselines, which is the case for most of the casual stitching imagery. In fact, one of the purposes of the projective transformation is to avoid the 3D reconstruction process in order to align the images, even though (as it is already known), it is necessary to constrain the image formation process, thus, limiting the applicability of the homographic warp.

Attempting to solve this issue, the most up-to-date research in image stitching aims to directly estimate non-rigid (image-to-image) warps in order to bring the images into alignment. However, aligning non-ideal image data through flexible methods is a challenging problem, which usually involves the use of complex algorithms (e.g., [51, 52]), and some of those algorithms are generally slow. On top of that, some of the currently available methods are either restricted to certain particular scene types [23] or make use of *as-similar-as-possible* warps [53] or *smoothly varying affine* warps which either constrain the flexibility of the image alignments or compromise the integrity of the stitched results. These are probably the reasons why most of the current stitching technologies do not make use of this type of flexible approaches yet.

The main purpose of the following Chapter is to introduce Moving DLT: the first method for the generation of as-projective-as-possible (i.e., *smoothly varying projective*) warps. The purpose of these warps is to perform projective image alignments that impose less constraints in the image formation processes while (similar to a basic projective warp) disregarding the use of 3D reconstructions or epipolar geometry estimates or multi-step or heuristic processes. For the case of image stitching, Moving DLT seamlessly bridges image regions that are *inconsistent with the projective model*. On top of that, Moving DLT is a very simple and fast method and it can also be implemented in parallel.

The second purpose of the next Chapter is to explain the main differences and advantages of projective regularisation over affine regularisation for image stitching.

Chapter 4

As-Projective-As-Possible Warps

4.1 Introduction

The *Moving Direct Linear Transformation* is this thesis' main contribution to knowledge. The method is inspired by the elegant and simple framework of Moving Least Squares (MLS) [49], inheriting its approximating capabilities and efficiency. Unlike any method for non-rigid image stitching, Moving DLT is able to directly produce flexible warps that make use of projective regularisations. Such property results into highly accurate image stitching with significantly reduced ghosting effects, which lowers the dependency on post hoc de-ghosting while also maintains the plausibility of the alignment results.

The first part of this Chapter presents the theory behind the novel Moving DLT method. In particular, this first part covers how to formulate and solve projective estimation problems with Moving DLT. This also explains how the as-projective-as-possible (APAP) warps are able to relax the restrictive assumptions of the basic projective warp, thus, making the proposed method particularly suitable for image stitching (and general projective estimation) tasks.

The second part of this Chapter presents two techniques that allow a fast computation for Moving DLT. These methods are very simple and effective. For example, one of these two methods allows the generation of an APAP warp for stitching a pair of images¹ (of size 2000×1329 pixels) with 5068 keypoint matches in ~ 5 seconds + ~ 1 second for compositing. Compared with the significant improvements in the image alignment results, this is an impressively low computational time.

¹The pair of images belongs to the *construction site* image set. From all of the images used in this work, this image pair contains the highest number of keypoint matches. These images along with the corresponding stitching results will be shown in Chapter 6 of this thesis.

This Chapter ends by explaining the differences between Moving DLT and Moving Least Squares (which also allows to explain the differences between smoothly varying affine and the newly proposed: smoothly varying projective warps).

4.2 Moving Direct Linear Transformation (Moving DLT)

Recalling from Section 2.5, a homography \mathbf{H} is an image-to-image mapping that is able to relate an arbitrary homogeneous pixel position or point \mathbf{x}_* from image I with a corresponding point \mathbf{x}'_* in I' via the following equation

$$\tilde{\mathbf{x}}'_* \sim \mathbf{H}\tilde{\mathbf{x}}_*. \quad (4.1)$$

Then, recovering the inhomogeneous pixel coordinates $\mathbf{x}'_* = [x'_* \ y'_*]^T$ can be achieved by

$$x'_* = \frac{\mathbf{r}_1[x_* \ y_* \ 1]^T}{\mathbf{r}_3[x_* \ y_* \ 1]^T} \quad \text{and} \quad y'_* = \frac{\mathbf{r}_2[x_* \ y_* \ 1]^T}{\mathbf{r}_3[x_* \ y_* \ 1]^T}, \quad (4.2)$$

where \mathbf{r}_j is the j -th row of \mathbf{H} . However, when the views I and I' do not differ purely by rotation or are not of a planar scene, using a basic homographic warp inevitably yields misalignment or parallax errors.

In order to alleviate this problem, the idea in this thesis is to warp each \mathbf{x}_* using a *location dependent* homography

$$\tilde{\mathbf{x}}'_* \sim \mathbf{H}_* \tilde{\mathbf{x}}_* \quad (4.3)$$

where \mathbf{H}_* is estimated from the following *weighted* (DLT) problem

$$\hat{\mathbf{h}}_* = \underset{\mathbf{h}}{\operatorname{argmin}} \sum_{i=1}^N \|w_*^i \mathbf{A}_i \mathbf{h}\|^2 \quad \text{s.t.} \quad \|\mathbf{h}\| = 1, \quad (4.4)$$

with \mathbf{A}_i defined as in Eq. 2.26 (copied below)

$$\mathbf{A}_i = \begin{bmatrix} \mathbf{0}_{1 \times 3} & -\tilde{\mathbf{x}}_i^T & y'_i \tilde{\mathbf{x}}_i^T \\ \tilde{\mathbf{x}}_i^T & \mathbf{0}_{1 \times 3} & -x'_i \tilde{\mathbf{x}}_i^T \end{bmatrix}.$$

The goal of the scalar weights $\{w_*^i\}_{i=1}^N$ in (4.4) is to give higher importance to data that are closer to \mathbf{x}_* . These weights are calculated as

$$w_*^i = \exp(-\|\mathbf{x}_* - \mathbf{x}_i\|^2/\sigma^2), \quad (4.5)$$

which is a *Gaussian* weighting function, although, other weighting functions can be used. In this function, σ is a scale or bandwidth parameter, and \mathbf{x}_i is the coordinate in the source image I of one-half of the i -th point match $\{\mathbf{x}_i, \mathbf{x}'_i\}$.

Intuitively, since the weighting function in (4.5) assigns higher weights to data closer to \mathbf{x}_* , the projective warp \mathbf{H}_* better respects the local structure around \mathbf{x}_* . By contrasting this new approach with respect to Eq. 4.1 one can realise that (4.1) uses a single and global \mathbf{H} for all \mathbf{x}_* , whereas the proposed formulation uses one locally weighted homography \mathbf{H}_* per each \mathbf{x}_* . Moreover, as \mathbf{x}_* is *moved* continuously in its domain I , the warp \mathbf{H}_* also varies smoothly [49]. This produces an overall warp that adapts flexibly to the data, yet attempts to preserve the projective trend of the warp, i.e., an *as-projective-as-possible* warp. This new method is called *Moving DLT*.

The problem in Eq. 4.4 can be re-written in its corresponding matrix form as

$$\hat{\mathbf{h}}_* = \underset{\mathbf{h}}{\operatorname{argmin}} \|\mathbf{W}_* \mathbf{A} \mathbf{h}\|^2 \quad \text{s.t.} \quad \|\mathbf{h}\| = 1, \quad (4.6)$$

where the weight matrix $\mathbf{W}_* \in \mathbb{R}^{2N \times 2N}$ is composed as

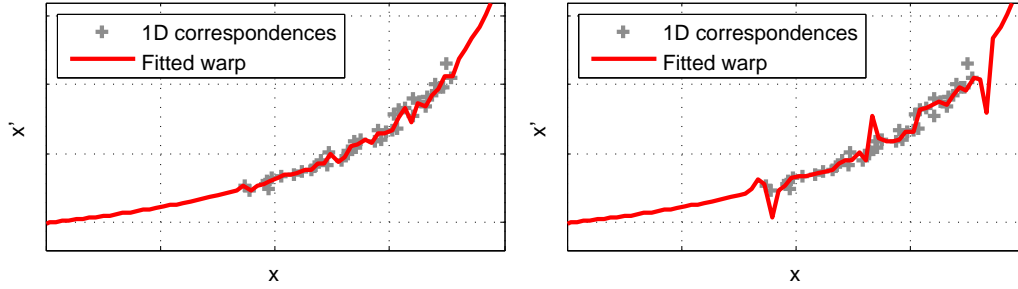
$$\mathbf{W}_* = \operatorname{diag}([w_*^1 w_*^1 w_*^2 w_*^2 \dots w_*^N w_*^N]) \quad (4.7)$$

and $\operatorname{diag}(\cdot)$ creates a diagonal matrix given a vector. This is a weighted SVD (WSVD) problem, and the solution is simply the least significant right singular vector of $\mathbf{W}_* \mathbf{A}$.

Problem (4.6) may be unstable when many of the weights are insignificant, e.g., when \mathbf{x}_* is in a data poor (extrapolation) region. To prevent numerical issues in the estimation, the weights are offset with a small value γ within 0 and 1

$$w_*^i = \max(\exp(-\|\mathbf{x}_* - \mathbf{x}_i\|^2/\sigma^2), \gamma). \quad (4.8)$$

This also serves to regularise the warp, whereby a high γ reduces the warp complexity. In fact as γ approaches 1 the resultant warp loses its flexibility and reduces to the original homographic warp. Fig. 4.1(b) illustrates a 1D as-projective-as-possible warp without regularization, while Fig. 4.1(a) shows the regularized warp using the weight offsetting on the same data (in order to



(a) A 1D analogy of an as-projective-as-possible warp for stitching data from two views that differ by rotation *and* translation. (b) Results from Moving DLT *without* regularisation for the 1D synthetic image stitching problem.

Figure 4.1: A 1D analogy of image stitching with as-projective-as-possible warps. (a) The red line shows a 1D as-projective-as-possible warp estimated by means of Moving DLT. The as-projective-as-possible warp interpolates the local deviations flexibly and extrapolates correctly following a global projective trend. (b) An as-projective-as-possible-warp without regularisation.

Algorithm 3 Moving Direct Linear Transformation algorithm for the estimation of a 2D as-projective-as-possible warp.

Require: A pair of images to stitch (image I and image I').

- 1: Obtain keypoint matches between image I and image I' .
 - 2: Remove mismatches i.e., outliers (refer to Section 4.3). This process will produce a set of N correct keypoint matches $\{\mathbf{x}_i, \mathbf{x}'_i\}_{i=1}^N$.
 - 3: Generate normalisation matrices \mathbf{N} and \mathbf{N}' (Eq. 2.30) with \mathbf{x} and \mathbf{x}' respectively.
 - 4: Normalise data, i.e., obtain normalised keypoint matches: $\tilde{\mathbf{z}} = \mathbf{N}\tilde{\mathbf{x}}$ and $\tilde{\mathbf{z}}' = \mathbf{N}'\tilde{\mathbf{x}}'$.
 - 5: Generate matrix \mathbf{A} with the set of normalised keypoint matches $\{\mathbf{z}_i, \mathbf{z}'_i\}_{i=1}^N$ (refer to Sec. 2.4).
 - 6: **for** each point (or pixel position) \mathbf{x}_* in image I **do**
 - 7: Obtain matrix \mathbf{W}_* which contains the weights between \mathbf{x}_* and all \mathbf{x}_i for $i = 1, \dots, N$.
 - 8: Perform SVD on matrix $\mathbf{W}_*\mathbf{A}$: $[\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}] = \text{svd}(\text{repmat}(\mathbf{W}_*, 1, 9) .* \mathbf{A})$.
 - 9: The solution $\hat{\mathbf{h}}_*$ is the least significant right singular vector of $\mathbf{W}_*\mathbf{A}$ (i.e., the last column of \mathbf{V}).
 - 10: Obtain (normalised) $\check{\mathbf{H}}_*$ reshaped from $\hat{\mathbf{h}}_*$.
 - 11: Obtain (de-normalised) \mathbf{H}_* from $\check{\mathbf{H}}_*$: $\mathbf{H}_* = \mathbf{N}'^{-1}\check{\mathbf{H}}_*\mathbf{N}$.
 - 12: **end for**
-

generate these 1D analogies, an instance of (4.4) with \mathbf{A}_i defined as in (2.35) is solved for each \mathbf{x}_* position along the x axis).

Algorithm 3 summarises the Moving DLT method for the estimation of 2D as-projective-as-possible warps. From Alg. 3, the size of matrix \mathbf{A} is $2N \times 9$ and the size of \mathbf{W}_* is $2N \times 1$, the “ $.*$ ” operator in line 6 of Alg. 3 means element-wise multiplication and $\text{repmat}(\cdot, \cdot, \cdot)$ is MATLAB’s *replicate and tile*² array function.

Fig. 4.2(a) presents a raw stitching result obtained with Alg. 3. The input images I and I' correspond to the data in Figs. 2.6(a) and 2.6(b). As can be seen in this result, the locally

²www.mathworks.com.au/help/matlab/ref/repmat.html

weighted homographies generated with Moving DLT have the ability to align the views without producing any visible alignment mistakes. Contrast this result with the result generated by a “global” homography in Fig. 2.6(c) where severe alignment mistakes are visible.

It is important to note that Moving DLT, or more specifically, the as-projective-as-possible warps do not aim at performing image stitching of images acquired under arbitrary motions. The purpose of Moving DLT is to perform local deformations to the projective warp in order to fit data that deviates from the projective model, as Fig. 4.1(a) illustrates.

It is also not the purpose of this approach (or this thesis) to dispense with the usage of deghosting or post-processing algorithms which are still very useful specially if serious misalignments remain or if there are moving objects in the images. The main point is that it is prudent to achieve higher accuracy alignments since this imposes a much lower dependency on the success of further compositing routines (as will be demonstrated in the experiments and results in Chapter 6).

4.3 Efficient Computation for Image Stitching

So far this work has assumed that no mismatches or outliers exist amongst the data. This idealised situation is extremely rare in practice since the keypoint matching algorithms are imperfect, and there is usually noise in the measured data. Therefore, it is necessary to resort to outlier removal approaches.

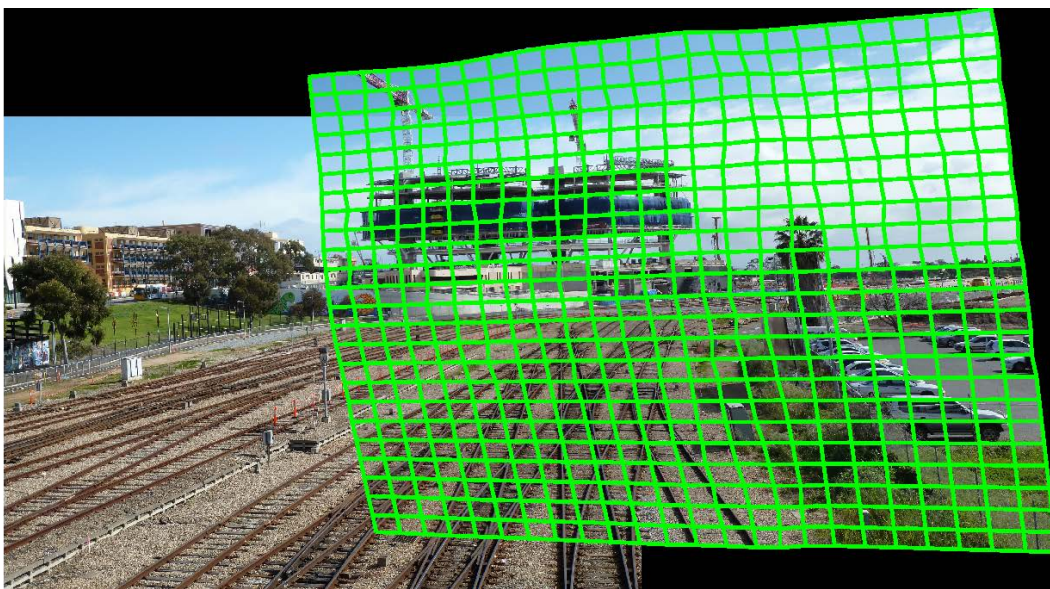
Before invoking Moving DLT it is possible to remove outliers by making use of RANSAC [20] with DLT as the minimal solver. One might argue against RANSAC since this work considers data where the inliers themselves may deviate from the projective trend. In practice though, the outlier errors are orders of magnitude larger than the inlier deviations. This observation was reported and justified in [83] where RANSAC was able to generate outlier detection rates similar (and sometimes better) than those of other, more complicated, methods e.g., SVM regression [50], annealed M-estimation [100] or local smoothness [68]. Thus, RANSAC can be effectively used.

4.3.1 Partitioning into cells

Solving (4.6) for each pixel position \mathbf{x}_* in the source image I is unnecessarily wasteful. This is because neighboring pixel positions will yield very similar weights (4.8) and, as consequence, very similar homography estimates. Instead of estimating a homography per pixel, one option is



(a) Image alignment results obtained by means of the proposed Moving DLT method. As can be seen in the figure, very few (if any) visible misalignments remain.



(b) Aligned images with transformed cells overlaid to visualise the warp. Observe that the warp is globally projective for extrapolation, but adapts flexibly in the overlap region for better alignment.

Figure 4.2: Demonstrating image stitching with Moving DLT. The input images correspond to views that differ by rotation *and* translation. The images are both of size 2000×1500 pixels. The number of SIFT matches $\{\mathbf{x}_i, \mathbf{x}'_i\}_{i=1}^N$ (not shown) after RANSAC is 2753. With $N = 2753$ keypoint matches, the number of entries in the \mathbf{A} matrix is $2N \times 9 = 49554$ and the number of entries in \mathbf{W}_* is $2N \times 1 = 5506$, assuming 8 bytes per entry in \mathbf{A} and \mathbf{W}_* , the total number of bytes is 440480. This is equal to 0.42007 MegaBytes, which is a low memory consumption requirement for each \mathbf{H}_* . On top of that, it is not necessary to save the values of \mathbf{W}_* for each \mathbf{H}_* . Thus, in principle, the computational cost does not grow with the number of weighted homography matrices \mathbf{H}_* . This stitching result was generated in ~ 4 seconds.

to uniformly partition the 2D domain of image I into a grid of $C_1 \times C_2$ cells, and take the center of each cell as \mathbf{x}_* . Pixels within the same cell are then warped using the same homography.

Fig. 4.2(b) illustrates a mosaic result where the image I is partitioned into 100×100 cells. In this figure it is possible to observe that the as-projective-as-possible warp is globally projective for extrapolation, but adapts flexibly in the overlap region for better alignment.

Partitioning into cells effectively reduces the number of Weighted SVD (WSVD) instances to $C_1 \times C_2$. Moreover, each of the WSVD instances are independent of each other (see (4.6)), thus a straightforward approach to further speed up the computation of the as-projective-as-possible warps is to solve the WSVDs in parallel. Although, even without parallel computations, Moving DLT is a very fast and efficient approach for image stitching. This will be further discussed in Chapter 6.

A potential concern is that discontinuities in the warp may occur between cells since the partitioning effectively downsamples the smoothly varying weights in Eq. 4.8. In practice, as long as the cell resolution is sufficiently high, the effects of warp discontinuities are minimal (100×100 is an adequate size for all of the images tested in the experiments of this work).

4.3.2 Updating weighted SVDs

Further speedups of Moving DLT are possible by realising that, for most of the cells, due to the offsetting (4.8), many of the weights do not differ from γ .

Based on the input images (I and I') in Figs. 2.6(a) and 2.6(b), Fig. 4.3 histograms across all cells the number of weights that differ from γ (in this example, $\gamma = 0.0025$). A vast majority of cells ($> 40\%$) have fewer than 20 weights (out of a total of 2753) that differ from the offset γ .

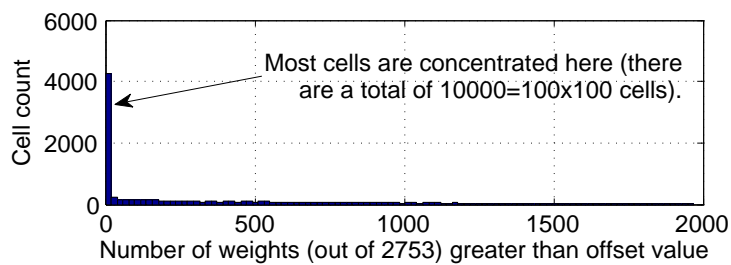


Figure 4.3: Histogram of number of weights $\neq \gamma$ for the cells in Fig. 2.6(b).

To exploit this observation a WSVD can be updated from a previous solution instead of being computed from scratch. Defining $\mathbf{W}_\gamma = \gamma \mathbf{I}$, let the columns of \mathbf{V} be the right singular vectors of $\mathbf{W}_\gamma \mathbf{A}$. Define the eigendecomposition

$$\mathbf{A}^T \mathbf{W}_\gamma^T \mathbf{W}_\gamma \mathbf{A} = \mathbf{V} \mathbf{D} \mathbf{V}^T \quad (4.9)$$

as the base solution. Let $\tilde{\mathbf{W}}$ equal \mathbf{W}_γ except the i -th diagonal element that has value \tilde{w}_i .

The eigendecomposition of $\mathbf{A}^T \tilde{\mathbf{W}}^T \tilde{\mathbf{W}} \mathbf{A}$ can be obtained as the rank-one update

$$\mathbf{A}^T \tilde{\mathbf{W}}^T \tilde{\mathbf{W}} \mathbf{A} = \mathbf{V} \mathbf{D} \mathbf{V}^T + \rho \mathbf{r}_i \mathbf{r}_i^T = \mathbf{V} (\mathbf{D} + \rho \bar{\mathbf{r}}_i \bar{\mathbf{r}}_i^T) \mathbf{V}^T,$$

where $\rho = (\tilde{w}_i^2 / \gamma^2 - 1)$, \mathbf{r}_i is the i -th row of \mathbf{A} , and $\bar{\mathbf{r}}_i = \mathbf{V}^T \mathbf{r}_i$. The diagonalisation of the new diagonal matrix

$$\mathbf{D} + \rho \bar{\mathbf{r}}_i \bar{\mathbf{r}}_i^T = \tilde{\mathbf{C}} \tilde{\mathbf{D}} \tilde{\mathbf{C}}^T \in \mathbb{R}^{m \times m} \quad (4.10)$$

can be done efficiently using secular equations [75].

Multiplying $\tilde{\mathbf{V}} \tilde{\mathbf{C}}$ effectively yields the right singular vectors of $\tilde{\mathbf{W}} \mathbf{A}$. This can be done efficiently by exploiting the Cauchy structure in $\tilde{\mathbf{C}}$ [75]. The cost of this rank-one update is $\mathcal{O}(m^2 \log^2 m)$.

The WSVD for each cell can thus be obtained via a small number of rank-one updates to the base solution, each costing $\mathcal{O}(m^2 \log^2 m)$. Overall this is cheaper than computing from scratch, where for $\mathbf{W}_* \mathbf{A}$ of size $n \times m$, would take $\mathcal{O}(4nm^2 + 8m^3)$ even if just the right singular vectors are computed [28]. Note that, in Eq. 4.6, $(n = 2N) \gg (m = 9)$.

4.4 Comparing Moving DLT against (affine) Moving Least Squares for Image Stitching

Conceptually, Moving DLT is the homogeneous version of Moving Least Squares (MLS) commonly used in surface approximation [4] and image manipulation. In the context of warping points in 2D for image manipulation [71], MLS estimates for each \mathbf{x}_* an *affine transformation* (refer to Table 2.1) defined by a 3×3 matrix \mathbf{F}_*

$$\tilde{\mathbf{x}}'_* \sim \mathbf{F}_* \tilde{\mathbf{x}}_*, \quad (4.11)$$

where

$$\mathbf{F}_* = \underset{\mathbf{F}}{\operatorname{argmin}} \sum_{i=1}^N w_i^i \left\| \mathbf{F} \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix} - \begin{bmatrix} \mathbf{x}' \\ 1 \end{bmatrix} \right\|^2. \quad (4.12)$$

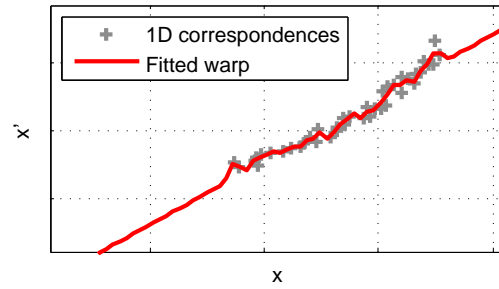


Figure 4.4: An affine-as-possible warp obtained by means of Moving Least Squares. The warp is able to flexibly interpolate the data that deviates from the projective trend. However, when there is no more data to stitch, the warp goes back to an affine, i.e., linear trend instead of reverting into a fully projective warp.

The problem in Eq. 4.12 is a weighted linear least squares problem [7, Pp. 42], which can be easily solved by obtaining the solution to the following *normal equations*

$$\begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ \vdots & \vdots & \vdots \\ x_N & y_N & 1 \end{bmatrix}^T \begin{bmatrix} w_*^1 & 0 & \dots & 0 \\ 0 & w_*^2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & w_*^N \end{bmatrix} \begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ \vdots & \vdots & \vdots \\ x_N & y_N & 1 \end{bmatrix} \mathbf{F}_*^T = \begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ \vdots & \vdots & \vdots \\ x_N & y_N & 1 \end{bmatrix}^T \begin{bmatrix} w_*^1 & 0 & \dots & 0 \\ 0 & w_*^2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & w_*^N \end{bmatrix} \begin{bmatrix} x'_1 & y'_1 & 1 \\ x'_2 & y'_2 & 1 \\ \vdots & \vdots & \vdots \\ x'_N & y'_N & 1 \end{bmatrix}. \quad (4.13)$$

Writing 4.13 in matrix form produces the following equation

$$\mathbf{B}^T \mathbf{W}_* \mathbf{B} \mathbf{F}_*^T = \mathbf{B}^T \mathbf{W}_* \mathbf{f}, \quad (4.14)$$

then, the solution to (4.12) is just

$$\mathbf{F}_* = ((\mathbf{B}^T \mathbf{W}_* \mathbf{B})^{-1} \mathbf{B}^T \mathbf{W}_* \mathbf{f})^T. \quad (4.15)$$

Including nonstationary weights $\{w_*^i\}_{i=1}^N$ produces flexible warps, but since each \mathbf{F}_* corresponds to an affine transformation, such warps are only *as-affine-as-possible*. Fig. 4.4 illustrates this warp.

The 1D analogy from Fig. 4.4 illustrates how MLS is able to interpolate or align non-ideal stitching data while going back to a global affine trend in the extrapolation area, thus, producing



Figure 4.5: Image stitching results obtained with as-affine-as-possible warps generated with MLS. Note the distortion of the alignments in the areas where the images do not overlap. The two red strokes drawn on top of the images illustrate the deformations of the warp. These strokes follow the lines of the tiles of the floor. Besides the image distortions, visible alignment mistakes remain (circled in red). This is probably a consequence of the implicit bias which constrains the flexibility of the alignments when there is few data to guide the warps. The images are taken with a camera undergoing rotation and translation. The size of the images is 730×487 pixels. The number of inliers after RANSAC is 415. Similar to Fig. 4.2, the domain of image I was partitioned into 100×100 cells.

the aforementioned as-affine-as-possible warps.

An affine transformation can be seen as a restricted form of a homography that preserves parallelism in the image alignments, this is the reason why an affine transformation is, in principle, not suitable for image stitching. However, one can argue that, given a large number of sufficiently small patches (i.e., given that the distances between each x_* are not too large), a locally varying affine warp should have similar degrees of freedom with respect to a locally varying homography. However, one would then need to determine how small the patches should be and how many of them. Even though the global behaviour is biased towards affine.

This “inadequate” affine bias tends to produce highly distorted results in the extrapolation region of the stitched images where there is no (keypoint) data to guide the local deformation and the warp reverts to the global affine model. This is illustrated in the stitching results from Fig. 4.5 where the affine prior from MLS introduces severe distortions in the aligned images.

Compare the previous results of MLS from Fig. 4.5 against the results obtained by the APAP warps generated by Moving DLT (Fig. 4.6), where the warps achieve perspective realism and no alignment mistakes remain.

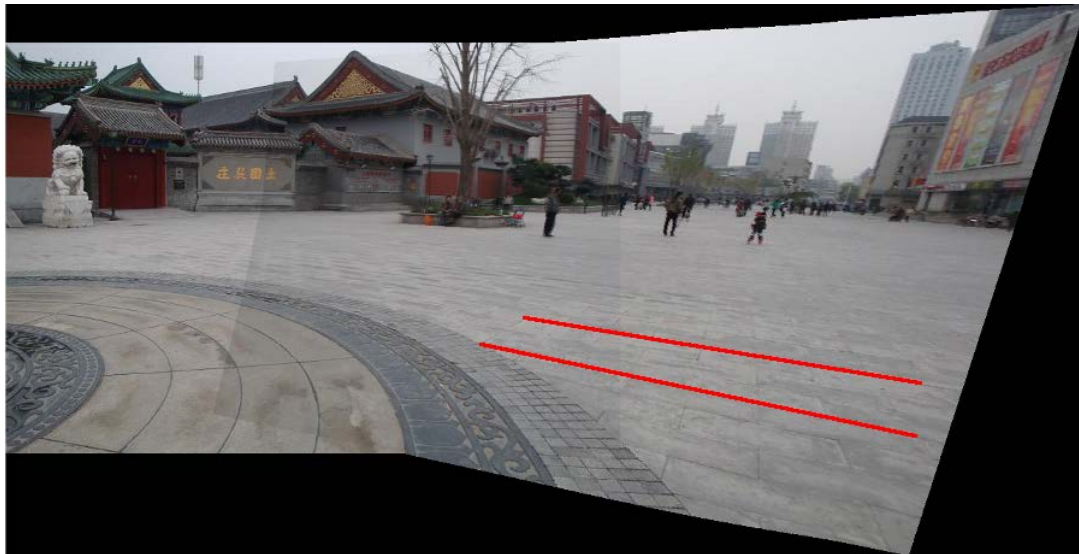


Figure 4.6: Image stitching results obtained with APAP warps generated with Moving DLT. In this image, few or no visible alignment mistakes remain and the scene is able to maintain geometric consistency. In particular, note how the two red strokes correspond to straight lines, thus, giving consistency and realism to the aligned mosaics. The domain of image I was partitioned into 100×100 cells.

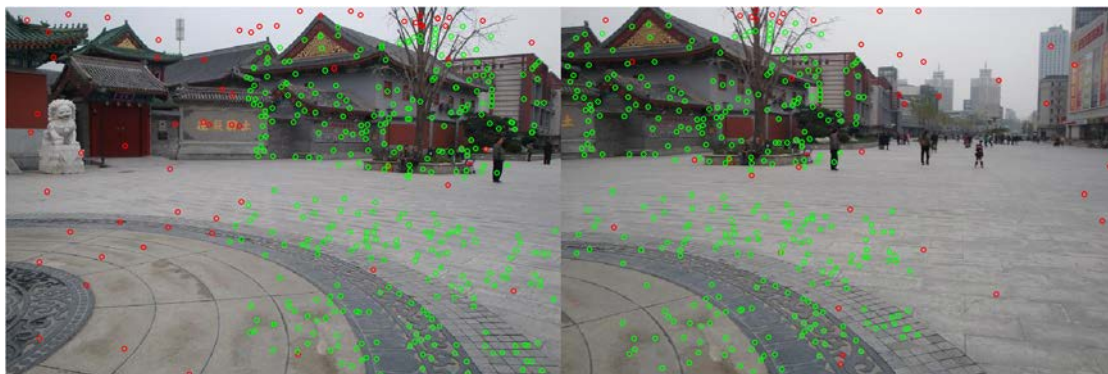


Figure 4.7: Feature points extracted from the *temple* image pair (this image pair appears in Fig. A.2). Inliers are shown in green, outliers in red. For clarity purposes the matching lines are omitted.

Also note that for both of the methods, the domain of image I was partitioned into the same number of cells: 100×100 . However, for MLS visible misalignments still remain. This is probably due to the affine bias that constrains the flexibility of the alignments in data-poor regions. Fig. 4.7 shows the keypoint matches between the two images I and I' in the previous mosaics. In particular, note how, for MLS, the misalignments persist in the regions of the images where the number of keypoint matches decrease. Moving DLT on the other hand, does not present this problem since it makes use of a more adequate and justifiable regularisation for image stitching.

It is important to mention that MLS is conceptually similar to the Smoothly Varying Affine

warps (SVA) [51], which were discussed during the literature review of this thesis (Section 3.5 from Chapter 3) and which is a state-of-the-art method for image stitching. MLS has not been proposed as an image stitching method *per se*. Thus, instead of comparing the proposed Moving DLT approach against MLS, experimental comparisons against SVA are performed. However, once again, the underlying principles and drawbacks of both SVA and MLS are the same: locally adapted affine warps with global affine priors. This is the reason why, not surprisingly, (similar to MLS) SVA also introduces very similar image deformations and presents similar alignment results in the stitched images (these results and comparisons will be reported in Chapter 6).

4.5 Summary

This Chapter presented the theory behind the novel Moving DLT approach for image stitching. Since the method is based on Moving Least Squares, there are some similarities between these two approaches. Specifically, both of these methods make use of weights and weighted piecewise smooth estimates in order to approximate complex, functions.

However, besides the similarities, there are also very important differences and advantages of Moving DLT over MLS and other similar approaches. The most important advantage of Moving DLT over MLS is the implicit *projective bias* for “non-rigid” homography estimation. This most important characteristic is what gives Moving DLT the ability to produce state-of-the-art stitching results as it will be seen in the experimental evaluations. Also important is the fact that, contrary to other similar approaches, Moving DLT remains a very simple and yet very powerful approach for flexible warp estimation.

This Chapter also presented different options for balancing the accuracy of the estimated warps against the speed of the whole estimation process. On top of this, since each homography *estimation process* is independent from each other, Moving DLT can be easily implemented in parallel by making use of GPUs or other similar hardware architectures. This can give the method even further speed improvements; maybe even achieving real-time computation, which can be extremely useful for task like live-3D reconstruction, simultaneous localisation and mapping (SLAM) and augmented reality.

The next contribution of this research is the development of a method for simultaneously estimating multiple as-projective-as-possible warps. This method, named *Bundled Moving DLT*, is given in the following Chapter.

Chapter 5

Simultaneous Refinement of Multiple As-Projective-As-Possible Warps

5.1 Introduction

This Chapter presents a novel formulation of bundle adjustment which is able to simultaneously refine multiple as-projective-as-possible warps. This method effectively avoids the pairwise error propagation of most of the incremental image stitching approaches while making use of the proposed non-rigid projective warps.

The novel formulation presented in this Chapter is the first bundle adjustment method for simultaneously aligning multiple flexible warps for image stitching.

Note that if one insists on incremental (pairwise) stitching, Moving DLT is still an excellent choice; as it will be shown in Section 6.2 from Chapter 6, the alignment error of Moving DLT is much smaller than other pairwise stitching methods. This means that the cumulative error of such technique will also be much smaller.

5.2 Selecting the Reference Frame

Given a set of input images $\{I^k\}_{k=1}^K$, the initial step is to map all the keypoints in the images onto a common reference frame I^R .

Though not necessary for bundle adjustment, for simplicity I^R is chosen from one of the input images. To this end, the keypoint-based panorama recognition method of [12, Sec. 3] is applied

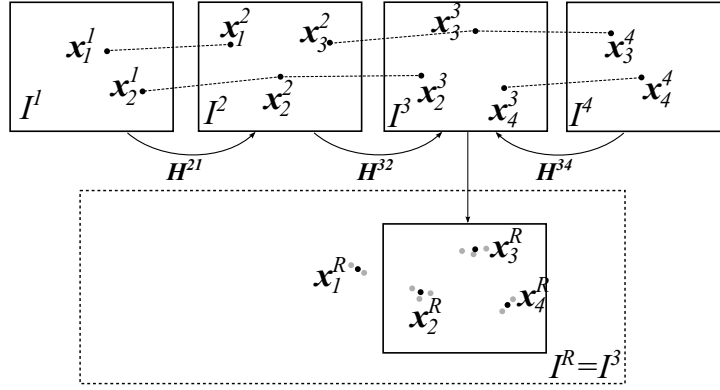


Figure 5.1: Warping the keypoints (through pairwise homography “chaining”) from the input images $\{I^k\}_{k=1}^K$ onto a common reference frame I^R . Points in black in I^R are the averaged projections of the keypoint mappings (in gray). I^R was chosen from input image I^3 .

in order to identify pairs of overlapping images and construct an image connection graph. The graph is traversed to find the node (image) with the largest number of edges which is chosen as the reference frame I^R .

The byproduct of the panorama recognition step is a set of planar homographies between each pair of overlapping images (similar to Section 3.2). The homographies are then chained and used to warp the keypoints in all the images onto I^R . To minimise propagation errors during this process, an optimal chaining order can be estimated e.g., the minimum spanning tree of the connection graph [58].

Within I^R , the coordinates of keypoints that are matched across several images, i.e., have the same identity (which is inferred from the pairwise image matching conducted in panorama recognition) are averaged. The result of this process is a set of coordinates $\{\mathbf{x}_i^R\}_{i=1}^N$ in I^R , where each \mathbf{x}_i^R is (potentially) matched to a keypoint \mathbf{x}_i^k in the k -th image I^k . Fig. 5.1 provides an illustration of this initialization step.

5.3 Bundled Moving Direct Linear Transformation (Bundled Moving DLT)

Once the reference frame I^R has been selected, given an arbitrary location \mathbf{x}_* in I^R , the goal is to estimate a set of location dependent homographies $\{\mathbf{H}_*^k\}_{k=1}^K$, where each \mathbf{H}_*^k maps \mathbf{x}_* from I^R to I^k following

$$\tilde{\mathbf{x}}_*^k \sim \mathbf{H}_*^k \tilde{\mathbf{x}}_*. \quad (5.1)$$

The pixel intensity at \mathbf{x}_* in I^R is composited from the original intensity at \mathbf{x}_* in I^R (if it exists) and the pixel intensities (if they exist) at locations $\{\mathbf{x}_*^k\}_{k=1}^K$ in $\{I^k\}_{k=1}^K$.

To estimate the required homographies $\{\mathbf{H}_*^k\}_{k=1}^K$ for position \mathbf{x}_* , it is necessary to simultaneously minimise the *transfer error* of all correspondences. Specifically, this thesis proposes minimising the following cost function

$$E_*(\Theta) = \sum_{i=1}^N \frac{w_*^i}{\sum_{k=1}^K \delta_{ik}} \sum_{k=1}^K \delta_{ik} \|\mathbf{x}_i^k - f(\mathbf{q}_i, \mathbf{H}_*^k)\|^2, \quad (5.2)$$

where $\Theta = [\mathbf{H}_*^1, \dots, \mathbf{H}_*^K, \mathbf{q}_1, \dots, \mathbf{q}_N]$ and $f(\mathbf{q}, \mathbf{H})$ is the projective warp (in inhomogeneous coordinates) defined as

$$f(\mathbf{q}, \mathbf{H}) = \begin{bmatrix} \mathbf{r}_1[\mathbf{q}^T \ 1]^T & \mathbf{r}_2[\mathbf{q}^T \ 1]^T \\ \mathbf{r}_3[\mathbf{q}^T \ 1]^T & \mathbf{r}_3[\mathbf{q}^T \ 1]^T \end{bmatrix}^T, \quad (5.3)$$

where $\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3$ are the three rows of homography \mathbf{H} .

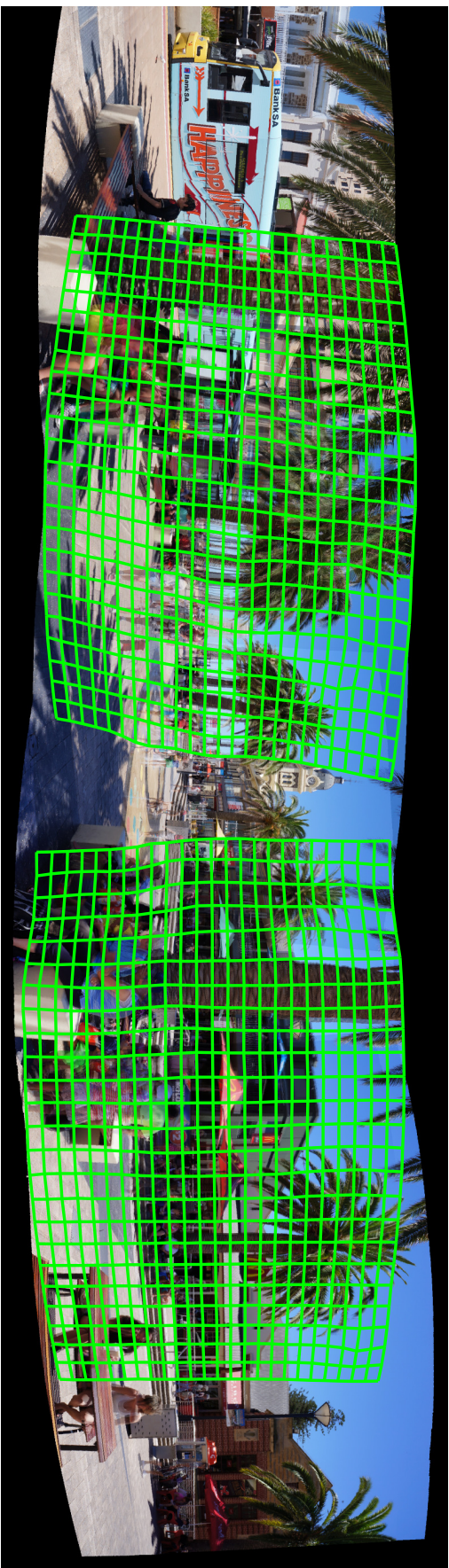
The optimized parameters include the point coordinates $\{\mathbf{q}_i\}_{i=1}^N$, which are essential to ‘‘couple’’ the homographies in bundle adjustment [84]. The coordinates $\{\mathbf{q}_i\}_{i=1}^N$ are initialized as the points $\{\mathbf{x}_i^R\}_{i=1}^N$ resulting from the homography chaining in Sec. 5.2. The k -th homography \mathbf{H}_*^k is initialized using Moving DLT on the correspondences between $\{\mathbf{x}_i^R\}_{i=1}^N$ and keypoints in I^k . Note that not all \mathbf{x}_i^R has a correspondence in I^k ; if the correspondence $\{\mathbf{x}_i^R, \mathbf{x}_i^k\}$ exists, the indicator $\delta_{ik} = 1$, else $\delta_{ik} = 0$. The division of each error term in (5.2) by $\sum_{k=1}^K \delta_{ik}$ ensures that points \mathbf{x}_i^R that are matched in many images do not dominate.

Note that, in this case, the local weights

$$w_*^i = \max(\exp(-\|\mathbf{x}_* - \mathbf{x}_i^R\|^2/\sigma^2), \gamma) \quad (5.4)$$

are computed by referring to the coordinates $\{\mathbf{x}_i^R\}_{i=1}^N$. This ensures that the optimized homographies are locally adapted to \mathbf{x}_* . One could also refer the weights to the points $\{\mathbf{q}_i\}_{i=1}^N$ which are iteratively updated. However, this simple scheme is sufficient to satisfactorily achieve the desired effect.

To reduce the number of instances of (5.2) to solve, as in Sec. 4.3 the domain of the reference image I^R is partitioned into cells. The center of each cell is taken as \mathbf{x}_* , and the estimated homographies for \mathbf{x}_* are applied to the pixels within the same cell. Further, the Moving DLT initialization across the cells can be accomplished as a series of efficient rank-one updates (see Section 4.3 in Chapter 4). Algorithm 4 summarizes the proposed Bundled Moving DLT method. Fig. 5.2(a) illustrates the multiple as-projective-as-possible warps estimated by means of Eq. 5.2.



(a) Aligned images with transformed cells overlaid to visualize the multiple as-projective-as-possible warps refined using the novel bundle adjustment scheme (Section 5.3).



(b) An example of a panorama generated through Bundled Moving DLT for image stitching. This panorama is composed of 7 images where each image is of size 2456×1632 pixels. The reference frame I^R is partitioned in 100×100 cells, and the number of points in I^R is 13380. The total time for generating this panorama was ~ 8 minutes. Other competing approaches take about the same time (8 minutes) for stitching only a pair of images of about $\frac{1}{4}$ of the size of the images used in this panorama.

Figure 5.2: Stitching results obtained with the proposed Bundled Moving DLT approach for panorama creation.

Algorithm 4 Simultaneous refinement of multiple as-projective-as-possible warps for panorama creation.

Require: Input images $\{I^k\}_{k=1}^K$ with overlaps.

- 1: Choose reference frame I^R from $\{I^k\}_{k=1}^K$.
 - 2: Map all keypoints from $\{I^k\}_{k=1}^K$ onto I^R .
 - 3: Match points $\{\mathbf{x}_i^R\}_{i=1}^N$ in I^R with points in $\{I^k\}_{k=1}^K$.
 - 4: Define $C_1 \times C_2$ cells in I^R .
 - 5: **for** each cell in I^R **do**
 - 6: Compute weights (5.4) for current cell center \mathbf{x}_* .
 - 7: **for** $k = 1, \dots, K$ **do**
 - 8: Apply Moving DLT (4.6) to yield homography \mathbf{H}_*^k .
 - 9: **end for**
 - 10: Refine all $\{\mathbf{H}_*^k\}_{k=1}^K$ with (weighted) bundle adjustment (5.2).
 - 11: Using $\{\mathbf{H}_*^k\}_{k=1}^K$, composite pixels in current cell.
 - 12: **end for**
 - 13: *Note: Moving DLT in Step 8 can be computed using rank-one updates.*
-

To give an idea of the size of problem (5.2), the size of the Jacobian is $(9K + 2N) \times (\sum_{i,k} \delta_{ik})$. However, each error term includes only one point \mathbf{q}_i , hence the Jacobian is extremely sparse. In order to efficiently solve this problem, this work makes use of the sparse Levenberg-Marquardt library of [1] for minimising (5.2). The 7-image panorama in Fig. 5.2(b) was created in ~ 7 minutes (time includes pixel compositing) in MATLAB. Each one of these 7 images is of size 2456×1632 pixels. The reference frame or canvas I^R is partitioned in 100×100 cells, and the total number of points in I^R is 13380. Of course, since the problems (5.2) are independent across the cells, they can be solved in parallel for speedup.

Previous methods on topology inference (e.g., [40, 58]) can also be applied to optimise the order of stitching. It should be noted that since the Bundled Moving DLT method can stitch multiple images simultaneously (provided they overlap with a reference image), it significantly reduces error propagation. This claim will be validated in the Experiment and Results Chapter of this thesis (Chapter 6).

5.4 Comparing Bundled Moving DLT and Bundle Adjustment

The equation in (5.2) is a *Weighted Bundle Adjustment* problem. Similar to Moving Least Squares and Moving DLT where for each pixel a weighted problem is solved, Bundled Moving DLT solves one weighted bundle adjustment problem for each one of the pixels (or cells) in I^R .

Contrary to this novel formulation, the classical approach for image stitching focuses on solving one single “non-weighted” bundle adjustment problem for all pixels in the reference frame. In particular, current bundle adjustment approaches solve the following (or some variant of the

following) problem:

$$E(\Theta) = \sum_{i=1}^N \frac{1}{\sum_{k=1}^K \delta_{ik}} \sum_{k=1}^K \delta_{ik} \|\mathbf{x}_i^k - f(\mathbf{q}_i, \mathbf{H}^k)\|^2, \quad (5.5)$$

where $\Theta = [\mathbf{H}^1, \dots, \mathbf{H}^K, \mathbf{q}_1, \dots, \mathbf{q}_N]$ and each \mathbf{H}^k is often initialised with the chained homographies obtained from the pairwise alignment process from Section 5.2. The first row in Fig 5.3 in page 70 presents an illustration of the stitching results, i.e., the initialisation obtained with such pairwise stitching approach.

After the problem in Eq. 5.5 has been solved (usually by means of the Levenberg-Marquardt algorithm), the estimated homographies $\{\mathbf{H}^k\}_{k=1}^K$ are used for mapping the pixels \mathbf{x}_* from the reference frame I^R to the corresponding $\{I^k\}_{k=1}^K$ via the following equation:

$$\tilde{\mathbf{x}}_*^k \sim \mathbf{H}^k \tilde{\mathbf{x}}_*, \quad (5.6)$$

this way, the pixel intensity at \mathbf{x}_* in I^R can be composited with the pixel intensities (if they exist) at locations $\{\mathbf{x}_*^k\}_{k=1}^K$ in $\{I^k\}_{k=1}^K$.

The main difference between the proposed Bundled Moving DLT formulation and the classical bundle adjustment approach in (5.5) is the fact that the latter has no weights (w_*^i) or, more specifically, all of the weights are equivalent (normally equal to 1). Thus, the estimated homographies are not location dependant, they correspond to a basic (non-weighted) projective warp or planar homography.

The second row in Fig. 5.3 shows a panorama obtained with the classical bundle adjustment formulation from Eq. 5.5. Even though bundle adjustment manages to evenly distribute the alignment errors across all of images in the panorama, obvious alignment errors remain due to the inability of the basic projective warp at characterising the data.

Bundled Moving DLT on the other hand, maps the pixels \mathbf{x}_* in the reference frame by means of the APAP warps. More specifically, each \mathbf{x}_* in I^R is mapped to images $\{I^k\}_{k=1}^K$ with the corresponding *locally weighted* homography $\{\mathbf{H}_*^k\}_{k=1}^K$ obtained through Eq. 5.2 (the mapping equation is shown in (5.1)) and, as mentioned before, these locally weighted homographies depend on the location of each particular \mathbf{x}_* . Because of this reason, Bundled Moving DLT is able to stitch the data that a basic projective model cannot characterise while evenly distributing the alignment errors across all images, thus, minimising or eliminating the dreaded error propagation of most the pairwise stitching approaches. As shown in Fig. 5.3 (third row), Bundled Moving DLT allows the generation of panoramas with almost no visible alignment mistakes.

Several other variants of the bundle adjustment problem from Eq. 5.5 have been proposed in the literature including approaches that make use of 3D reconstructions, pairs or triplets of inter-image homographies, rotation and intrinsic parameters' matrices, etc. Some of these variants were given in Section 3.3 of the literature review of this thesis and some others can be found in [77, Chapter 9.2]. However, it is important to realise that all of these approaches make use of 2D projective transformations for performing the image alignments. Thus, in principle, all of these methods are restricted to performing basic homography-based image stitching.

5.5 Summary

This Chapter presented a new formulation of bundle adjustment that allows the simultaneous estimation of multiple as-projective-as-possible warps for image stitching. Through the simultaneous estimation of multiple APAP warps, it is possible to obtain non-rigid image alignments that also minimise the error propagation that occurs with most of the incremental or pairwise stitching approaches.

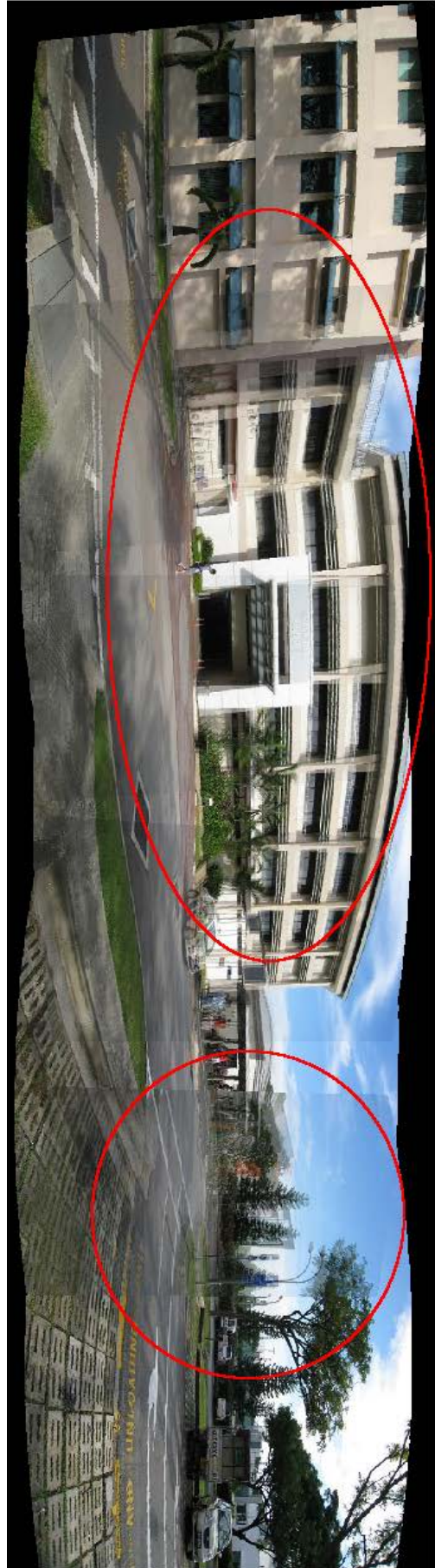
This Chapter also compared the “classical” formulation of image stitching with bundle adjustment and the novel Bundled Moving DLT method. In particular this Chapter described the main difference between these two approaches, which is the use of planar homographies (in bundle adjustment) compared with locally weighted homographies for warping the pixels in the images in Bundled Moving DLT.

The computational cost of, or size of the matrices required by the proposed Bundled Moving DLT formulation was also analysed in this Chapter and it was also noted that, since not all of the keypoint matches appear in all of the images, such matrices are generally sparse. This sparsity characteristic justifies the use of sparse solvers, which makes the method fast when compared against other current methods. For example, the time that Bundled Moving DLT requires for stitching a set of 7 high resolution images is similar to the time that other state-of-the-art approaches require for stitching only a *pair* of much smaller images. On top of that, every instance that the Bundled Moving DLT approach has to solve is independent from each other, thus, just like with Moving DLT, further speedups are possible by making use of GPU or other similar programming.

The purpose of the following Chapter is to present several experimental evaluations, which allow to demonstrate the effectiveness and efficiency of the methods proposed in this thesis over other, currently available, solutions for image stitching.



Figure 5.3: (Two page figure) Comparing the “raw” image alignment results of different techniques for panorama creation. The 5 images that compose the panorama were taken with a rotating and translating camera. In all of the cases, the central image is selected as the reference frame. First row: pairwise image stitching with projective transformations. Note how, as the images are further away from the reference frame, alignment mistakes are more obvious due to error propagation and amplification. Second row: Image stitching with (classical) Bundle Adjustment. The errors mistakes are evenly distributed amongst all of the images that compose the panorama. However, alignment mistakes can still be appreciated due to the “planarity” of the projective warps. Third row: Image stitching with Bundled Moving DLT. No obvious alignment mistakes are visible due to the flexibility of the APAP warps.



Chapter 6

Experiments and Results

6.1 Introduction

This Chapter compares the proposed Moving DLT and Bundled Moving DLT approaches against different state-of-the-art methods and against commercial software for image stitching. In particular, two types of comparisons are performed. First, *Moving DLT* is compared against other (pairwise) non-rigid methods for image stitching. In this case two types of experimental evaluations are obtained: *quantitative experiments* which numerically measure the accuracy of each one of the different stitching methods; and *qualitative experiments*, which allow a “visual inspection” of the advantages and disadvantages of each approach. On the second part of the experimental evaluation *Bundled Moving DLT* is compared against other state-of-the-art stitching tools that make use of bundle adjustment for panorama creation.

The main purpose of this Chapter is to demonstrate that better image alignments effectively impose less dependence on further post-processing and de-ghosting techniques, which translates into the generation of significantly better results.

In all of the experiments, input images that correspond to views that differ by rotation *and* translation were either generated or selected from recent stitching papers.

Similar to previous Chapters, the methods proposed in this thesis are referred as APAP (for as-projective-as-possible warps).

6.2 Comparisons with Flexible Warp Methods

First, APAP warps are compared against other flexible warp methods for image stitching, namely, Content Preserving Warps (CPW) [53], Dual Homography Warps (DHW) [23], and Smoothly Varying Affine warps (SVA) [51]. As mentioned in Sections 3.5 and 5.4, these methods can only stitch two images at a time, since they either cannot be easily extended to simultaneous estimation, or at least no such extensions exist in the literature.

The aim of these experiments is to cogently compare the alignment accuracy of the different image warping methods, thus, these results avoid sophisticated post-processing routines like seam cutting [2] and straightening [23], and simply blend the aligned images by intensity averaging such that any misalignments remain relatively obvious. For completeness, these experiments also include the results from the commercial tools Autostitch¹ [12] and Photosynth results, which were obtained by inputting two images at once. For Photosynth, the final postprocessed results are used since “raw” alignment results are not obtainable in the standard version of the software.

6.2.1 Preprocessing and Parameter Settings

Given a pair of input images, the first step is to detect and match SIFT keypoints using the VLFeat library [88]. Then RANSAC is used in order to remove mismatches, so that the remaining inliers are given to CPW, DHW, SVA and APAP. The performance of these methods depends on having the correct parameters. For CPW, DHW and SVA, the required parameters are tuned for best results²; refer to the respective papers for the list of required parameters. For APAP, the scale σ is varied within the range [8 12] for images of sizes 320×240 to 2456×1632 pixels. The number of keypoint matches per image pair varied within the range [161 5068]. The offset γ was empirically chosen from [0.0025 0.025] (even though the offset value can be tuned in order to achieve better results, such parameter is not crucial for performance so, alternatively, γ can be set to a fixed small value e.g., 0.01, which in this work was observed to provide “good” overall alignment results). The grid sizes C_1 and C_2 were both set to 100 on each dataset; the same grid resolution was also used in the CPW grid. In addition, following [53], for CPW the source image was pre-warped with the global homography estimated via DLT on the inliers returned by RANSAC. For Photosynth and Autostitch the original input images (with EXIF tags) were given.

¹The commercial version of Autostitch: “Autopano” was used in these experiments.

²Through personal communication, the correctness of the implementations of CPW, DHW and SVA and their parameter settings have been verified.

6.2.2 Qualitative Comparisons

Figs. 6.1 and 6.2 (pages 78 and 80) depict results on the *railtracks* and *temple* image pairs. The baseline warp (single homography via DLT on inliers) is clearly unable to satisfactorily align the images (since the views do not differ purely by rotation). SVA, DHW and Autostitch are marginally better, but significant ghosting remains. Further, note the highly distorted warp produced by SVA, especially in the extrapolation regions (e.g., the railtracks do not follow a straight line in Fig. 6.1 and there are visible deformations in the left rooftop and bottom tiles of Fig. 6.2). The errors made by Photosynth seem less “ghostly”, suggesting the usage of advanced blending or pixel selection [76] to conceal the misalignments. Nonetheless it is clear that the post-processing was not completely successful; observe the misaligned rail tracks and tiles on the ground. Contrast the above methods with APAP, which cleanly aligned the two images with few artifacts, thus, reducing the burden on post-processing. While CPW with pre-warping is able to produce good results, the rigidity constraints (a grid like in Fig. 4.2(b) is defined and discouraged from deforming) may counterproductively limit the flexibility of the warp (observe the only slightly nonlinear outlines of the warped images³). Thus, although the rail tracks and tiles are aligned correctly (more keypoint matches exist in these relatively texture-rich areas to influence the warp), ghosting occurs in regions near the skyline. Note that although APAP introduces a grid, it is for computational efficiency and not to impose rigidity.

As noted in the Appendix B-8.2 of [51], a major challenge for most of the flexible image stitching methods is that of aligning images that contain scenes with large depth discontinuities. The proposed APAP stitching warps were tested with such a scene (see Fig. 6.3 in page 82). In this scene, the trees, lamp post and bollards in the foreground cause sharp depth discontinuities. The results show that the APAP warps do not fail in such data. This is because the amount of camera translation is small compared to the overall scene depth (nonetheless, this small translation is sufficient to cause the baseline projective warp to break down - see Row 1 Fig. 6.3). Thus, while areas with sharp depth discontinuities may cause large deviations from the projective warp, the amount of deviation (mainly) depends on the camera translation distance.

Fig. 6.4 in page 84 shows the stitching results obtained on the *construction site* image pair. DHW and Autostitch present strong misalignments followed by the baseline projective warp and CPW where less alignment errors can be noticed. For SVA strong misalignments remain and the warped images look distorted in the extrapolation region. This distortion is probably because SVA reverts to a global affinity model instead of a projective model. For Photosynth

³As explained in Sec. 3.5, imposing warp rigidity is essential to prevent wobbling in video stabilisation [53]. Note that the original purpose of CPW was for video stabilisation and not image stitching.

no misalignments remain. However, the post-processing techniques from Photosynth failed to produce a convincing result, in particular, the hose around the left building does not follow a continuous line, it has been “cut” into two non-continuous pieces. Probably, the post-processing techniques from Photosynth could not conceal the errors from the image alignment stage. For APAP warps few visible misalignments remain.

Fig. 6.5 (page 86) presents the results obtained in the *train* image pair. For this image pair strong misalignments can be observed for the baseline warp and Autostitch. SVA produces better alignments than the baseline warp and Autostitch (note the alignments around the street light and palm trees) but still, misalignments and some distortion can be noticed (in particular, look how the lines of the bench are not straight). DHW was not able to align the images in the region around the bench. CPW presents good alignments around the bench and palm tree areas, but not around the street light. For Photosynth errors can be seen in the final post-processed result (note how the faces of the people have been erased and note how the bench and the tiles around the bench are not properly aligned). The APAP warps manage to generate better alignments in the street light, bench and palm trees, thus, producing fewer alignment mistakes and ghosting.

Fig. 6.6 in page 88 shows the stitching results for the *garden* image pair. The baseline warp, SVA and DHW results present misalignments around the building and side-walk areas. For Autostitch, misalignments are far more obvious. CPW and APAP present similar results, although, APAP makes less noticeable alignment mistakes. For Photosynth, there is a small error: the side-walk is not properly aligned in one section; other than that, no obvious errors remain (this is probably because of the post-processing and de-ghosting methods used by Photosynth).

In page 90, Fig. 6.7 depicts the stitching results on the *carpark* image pair. For the baseline method and DHW serious misalignments can be observed in the images. SVA introduced some distortion in the extrapolation regions (observe the ground between the building and the grass patch). Autostitch presents errors around the grass patch and side-walk areas. Photosynth produces an unexpected bending effect in the grass patch; this is most probably the side effect of the post-processing techniques. Here, CPW and APAP present some minor errors.

Fig. 6.8 shows the results in the *apartments* image pair. This image appears in page 92. SVA and Autostitch produce very obvious alignment errors. The baseline warp, DHW and CPW present similar results with the most visible misalignments around the car plate and around the windows of the apartments. APAP presents some alignment errors around the rooftops of the apartments and around the tyres of the car. For Photosynth no visible errors remain but this is due to the post-processing and de-ghosting methods that conceal the alignment mistakes.

Fig. 6.9 (in page 94) shows the stitching results obtained on the *chess/girl* image pair. As can be seen in Fig. 6.9, the baseline warp could not align the images properly, thus, significant ghosting remains. The APAP stitching method manages to align the images with little visible artifacts. The results from APAP are similar to those of obtained from Photosynth (with post-processing). For SVA, DHW and CPW significant alignment errors remain. On this dataset Autostitch was not able to produce a good result, as observed also in [51], therefore, the results from Autostitch are omitted.

Moving DLT, like most image stitching methods, relies on feature detection and matching. When the number of matched feature points is low, none of the methods compared here will be able to generate satisfactory stitching results. This can be observed in page 96 where Fig. 6.10 shows the stitching results on the *rooftops* image pair. In this image pair the number of matched feature points is very small in certain regions (specifically, around the squared rooftops). The best stitching result in Fig. 6.10 is produced by Photosynth which most likely makes use of post-processing and de-ghosting techniques.

Lastly, Fig. 6.11 in page 98 depicts results in the *couch* image pair. Here, SVA and Autostitch present the most obvious misalignments. For SVA the images look distorted, again. The baseline warp, DHW and CPW present similar results with the most obvious alignment errors around the left side of the couch. For Photosynth some artifacts remain after post-processing the images. In particular, the frame of the window and one of the bags on the couch are not properly aligned. The APAP warps introduce some alignment errors around the baggage area.

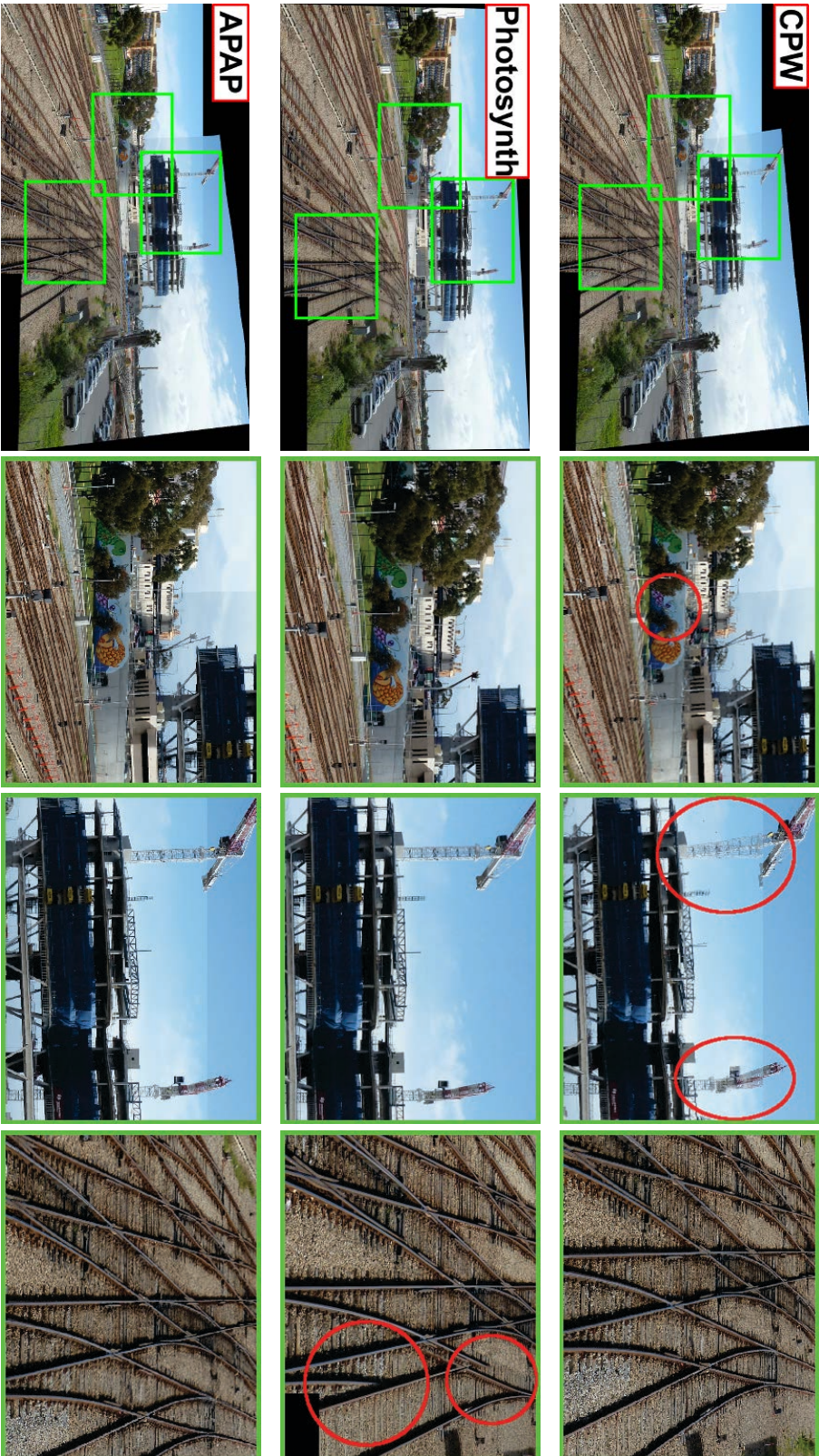
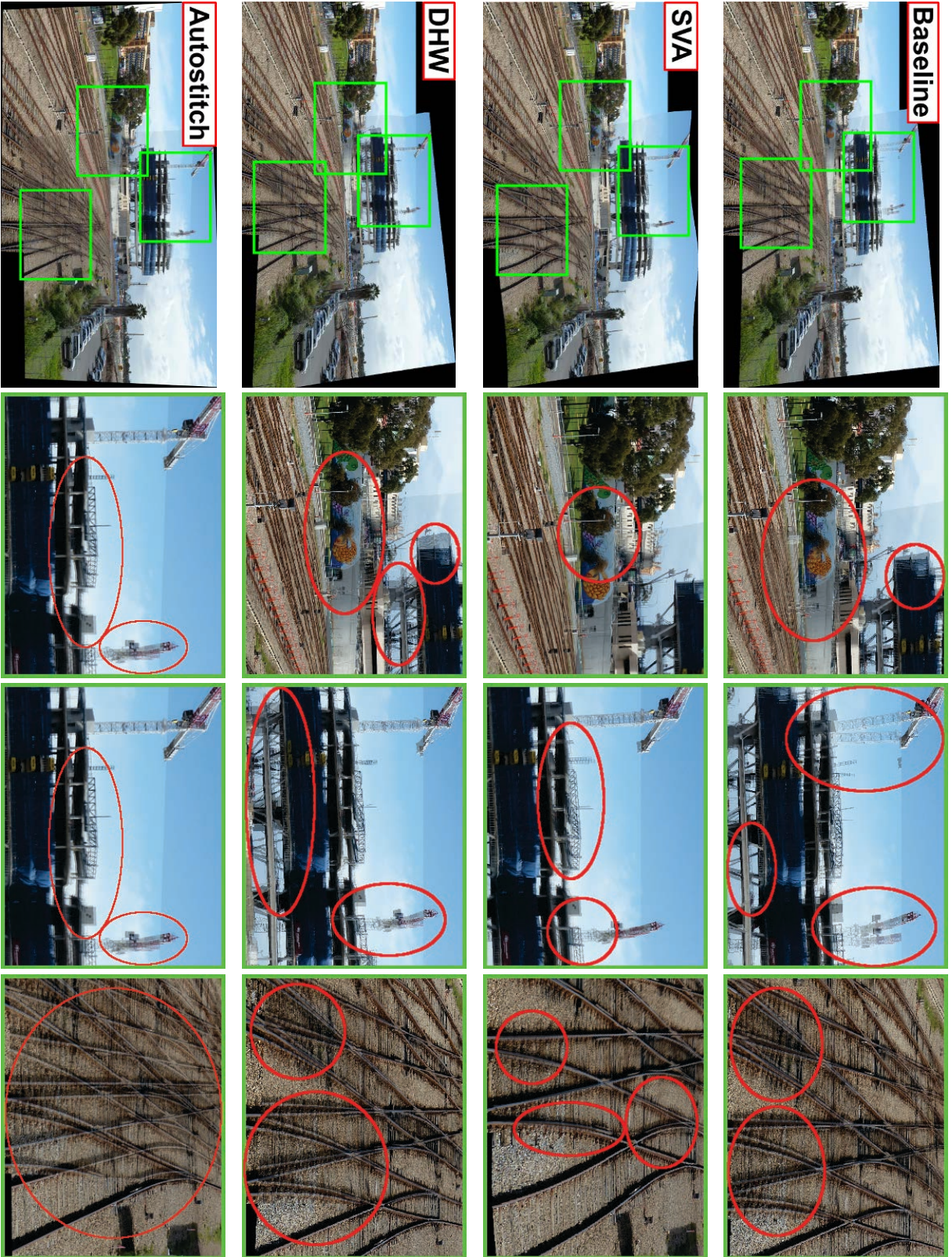


Figure 6.1: (Two page figure) Qualitative comparisons (best viewed on screen) on the *railtracks* image pair. The green boxes denote the image areas that are shown in detail. Red circles highlight errors. List of abbreviations: SVA-Smoothly Varying Affine, DHW-Dual Homography Warps, CPW-Content Preserving Warps, APAP-As Projective As Possible Warps.



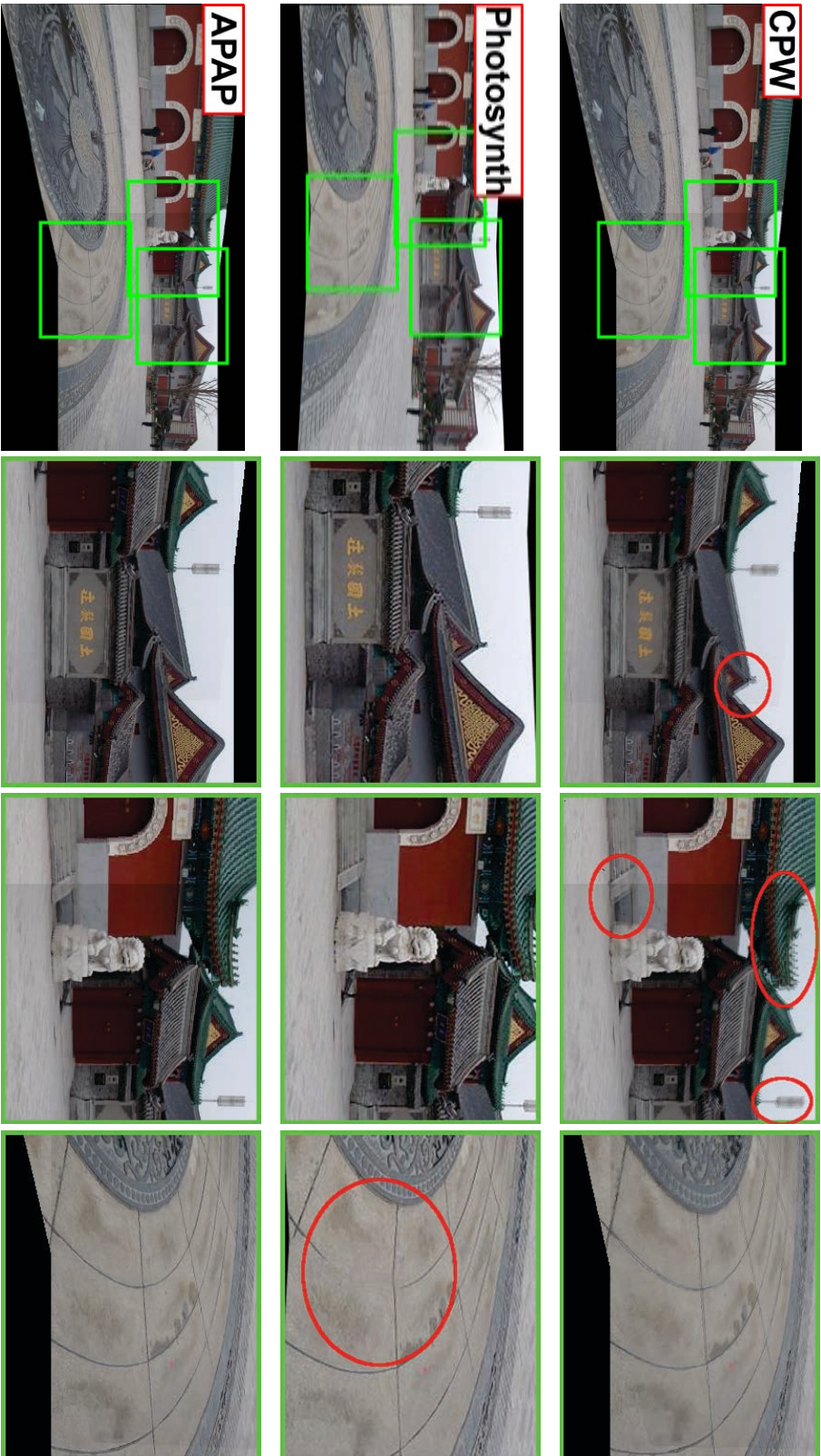


Figure 6.2: (Two page figure) Qualitative comparisons (best viewed on screen) on the *temple* image pair. The green boxes denote the image areas that are shown in detail. Red circles highlight errors. List of abbreviations: SVA-Smoothly Varying Affine, DHW-Dual Homography Warps, CPW-Content Preserving Warps, APAP-As Projective As Possible Warps.

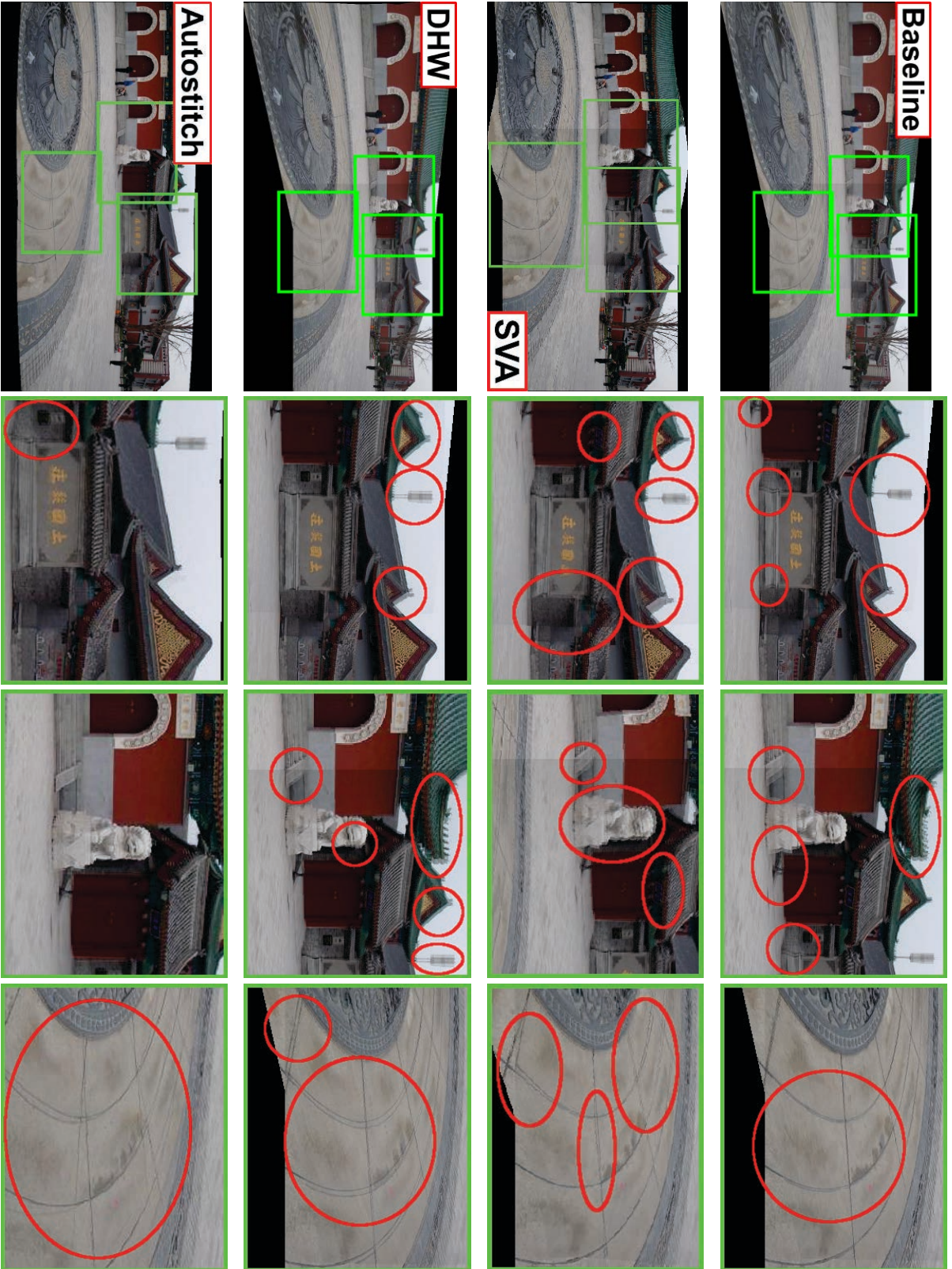
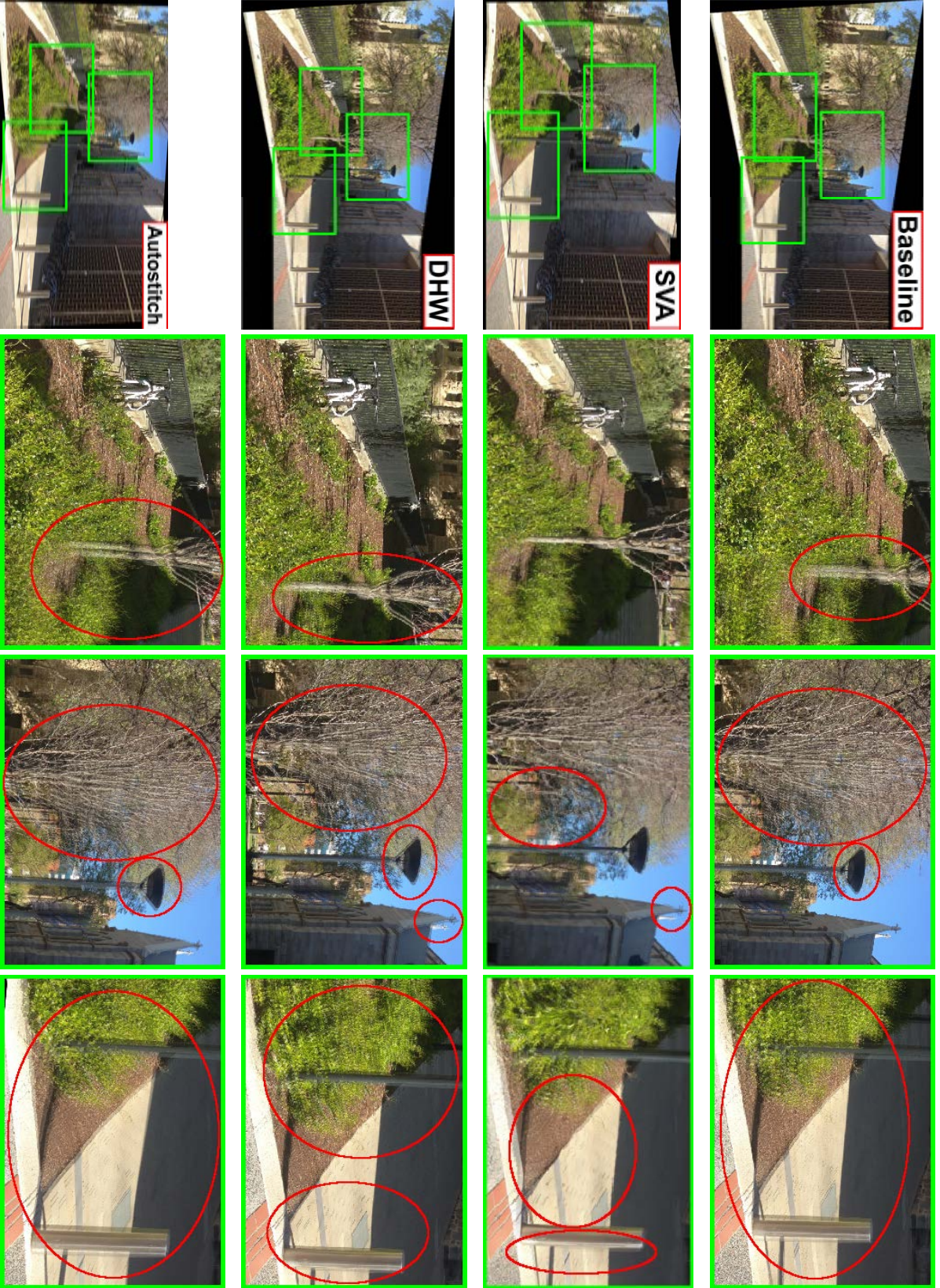




Figure 6.3: (Two page figure) Qualitative comparisons (best viewed on screen) on the *bikes* image pair. The green boxes denote the image areas that are shown in detail. Red circles highlight errors. List of abbreviations: SVA-Smoothly Varying Affine, DHW-Dual Homography Warps, CPW-Content Preserving Warps, APAP-As Projective As Possible Warps.



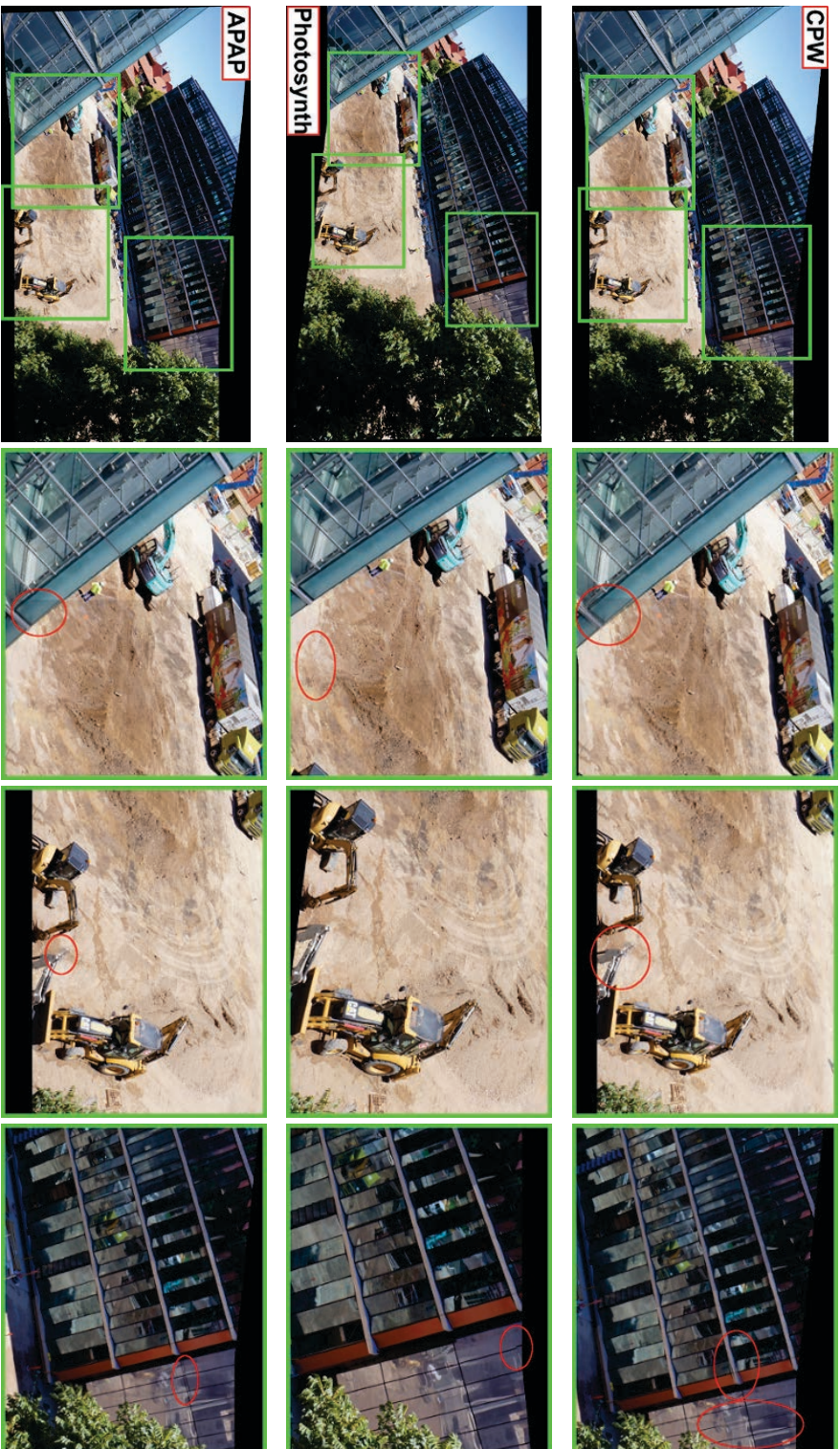


Figure 6.4: (Two page figure) Qualitative comparisons (best viewed on screen) on the *construction site* image pair. The green boxes denote the image areas that are shown in detail. Red circles highlight errors. List of abbreviations: SVA-Smoothly Varying Affine, DHW-Dual Homography Warps, CPW-Content Preserving Warps, APAP-As Projective As Possible Warps.

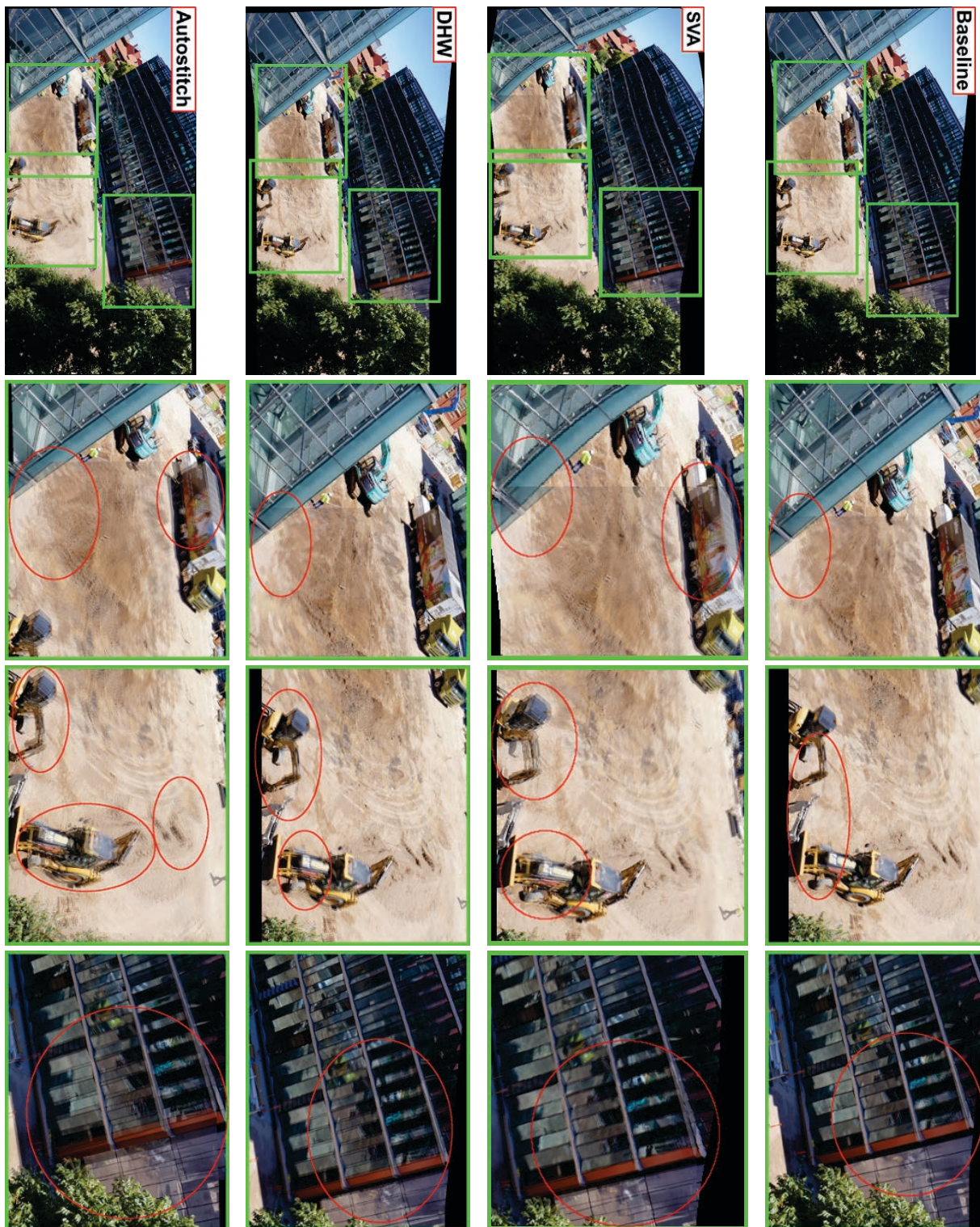




Figure 6.5: (Two page figure) Qualitative comparisons (best viewed on screen) on the *train* image pair. The green boxes denote the image areas that are shown in detail. Red circles highlight errors. List of abbreviations: SVA-Smoothly Varying Affine, DHW-Dual Homography Warps, CPW-Content Preserving Warps, APAP-As Projective As Possible Warps.

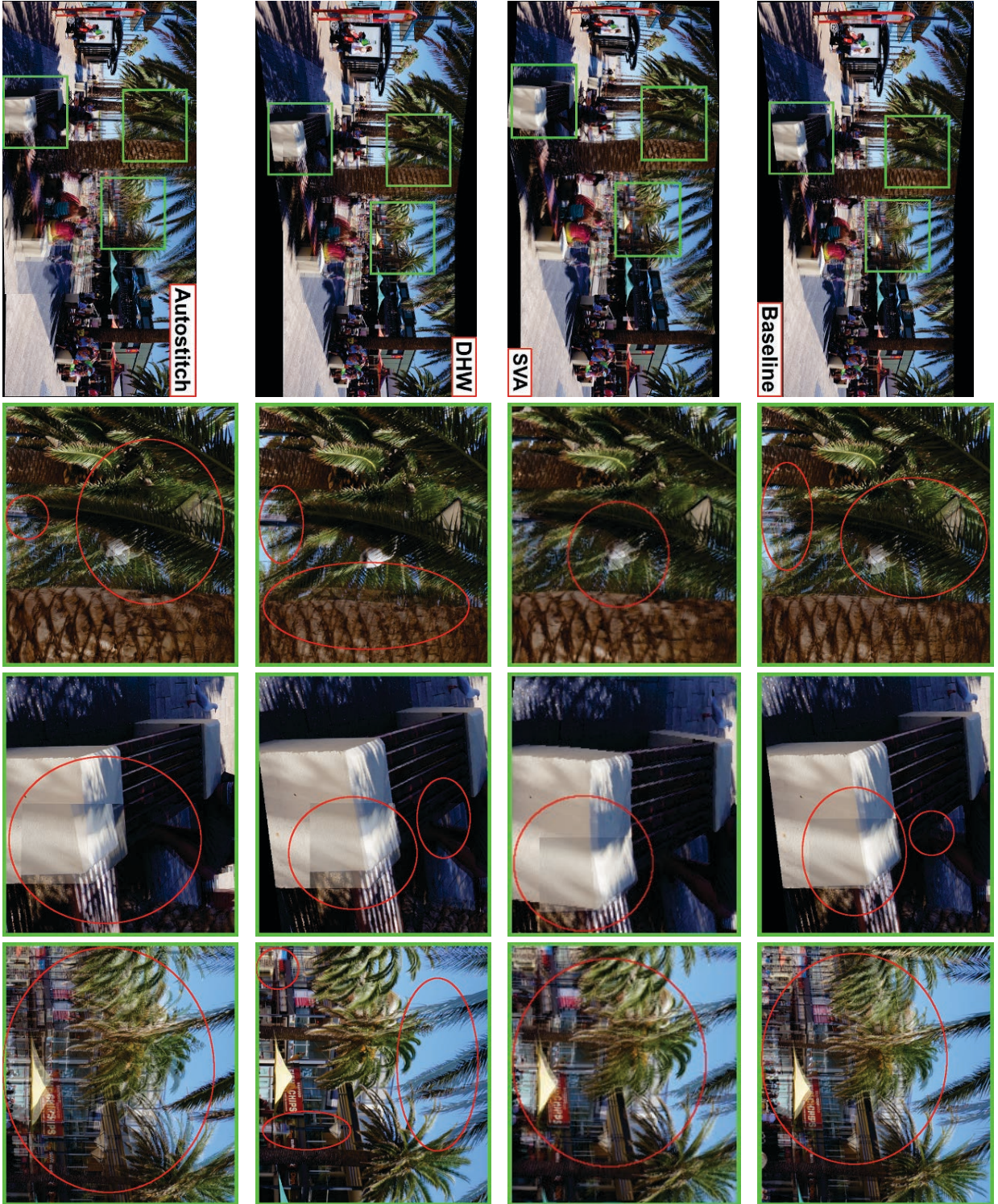
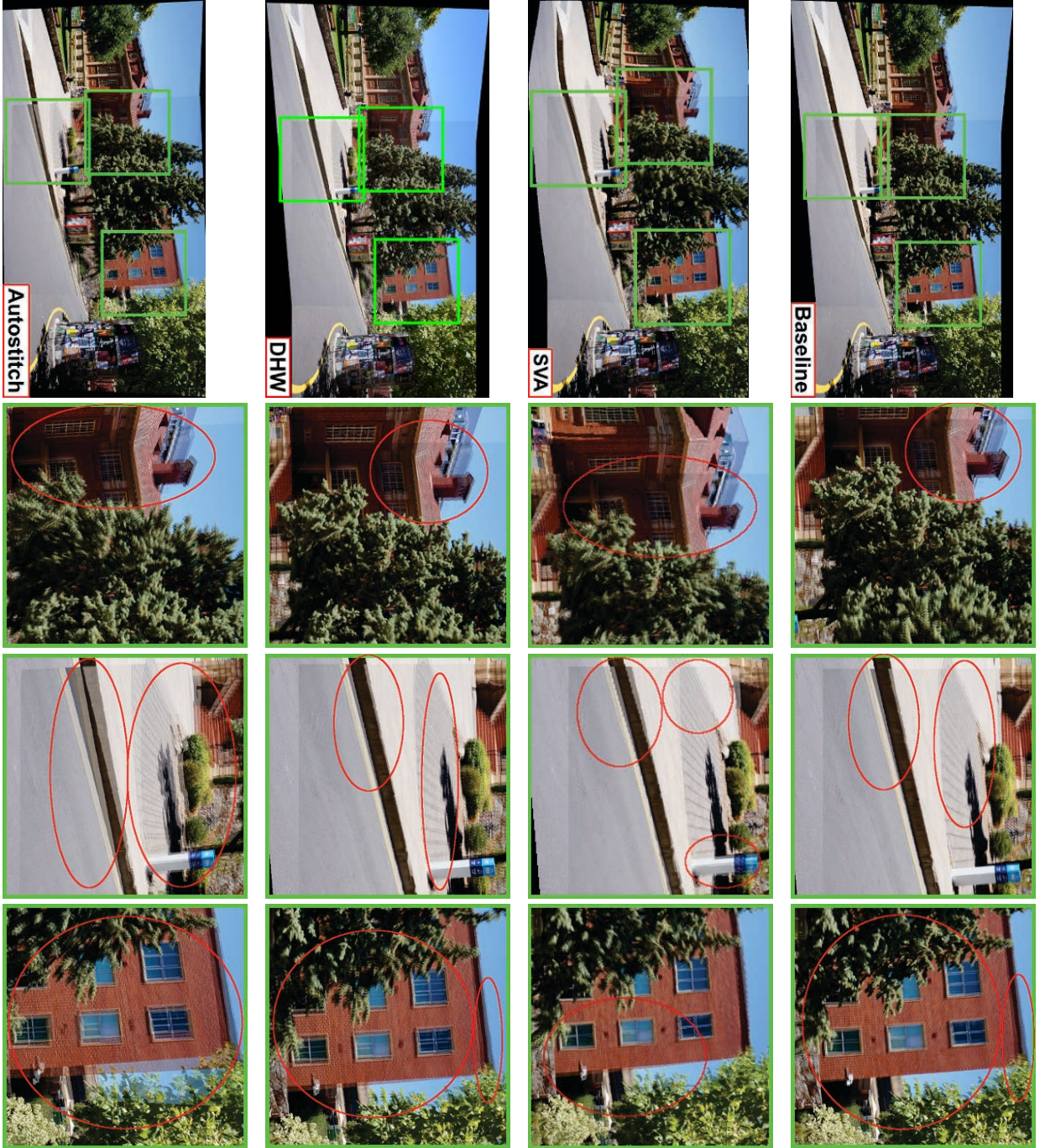




Figure 6.6: (Two page figure) Qualitative comparisons (best viewed on screen) on the *garden* image pair. The green boxes denote the image areas that are shown in detail. Red circles highlight errors. List of abbreviations: SVA-Smoothly Varying Affine, DHW-Dual Homography Warps, CPW-Content Preserving Warps, APAP-As Projective As Possible Warps.



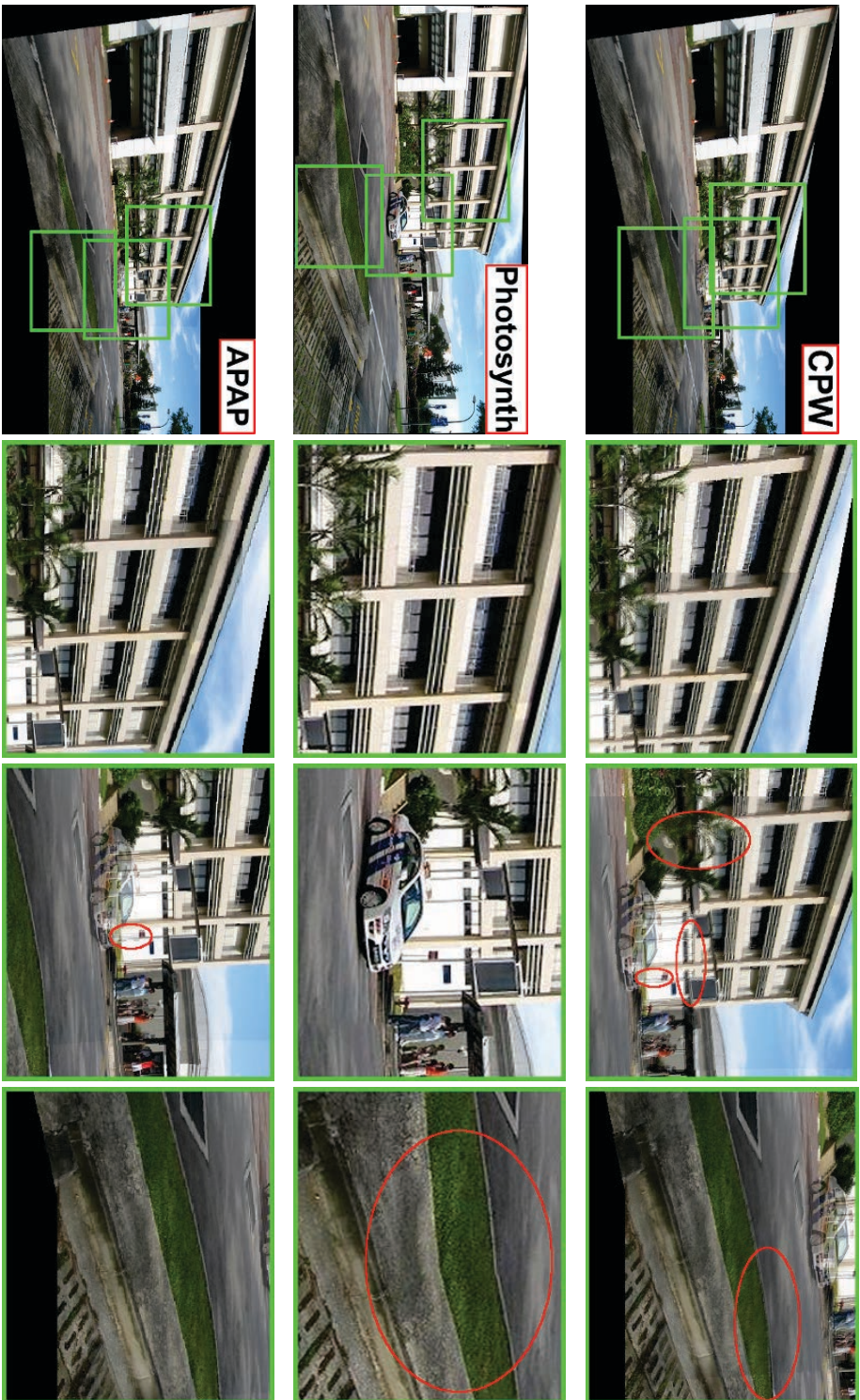


Figure 6.7: (Two page figure) Qualitative comparisons (best viewed on screen) on the *carpark* image pair. The green boxes denote the image areas that are shown in detail. Red circles highlight errors. List of abbreviations: SVA-Smoothly Varying Affine, DHW-Dual Homography Warps, CPW-Content Preserving Warps, APAP-As Projective As Possible Warps.

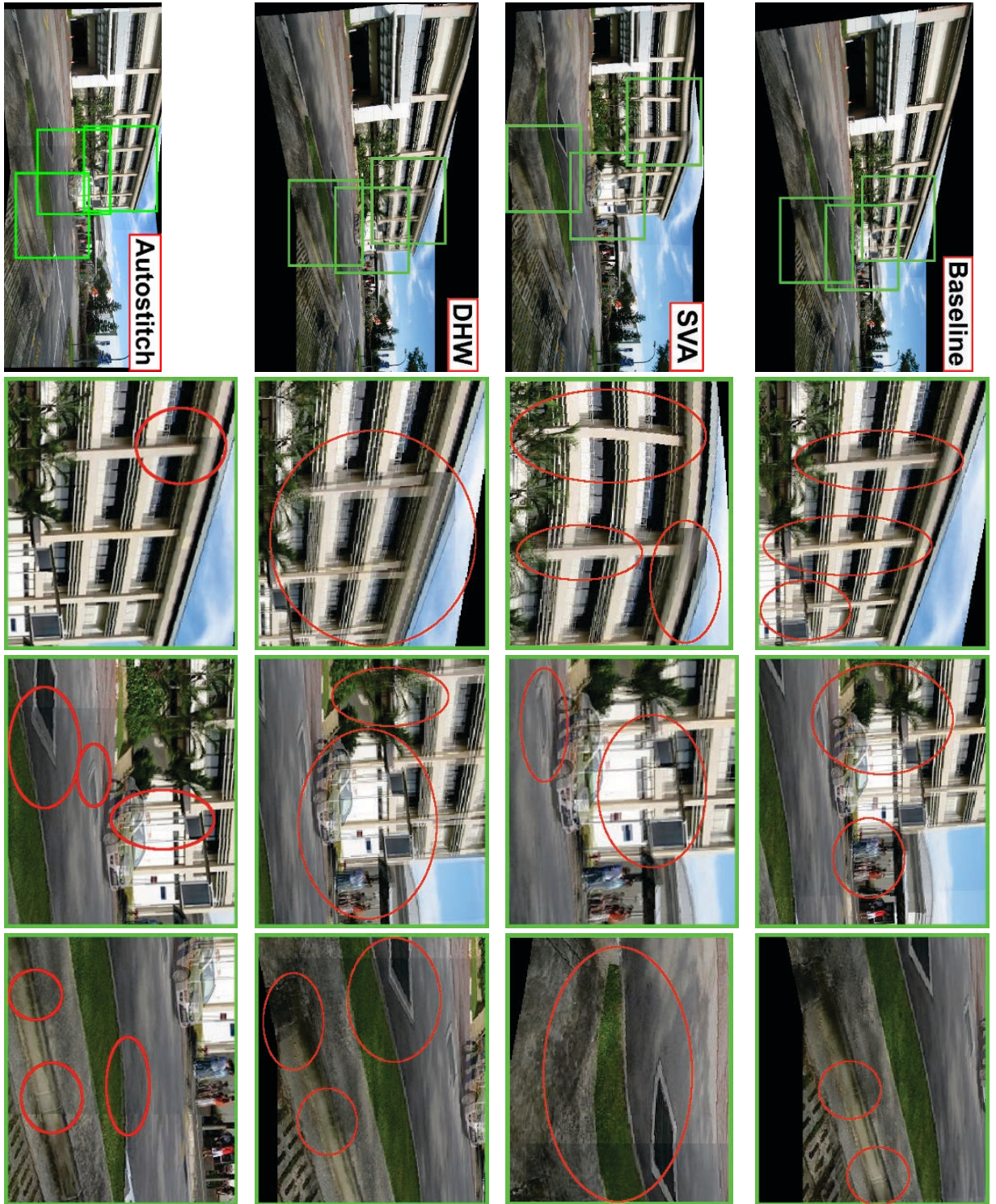
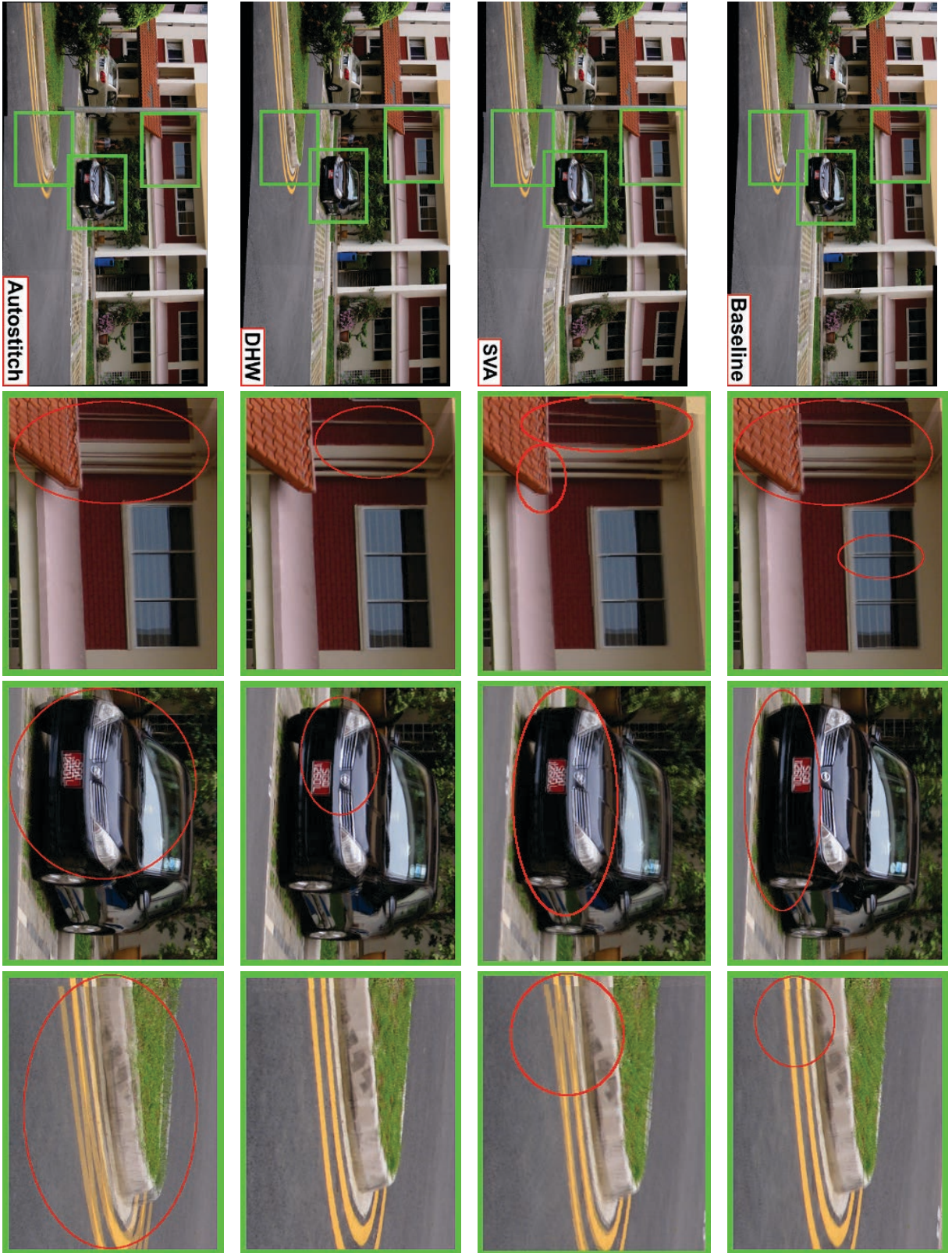




Figure 6.8: (Two page figure) Qualitative comparisons (best viewed on screen) on the *apartments* image pair. The green boxes denote the image areas that are shown in detail. Red circles highlight errors. List of abbreviations: SVA-Smoothly Varying Affine, DHW-Dual Homography Warps, CPW-Content Preserving Warps, APAP-As Projective As Possible Warps.



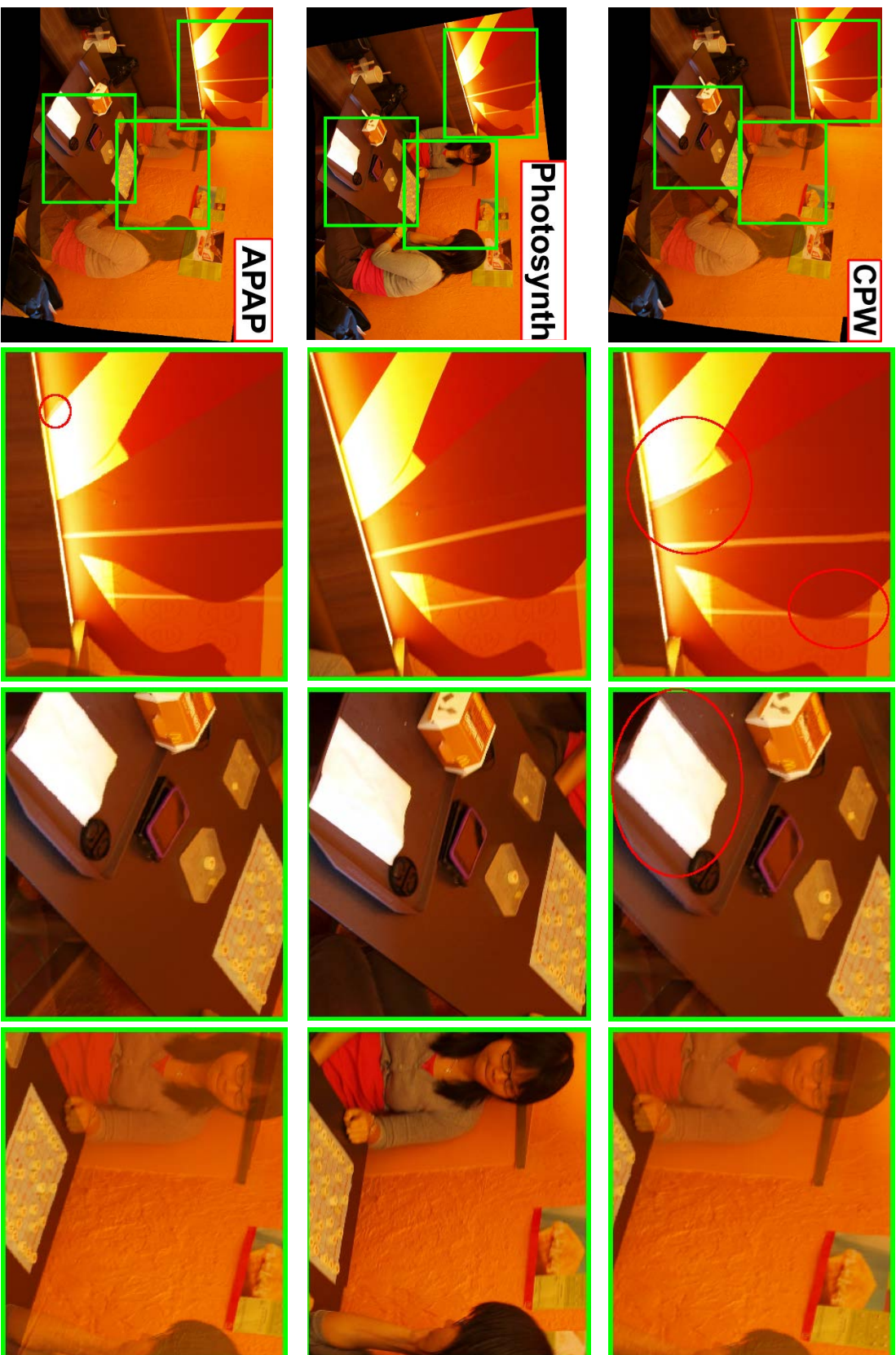


Figure 6.9: (Two page figure) Qualitative comparisons (best viewed on screen) on the *chess/girl* image pair. The green boxes denote the image areas that are shown in detail. Red circles highlight errors. List of abbreviations: SVA-Smoothly Varying Affine, DHW-Dual Homography Warps, CPW-Content Preserving Warps, APAP-As Projective As Possible Warps.

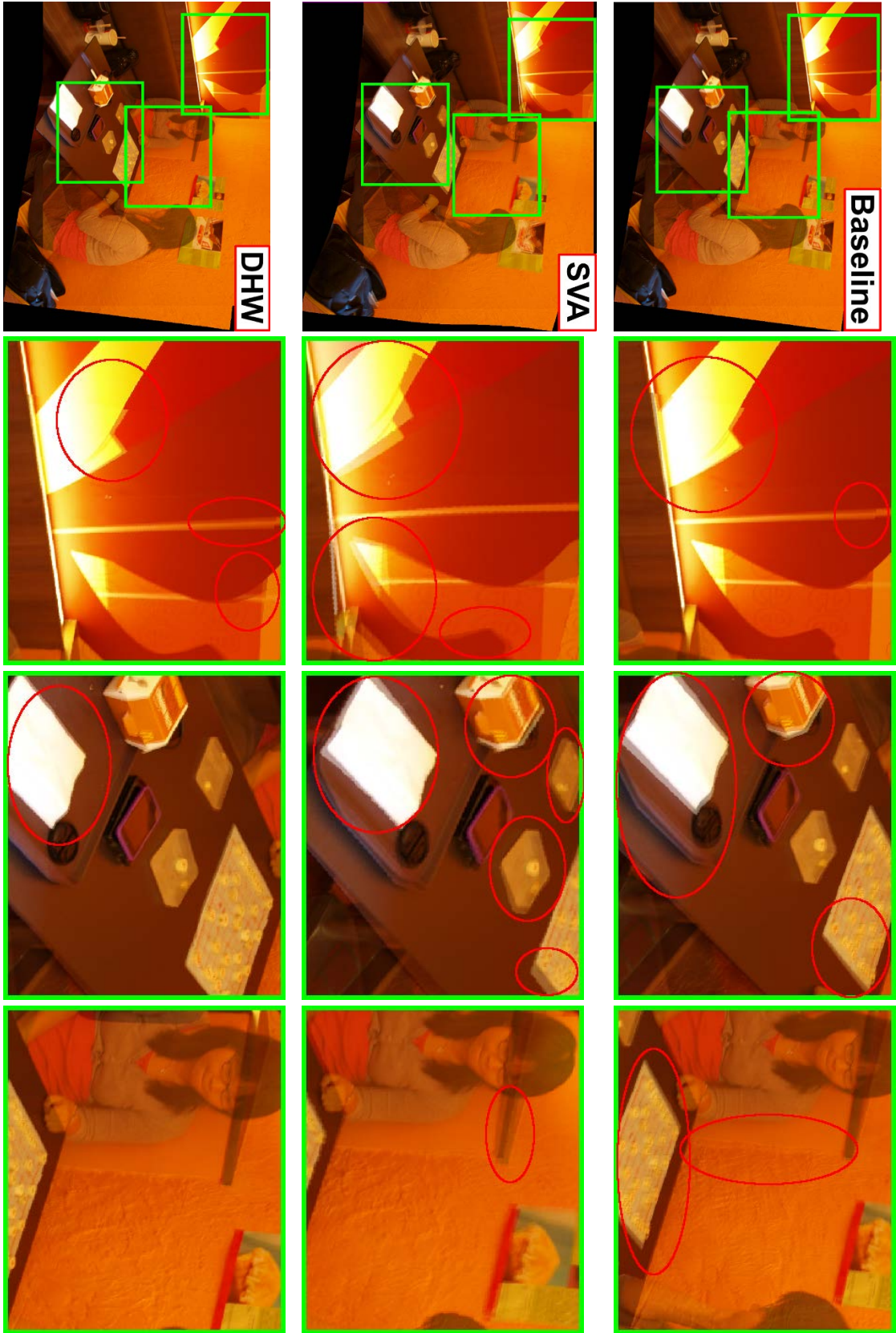
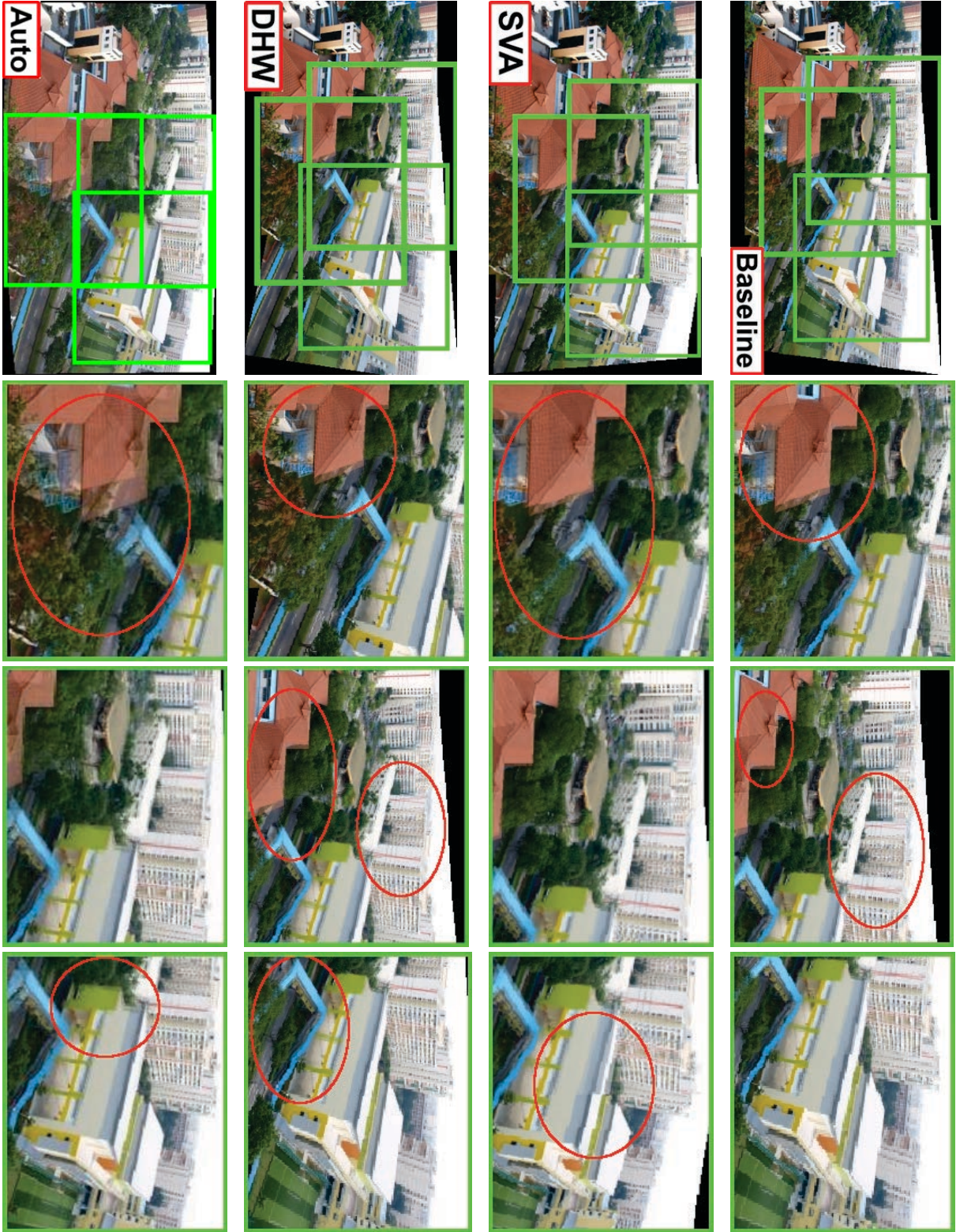




Figure 6.10: (Two page figure) Qualitative comparisons (best viewed on screen) on the *rooftops* image pair. The green boxes denote the image areas that are shown in detail. Red circles highlight errors. List of abbreviations: SVA-Smoothly Varying Affine, DHW-Dual Homography Warps, CPW-Content Preserving Warps, APAP-As Projective As Possible Warps.



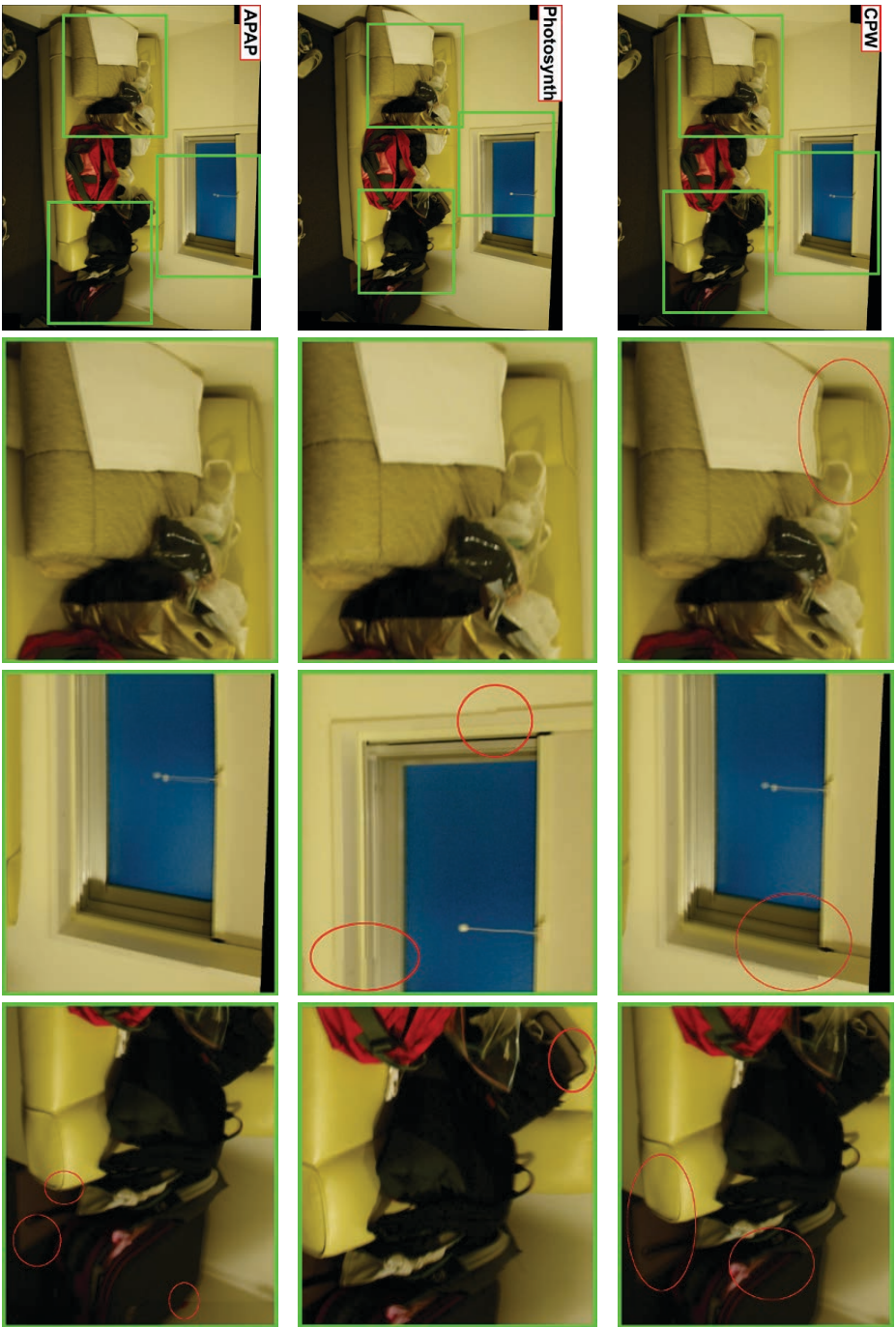
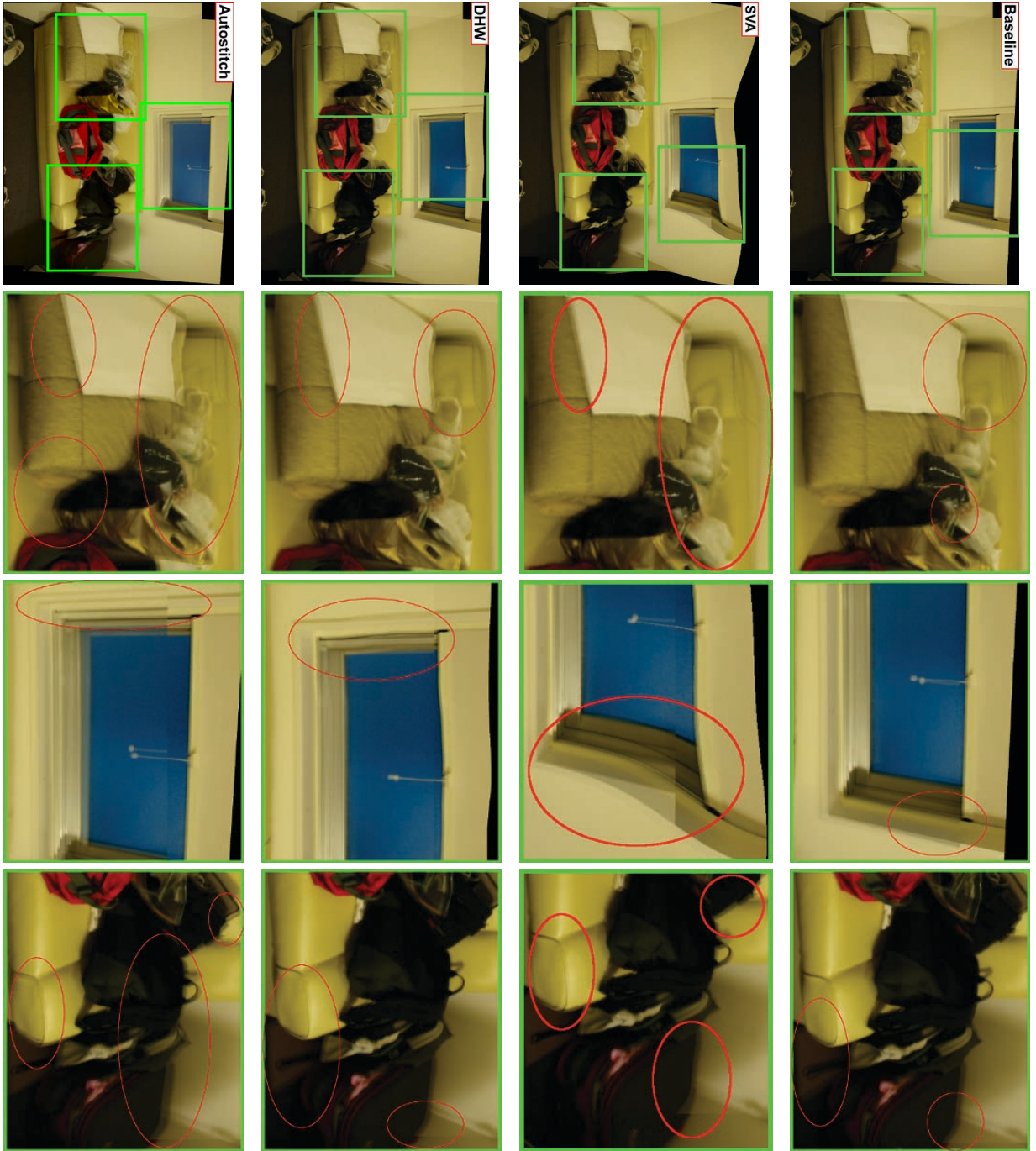


Figure 6.11: (Two page figure) Qualitative comparisons (Best viewed on screen) on the *couch* image pair. The green boxes denote the image areas that are shown in detail. Red circles highlight errors. List of abbreviations: SVA-Smoothly Varying Affine, DHW-Dual Homography Warps, CPW-Content Preserving Warps, APAP-As Projective As Possible Warps.



6.2.3 Runtime Information

For DHW, CPW, SVA and APAP, the total duration time for warp estimation (plus any data structure preparation), pixel warping and blending was recorded. All methods were run in MATLAB with C Mex acceleration for warping and blending. APAP takes in the order of seconds (< 6 in the current experiments), while CPW typically requires tens of seconds to few (3-5) minutes. In contrast, SVA scales badly with the number of keypoint matches, most likely due to the underlying point set registration method [59]. While 8 minutes was reported in [51] for 500×500 -pixel images, in these experiments SVA takes between 1 minute for the *rooftops* image pair (320×240 pixels per image with 161 keypoint matches) and ~ 2 hours for the *construction site* (2000×1329 pixels per image with 5068 keypoint matches). DHW, Autostitch and Photosynth typically take ~ 2 seconds.

6.2.4 Quantitative Benchmarking

To quantify the alignment accuracy of an estimated warp $f : \mathbb{R}^2 \mapsto \mathbb{R}^2$, the following experiments obtain the root mean squared error (RMSE) of f on a set of keypoint matches $\{\mathbf{x}_i, \mathbf{x}'_i\}_{i=1}^N$, i.e., $\text{RMSE}(f) = \sqrt{\frac{1}{N} \sum_{i=1}^N \|f(\mathbf{x}_i) - \mathbf{x}'_i\|^2}$. For an image pair the available SIFT keypoint matches are partitioned into a “training” and “testing” set. The training set is used to learn the warps, and the RMSE is evaluated over both sets.

These experiments also employed the error metric of [52]: a pixel \mathbf{x} in the source image is labeled as an outlier if there are no similar pixels in the neighbourhood of $f(\mathbf{x})$ in the target image. Following [52], neighbourhood is defined by a 4-pixel radius, and two pixels are judged similar if their intensities differ by less than 10 gray levels. The percentage of outliers resulting from f is regarded as the warping error. Note that pixels that do not exist in the overlapping region are excluded from this measure. f is estimated using only the data in the training set. Table 6.1 depicts the average errors (over 20 repetitions) on 11 challenging real image pairs, 6 of which were provided by the authors of [23, 51]. It is clear that APAP provides the lowest errors (RMSE and % outliers) in most of the image pairs.

To further investigate the accuracy of the different stitching methods under non-ideal conditions, synthetic 2D images were produced by projecting 3D point clouds onto two virtual cameras. The point clouds were laser scanned from parts of buildings in a university campus; see Column 1 in Fig. 6.12 for the point clouds used. The camera intrinsics and poses were controlled such that the projections fit within 200×200 -pixel images. The projections yield a set of two-view point matches that permit the direct application of the various warp estimation methods. For each

Dataset		Base	DHW	SVA	CPW	APAP
<i>railtracks</i>	-Training set error	13.91	14.09	7.48	6.69	4.51
	-Testing set error	13.95	14.12	7.30	6.77	4.66
	-% outliers	21.14	20.48	16.73	16.42	16.86
<i>temple</i> (from [23])	-Training set error	2.66	6.64	12.30	2.48	1.36
	-Testing set error	2.90	6.84	12.21	2.54	2.04
	-% outliers	11.65	12.27	12.29	12.65	11.52
<i>bikes</i>	-Training set error	4.33	5.89	5.10	4.25	4.22
	-Testing set error	4.61	6.16	5.28	4.57	4.52
	-% outliers	26.13	25.71	25.11	25.14	25.12
<i>construction site</i>	-Training set error	12.43	11.24	11.36	7.06	5.16
	-Testing set error	12.88	11.87	11.56	7.43	5.88
	-% outliers	11.28	11.24	10.79	11.18	10.38
<i>train</i>	-Training set error	14.76	13.38	9.16	6.33	5.24
	-Testing set error	15.16	13.52	9.84	6.83	6.06
	-% outliers	18.17	21.01	10.61	11.83	11.11
<i>garden</i>	-Training set error	9.06	8.76	8.98	6.36	5.19
	-Testing set error	9.12	9.01	9.47	7.06	5.31
	-% outliers	14.03	16.01	13.20	13.54	13.15
<i>carpark</i> (from [23])	-Training set error	4.77	4.36	4.19	3.60	1.38
	-Testing set error	4.85	5.67	4.05	3.86	1.67
	-% outliers	9.50	9.32	9.01	9.28	8.04
<i>apartments</i> (from [23])	-Training set error	10.23	9.06	9.84	6.86	6.23
	-Testing set error	10.48	9.76	10.12	7.02	6.40
	-% outliers	4.16	3.10	3.69	3.27	2.83
<i>chess/girl</i> (from [51])	-Training set error	7.92	10.72	21.28	9.45	2.96
	-Testing set error	8.01	12.38	20.78	9.77	4.21
	-% outliers	23.35	22.87	22.98	23.44	21.80
<i>rooftops</i> (from [51])	-Training set error	2.90	4.80	3.96	3.16	1.92
	-Testing set error	3.48	4.95	4.11	3.45	2.82
	-% outliers	8.66	10.48	10.17	8.24	8.44
<i>couch</i> (from [51])	-Training set error	11.46	10.57	12.04	5.75	5.66
	-Testing set error	11.84	10.86	12.93	5.92	5.68
	-% outliers	39.10	38.80	37.20	39.56	36.68

Table 6.1: Average RMSE (in pixels) and % outliers over 20 repetitions for 5 methods on 11 image pairs. See Figs. 6.1 to 6.11 to view the qualitative stitching results.

point cloud, the relative rotation between the cameras was fixed at 30° , and varied the *distance* between the camera centers along a fixed direction. As before, the point matches of a point cloud are partitioned into a training and testing set.

To generate different data instances, 1500 points were randomly sampled from each point cloud. Fig. 6.12 shows the average (over 50 repetitions) training and testing RMSE plotted against camera distance (% outlier measure [52] cannot be used here since there are no image pixels). Expectedly, all methods deteriorate with the increase in camera distance. Note that the errors of SVA and CPW do not diminish as the translation tends to zero. For SVA, this is due to its affine

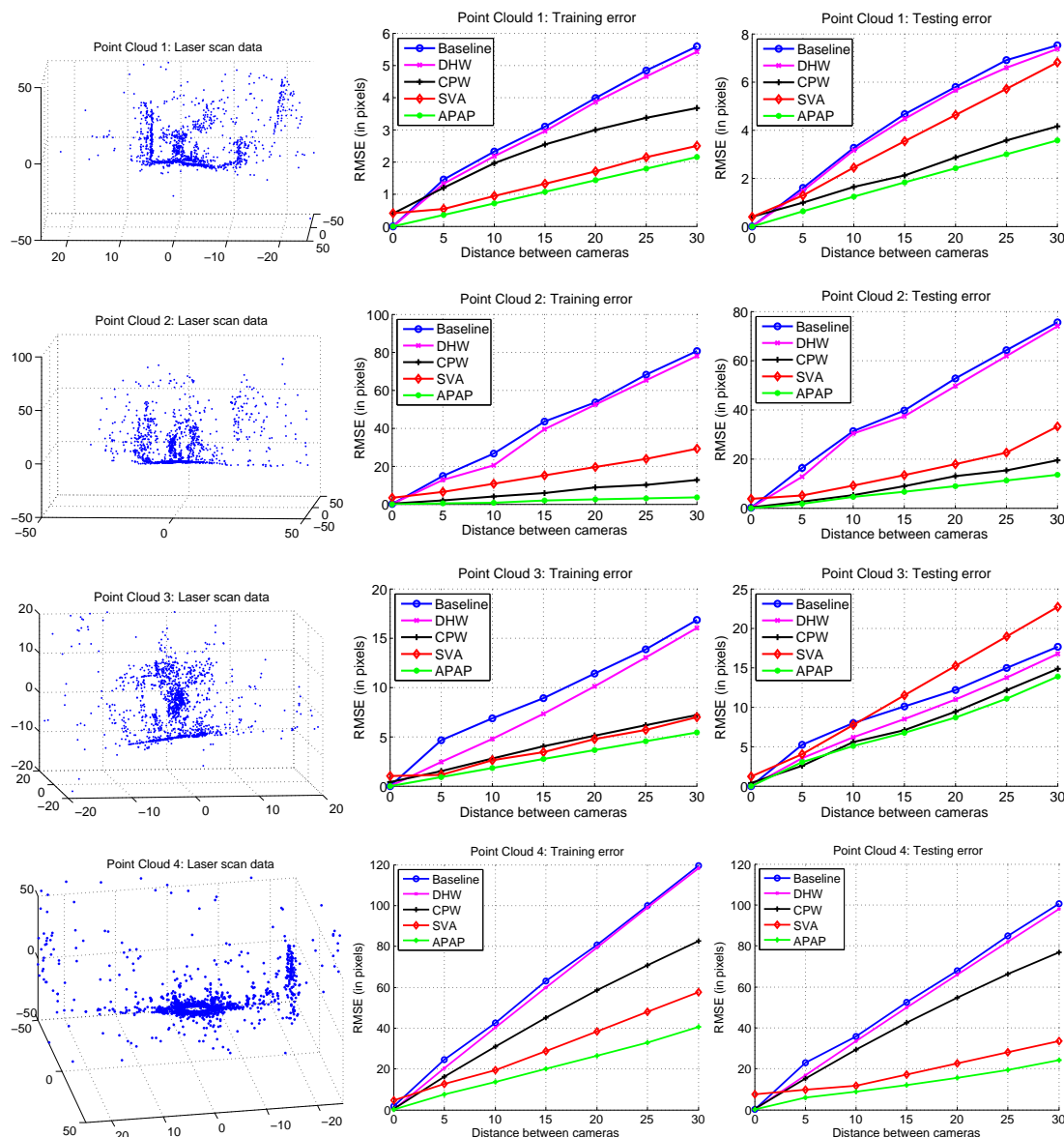


Figure 6.12: Point cloud (left) and average RMSE on the training set (middle) and the testing set (right) as a function of inter-camera translational distance.

regularisation (in other words, the affine model is the incorrect model, even with no camera translations). For CPW, this indicates that its rigidity preserving distortions may sometimes overly perturb the pre-warping by the homography. In contrast, APAP warps reduce to a global homography as the camera centres coincide, and provides overall the lowest error.

6.3 Comparisons with Bundle Adjustment

Here, the novel APAP bundle adjustment scheme (i.e., Bundled Moving DLT) is compared against the tools Autostitch [12] and Photosynth. Both of these solutions make use of bundle adjustment in order to optimise the relative rotations of the set of overlapping images, which in turn give rise to a set of aligning homographies. In this section, two types of comparisons are performed: stitching of panoramas without post-processing and stitching of panoramas with post-processing.

6.3.1 Stitching Full Panoramas without post-processing

This section compares the novel APAP bundle adjustment scheme against Autostitch [12] in simultaneously refining multiple alignment functions. Again, to directly compare alignment accuracy, any advanced compositing techniques are avoided, and the results are obtained by simply blending the aligned images with intensity averaging (the commercial version of Autostitch: “AutoPano”, allows post-processing to be switched off). Since Autostitch prewarps the images onto a cylindrical surface, the same prewarping was also performed for APAP.

Figs. 6.13, 6.14 and 6.15 show the alignment results respectively on the *construction site*, *garden* and *train* image sets (these figures appear in pages 104, 105 and 106, respectively). For Bundled Moving DLT, these panoramas were generated in 7, 5 and 7 minutes respectively. For Autostitch the panorama generation time is < 15 seconds.

The images correspond to views that differ by more than pure rotation, as one would expect from a typical tourist’s photo collection. The Autostitch results exhibit obvious misalignments; these are highlighted with red circles in the figures. Fundamentally, this is due to being restricted to using homographies for alignment. In contrast, the new Bundled Moving DLT method produces much more accurate alignments that maintains a geometrically plausible overall result.

However, both methods (without photometric post-processing) cannot handle moving objects, which give rise to motion parallax in the mosaic. This is evident in the *train* scene (Fig. 6.15), where there are many walking pedestrians. Nonetheless, the APAP method is able to handle the static components of the scene much better than Autostitch.

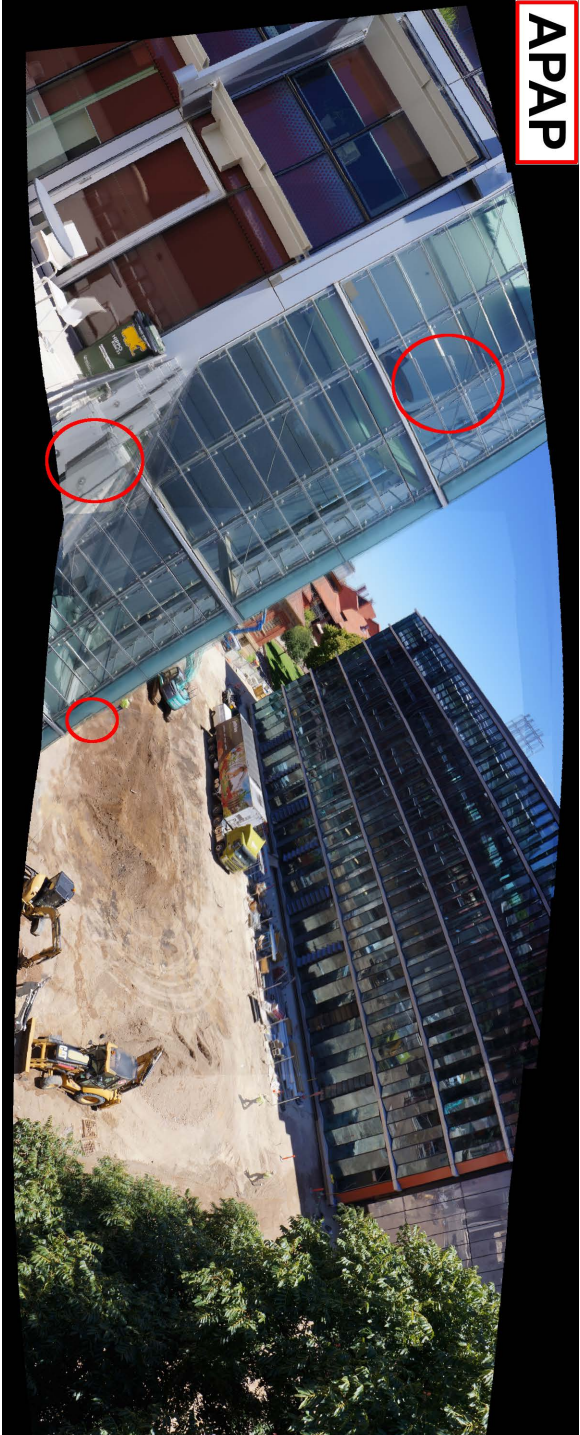
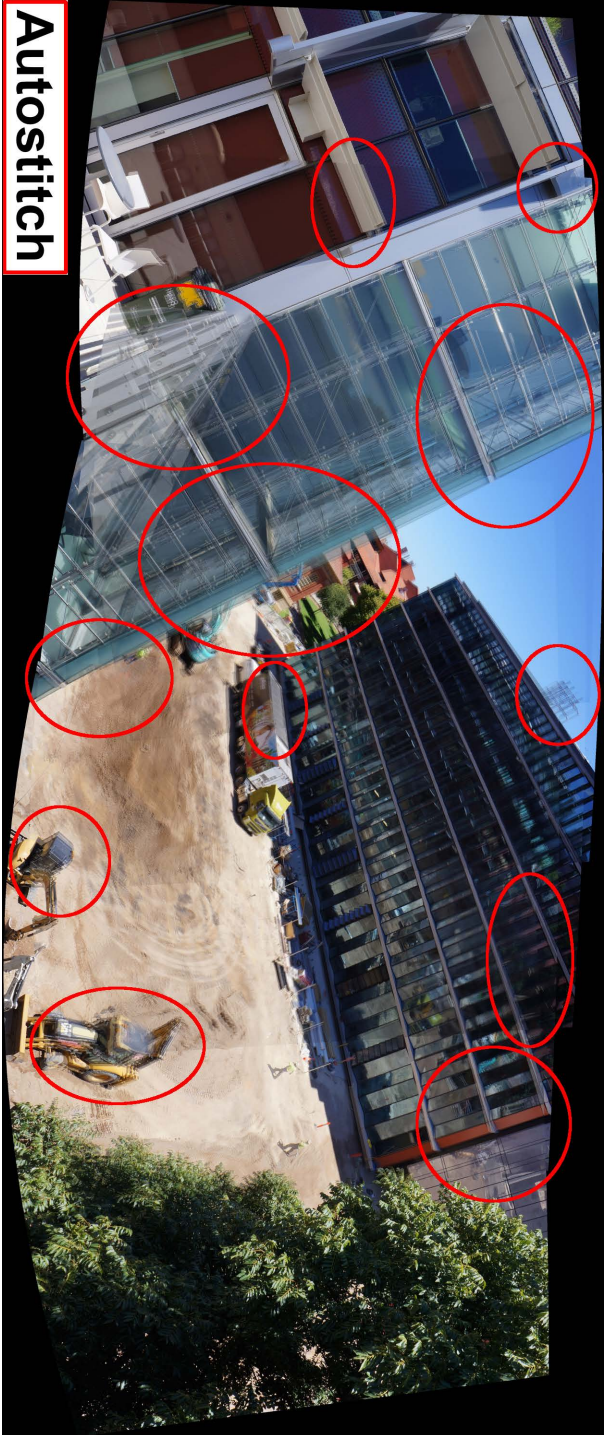


Figure 6.13: Panorama results without post-processing on the *construction site* image set (best viewed on screen). Red circles highlight errors. Note that photometric post-processing was *not* applied on Autostitch and APAP.



Figure 6.14: Panorama results without post-processing on the *garden* image set (best viewed on screen). Red circles highlight alignment errors. Note that photometric post-processing was *not* applied on Autostitch and APAP.

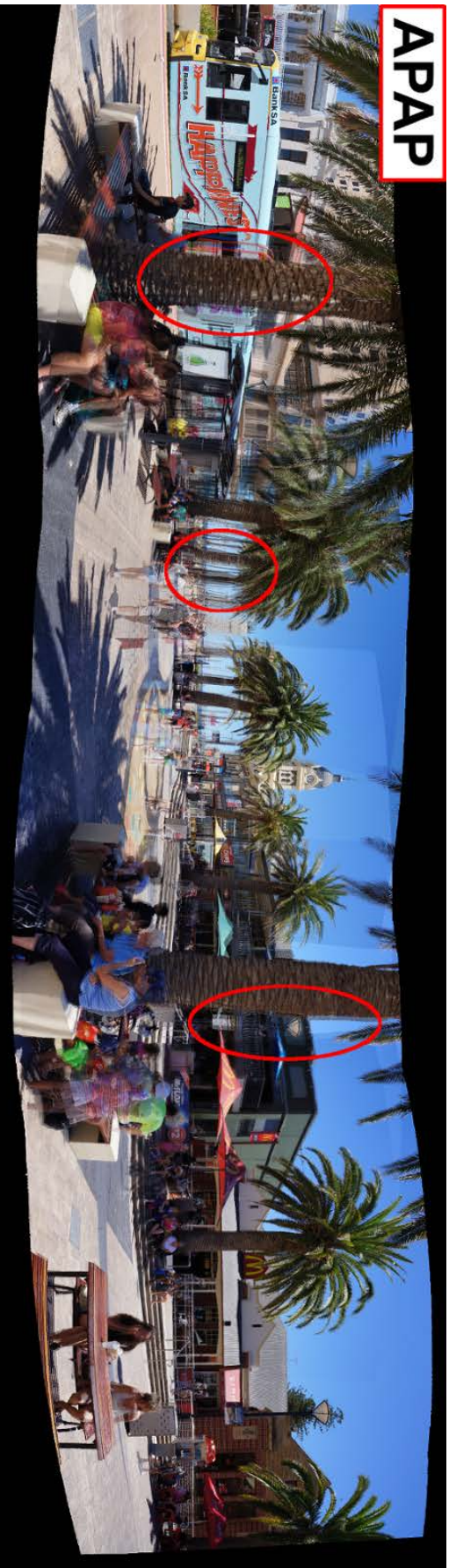


Figure 6.15: Panorama results without post-processing on the *train* image set (best viewed on screen). Red circles highlight errors. Note that photometric post-processing was *not* applied on Autostitch and APAP.

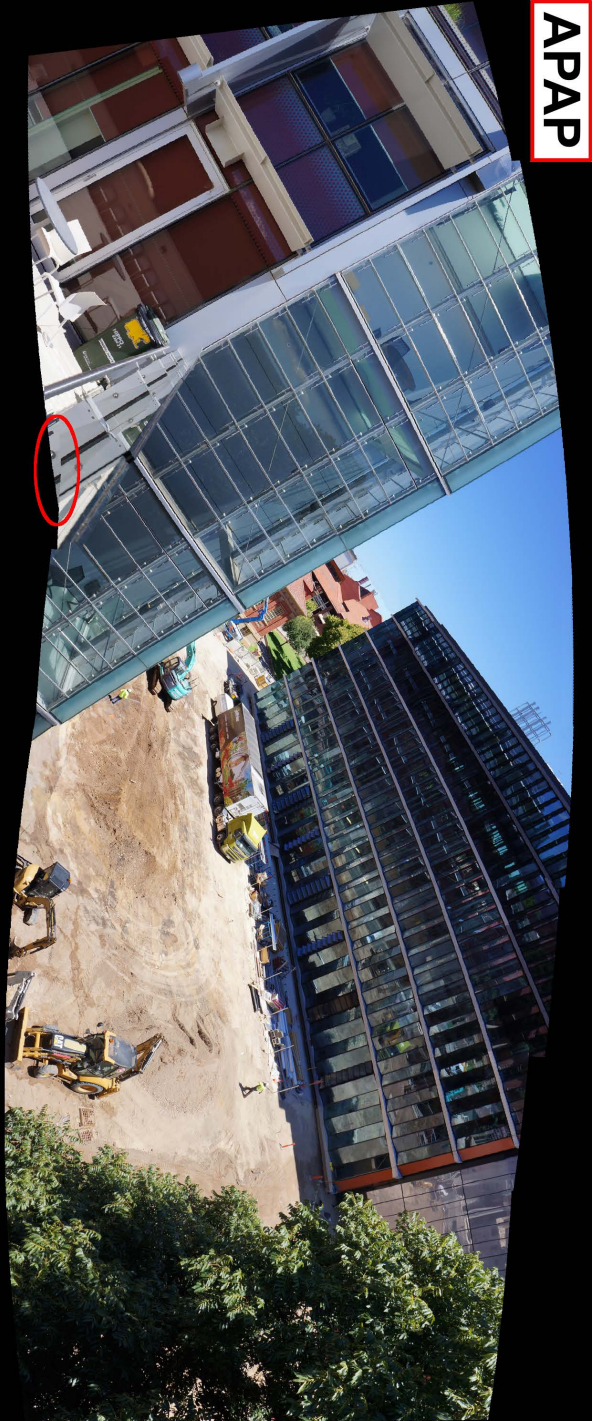
6.3.2 Stitching Full Panoramas with post-processing

As mentioned throughout this work, the underpinning premise of this thesis is that accurate image alignments impose much lower expectations or requirements on the effectiveness of de-ghosting and photometric post-processing methods. Since such post-processing techniques are imperfect and may not work all the time [39], it is vital to minimise errors in the alignment step. Here, this point is illustrated by post-processing the stitched images obtained by the as-projective-as-possible warps, which provides the most accurate alignment amongst the compared techniques. These experiments focus on stitching large panoramas from multiple images, since stitching multiple images naturally presents many opportunities for alignment errors to surface.

In these experiments, full panoramas are generated by stitching multiple images onto a canvas using Bundled Moving DLT (Chapter 5). After the warps are generated, each image is warped onto a canvas. After each image is warped onto the canvas, seam cutting and feathering blending techniques are applied in order to composite the pixels. This allows a better demonstration of the accuracy of the proposed image alignment since any misalignment errors will be propagated and amplified. These post-processed panoramas obtained by Bundled Moving DLT are compared against Autostitch and Photosynth with photometric post-processing enabled. Figs. 6.16, 6.17 and 6.18 present results on (respectively) the *construction site*, *garden* and *train* image sets (pages 108, 110 and 112).

It is evident that significant artifacts remain in the results of Autostitch. The results from Photosynth show signs of the usage of seam cutting-like techniques and sophisticated pixel blending methods. However noticeable artifacts can still be observed, since the post-processing failed to conceal the misalignments. Comparatively, the results obtained by Bundled Moving DLT present less obvious alignment mistakes and artifacts. In particular, in *train* the motion parallax errors have been dealt with by seam cutting after Moving DLT, without introducing noticeable alignment errors in the other parts of the scene.

Notwithstanding the potentially bad errors from using basic homography alignment, the results of Photosynth show the remarkable ability of post-processing methods to reduce or conceal much of the misalignment artifacts. The practical contribution of the methods proposed in this thesis is, therefore, to allow the remaining errors to be eliminated thoroughly via improved image alignment.



APAP

Figure 6.16: (Two page figure) Panorama results with post-processing on the *construction site* image set. Red circles highlight errors.

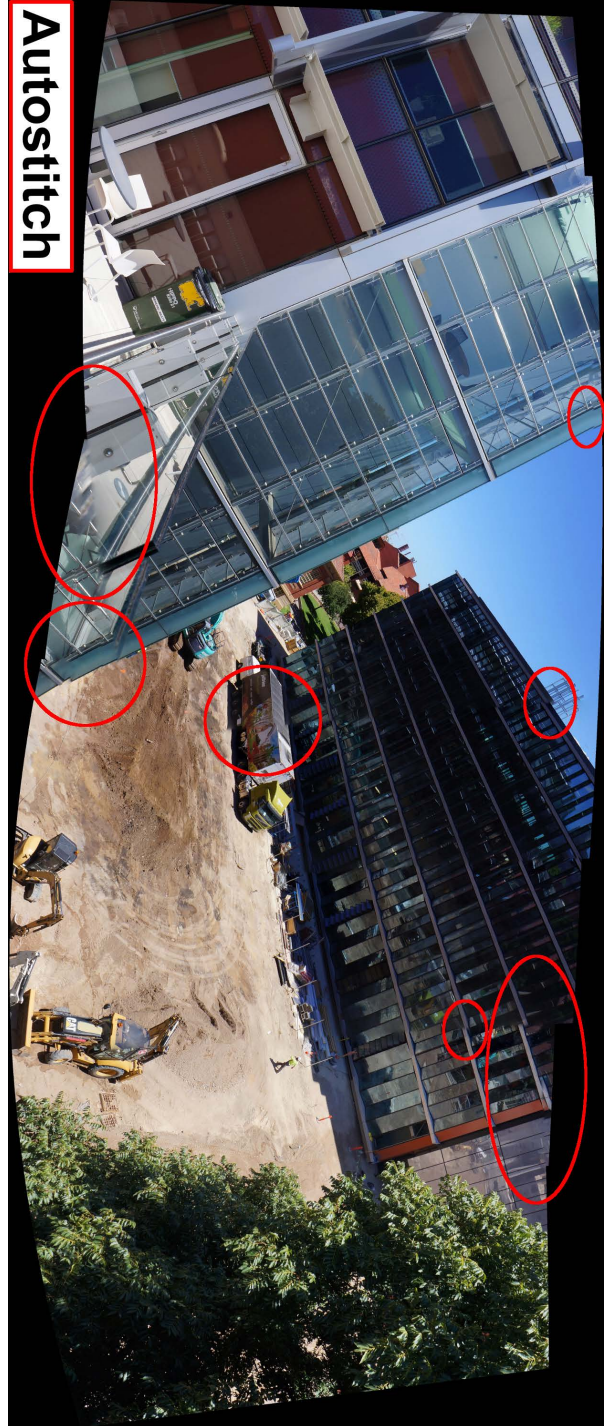




Figure 6.17: (Two page figure) Panorama results with post-processing on the *garden* image set. Red circles highlight errors.

Autostitch



Photosynth



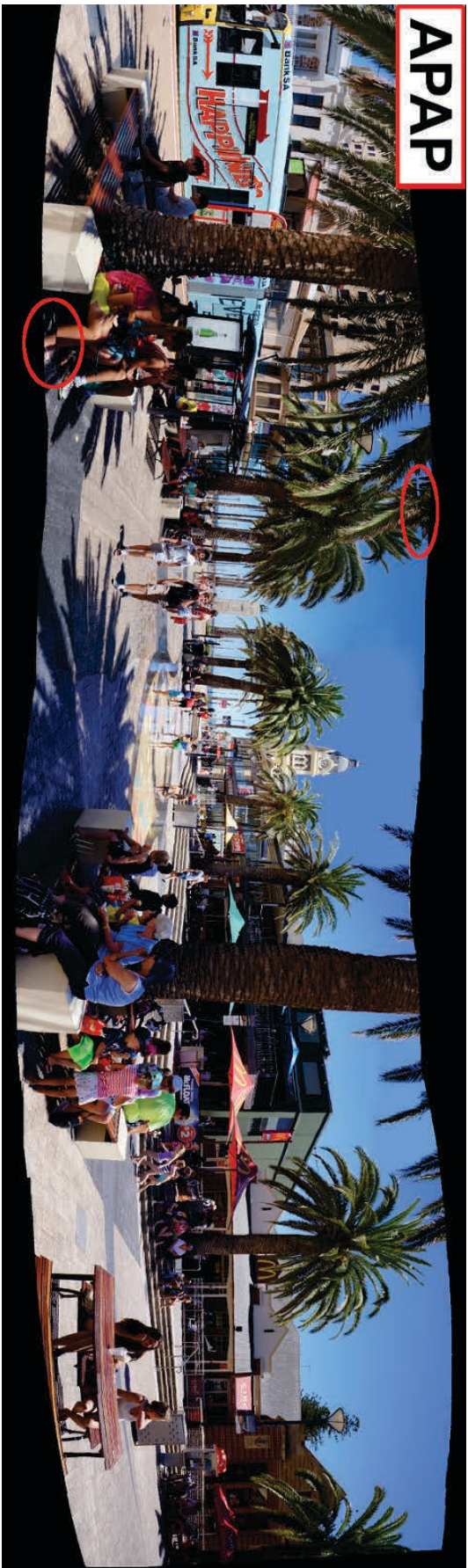
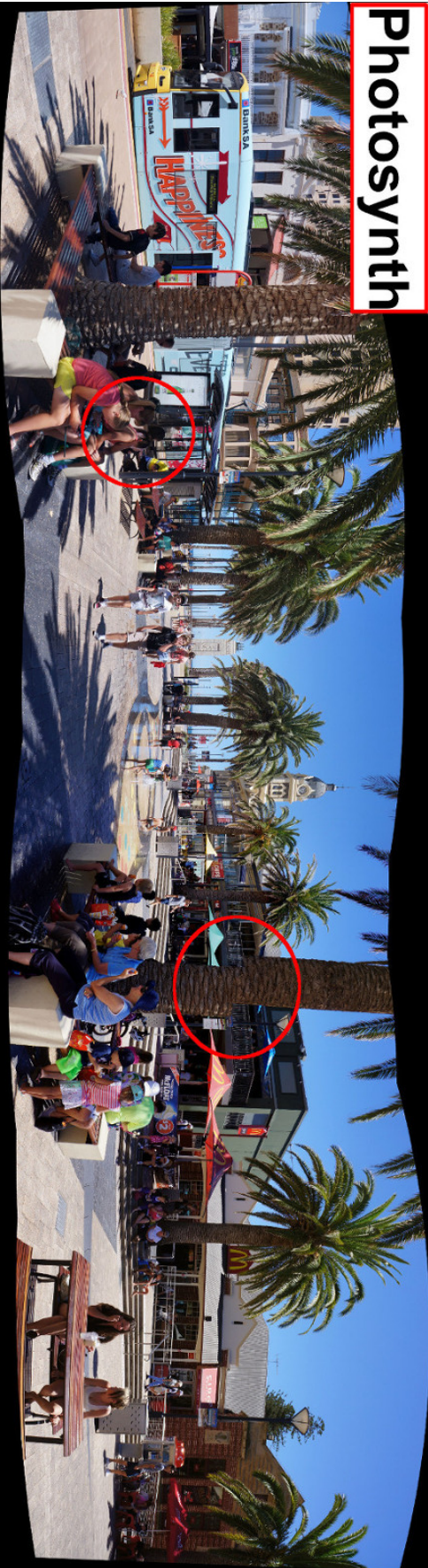


Figure 6.18: (Two page figure) Panorama results with post-processing on the *train* image set. Red circles highlight errors.

Autostitch



Photosynth



6.4 Summary

This Chapter presented several experiments and results that allowed to compare the proposed Moving DLT and its APAP warps, against other recent methods for flexible warp estimation and against commercial solutions for panorama creation. It is important to note that this thesis is the first work that manages to compare such a “large” number of recent solutions for image stitching with different types of (real and simulated) evaluations. In most of these evaluations Moving DLT was able to achieve results with less visible alignments mistakes, less alignment errors and, most importantly, less visible artifacts.

In fact, some of the raw alignment results obtained through Moving DLT, resemble the results obtained with tools like Photosynth which *includes post-processing* (see for example Figs. 6.1, 6.2, 6.6 and 6.9). Such impressive results support the underpinning assumption behind this work, that better approaches for image stitching should be oriented towards the generation of much better image alignment functions since they impose far less constraints on further post-processing or compositing techniques.

The purpose of the next Chapter is to conclude this thesis and to propose future lines of work. Since Moving DLT is a method for the generation of non-rigid projective warps (which is in this work were applied to the task of image stitching), the following Chapter also presents other interesting applications where Moving DLT can be further applied.

Chapter 7

Conclusions

Over the years, two dimensional projective transformations have become one of the standard tools for performing tasks that require some form of image alignment. These task include (but are not limited to) 3D reconstruction, augmented reality, image-based rendering and, of course, image stitching.

The main argument in this thesis is that even though projective transformations are very popular and have helped accomplishing several successful tasks in a wide range of fields, they are very limited models. What is surprising is the fact that, even though these limitations have been known for several years, basic projective warps are still widely used. However, such popularity may have stemmed from the fact that other approaches are much more complicated, slow or computationally expensive. Thus, the majority of the applications involving projective estimations, prefer to fix or conceal the mistakes produced by homographic alignments in subsequent stages of their pipelines or processes.

In order to solve the main limitations of a basic projective warp, this thesis proposed *Moving DLT*. Moving DLT is the first method for the generation of smoothly varying projective warps. These novel warps, which throughout this text have been called, *as-projective-as-possible* warps are the result of the successful integration of the Direct Linear Transformation into the framework of Moving Least Squares.

Moving DLT and its as-projective-as-possible (APAP) warps aim at accounting for the parallax and scene depth discontinuities in stitching imagery (which cause the basic homographic warp to “break”). But what is more important is the fact that Moving DLT manages to achieve this goal without explicitly modelling the camera motions, the 3D structure of the scene, the camera

matrices or the depth information of the image pixels. All of these characteristics make the proposed method a very versatile approach for flexible projective warp estimation.

Another important advantage of Moving DLT is the fact that the method is a very simple, computationally fast and very effective approach for smoothly varying homography generation. These are characteristics that no other recent solution for non-rigid image stitching has offered (combined) before.

This thesis also introduced Bundled Moving DLT, which is a novel bundle adjustment formulation that allows the refinement of multiple as-projective-as-possible warps (in the same way that the “classical” bundle adjustment allows the refinement of multiple projective warps). This unique formulation allows the generation of long panoramas while avoiding the error propagation of the incremental or pairwise stitching methods.

Combined with commonly used post-processing methods, the image alignments of Bundled Moving DLT are able to generate state-of-the-art stitching results with far less visible artifacts and no visible image distortions or deformations.

Besides image stitching, there is a wide and very exciting range of applications where Moving DLT and its as-projective-as-possible warps can be used. Some of these applications, along with possible avenues for future work, are described next.

7.1 Future Work

7.1.1 Non-Rigid Image Registration

The most straightforward task where Moving DLT can be applied is Non-Rigid Image Registration [24, 26, 83] (NRIR). In NRIR the goal is to align a *template* image with an *input* or *support* image, which (usually) contains the template image undergoing a non-rigid deformation (Fig. 7.1 illustrates).

NRIR can be seen as a special case of image stitching where one of the images completely overlaps or subsumes the other. This type of problems usually involve non-linear formulations solved by means of Gauss-Newton algorithms or Thin-Plate-Splines [8].

In order to solve this problem, the Moving DLT algorithm from Alg. 3 can be applied to the template and input images. But, for the NRIR case, instead of stitching the images, a mesh overlaid on top of the template image is warped onto the input image, thus, describing the

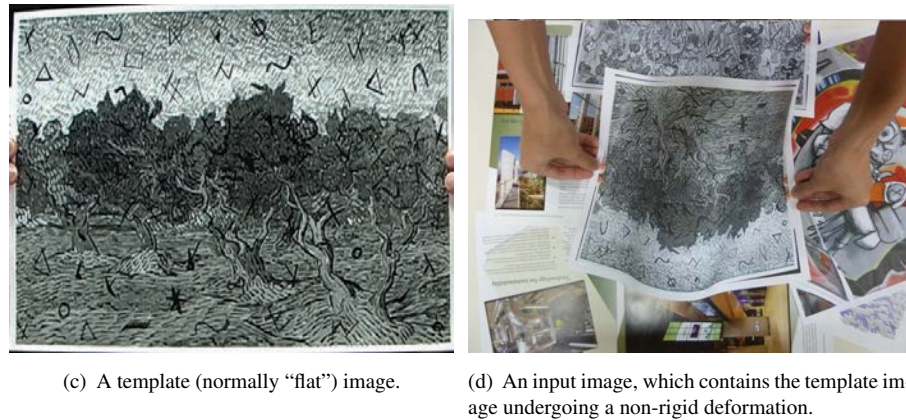


Figure 7.1: Typical input data for Non-Rigid Image Registration tasks.

deformation of the template image. Fig. 7.2 shows examples of non-rigid image registration results obtained through Moving DLT.

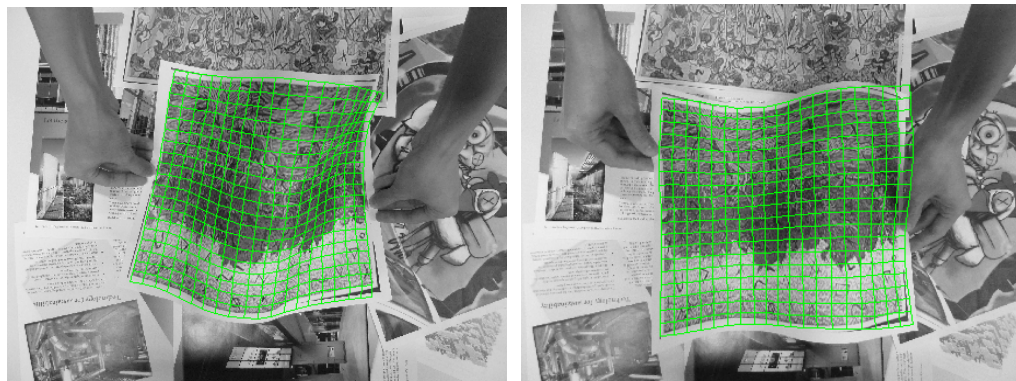


Figure 7.2: Examples of Non-Rigid Image Registration results obtained with Moving DLT.

One of the possible lines of future work is to analyse the advantages of projective regularisations for the task of NRIR. However, a much more interesting possibility is to develop a solution for giving Moving DLT the ability to register not only smooth but also "sharp" image foldings and discontinuities, as well as self occlusions [68] in the input images. In fact, giving Moving Least Squares methods the ability to approximate piecewise-smooth functions has become an active area of research in the computer graphics and computer vision communities in recent years. Some of the current solutions make use of two-stage processes where discontinuities are first identified (by means of e.g., robust methods) from the input data and then Moving Least Squares is used for approximating each one of the remaining, continuous "pieces" [61, 90].

Other methods make use of multiple weighting functions in order to incorporate additional information (besides the distance between two points \mathbf{x} and \mathbf{x}_i) into the Moving Least Squares function approximation process. For example, [43] incorporates a second weighting function

that also measures the distance between pixel colours, i.e., if a point \mathbf{x}_i has a colour that is similar to that of the point of interest \mathbf{x} , the weight of \mathbf{x}_i will be higher than that of a point \mathbf{x}_j , which has a colour that is different from that of \mathbf{x} . Instead of the colour information, [21] developed an iterative, robust Moving Least Squares framework for 3D reconstruction, which incorporates the difference in the orientation of the normals of the 3D points that are used as input for the reconstruction process. By making use of these normals, this iterative framework is able to automatically distinguish between points that belong to different surfaces, thus, giving the method the ability to identify and reconstruct discontinuities and sharp foldings from the sparse input data.

Instead of making use of multi-stage processes or multiple weighting functions (based on the work of Lancaster and Salkauskas [45]), Liu and Shi [54] propose a MLS approach that makes use of the image gradient information in order to alter the nodal point distribution at the influence domain around a particular point of interest \mathbf{x} . By altering the weight distribution around each point of interest, the proposed approach is able to detect “singularities” in the input data; this gives the method the ability to automatically alternate between performing smooth function approximations and interpolations (for discontinuity detection and modelling).

Some of the previous techniques can be incorporated into Moving DLT in order to generate piecewise-smooth, NRIR functions. For example, multiple weighting functions can be easily incorporated into Moving DLT or a pixel labelling method, e.g., graph-cut [9], can be first applied to the images in order to segment the different regions of the scene, and then Moving DLT can be applied to each independent region (a similar pixel-labelling approach was followed by [22] for the task of scene reconstruction). However, multi-stage processes can be sub-optimal, pixel-labelling processes can be overkill or can overcomplicate the task of “simple” image registration and *ad-hoc* solutions like including multiple weighting functions based on e.g., colour features are hard to generalise. Thus, analysing the feasibility and the advantages of more fundamental approaches (like the approaches presented in [45, 54]) for discontinuity preserving Moving DLT methods, is an interesting and challenging line of possible future work.

7.1.2 Non-Rigid Structure from Motion

Non-Rigid Structure from Motion (NRSfM) [10] is the extension of the classical Structure from Motion approaches for 3D reconstruction, to the non-rigid case. This process usually requires a sequence of images that contain a (moving) non-rigid object (e.g., a sequence of images of a waving flag, a moving face or a beating heart), and the goal is to reconstruct or estimate the 3D point positions \mathbf{p} of this object in each one of the frames of the sequence.

One possibility for solving the NRSfM problem is to make use of piecewise homographies [87]. By making use of methods like [98] it is possible to estimate the values of the rotation matrices (\mathbf{R} and \mathbf{R}'), translation vectors (\mathbf{t} and \mathbf{t}'), 3D plane normal \mathbf{n} and plane depth c of the homography matrix equation, presented in (2.13):

$$\mathbf{H} = \mathbf{K}'\mathbf{R}'[\mathbf{I} - (\mathbf{n}^T\mathbf{t} + c)^{-1}(\mathbf{t} - \mathbf{t}')\mathbf{n}^T]\mathbf{R}^{-1}\mathbf{K}^{-1},$$

where \mathbf{I} is a 3×3 identity matrix.

The calibration matrices \mathbf{K} and \mathbf{K}' can be estimated through other methods, e.g., [97], [31, Pp. 211–212] and [85] (although it is a common practice to make them equal to the identity matrix $\mathbf{K} = \mathbf{K}' = \mathbf{I}$). Also, since the homography matrix \mathbf{H} warps points from I to I' (i.e., image I' is not warped), $\mathbf{R}' = \mathbf{I}$ and $\mathbf{t}' = [0 \ 0 \ 0]^T$. Thus, the previous process produces the following camera matrices:

$$\mathbf{P} = \mathbf{K}[\mathbf{R} \ -\mathbf{R}\mathbf{t}] \quad \text{and} \quad \mathbf{P}' = \mathbf{K}'[\mathbf{R}' \ -\mathbf{R}'\mathbf{t}'] = \mathbf{K}' \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \quad (7.1)$$

Appendix B explains how, by means of the camera matrices \mathbf{P} and \mathbf{P}' , the 3D positions \mathbf{p} of the points from the images I and I' can be recovered. However, since in this case, the camera matrix is recovered by means of 2D homography, the obtained 3D reconstruction will correspond to a 3D plane.

In order to reconstruct a non-rigid surface, Varol *et al.* [87] estimate a set of inlier keypoint matches $\{\mathbf{x}, \mathbf{x}'\}_{i=1}^N$ between the images I and I' , afterwards, they *manually* split the input image I into M patches and, for each patch, they obtain a homography \mathbf{H}_j that aligns the patch with the support image I' . Each one of these homographies \mathbf{H}_j is then decomposed into its corresponding \mathbf{R}_j , \mathbf{t}_j and \mathbf{n}_j elements, which allow them to generate the corresponding camera matrix \mathbf{P}_j . The idea behind such work is to perform a piecewise reconstruction where, through each \mathbf{P}_j , a small 3D planar patch of the non-rigid surface is reconstructed.

The main problem with this type of approaches is to make sure that all of the patches are reconstructed up to the same global scale, i.e., to ensure that the piecewise reconstructions are properly aligned. In order to achieve such globally consistent solutions, there must be smoothness between the estimated homographies \mathbf{H}_j . The work in [87] achieves this smoothness by manually splitting the input image while making sure that neighbouring patches share a number of the keypoint matches.

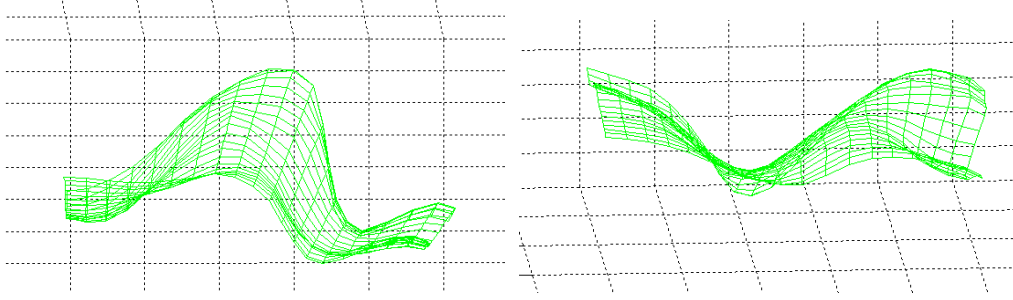


Figure 7.3: Non-Rigid Structure from Motion results obtained with Moving DLT. The results correspond to the images shown in Fig. 7.2.

Moving DLT is inherently able to generate smoothly varying homographies. Thus, in principle, decomposing each homography \mathbf{H}_j generated through Moving DLT and then applying the approach of [87] for reconstructing the images, generates globally consistent 3D reconstructions. In fact, Fig. 7.3 shows two different 3D reconstruction results obtained with this approach. More specifically, in order to generate the previous 3D reconstruction results through Moving DLT, the following cost function is minimised

$$E(\mathbf{p}, \mathbf{c}) = \sum_{i=1}^N \sum_{j=1}^M \|w_{ij}(\mathbf{B}_{ij}\mathbf{p}_i - \mathbf{b}_{ij})\|^2 + \|w_{ij}(\mathbf{n}_j^T \mathbf{p}_j + c_j)\|^2, \quad (7.2)$$

where $\mathbf{p} = \{\mathbf{p}_1, \dots, \mathbf{p}_N\}$ is the 3D point cloud to be estimated, $\mathbf{c} = \{c_1, \dots, c_M\}$ contains the depths of all of the 3D planes (or “patches”) that need to be re-scaled in order to obtain a globally consistent solution, \mathbf{b}_{ij} is defined as

$$\mathbf{b}_{ij} = \begin{bmatrix} \mathbf{P}'_{14} - \mathbf{x}'_i \mathbf{P}'_{34} \\ \mathbf{P}'_{24} - \mathbf{x}'_i \mathbf{P}'_{34} \\ \mathbf{P}^j_{14} - \mathbf{x}_i \mathbf{P}^j_{34} \\ \mathbf{P}^j_{24} - \mathbf{x}_i \mathbf{P}^j_{34} \end{bmatrix}, \quad (7.3)$$

and

$$\mathbf{B}_{ij} = \begin{bmatrix} \mathbf{P}'_{11} - \mathbf{x}'_i \mathbf{P}'_{31} & \mathbf{P}'_{12} - \mathbf{x}'_i \mathbf{P}'_{32} & \mathbf{P}'_{13} - \mathbf{x}'_i \mathbf{P}'_{33} \\ \mathbf{P}'_{21} - \mathbf{x}'_i \mathbf{P}'_{31} & \mathbf{P}'_{22} - \mathbf{x}'_i \mathbf{P}'_{32} & \mathbf{P}'_{23} - \mathbf{x}'_i \mathbf{P}'_{33} \\ \mathbf{P}^j_{11} - \mathbf{x}_i \mathbf{P}^j_{31} & \mathbf{P}^j_{12} - \mathbf{x}_i \mathbf{P}^j_{32} & \mathbf{P}^j_{13} - \mathbf{x}_i \mathbf{P}^j_{33} \\ \mathbf{P}^j_{21} - \mathbf{x}_i \mathbf{P}^j_{31} & \mathbf{P}^j_{22} - \mathbf{x}_i \mathbf{P}^j_{32} & \mathbf{P}^j_{23} - \mathbf{x}_i \mathbf{P}^j_{33} \end{bmatrix}, \quad (7.4)$$

where \mathbf{P}^k_{ij} is the (i, j) -th entry of the k -th projection matrix. \mathbf{P}' is the projection matrix for the input image defined in (7.1) (the matrices \mathbf{B} and \mathbf{b} are defined as in [87]).

Even though, the previous Moving DLT approach is able to obtain consistent 3D reconstructions,

the current literature on NRSfM offers a plethora of methods for reconstructing waving clothes and bending pieces of paper from a set of sparse (SIFT) matches. It was until more recently that the work of Garg *et al.* [25] provided an excellent example of the type of reconstruction results that these approaches should aim at obtaining in future iterations (Fig. 7.4 offers an example of the 3D reconstruction results from Garg *et al.*). Unfortunately, such type of methods

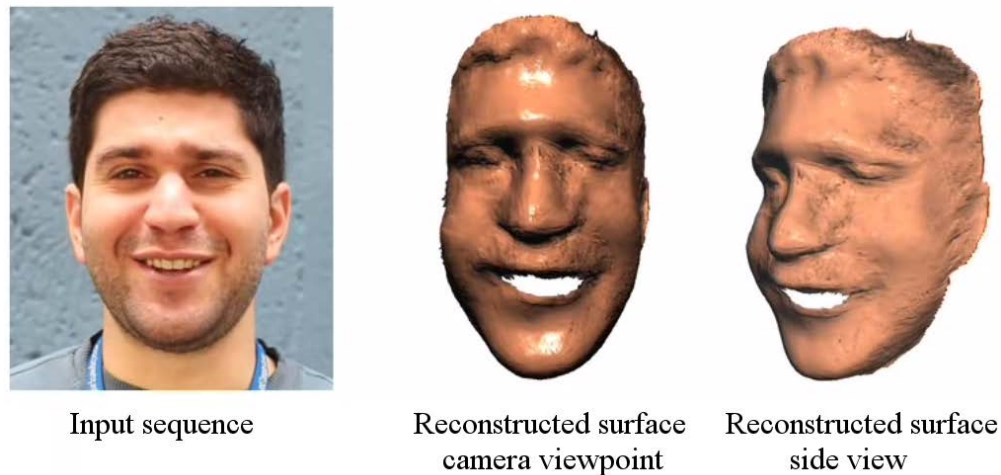


Figure 7.4: An example of the dense 3D reconstruction results of [25]. This image is taken from [25].

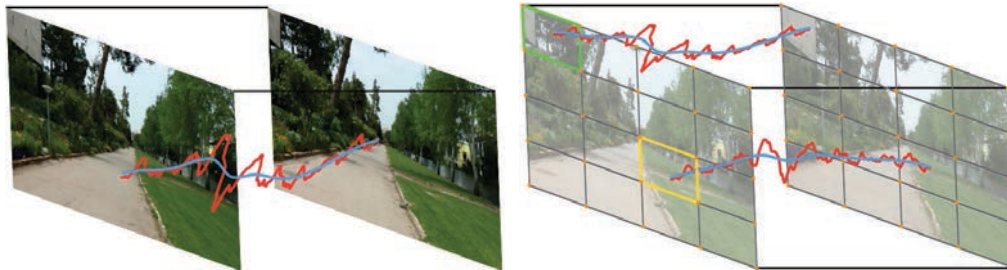
are, in general, very time consuming and computationally demanding. Thus, a very interesting challenge in this area is to make use of the NRSfM scheme of Moving DLT in order to perform dense reconstructions of general scenes, i.e., including rigid and non-rigid objects, but with lower computational costs and lower running times.

It is also worth mentioning that, instead of making use of homographies in order to perform the 3D piecewise planar reconstructions, other works, e.g., [69] make use of the *piecewise Quadratic Reconstruction model* of Fayad *et al.* [19]. Thus, another interesting possibility for future work is to make use of the Moving Least Squares and Moving DLT frameworks and apply them to the piecewise quadratic reconstruction model of Fayad *et al.* [19] for NRSfM. The challenge of such task is to ensure that the resulting (“moving”) framework is also as simple, fast and effective as MLS and Moving DLT.

7.1.3 Video Stabilisation

During the literature review of Chapter 3, the Content Preserving Warps (CPW) [53] method for video stabilisation was briefly described. The goal of video stabilisation is to smooth the camera motion or *path* of a shaky video. In order to do this the initial “shaky” camera path needs to be estimated and then smoothed (the red and blue lines in Fig. 7.5(a) show an example of a shaky

path and its corresponding smooth camera path). After the shaky and the smooth camera paths have been estimated, the frames from the shaky path are projected or warped into the smooth camera path. In other words, video stabilisation is a “stitching problem” where the images from the shaky video are aligned with the (empty) images from the smooth video (refer to [53] for details).



(a) Traditional methods for video stabilisation (including CPW) make use of a single camera path for stabilising whole videosequences. (b) Bundled camera paths make use of multiple camera paths in order to stabilise the video.

Figure 7.5: Comparing *single* and *bundled* camera paths for video stabilisations. The figure shows the camera trajectories (visualized by the y axis translation over time) and plots the original path (in red) and the smoothed path (in blue) for both methods. This image is taken from [55].

In order to perform this alignment, CPW divides each one of the images to be stabilised into a mesh. Then, for each cell in the mesh, a similarity transformation is obtained. The purpose of these similarity transformations is to bring the input image into alignment with the stabilised path.

CPW and other similar approaches for video stabilisation assume that there is only a single global camera path. Another, more recent work [55] focuses on making use of several camera paths, one for each one of the cells in the input image (Fig. 7.5 compares both approaches). The goal of such approach is then to simultaneously stabilise all of the paths in a bundled manner. The authors call this method “Bundled Camera Paths”.

As opposed to CPW that makes use of similarity transformations, Bundled Camera Paths performs the image-to-image (or more specifically, the cell-to-cell) mappings for video stabilisation by means of homographies. However, such homographies are estimated by means of a mesh (refer to [55] for details). Fig. 7.6 depicts this process. Thus, in principle, analysing the advantages of the “mesh-less” Moving DLT method¹ for single or bundled paths for video stabilisation is also an interesting possibility for future work.

¹As previously mentioned, the proposed Moving DLT method also makes use of a mesh, but in this case the mesh is used for speed purposes only; Moving DLT is a mesh free method.

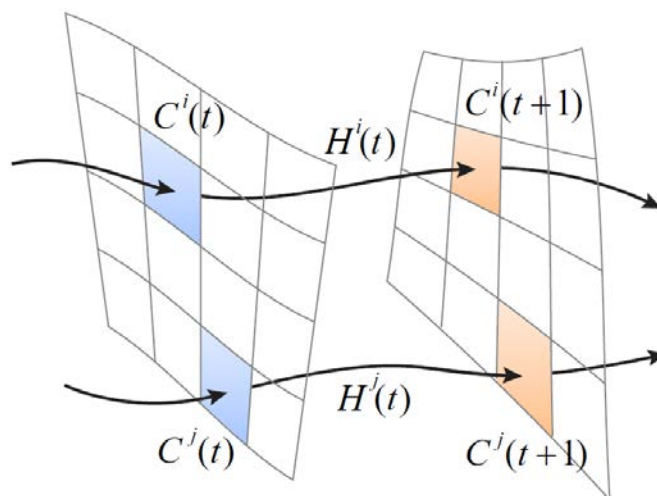


Figure 7.6: The image stabilisation process of Bundled Camera Paths. Each cell C^i in the image at time t is warped to image at time $t+1$ by means of a *stabilised* homography mapping H^i .

7.2 Drawbacks and Limitations

It is important to emphasize the fact that Moving DLT is *not* a method that aligns images obtained under arbitrary camera motions. The generated as-projective-as-possible warps will eventually “break” as the distance between the cameras increases. This is, in principle what the results from Section 6.2.4 are showing.

For image stitching under arbitrary camera motions, approaches like the pushbroom camera model (presented in Section 3.6) can be used. However, as it is already known, such methods require “dense data” or videos in order to perform. Moving DLT is suited for the cases where the images to align *deviate* from the assumptions of projective estimation. In practice, this is one of the most common scenarios when dealing with images taken by casual users.

A limitation of most of the current image stitching methods occurs when there is a lack of features in the images to align. As shown in Fig. 6.10 in the experiments and results Chapter, if there are few keypoint matches in the images to align, most of the feature based approaches will perform poorly. This is because, in those cases, there is no data to guide the warps. A possible solution to this problem is the development of *pixel based* methods that are not too heavily dependant on the quality and amount of keypoint matches. Developing a fast pixel based version of Moving DLT is another interesting area of possible future work.

Another aspect to be improved in Moving DLT could be speed. Moving DLT as been shown to be a very fast method for image stitching, however, other tasks like augmented reality or Simultaneous Localisation and Mapping, demand real time performance. Even though this work

already suggested making use of GPU's or faster programming languages like C++, a much better approach for achieving real-time performance is to find or develop more principled solutions or faster methods in order to solve the formulations of Moving DLT and Bundled Moving DLT. Developing a real-time version of Moving DLT is left as future work as well.

This thesis would like to conclude by saying that, as seen in the previous paragraphs, image stitching is only one of the possible applications where Moving DLT has proved to be very useful. However, there are several tasks where the proposed methods can still be applied. Basically, it is possible to try in each one of the applications that make use of a basic projective transformation and develop new and interesting formulations and obtain new results. However, what is more exciting about future work is the range of new possibilities and new applications that are waiting to be discovered and, hopefully, Moving DLT is the engine that propels such amazing discoveries.

Appendix A

Image Sets Used in the Experiments

Some of the images used in the experimental evaluations of this thesis were taken from different datasets used in recent stitching papers. This Appendix shows the input (unwarped) images and the corresponding reference that makes use of them.

Figs. [A.2](#), [A.3](#) and [A.4](#) show the images from the *temple*, *carpark* and *apartments* datasets. These images are used by Gao et al. in [23]. Figs. [A.5](#), [A.6](#) and [A.7](#) show the images from the *chess/girl*, *couch* and *rooftops* datasets used by Lin et al. in [51]. Finally, the images generated for this thesis, namely, the *railtracks*, *bikes*, *construction site*, *garden* and *train* datasets are shown in Figs. [A.1](#), [A.8](#), [A.9](#), [A.10](#) and [A.11](#).

A.1 Images Used in Pairwise Stitching



Figure A.1: *railtracks* image pair. Size of images: 2000×1500 pixels. Number of inliers after RANSAC is: 2753.



Figure A.2: *temple* image pair [23]. Size of images: 730×487 pixels. Number of inliers after RANSAC is: 415.



Figure A.3: *carpark* image pair [23]. Size of images: 653×490 pixels. Number of inliers after RANSAC is: 359.



Figure A.4: *apartment* image pair [23]. Size of images: 1632×1224 pixels. Number of inliers after RANSAC is: 2634.



Figure A.5: *chess/girl* image pair [51]. Size of images: 1824×1368 pixels. Number of inliers after RANSAC is: 1381.



Figure A.6: *couch* image pair [51]. Size of images: 1824×1368 pixels. Number of inliers after RANSAC is: 1329.



Figure A.7: *rooftops* image from [51]. Size of images: 320×240 pixels. Number of inliers after RANSAC is: 161.



Figure A.8: *bikes* image pair. Size of images: 1632×1224 pixels. Number of inliers after RANSAC is: 2561.

A.2 Images Used For Stitching Full Panoramas

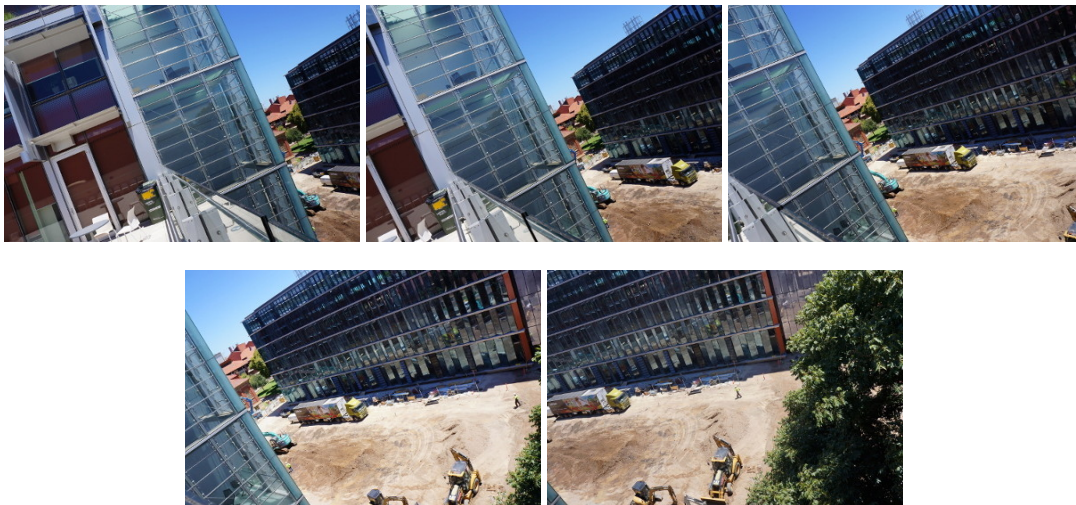


Figure A.9: *construction site* dataset. Size of images: 2000×1329 pixels. Number of inliers between the pair of images used for the pairwise experiments in Section 6.2: 5068. Number of inliers between the images used for the panorama experiments in Section 6.3: 12616.



Figure A.10: *garden* dataset. Size of images: 2000×1329 pixels. Number of inliers between the pair of images used for the pairwise experiments in Section 6.2: 4567. Number of inliers between the images used for the panorama experiments in Section 6.3: 10693.



Figure A.11: *train* dataset. Size of images: 2456×1632 pixels. Number of inliers between the pair of images used for the pairwise experiments in Section 6.2: 4231. Number of inliers between the images used for the panorama experiments in Section 6.3: 13380.

Appendix B

The Image Formation Process

Transforming the 3D world into a 2D image involves a projection process in which one dimension, corresponding to the “depth” of the world, is removed. In computer vision, this process is usually explained in terms of the central projection model (also called the pinhole camera model).

In central projection 3D points are projected by rays or lines passing through a common center of projection or focal point onto a particular 2D plane; this 2D plane is called the *image plane*. The intersection of the lines with the image plane are the “pictures” of the 3D points. Fig. B.1 depicts this process. Mathematically, this process is described with the use of projective geometry.

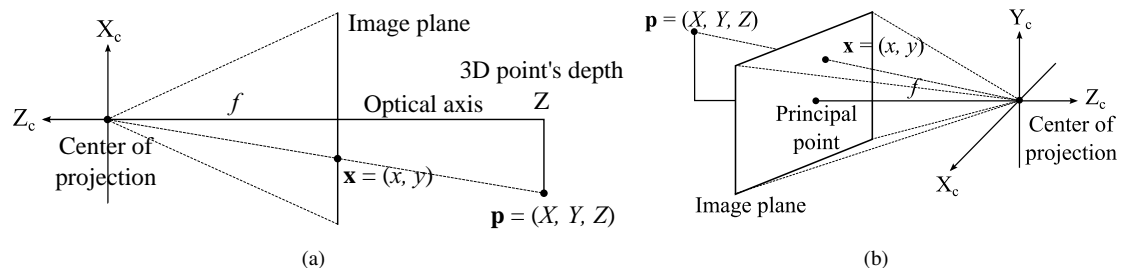


Figure B.1: Two different perspectives of the image formation process through the pinhole camera model. In the pinhole camera model, points p from the 3D world are mapped by lines passing through a common center of projection into its corresponding image point x . In this process, the “depth” of the points from the world is removed. The optical axis is the ray extending through the focal point, perpendicular to the image plane. The point where the optical axis and the image plane intersect is called the *principal point* of the image. The length between the focal point and the image plane is the *focal length* f .

In projective geometry the world is modelled in the 3D *projective space*. The 3D projective space includes points from the 3D Euclidean space \mathbb{R}^3 along with points at infinity. The 3D projective space is denoted by \mathbb{P}^3 . In the 3D projective space, points are represented in homogeneous

coordinates as the 4-tuple $\tilde{\mathbf{p}} = [X \ Y \ Z \ W]^T$. If two tuples or points differ only by scale, they are considered to be equivalent. A homogeneous point $\tilde{\mathbf{p}}$ can be converted back into an inhomogeneous or Cartesian vector \mathbf{p} by dividing it through by the last element W , i.e., $\mathbf{p} = [X/W \ Y/W \ Z/W]^T$.

A homogeneous point whose last element is $W = 0$ corresponds to a *point at infinity* (also called ideal point). Points at infinity do not have an equivalent inhomogeneous representation.

In a similar way, the model for the image is a 2D projective plane \mathbb{P}^2 with points in the 2D image plane represented in homogeneous coordinates as 3-tuples $\tilde{\mathbf{x}} = [x \ y \ z]^T$. Similar to the 3D projective space, in inhomogeneous coordinates $\mathbf{x} = [x/z \ y/z]^T$.

Central projection is a simple mapping from the 3D projective space \mathbb{P}^3 to the 2D projective plane \mathbb{P}^2 . The mapping is performed by a 3×4 matrix \mathbf{P} which is called the *camera matrix*. Thus, under central projection any (homogeneous) 3D point $\tilde{\mathbf{p}}$ from the “real world” is projected to a 2D image plane via the following relationship:

$$\tilde{\mathbf{x}} \sim \mathbf{P}\tilde{\mathbf{p}}. \quad (\text{B.1})$$

where \sim indicates equality up to scale. The camera matrix is composed by a 3×4 *external parameters* matrix and a 3×3 *internal parameters* matrix. Each one of these elements is described next.

B.1 The Camera Matrix

B.1.1 External Parameters

Imagine you take a picture with a camera; what are the elements or parameters that affect the final image you obtain? Probably, the most obvious parameters are the location and orientation of the camera, i.e., what is the camera “seeing” right now? The relative position of the 3D points in the world with respect to the position of the camera will determine where the 3D points will appear in the image. Different camera positions will produce different images.

In order to determine where a 3D point will appear in the final image, it is necessary to establish the location and orientation of this point with respect to the camera. This is usually done by taking the position of the 3D point whose coordinates are represented in the world coordinate system and assign it coordinates with respect to the camera coordinate system. In the camera coordinate system, the center of projection is denoted as the origin. Such process can be described

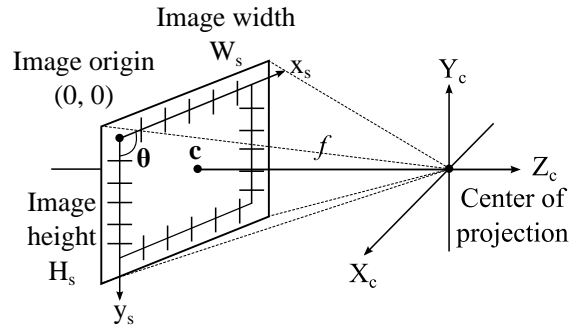


Figure B.2: Example of the simplified camera intrinsic parameters. The principal point corresponds to the centre of the image and there is no skew between the image axes. The only unknown parameter in this case is the focal length f .

in matrix notation by

$$\mathbf{p}' = \begin{bmatrix} \mathbf{R} & -\mathbf{R}\mathbf{t} \end{bmatrix} \tilde{\mathbf{p}} = \mathbf{E}\tilde{\mathbf{p}}, \quad (\text{B.2})$$

where \mathbf{p} is a 3D point and \mathbf{p}' is point \mathbf{p} with the camera focal point as the frame of reference, \mathbf{R} is a 3×3 rotation matrix and $\mathbf{t} = [t_x \ t_y \ t_z]^T$ is a 3D translation vector. The matrix $\mathbf{E} = \begin{bmatrix} \mathbf{R} & -\mathbf{R}\mathbf{t} \end{bmatrix}$ corresponds to the *external parameters* of the camera.

B.1.2 Camera Intrinsics

Besides the position of the camera there is another set of properties that determines the appearance of the final picture. Depending on the particular manufacturer, different cameras can have different fields of view, different aspect ratios and different focal lengths and all of these elements change the way the image is generated. In the image formation process it is possible to model these properties by means of the camera *intrinsics* or the camera *internal parameters*.

In projective geometry, these internal parameters are represented by a 3×3 matrix defined as

$$\mathbf{K} = \begin{bmatrix} f_x & r & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}, \quad (\text{B.3})$$

where f_x and f_y are the independent focal lengths along the x_s and y_s image axes (the aspect ratio of the image is reflected in the different scaling of these focal lengths along x_s and y_s). $r = \cos(\theta)$ encodes the skew (more specifically, the angle) between the image coordinate axes and $\mathbf{c} = [c_x \ c_y]^T$ denotes the coordinates of the principal point.

In practice, most of the recent cameras have square pixels (i.e., there is no skew), have the same aspect ratios along the x_s and y_s image axes (i.e., the focal lengths are equivalent) and assume that the location of the principal point corresponds to the centre point of the image. Thus, a simplified version of the intrinsic camera matrix can be obtained by setting $r = 0$ and $f_x = f_y = f$, which produces the commonly used intrinsic camera matrix

$$\mathbf{K} = \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix}, \quad (\text{B.4})$$

with $c_x = W_s/2$ and $c_y = H_s/2$ where W_s and H_s are, respectively, the width and height of the image (Fig. B.2 illustrates). Since, most of the times, the location of the principal point is assumed to be known, some other versions of the camera matrix have $c_x = c_y = 0$.

By putting the camera intrinsics and extrinsic parameters together it is then possible to obtain the 3×4 camera matrix \mathbf{P} as

$$\mathbf{P} = \mathbf{K} \begin{bmatrix} \mathbf{R} & -\mathbf{R}\mathbf{t} \end{bmatrix} = \mathbf{K}\mathbf{E}. \quad (\text{B.5})$$

B.2 Relating Points in Two Views through the Camera Matrix

Through the camera matrix it is possible to relate points between two (or more) views I and I' . In order to describe such process, (B.5) will be written in the following form:

$$\tilde{\mathbf{P}} = \begin{bmatrix} \mathbf{K} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \end{bmatrix} \begin{bmatrix} \mathbf{R} & -\mathbf{R}\mathbf{t} \\ \mathbf{0} & \mathbf{1} \end{bmatrix} = \tilde{\mathbf{K}}\tilde{\mathbf{E}}, \quad (\text{B.6})$$

which corresponds to a 4×4 invertible camera matrix $\tilde{\mathbf{P}}$ [76] with $\tilde{\mathbf{E}}$ being a 4×4 external parameters matrix and $\tilde{\mathbf{K}}$ being a 4×4 full rank intrinsics matrix.

Writing the camera matrix as in (B.6) allows the mapping

$$\tilde{\mathbf{x}} \sim \tilde{\mathbf{P}}\tilde{\mathbf{p}} = \tilde{\mathbf{K}}\tilde{\mathbf{E}}\tilde{\mathbf{p}}, \quad (\text{B.7})$$

of any homogeneous 3D point $\tilde{\mathbf{p}} = [X \ Y \ Z \ 1]^T$ to a particular type of homogeneous image point $\tilde{\mathbf{x}} = [x \ y \ 1 \ d]^T$, where d is the inverse depth of point \mathbf{x} [77, Pp. 49]. This depth value is necessary to reason about mappings between images of a 3D scene.

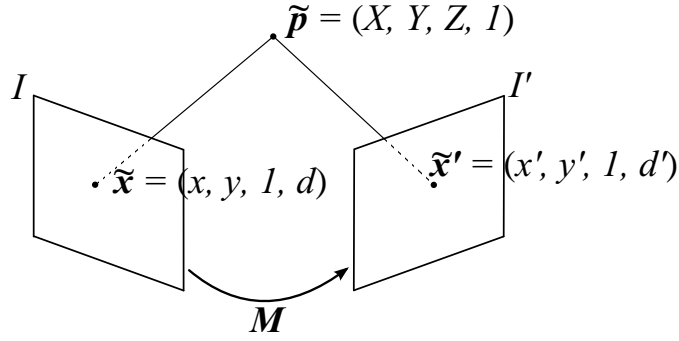


Figure B.3: Relationship between the projections or pictures of a 3D point in two different images. These relationships can be described by making use of the camera matrices from both images (which requires knowing the depth values of each pixel in the images).

Given a homogeneous 3D point $\tilde{\mathbf{p}} = [X \ Y \ Z \ 1]^T$, a 2D representation or picture of \mathbf{p} can be obtained by means of the 4×4 camera matrix $\tilde{\mathbf{P}}$ in (B.6). This can be expressed as

$$\tilde{\mathbf{x}} \sim \tilde{\mathbf{P}}\tilde{\mathbf{p}} = \tilde{\mathbf{K}}\tilde{\mathbf{E}}\tilde{\mathbf{p}}, \quad (\text{B.8})$$

where $\tilde{\mathbf{x}} = [x \ y \ 1 \ d]^T$ corresponds to the homogeneous image coordinates and depth of the 3D homogeneous point $\tilde{\mathbf{p}}$. If a second picture of \mathbf{p} is taken from a different location, which of course involves the use of a different camera matrix $\tilde{\mathbf{P}}'$, there will be two overlapping images of the same scene. Fig. B.3 illustrates this scenario.

From Eq. B.8 it is possible to observe that any 3D point \mathbf{p} can be obtained via the following equation

$$\tilde{\mathbf{p}} \sim \tilde{\mathbf{E}}^{-1}\tilde{\mathbf{K}}^{-1}\tilde{\mathbf{x}}. \quad (\text{B.9})$$

Afterwards, it is then possible to project the point $\tilde{\mathbf{p}}$ recovered from image I , into the second image I' via the equation

$$\tilde{\mathbf{x}}' = \tilde{\mathbf{K}}'\tilde{\mathbf{E}}'\tilde{\mathbf{p}} = \tilde{\mathbf{K}}'\tilde{\mathbf{E}}'\tilde{\mathbf{E}}^{-1}\tilde{\mathbf{K}}^{-1}\tilde{\mathbf{x}} = \tilde{\mathbf{P}}\tilde{\mathbf{P}}'^{-1}\tilde{\mathbf{x}} = \mathbf{M}\tilde{\mathbf{x}}, \quad (\text{B.10})$$

where \mathbf{x}' is point \mathbf{p} in image I' , $\tilde{\mathbf{K}}$, $\tilde{\mathbf{K}}'$ and $\tilde{\mathbf{E}}$, $\tilde{\mathbf{E}}'$ are the 4×4 calibration and external parameters matrices for images I and I' respectively, and \mathbf{M} is a 4×4 matrix that maps point \mathbf{x} to \mathbf{x}' . Fig. B.3 illustrates this process.

If this same process is applied to every point location \mathbf{x}_* in I , it is then possible to map any point \mathbf{x}_* from image I to image I' through the equation

$$\tilde{\mathbf{x}}'_* \sim \mathbf{M}\tilde{\mathbf{x}}_*. \quad (\text{B.11})$$

Unfortunately, Eq. B.10 assumes that the inverse depth d for each pixel in the image are known but, for most the conventional images or digital cameras, it is not possible to obtain such information. In fact, the goal of some of the 3D reconstruction and plane+parallax approaches, is to estimate these parameters (or some of them) in order to align the images through this or a similar process. One of the most practical options for dispensing the use of this “depth information” for relating two views is through the use of homographies. But, as seen in Chapter 2, homographies are limited to special cases only. The purpose of the proposed Moving DLT (Chapter 4) is to relax such limitations. Thus, making homography warps applicable to more general imaging scenarios (as seen in Chapters 4-6 of this thesis).

Bibliography

- [1] Sameer Agarwal and Keir Mierle. Ceres solver. <https://code.google.com/p/ceres-solver/>.
- [2] Aseem Agarwala, Mira Dontcheva, Manesh Agrawala, Steven Drucker, Alex Colburn, Brian Curless, David Salesin, and Michael Cohen. Interactive digital photomontage. In *Proceedings of the ACM Special Interest Group on Graphics and Interactive Techniques (SIGGRAPH)*, 2004.
- [3] Aseem Agarwala, Maneesh Agarwala, Michael Cohen, David Salesin, and Richard Szeliski. Photographing long scenes with multi-viewpoint panoramas. In *Proceedings of the ACM Special Interest Group on Graphics and Interactive Techniques (SIGGRAPH)*, 2006.
- [4] Marc Alexa, Johannes Behr, Dabiel Cohen-Or, Shachar Fleishman, David Levin, and Claudio T. Silva. Computing and rendering point set surfaces. *IEEE TVCG*, 9(1):3–15, 2003.
- [5] Simon Baker and Iain Matthews. Lucas-Kanade 20 years on: A unifying framework. *International Journal of Computer Vision (IJCV)*, 56(3):221–255, 2004.
- [6] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B. Goldman. PatchMatch: a randomized correspondence algorithm for structural image editing. In *Proceedings of the ACM Special Interest Group on Graphics and Interactive Techniques (SIGGRAPH)*, pages 24:1–24:11. ACM, 2009. ISBN 978-1-60558-726-4.
- [7] Åke Björck. *Numerical methods for least squares problems*. SIAM, 1996.
- [8] Fred L. Bookstein. Principal warps: thin-plate splines and the decomposition of deformations. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 11(6): 567–585, Jun 1989.

- [9] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 23(11):1222–1239, 2001.
- [10] Christoph Bregler, Aaron Hertzmann, and Henning Biermann. Recovering non-rigid 3D shape from image streams. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 300–314, 2000.
- [11] Matthew Brown and David G. Lowe. Recognising panoramas. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, volume 2, pages 1218–1225, October 2003.
- [12] Matthew Brown and David G. Lowe. Automatic panoramic image stitching using invariant features. *International Journal of Computer Vision (IJCV)*, 74(1):59–73, 2007.
- [13] Peter J. Burt and Edward H. Adelson. A multiresolution spline with applications to image mosaics. *ACM Transactions on Graphics*, 2(4):217–236, 1983.
- [14] Noppadol Chumchob and Ke Chen. A robust affine image registration method. *International Journal of Numerical Analysis and Modeling*, 6(2):311–334, 2009.
- [15] Soheil Darabi, Eli Shechtman, Connelly Barnes, Dan Goldman, and Pradeep Sen. Image melding: Combining inconsistent images using patch-based synthesis. *ACM Transactions on Graphics*, 31-4:1–10, July 2012. ISSN 0730-0301.
- [16] Fadi Dornaika and Ronald Chung. Mosaicking images with parallax. *Signal Processing: Image Communication*, 19(8):771 – 786, 2004.
- [17] Ashley Eden, Matthew Uyttendaele, and Richard Szeliski. Seamless image stitching of scenes with large motions and expoure differences. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006.
- [18] Olivier Faugeras and Francis Lustman. Motion and structure from motion in a piecewise planar environment. *International Journal of Pattern Recognition and Artificial Intelligence*, 02(03):485–508, 1988.
- [19] João Fayad, Lourdes Agapito, and Alessio Bue. Piecewise quadratic reconstruction of non-rigid surfaces from monocular sequences. In Kostas Daniilidis, Petros Maragos, and Nikos Paragios, editors, *Proceedings of the European Conference on Computer Vision (ECCV)*, volume 6314 of *Lecture Notes in Computer Science*, pages 297–310. Springer Berlin Heidelberg, 2010.

- [20] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.
- [21] Shachar Fleishman, Daniel Cohen-Or, and Cláudio T. Silva. Robust moving least-squares fitting with sharp features. In *Proceedings of the ACM Special Interest Group on Graphics and Interactive Techniques (SIGGRAPH)*, pages 544–552, New York, NY, USA, 2005. ACM.
- [22] David Gallup, Jan-Michael Frahm, and Marc Pollefeys. Piecewise planar and non-planar stereo for urban scene reconstruction. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1418–1425, June 2010.
- [23] Junhong Gao, Seon J. Kim, and Michael S. Brown. Constructing image panoramas using dual-homography warping. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.
- [24] Ravi Garg, Anastasios Roussos, and Lourdes Agapito. Robust trajectory-space TV-L1 optical flow for non-rigid sequences. In *Proceedings of the International Conference on Energy Minimization Methods in Computer Vision and Pattern Recognition (EMM-CVPR)*, pages 300–314, Berlin, Heidelberg, 2011. Springer-Verlag.
- [25] Ravi Garg, Anastasios Roussos, and Lourdes Agapito. Dense variational reconstruction of non-rigid surfaces from monocular video. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013.
- [26] Vincent Gay-Bellile, Adrien Bartoli, and Patrick Sayd. Direct estimation of nonrigid registrations with image-based self-occlusion reasoning. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(1):87–104, Jan 2010.
- [27] Dan B. Goldman, Brian Curless, David Salesin, and Steven M. Seitz. Schematic storyboarding for video visualization and editing. In *Proceedings of the ACM Special Interest Group on Graphics and Interactive Techniques (SIGGRAPH)*, pages 862–871, 2006. ISBN 1-59593-364-6.
- [28] Gene H. Golub and Charles F. van Loan. *Matrix computations*. The Johns Hopkins University Press, 3rd edition, 1996.
- [29] M. Hansen, Prabu Anandan, K. Dana, G. Van der Wal, and P. Burt. Real-time scene stabilization and mosaic construction. In *Proceedings of the Second IEEE Workshop on Applications of Computer Vision*, pages 54–62, 1994.

- [30] Richard Hartley. In defense of the eight-point algorithm. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 19(6):580–593, 1997.
- [31] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004.
- [32] Kaiming He and Jian Sun. Statistics of patch offsets for image completion. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 16–29, Berlin, Heidelberg, 2012. Springer-Verlag. ISBN 978-3-642-33708-6.
- [33] Peter J. Huber. *Robust Statistics*. Wiley, 1981.
- [34] Takeo Igarashi, Tomer Moscovich, and John F. Hughes. As-rigid-as-possible shape manipulation. *ACM Transactions on Graphics*, 24(3), 2005.
- [35] Michal Irani and Prabu Anandan. Parallax geometry of pairs of points for 3D scene analysis. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 17–30, 1996.
- [36] Michal Irani and Prabu Anandan. Video indexing based on mosaic representations. *Proceedings of the IEEE*, 86(5):905–921, 1998. ISSN 0018-9219.
- [37] Michal Irani, Steve Hsu, and Prabu Anandan. Video compression using mosaic representations. *Signal Processing: Image Communication*, 7(4–6):529–552, 1995.
- [38] Mark Jenkinson and Stephen Smith. A global optimisation method for robust affine registration of brain images. *Medical Image Analysis*, 5(2):143–156, 2001.
- [39] Jiaya Jia and Chi-Keung Tang. Image stitching using structure deformation. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 30(4):617–631, 2008.
- [40] Eun-Young Kang, Isaac Cohen, and Gerard Medioni. A graph-based global registration for 2D mosaics. In *ICPR*, 2000.
- [41] Rakesh Kumar, Prabu Anandan, and Keith Hanna. Direct recovery of shape from multiple views: a parallax based approach. In *Proceedings of the 12th IAPR International Conference on*, volume 1, pages 685–688, 1994.
- [42] Rakesh Kumar, Prabu Anandan, Michal Irani, James Bergen, and Keith Hanna. Representation of scenes from collections of images. In *International Workshop on Representation of Visual Scenes*, pages 10–17, 1995.

- [43] Claudia Kuster, Jean-Charles Bazin, Cengiz ztireli, Teng Deng, Tobias Martin, Tiberiu Popa, and Markus Gross. Spatio-temporal geometry fusion for multiple hybrid cameras using moving least squares surfaces. *Computer Graphics Forum*, 33(2):1–10, 2014. ISSN 1467–8659.
- [44] Vivek Kwatra, Arno Schödl, Irfan Essa, Greg Turk, and Aaron Bobick. Graphcut textures: image and video synthesis using graph cuts. *ACM Transactions on Graphics*, 22(3):277–286, July 2003. ISSN 0730-0301.
- [45] Peter Lancaster and Kestutis Salkauskas. Surfaces generated by moving least squares methods. *Mathematics of Computation*, 37(155):141–158, 1981.
- [46] Didier Le Gall. MPEG: A video compression standard for multimedia applications. *ACM Communications*, 34(4):46–58, April 1991.
- [47] Ming-Chieh Lee, Wei-ge Chen, Chih-lung B. Lin, Chuang Gu, Tomislav Markoc, Steven I. Zabinsky, and Richard Szeliski. A layered video object coding system using sprite and affine motion model. *IEEE Trans. Cir. and Sys. for Video Technol.*, 7(1):130–145, 1997.
- [48] Kenneth Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly Journal of Applied Mathematics*, II(2):164–168, 1944.
- [49] David Levin. The approximation power of moving least-squares. *Mathematics of Computation*, 67:1517–1531, 1998.
- [50] Xiangru Li and Zhanyi Hu. Rejecting mismatches by correspondence function. *International Journal of Computer Vision*, 89(1):1–17, 2010.
- [51] Wen-Yan Lin, Siying Liu, Yasuyuki Matsushita, Tian-Tsong Ng, and Loong-Fah Cheong. Smoothly varying affine stitching. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.
- [52] Wen-Yan Lin, Linlin Liu, Yasuyuki Matsushita, and Kok-Lim Low. Aligning images in the wild. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.
- [53] Feng Liu, Michael Gleicher, Hailin Jin, and Aseem Agarwala. Content-preserving warps for 3D video stabilization. In *Proceedings of the ACM Special Interest Group on Graphics and Interactive Techniques (SIGGRAPH)*, 2009.

- [54] Huafeng Liu and Pengcheng Shi. Discontinuity-preserving moving least squares method. In Jun Zhang, Ji-Huan He, and Yuxi Fu, editors, *Computational and Information Science*, volume 3314 of *Lecture Notes in Computer Science*, pages 562–569. Springer Berlin Heidelberg, 2005.
- [55] Shuaicheng Liu, Lu Yuan, Ping Tan, and Jian Sun. Bundled camera paths for video stabilization. *ACM Transactions on Graphics*, 32(4):1–10, Jul 2013. ISSN 0730-0301.
- [56] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal on Computer Vision*, 60(2):91–110, 2004.
- [57] Donald W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial and Applied Mathematics*, 11(2):431–441, 1963.
- [58] Roberto Marzotto, Andre Fusiello, and Vittorio Murino. High resolution video mosaicing with global alignment. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004.
- [59] Andriy Mynorenko, Xubo Song, and Miguel Á. Carreira-Perpinán. Non-rigid point set registration: coherent point drift. In *Neural Information Processing Systems Foundation (NIPS)*, 2007.
- [60] Yoshikuni Nomura, Li Zhang, and Shree K. Nayar. Scene collages and flexible camera arrays. In *Proceedings of the Eurographics Conference on Rendering Techniques*, pages 127–138, 2007. ISBN 978-3-905673-52-4.
- [61] A. C. Oztireli, G. Guennebaud, and M. Gross. Feature preserving point set surfaces based on non-linear kernel regression. *Computer Graphics Forum*, 28(2):493–501, 2009. ISSN 1467–8659.
- [62] Shmuel Peleg and Moshe Ben-Ezra. Stereo panorama with a single camera. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, 1999.
- [63] Shmuel Peleg, Moshe Ben-Ezra, and Yael Pritch. Omnidirectional stereo imaging. <http://www.ben-ezra.org/omnistereo/omni.html>.
- [64] Shmuel Peleg, Yael Pritch, and Moshe Ben-Ezra. Cameras for stereo panoramic imaging. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2000.

- [65] Shmuel Peleg, Benny Rousso, Alex Rav-Acha, and Assaf Zomet. Mosaicing on adaptive manifolds. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 22(10), 2000.
- [66] Shmuel Peleg, Moshe Ben-Ezra, and Yael Pritch. Omnistere: Panoramic stereo imaging. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 23:279–290, 2001.
- [67] Patrick Pérez, Michael Gangnet, and Andrew Blake. Poisson image editing. In *Proceedings of the ACM Special Interest Group on Graphics and Interactive Techniques (SIGGRAPH)*, 2003.
- [68] Daniel Pizarro and Adrien Bartoli. Feature-based deformable surface detection with self-occlusion reasoning. *International Journal of Computer Vision (IJCV)*, 97(1):54–70, 2012.
- [69] Chris Russell, João Fayad, and Lourdes Agapito. Energy based multiple model fitting for non-rigid structure from motion. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3009–3016, June 2011.
- [70] Harpreet S. Sawhney. 3d geometry from planar parallax. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 929–934, 1994.
- [71] Scott Schaefer, Travis McPhail, and Joe Warren. Image deformation using moving least squares. In *Proceedings of the ACM Special Interest Group on Graphics and Interactive Techniques (SIGGRAPH)*, 2006.
- [72] Oliver Schmitt, Jan Modersitzki, Stefan Heldmann, Stefan Wirtz, and Bernd Fischer. Image registration of sectioned brains. *International Journal of Computer Vision (IJCV)*, 73(1):5–39, 2007.
- [73] Amnon Shashua and Nassir Navab. Relative affine structure: Theory and application to 3D reconstruction from perspective views. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 483–489, 1994.
- [74] Heung-Yeung Shum and Richard Szeliski. Construction of panoramic mosaics with global and local alignment. *International Journal of Computer Vision (IJCV)*, 36(2), 2000.
- [75] Peter Stange. On the efficient update of the singular value decomposition. In *Applied Mathematics and Mechanics*, 2008.

- [76] Richard Szeliski. Image alignment and stitching. *Handbook of Mathematical Models in Computer Vision*, pages 273–292, 2005.
- [77] Richard Szeliski. *Computer Vision: Algorithms and Applications*. Springer, 2009.
- [78] Richard Szeliski and Heung-Yeung Shum. Creating full view panoramic image mosaics and environment maps. In *Proceedings of the ACM Special Interest Group on Graphics and Interactive Techniques (SIGGRAPH)*, pages 251–258, 1997.
- [79] Richard Szeliski, Michael Uyttendaele, and Drew Steedly. Fast poisson blending using multi-splines. In *Proceedings of the International Conference on Computational Photography (ICCP)*, pages 1–8, 2011.
- [80] Trung Thanh-Pham, Tat-Jun Chin, Jin Yu, and David Suter. Simultaneous sampling and multi-structure fitting with adaptive reversible jump MCMC. In *Neural Information Processing Systems Foundation (NIPS)*, 2011.
- [81] Trung Thanh-Pham, Tat-Jun Chin, Jin Yu, and David Suter. The random cluster model for robust geometric fitting. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 0:710–717, 2012. ISSN 1063-6919.
- [82] Philippe Thévenaz, Urs E. Ruttimann, and Michael Unser. A pyramid approach to sub-pixel registration based on intensity. *Transactions on Image Processing (TIP)*, 7:27–41, 1998.
- [83] Quoc-Huy Tran, Tat-Jun Chin, Gustavo Carneiro, Michael S. Brown, and David Suter. In defense of RANSAC for outlier removal in deformation registration. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2012.
- [84] Bill Triggs, Philip Mclauchlan, Richard Hartley, and Andrew Fitzgibbon. Bundle adjustment – a modern synthesis. In *Vision Algorithms: Theory and Practice*, pages 298–375. Springer Verlag, 2000.
- [85] Roger Y. Tsai. A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf TV cameras and lenses. *Journal of Robotics and Automation*, pages 221–244, 1992.
- [86] Shinji Umeyama. Least-squares estimation of transformation parameters between two point patterns. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 13(4):376–380, 1991.

- [87] Aydin Varol, Mathieu Salzmann, Engin Tola, and Pascal Fua. Template-free monocular reconstruction of deformable surfaces. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1811–1818, Sept 2009.
- [88] Andrea Vedaldi and Brian Fulkerson. VLFeat: An open and portable library of computer vision algorithms. <http://www.vlfeat.org/>, 2008.
- [89] Lejing Wang, Joerg Traub, Simon Weidert, Sandro Michael Heining, Ekkehard Euler, and Nassir Navab. Parallax-free long bone X-ray image stitching. In *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pages 173–180, Berlin, Heidelberg, 2009. Springer-Verlag. ISBN 978-3-642-04267-6.
- [90] Christopher Weber, Stefanie Hahmann, Hans Hagen, and Georges-Pierre Bonneau. Sharp feature preserving MLS surface reconstruction based on local feature line approximations. *Graphical Models*, 74(6):335–345, November 2012.
- [91] Yonatan Wexler, Eli Shechtman, and Michal Irani. Space-time completion of video. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 29(3):463–476, 2007.
- [92] Hoi-Sim Wong, Tat-Jun Chin, Jin Yu, and David Suter. Efficient multi-structure robust fitting with incremental top-k lists comparison. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, 2010.
- [93] Hoi-Sim Wong, Tat-Jun Chin, Jin Yu, and David Suter. Dynamic and hierarchical multi-structure geometric model fitting. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2011.
- [94] Lihi Zelnik-Manor and Pietro Perona. Automating joiners. In *Proceedings of the International Symposium on Non-photorealistic Animation and Rendering*, pages 121–131. ACM, 2007. ISBN 978-1-59593-624-0.
- [95] Hong-Jiang Zhang, Atreyi Kankanhalli, and Stephen W. Smoliar. Automatic partitioning of full-motion video. *Multimedia Systems*, 1(1):10–28, 1993.
- [96] Zhengyou Zhang. Parameter estimation techniques: a tutorial with application to conic fitting. *Image and Vision Computing*, 15(1):59–76, 1997.
- [97] Zhengyou Zhang. A flexible new technique for camera calibration. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 22(11):1330–1334, November 2000. ISSN 0162-8828.

-
- [98] Zhongfei Zhang and Allen R. Hanson. 3D reconstruction based on homography mapping. In *Proceedings of the Advanced Research Projects Agency (ARPA) Image Understanding Workshop*, pages 0249–6399, 1996.
- [99] Qi Zhi and Jeremy R. Cooperstock. Overcoming parallax and sampling density issues in image mosaicing of non-planar scenes. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2007.
- [100] Jianke Zhu and Michael R. Lyu. Progressive finite Newton approach to real-time nonrigid surface detection. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2007.