Linköping Studies in Science and Technology
Thesis No. 1535

Geometric Computer Vision for Rolling-shutter and Push-broom Sensors

Erik Ringaby

Linköping University
INSTITUTE OF TECHNOLOGY

Department of Electrical Engineering
Linköping University, SE-581 83 Linköping, Sweden

Linköping June 2012

Geometric Computer Vision for Rolling-shutter and Push-broom Sensors

Department of Electrical Engineering
Linköping University
SE-581 83 Linköping
Sweden

ISBN 978-91-7519-872-9 ISSN 0280-7971 LIU-TEK-LIC-2012:20

Abstract

Almost all cell-phones and camcorders sold today are equipped with a CMOS (Complementary Metal Oxide Semiconductor) image sensor and there is also a general trend to incorporate CMOS sensors in other types of cameras. The sensor has many advantages over the more conventional CCD (Charge-Coupled Device) sensor such as lower power consumption, cheaper manufacturing and the potential for on-chip processing. Almost all CMOS sensors make use of what is called a *rolling shutter*. Compared to a *global shutter*, which images all the pixels at the same time, a rolling-shutter camera exposes the image row-by-row. This leads to geometric distortions in the image when either the camera or the objects in the scene are moving. The recorded videos and images will look wobbly (jello effect), skewed or otherwise strange and this is often not desirable. In addition, many computer vision algorithms assume that the camera used has a global shutter, and will break down if the distortions are too severe.

In airborne remote sensing it is common to use push-broom sensors. These sensors exhibit a similar kind of distortion as a rolling-shutter camera, due to the motion of the aircraft. If the acquired images are to be matched with maps or other images, then the distortions need to be suppressed.

The main contributions in this thesis are the development of the three dimensional models for rolling-shutter distortion correction. Previous attempts modelled the distortions as taking place in the image plane, and we have shown that our techniques give better results for hand-held camera motions.

The basic idea is to estimate the camera motion, not only between frames, but also the motion during frame capture. The motion can be estimated using inter-frame image correspondences and with these a non-linear optimisation problem can be formulated and solved. All rows in the rolling-shutter image are imaged at different times, and when the motion is known, each row can be transformed to the rectified position.

In addition to rolling-shutter distortions, hand-held footage often has shaky camera motion. It has been shown how to do efficient video stabilisation, in combination with the rectification, using rotation smoothing.

In the thesis it has been explored how to use similar techniques as for the rolling-shutter case in order to correct push-broom images, and also how to rectify 3D point clouds from e.g. the Kinect depth sensor.

Acknowledgments

First of all I would like to thank my supervisor Per-Erik Forssén for excellent guidance in the inspiring projects.

I would also like to thank my co-supervisor Michael Felsberg for sharing his knowledge and allowing me to work in his research group.

All of the members of the Computer Vision Laboratory have contributed, both by creating an inspiring atmosphere and by having interesting discussions. I would especially like to thank Per-Erik Forssén, Johan Hedborg, Marcus Wallenberg and Vasileios Zografos.

I would like to thank the people at FOI for our collaboration and Johan Wiklund for all the technical and hardware support.

Thank you Per-Erik and Vasileios for proof-reading the manuscript.

I would also like to thank my friends and family, especially my mother for life long support and encouragement, and interest in trying to understand what I am doing.

Anne Liljedahl, thank you for all what you have given me, the experiences we have shared, and letting me see things I may otherwise not have seen.

The research leading to this thesis has received funding from CENIIT through the Virtual Global Shutters for CMOS Cameras project.

Erik Ringaby June 2012

Contents

Part I

Background

Chapter 1

Introduction

1.1 Motivation

Almost all cell-phones and camcorders sold today are equipped with a CMOS (Complementary Metal Oxide Semiconductor) image sensor and there is also a general trend to incorporate CMOS sensors in other types of cameras. The sensor has many advantages over the more conventional CCD (Charge-Coupled Device) sensor such as lower power consumption, cheaper manufacturing and the potential for on-chip processing. Almost all CMOS sensors make use of what is called a *rolling shutter*. Compared to a *global shutter*, which images all the pixels at the same time, a rolling-shutter camera exposes the image row-by-row. This leads to geometric distortions in the image when either the camera or the objects in the scene are moving. Figure 1.1 shows some examples of different rolling-shutter distortions. The top left shows skew caused by a panning motion, the top right shows distortions caused by a 3D rotation and the bottom left shows distortions from a fast moving object (note that the car and the wheels are distorted differently). Almost all computer vision algorithms assume that the camera used has a global shutter. The work in this thesis will enable people to also use rolling-shutter cameras and is focused on distortions caused by camera motion, e.g. top row in figure 1.1.

In airborne remote sensing it is common to use push-broom sensors. These sensors exhibit a similar kind of distortion as a rolling-shutter camera, due to the motion of the aircraft, see figure 1.1 for an example. If the acquired images are to be matched with maps or other images, then the distortions need to be suppressed. In this thesis it has been explored how to use similar techniques as for the rolling-shutter case in order to correct push-broom images.

The work leading to this thesis was conducted within the *Virtual Global Shutters for CMOS Cameras* project, and papers D and E in collaboration with the Swedish Defence Research Agency (FOI).

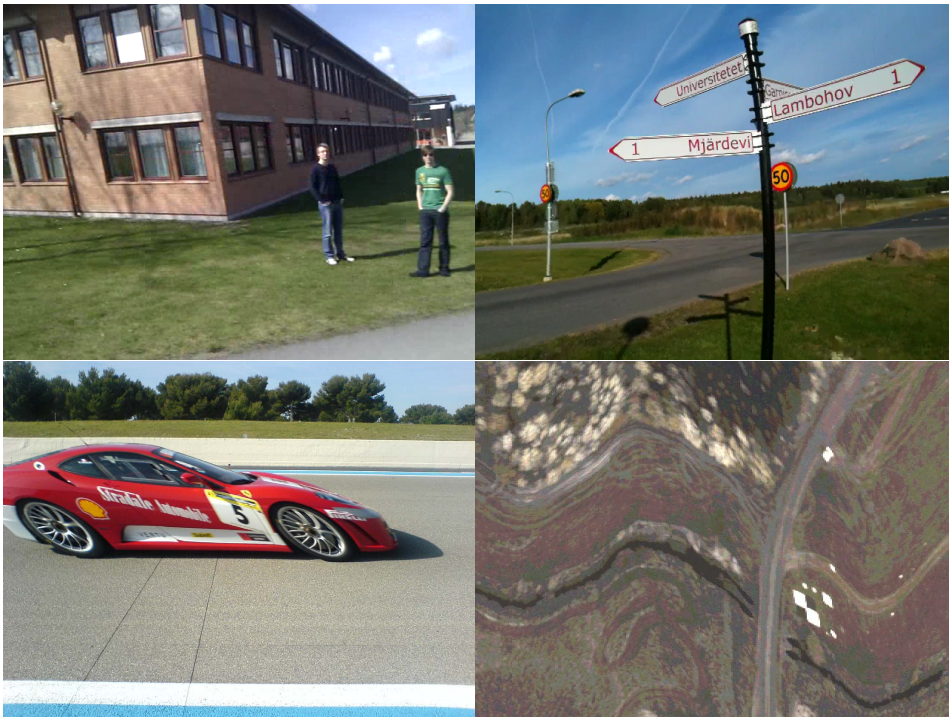

Figure 1.1: Geometrical distortions in images. Top left: slanted house due to camera pan. Top right: bent pole due to camera 3D rotation. Bottom left: slanted car and curved wheels due to fast object motion. Bottom right: curved path due to aircraft motion.

1.2 Outline

The thesis is divided into two parts. The first part gives a background to the theory and sensors used in my work. The second part consists of five publications covering rolling shutter and push-broom distortions.

1.2.1 Outline Part I: Background

The background part starts with chapter 2 which describes the sensors used in the publications. Chapter 3 introduces the camera models. Chapter 4 describes sensor motion estimation and how to correct for the geometrical distortions. Chapter 5 describes the evaluation measures used, and how the ground-truth dataset was generated. The first part ends with chapter 6, concluding remarks.

1.2.2 Outline Part II: Included Publications

Preprint versions of five publications are included in Part II. The full details and abstracts of these papers, together with statements of the contributions made by the authors, are given below.

Paper A: Rectifying rolling shutter video from hand-held devices

> Per-Erik Forssén and Erik Ringaby. Rectifying rolling shutter video from hand-held devices. In *CVPR'10*, 2010.

Abstract:
This paper presents a method for rectifying video sequences from *rolling shutter* (RS) cameras. In contrast to previous RS rectification attempts we model distortions as being caused by the 3D motion of the camera. The camera motion is parametrised as a continuous curve, with knots at the last row of each frame. Curve parameters are solved for using non-linear least squares over inter-frame correspondences obtained from a KLT tracker. We have generated synthetic RS sequences with associated ground-truth to allow controlled evaluation. Using these sequences, we demonstrate that our algorithm improves over to two previously published methods. The RS dataset is available on the web to allow comparison with other methods.
Contribution:
This paper was the first to correct rolling-shutter distortions by modeling the 3D camera motion. It also introduced the first rolling-shutter dataset. The author contributed to the rotation motion model, produced the dataset, and conducted the experiments.

Paper B: Efficient Video Rectification and Stabilisation for Cell-Phones

> Erik Ringaby and Per-Erik Forssén. Efficient video rectification and stabilisation for cell-phones. *International Journal of Computer Vision*, 96(3):335–352, 2012. http://dx.doi.org/10.1007/s11263-011-0465-8.

Abstract:
This article presents a method for rectifying and stabilising video from cell-phones with *rolling shutter* (RS) cameras. Due to size constraints, cell-phone cameras have constant, or near constant focal length, making them an ideal application for calibrated projective geometry. In contrast to previous RS rectification attempts that model distortions in the image plane, we model the 3D rotation of the camera. We parameterise the camera rotation as a continuous curve, with knots distributed across a short frame interval. Curve parameters are found using non-linear least squares over inter-frame correspondences from a KLT tracker. By smoothing a sequence of reference rotations from the estimated curve, we can at a small extra cost, obtain a high-quality image stabilisation. Using synthetic RS sequences with associated ground-truth, we demonstrate that our rectification improves over two other methods. We also compare our video stabilisation with the methods in iMovie and Deshaker.
Contribution:
This paper extends paper A, by allowing camera motions that are non constant during a frame capture, a new GPU-based forward interpolation, and the application of video stabilisation. The author was the main source of the findings for the importance of spline knot positions, the GPU based interpolation, and implemented the stabilisation.

Paper C: Scan Rectification for Structured Light Range Sensors with Rolling Shutters

> Erik Ringaby and Per-Erik Forssén. Scan rectification for structured light range sensors with rolling shutters. In *IEEE International Conference on Computer Vision*, Barcelona, Spain, November 2011. IEEE, IEEE Computer Society

Abstract:
Structured light range sensors, such as the Microsoft Kinect, have recently become popular as perception devices for computer vision and robotic systems. These sensors use CMOS imaging chips with electronic rolling shutters (ERS). When using such a sensor on a moving platform, both the image, and the depth map, will exhibit geometric distortions. We introduce an algorithm that can suppress such distortions, by rectifying the 3D point clouds from the range sensor. This is done by first estimating the time continuous 3D camera trajectory, and then transforming the 3D points to where they would have been, if the camera had been stationary. To ensure that image and range data are synchronous, the camera trajectory is computed from KLT tracks on the structured-light frames, after suppressing the structured-light pattern. We evaluate our rectification, by measuring angles between the visible sides of a cube, before and after rectification. We also measure how much better the 3D point clouds can be aligned after rectification. The obtained improvement is also related to the actual rotational velocity, measured using a MEMS gyroscope.
Contribution: This paper was the first to address the rolling-shutter problem on range scan sensors. Compared to paper A and paper B, the cost function is

defined on 3D features, and the full 6 DOF motion can be estimated and corrected for. The author contributed to the motion estimation, feature rejection steps, and the experiments.

Paper D: Co-alignment of Aerial Push-Broom Strips using Trajectory Smoothness Constraints

> Erik Ringaby, Jörgen Ahlberg, Per-Erik Forssén, and Niclas Wadströmer. Co-alignment of aerial push-broom strips using trajectory smoothness constraints. In *Proceedings SSBA'10 Symposium on Image Analysis*, pages 63–66, March 2010

Abstract:
We study the problem of registering a sequence of scan lines (a *strip*) from an airborne push-broom imager to another sequence partly covering the same area. Such a registration has to compensate for deformations caused by attitude and speed changes in the aircraft. The registration is challenging, as both strips contain such deformations.

Our algorithm estimates the 3D rotation of the camera for each scan line, by parametrising it as a linear spline with a number of knots evenly distributed in one of the strips. The rotations are estimated from correspondences between strips of the same area. Once the rotations are known, they can be compensated for, and each line of pixels can be transformed such that ground trace of the two strips are registered with respect to each other.
Contribution: This paper explored the possibility of using the previously introduced rolling-shutter correction scheme to register push-broom strips, by using smoothness constraints. The author contributed to the registration and conducted the experiments.

Paper E: Co-aligning aerial hyperspectral push-broom strips for change detection

> Erik Ringaby, Jörgen Ahlberg, Niclas Wadströmer, and Per-Erik Forssén. Co-aligning aerial hyperspectral push-broom strips for change detection. In *Proceedings of SPIE Security+Defence*, volume 7835, Tolouse, France, September 2010. SPIE, SPIE Digital Library

Abstract:
We have performed a field trial with an airborne push-broom hyperspectral sensor, making several flights over the same area and with known changes (e.g., moved vehicles) between the flights. Each flight results in a sequence of scan lines forming an image strip, and in order to detect changes between two flights, the two resulting image strips must be geometrically aligned and radiometrically corrected. The focus of this paper is the geometrical alignment, and we propose an image- and gyro-based method for geometric co-alignment (registration) of two image strips. The method is particularly useful when the sensor is not stabilized, thus reducing the need for expensive mechanical stabilization. The method works in several

steps, including gyro-based rectification, global alignment using SIFT matching, and a local alignment using KLT tracking. Experimental results are shown but not quantified, as ground truth is, by the nature of the trial, lacking.
Contribution: This paper extends paper D by using gyroscope measurements for the strip rectification. In addition to this a non-rigid registration is performed. The author contributed to the gyro-camera transformation, to the strip registration and conducted the experiments.

Other Publications

The following publications by the author are related to the included papers.

> Gustav Hanning, Nicklas Forslöw, Per-Erik Forssén, Erik Ringaby, David Törnqvist, and Jonas Callmer. Stabilizing cell phone video using inertial measurement sensors. In *The Second IEEE International Workshop on Mobile Vision*, Barcelona, Spain, November 2011. IEEE.

> Johan Hedborg, Erik Ringaby, Per-Erik Forssén, and Michael Felsberg. Structure and motion estimation from rolling shutter video. In *IWMV workshop at ICCV'11*, 2011.

Chapter 2

Sensors

All sensors used in this thesis share the property of sequential acquisition of an image frame. How the sensors work will be described in the following sections.

2.1 Rolling-shutter sensors

The function of a camera shutter is to allow light to pass through for a determined period of time. The shutter used can either be mechanical or electronic and have a global, block or rolling exposure method. In a global shutter camera, all pixels in a frame are imaged at a single time instance. Rolling shutter on the other hand is a technique used when acquiring images by scanning the frame. Instead of imaging the scene at a single time instance, the image rows are sequentially reset and read out. The rows which are not being read out are continued to be exposed. Figure 2.1 shows the difference between image integration with a global-shutter and rolling-shutter camera. The rolling-shutter method has the advantage of longer integration times, as shown in the bottom figure, which increases the sensitivity.

The two most common image sensors used in digital cameras are the CCD (Charge-Coupled Device) and the CMOS (Complementary Metal Oxide Semiconductor) image sensors. Generally, CCD sensors use global shutters and CMOS use rolling shutters. There are CMOS sensors with a global shutter, where all the pixels are exposed to light at the same time and at the end of integration time they are transferred to a light-shielded storage area simultaneously. After this the signals are read out.

In addition to increased sensitivity, the CMOS sensors are also cheaper to manufacture, they use less power and it is also simple to integrate other kind of electronics on the chip. Almost all camera equipped cellphones make use of a rolling shutter and the CMOS sensor is gradually replacing the CCD sensor in other segments such as camcorders and video capable SLR's. The rolling shutter will however introduce distortions when the scene or camera is moving, and the amount of these distortions depend on how fast the shutter "rolls". A rule of thumb

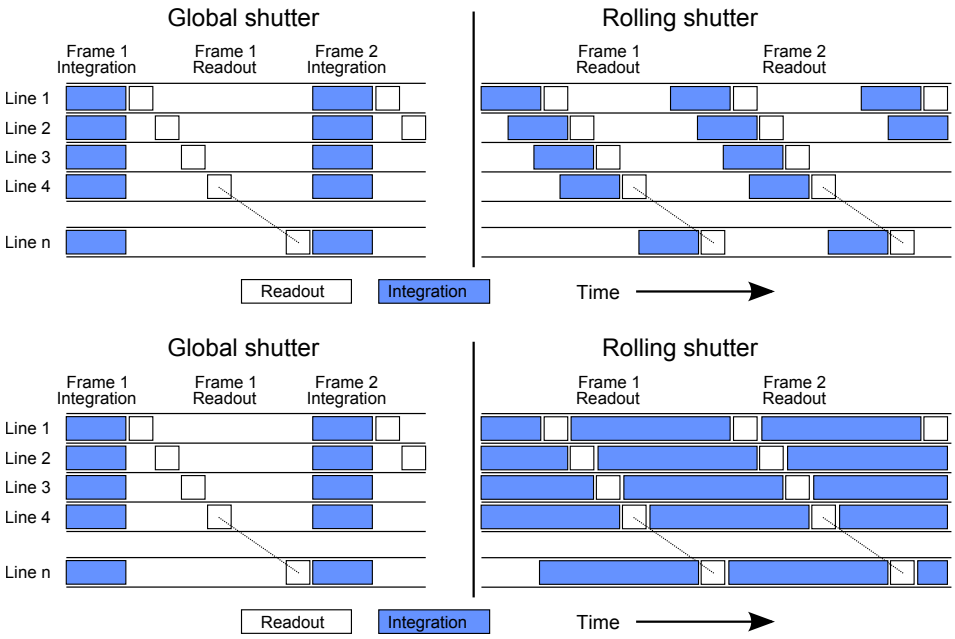

Figure 2.1: Rolling-shutter image integration.

is that the higher the resolution is, the slower the sensor will be, and furthermore expensive sensors are usually faster. Almost all computer vision algorithms assume a global shutter camera, but techniques from this thesis will enable researchers and others to also use rolling shutter cameras.

2.2 Kinect

In 2010, Microsoft released the Kinect sensor which is designed to provide motion input to the Xbox 360 gaming device. The sensor has gained popularity in the vision community due to its ability to deliver quasi-dense depth maps in 30 Hz, combined with a low price. The hardware consists of a near infrared (NIR) laser projector (A), a CMOS colour sensor (B) and a NIR CMOS sensor (C), see figure 2.2.

The laser projector is used to project a structured light pattern onto the scene. The NIR CMOS sensor images this pattern and the device uses triangulation to create a depth map. The image resolution is 640×480 when using an update of 30 Hz, but it is also possible to receive NIR and colour frames in 1280×1024 resolution. The depth map can be obtained at the same time as either the NIR image or the colour image, but the colour and NIR images cannot be obtained at the same time.

Both the NIR and colour sensors have electronic rolling shutters. Since the

Figure 2.2: The Kinect Sensor, (A) NIR laser projector, (B) CMOS colour sensor, (C) CMOS NIR sensor

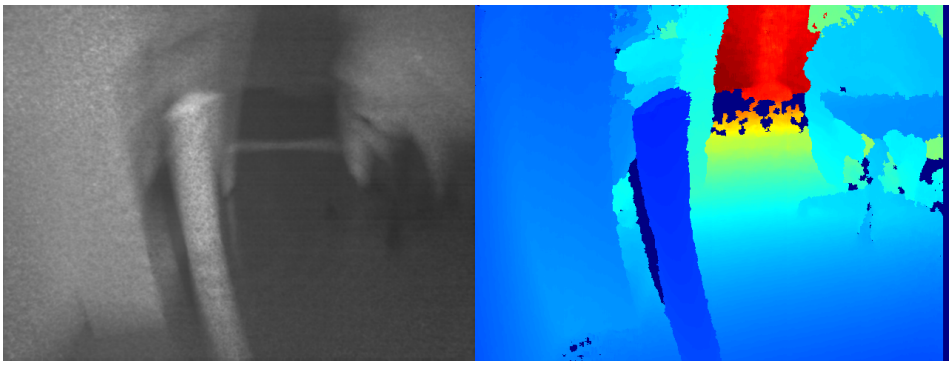

Figure 2.3: Distortions in the NIR and depth images caused by fast sensor motion.

Kinect sensor is designed to be stationary and objects in front of it do not move that fast (or very close to the sensor), the rolling-shutter distortions are usually not a big problem. If on the other hand the sensor is used on a mobile platform it will have noticeable distortions. See figure 2.3 for an example of a fast rotation. The two image sensors are not synchronised, so the same rows in the depth image and the colour image are, in general, not imaged at the same time.

2.3 Push-broom sensor

Push-broom sensors are commonly used in airborne remote sensing. The images, also called *strips* or *swaths*, from a push-broom sensor have similar geometrical

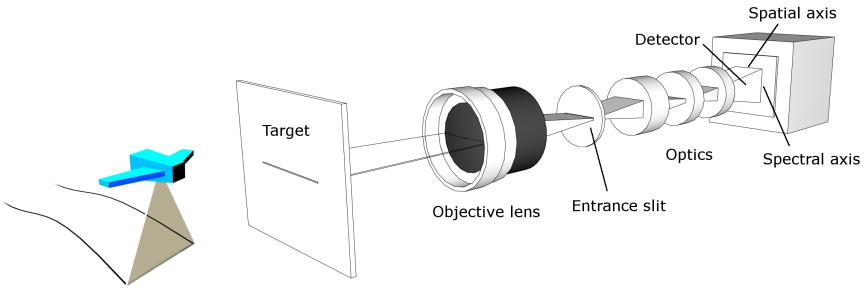

Figure 2.4: Left: How the 1D sensor "paints" the image. Right: Different spectral bands separated on the sensor using a prism.

distortions as those from a rolling-shutter sensor, but the sensors differ a lot in their design.

Instead of capturing a two dimensional image, the sensor is one dimensional in the spatial domain and "paints" the image by exploiting the ego-motion of the moving platform, see figure 2.4 left. The sensor itself is two dimensional and a prism refracts the light into different wavelengths along one of the axes of the hyper-spectral sensor (figure 2.4, right). The number of spectral bands depends on the sensor used.

If the imaging platform (e.g. aircraft) moves in a linear trajectory we would have to solve a simple problem, but this rarely the case. When the aircraft rotates, or moves away from the path, geometric distortions will be present in the image.

There are also hyper-spectral sensors which use two spatial dimensions, but record the different wavelengths at different time steps. In this case, the registration has to be done across different spectral bands instead, but this is not considered here.

2.4 Other sensors

Other similar sensors not covered in this thesis are crossed-slits [16], and moving LIDAR[2].

Chapter 3

Camera models

Some computer vision algorithms operate only in the image plane and do not care which camera has been used to record the image. We need a camera model for our algorithm and use the pin-hole camera model. The following sections will describe the standard (global-shutter) model and our rolling-shutter version. Lens distortions are not considered in this work.

3.1 Pin-hole camera with global shutter

The pin-hole camera model is a simple model which describes how 3D points in the world project onto the image plane. The camera aperture corresponds to a point and no lenses are used to describe the focusing of light. Figure 3.1 shows how a 3D object projects onto an image plane.

The relationship seen in figure 3.1 can be expressed as:

$$\frac{x}{f} = \frac{X}{Z} \tag{3.1}$$

$$\frac{y}{f} = \frac{Y}{Z} \tag{3.2}$$

This relationship, together with a translation of the origin, skew and aspect ratio can also be described in matrix notation using homogeneous coordinates:

$$\begin{pmatrix} \lambda x \\ \lambda y \\ \lambda \end{pmatrix} = \begin{pmatrix} f & s & c_x \\ 0 & f\alpha & c_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \tag{3.3}$$

$$\mathbf{x} = \mathbf{K}\mathbf{X} \tag{3.4}$$

The matrix K contains the *intrinsic* or *internal* camera parameters, and describes the camera. c_x and c_y describe the translation of the principal point required to move the origin into image coordinates. The focal length f, in x and

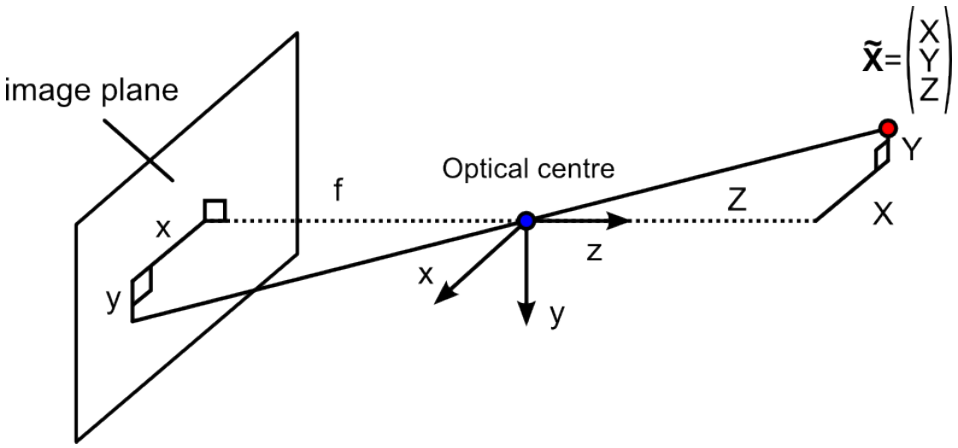

Figure 3.1: The pinhole camera model projects a 3D point $\tilde{\mathbf{X}}$ onto the image plane.

y direction may be different due to the aspect ratio α. The pixels may also be skewed, but in most cases $s = 0$.

Cameras used in this thesis, e.g. the one in iPhone 3GS, have a (near) constant focal length, which enables us to calibrate the camera once. We have also seen that transferring the intrinsic camera parameters between cameras of the same model works well. See section 3.3 for how the parameters are calibrated.

The *extrinsic* or *external* camera parameters describe how the camera relates to a world coordinate system. This relation, or transformation, can be described as a translation $\mathbf{d}$ and a rotation $\mathbf{R}$ and expressed as a matrix multiplication:

$$\mathbf{x} = \mathbf{K}[\mathbf{R}|\mathbf{d}]\mathbf{X}. \tag{3.5}$$

3.2 Pin-hole camera with rolling shutter

When a rolling-shutter camera is stationary and is imaging a rigid scene, the same model as the global-shutter case may be used. The model must however be changed when the camera is moving. The internal camera parameters are still the same (we have fixed focal lengths), but the external parameters are now time dependent. By assuming that the scanning begins at the top row down to the bottom row we get:

$$\mathbf{x} = \mathbf{K}[\mathbf{R}(t)|\mathbf{d}(t)]\mathbf{X}, \tag{3.6}$$

where $t = 0$ represents the first row of the frame.

With this representation we can describe the camera's positions and orientations during a frame capture, and can correct for the geometrical distortions due to the camera motion.

3.2.1 Different motion models

Instead of modelling the full camera motion as the source of the distortions one can simplify the model to three different special cases: pure rotation, pure translation, and imaging of a planar scene. By choosing one of these models the estimation is simplified, which will be described in section 4.2. The pure rotation case assumes that the camera only rotates around the optical centre, which simplifies equation 3.6 to:

$$\mathbf{x} = \mathbf{K}\mathbf{R}(t)\mathbf{X}. \tag{3.7}$$

If the camera is imaging a planar scene the motion can be described by:

$$\mathbf{x} = \mathbf{K}\mathbf{R}(t)(\mathbf{X} + \mathbf{d}(t)) = \mathbf{K}\mathbf{R}(t)\mathbf{D}(t)\tilde{\mathbf{X}}\,, \tag{3.8}$$

$$\text{where} \quad \mathbf{D} = \begin{pmatrix} 1 & 0 & d_1 \\ 0 & 1 & d_2 \\ 0 & 0 & d_3 \end{pmatrix}\,, \tag{3.9}$$

and $\tilde{\mathbf{X}}$ is a three element vector containing the non-zero elements of $\mathbf{X}$, and a 1 in the third position. If the motion is a pure translation, 3.8 simplifies to:

$$\mathbf{x} = \mathbf{K}\mathbf{D}(t)\tilde{\mathbf{X}}\,. \tag{3.10}$$

In paper A we came to the conclusion that the rotation model was the best for hand-held camera motions. When a user holds the camera, the main cause for the motion (and also the cause for the distortions) is rotation. If we only look at changes during a short time interval, e.g. 2-3 frames, the camera does not translate significantly. A notable exception to this is when translation is the dominant component e.g. footage from a moving platform, such as a car.

3.3 Camera calibration

The algorithms in this thesis require calibrated cameras. We use the OpenCV implementation of Zhang's method [15] for camera calibration, which requires a number of images of a planar checkerboard pattern from different orientations. The intrinsic parameters are acquired this way and the lens distortion parameters are neglected.

On a rolling-shutter camera, an additional parameter needs to be estimated also, the readout time. The rolling-shutter chip frame period $1/f$ (where f is the *frame rate*) is divided into a *readout time* t_r, and an *inter-frame delay*, t_d as:

$$1/f = t_r + t_d\,. \tag{3.11}$$

Figure 3.2 shows this relation. The inter-frame delay is useful to know when the continuous camera motion is estimated. For more details on the readout time calibration, see Appendix A in Paper B.

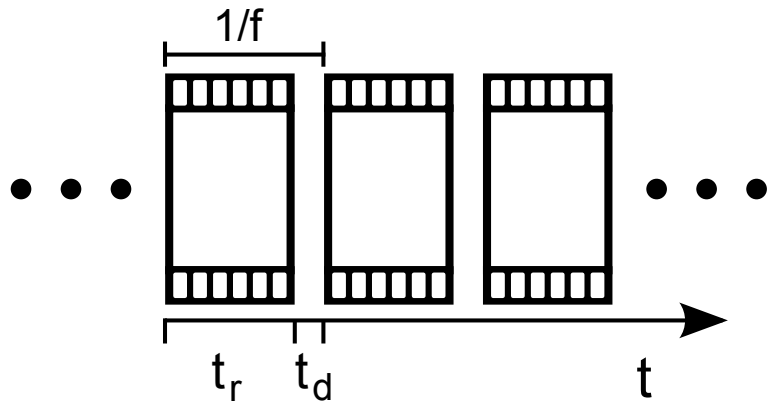

Figure 3.2: Relation between the frame period $1/f$, readout time t_r, and inter-frame delay t_d.

3.4 Push-broom

The push-broom sensor exploits the ego-motion of the moving platform when creating the image. We do however neglect the translational component of the motion and model the distortion of a strip as a sequence of rotation homographies:

$$\mathbf{H}(t) = \mathbf{K}\mathbf{R}(t)\mathbf{K}^{-1}\,, \tag{3.12}$$

This means that we model the sensor as rotating purely about its optical centre and thus the imaged ground patch is modelled as being on the interior surface of a sphere. This will cause some distortions in the reconstruction, but if the radius of the sphere (i.e. the aircraft altitude is large enough (compared to the strip length), this distortion is small.

Chapter 4

Geometric distortion correction

The distortions corrected for in this thesis are those caused by motion of the sensor. This is done by exploiting the continuity of the camera motion in rolling-shutter video. Feature points are detected and tracked across frames and used to estimate the camera ego-motion. The distortions are more severe when shooting video compared to pictures, since the user usually tries to hold the camera steady for pictures. When depth is available, as for the Kinect sensor, the 3D points can also be used to estimate the motion.

Co-alignment of push-broom strips is a bit different since each strip comes from a single flight and we typically only have a few strips (compared to many frames in a video). Also, they might not overlap as much as two consecutive frames in a video, but within each strip the sensor has a continuous motion.

4.1 Point correspondences

For rolling-shutter video we detect points using the good features to track detector [13]. These are then tracked using the KLT-tracker [8] in order to acquire correspondences across frames. The KLT-tracker uses an image patch in one image and estimates the patch position in the next frame. It does so by using a spatial intensity gradient search which minimises the Euclidean distance between the corresponding patches. To be able to cope with large motions we use a scale pyramid approach.

We employ a cross-checking step, as in [1], which uses an additional tracking from the second image back to the first one. Only those points which return to their original position are regarded as inliers. Figure 4.1 shows points rejected using a threshold of 0.5 pixels in red and accepted points in green.

Since push-broom strips are acquired at different times, tracking is difficult to do. Less overlap than between video frames and also larger changes in illumination

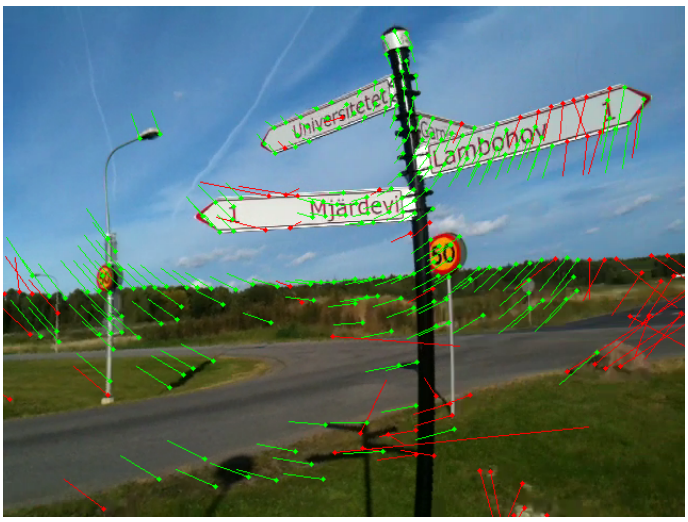

Figure 4.1: Tracked points between two frames. Rejected points in red, and accepted points in green.

makes feature matching a more suitable method for correspondence search than e.g. KLT. We use SIFT features and match them to acquire correspondences for an initial registration of the strips.

4.2 Camera Motion estimation

The sparse point correspondences can be used to estimate the camera motion. The assumption is that the camera is moving in a static scene, so all displacement vectors are due to camera motion.

The camera motion is estimated through iterative non-linear least squares (Levenberg- Marquardt) by minimisation of the cost function associated with the camera motion model.

Since the image rows are exposed at different times, one would like to have the camera pose for each of them. This will result in a high number of parameters to be estimated and we therefore model the motion as a spline. In that way, we only estimate the parameters for a certain number of points along this curve, called *knots*. This spline exploits that the motion is smooth and interpolates all poses between the knots.

4.2.1 Motion parametrisation

In section 3.2 the different motion models were described and for the full model the motion is represented as a sequence of rotations and translations (the knots). The translations are represented as a three element vector and the rotation can be

represented as a 3×3 matrix $\mathbf{R}$, a unit quaternion, or a three element axis angle vector $\mathbf{n}$. During the optimisation the axis angle representation is used since it is a minimal representation of a 3D rotation. Converting from this representation to a rotation matrix is done using the matrix exponent, which for rotations simplifies to Rodrigues formula:

$$\mathbf{R} = \text{expm}(\mathbf{n}) = \mathbf{I} + [\hat{\mathbf{n}}]_x \sin\phi + [\hat{\mathbf{n}}]_x^2(1 - \cos\phi) \tag{4.1}$$

$$\text{where } [\hat{\mathbf{n}}]_x = \frac{1}{\phi}\begin{pmatrix} 0 & -n_3 & n_2 \\ n_3 & 0 & -n_1 \\ -n_2 & n_1 & 0 \end{pmatrix}. \tag{4.2}$$

$\hat{\mathbf{n}}$ is the corresponding unit vector to $\mathbf{n}$, which defines the axis where the rotation is taking place and ϕ is the magnitude of $\mathbf{n}$ which corresponds to the rotation angle around the axis. To convert a rotation matrix back to vector form, the matrix logarithm can be used and for rotations the following closed form exists:

$$\mathbf{n} = \text{logm}(\mathbf{R}) = \phi\hat{\mathbf{n}}\,, \quad \text{where} \quad \begin{cases} \tilde{\mathbf{n}} = \begin{pmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{pmatrix} \\ \phi = \tan^{-1}(||\tilde{\mathbf{n}}||, \text{tr}\mathbf{R} - 1) \\ \hat{\mathbf{n}} = \tilde{\mathbf{n}}/||\tilde{\mathbf{n}}||\,. \end{cases} \tag{4.3}$$

For interpolation of translations we are using a linear interpolation:

$$\mathbf{d}_{\text{interp}} = (1 - w)\mathbf{d}_1 + w\mathbf{d}_2\,, \tag{4.4}$$

where $\mathbf{d}_1$ and $\mathbf{d}_2$ are two translation vectors (three elements) and $w \in [0, 1]$ is the weight parameter. Interpolation of rotations is slightly more complicated due to the periodic structure of SO(3). We use SLERP (Spherical Linear intERPolation) [14] with an interpolation parameter $\tau \in [0, 1]$ between two knot rotations:

$$\mathbf{n}_{\text{diff}} = \text{logm}\left(\text{expm}(-\mathbf{n}_1)\text{expm}(\mathbf{n}_2)\right) \tag{4.5}$$

$$\mathbf{R}_{\text{interp}} = \text{expm}(\mathbf{n}_1)\text{expm}(\tau\mathbf{n}_{\text{diff}}). \tag{4.6}$$

$\mathbf{n}_1$ and $\mathbf{n}_2$ are two rotation axis-angle vectors and $\mathbf{R}_{\text{interp}}$ is the resulting rotation matrix.

4.2.2 Optimisation

By assuming that the row which is exposed first is the top one, the row number is proportional to time. When using the rotation only model, two corresponding homogeneous image points $\mathbf{x}$, and $\mathbf{y}$ are projected from the 3D point $\mathbf{X}$ as:

$$\mathbf{x} = \mathbf{K}\mathbf{R}(N_x)\mathbf{X}\,, \text{ and } \quad \mathbf{y} = \mathbf{K}\mathbf{R}(N_y)\mathbf{X} \tag{4.7}$$

where N_x and N_y correspond to the time parameters, e.g. the row number for point $\mathbf{x}$ and $\mathbf{y}$ respectively. This gives us the relation:

$$\mathbf{x} = \mathbf{K}\mathbf{R}(N_x)\mathbf{R}^T(N_y)\mathbf{K}^{-1}\mathbf{y} = \mathbf{H}\mathbf{y}\,. \tag{4.8}$$

The positions of the knots are discussed in paper B. When these positions have been decided, the rotation from an arbitrary row N_{curr} (relative to the first row in the first image) is acquired by:

$$\mathbf{R} = \texttt{SLERP}(\mathbf{n}_m, \mathbf{n}_{m+1}, \tau)\,, \text{ for} \tag{4.9}$$

$$\tau = \frac{N_{\text{curr}} - N_m}{N_{m+1} - N_m}\,, \quad \text{where} \quad N_m \leq N_{\text{curr}} \leq N_{m+1}, \tag{4.10}$$

and N_m, N_{m+1} are the two neighbouring knot times.

The cost function to be minimised is the summed (symmetric) image-plane residuals of a set of corresponding points $\mathbf{x}_k \leftrightarrow \mathbf{y}_k$:

$$J = \sum_{k=1}^{K} d(\mathbf{x}_k, \mathbf{H}\mathbf{y}_k)^2 + d(\mathbf{y}_k, \mathbf{H}^{-1}\mathbf{x}_k)^2, \tag{4.11}$$

$$\text{where} \quad d(\mathbf{x}, \mathbf{y})^2 = (x_1/x_3 - y_1/y_3)^2 + (x_2/x_3 - y_2/y_3)^2\,. \tag{4.12}$$

Here K is the total number of correspondences between two images. It is also possible to use correspondences from more than two images in the cost function. When using the rotation only model, $\mathbf{H}$ is defined in (4.8), and here it would be beneficial to use a small number of frames per optimisation, in case the motion also includes translations. When using the planar scene model, $\mathbf{H}$ is defined by:

$$\mathbf{H} = \mathbf{K}\mathbf{R}(N_x)\mathbf{D}(N_x)\mathbf{D}(N_y)^{-1}\mathbf{R}^T(N_y)\mathbf{K}^{-1}. \tag{4.13}$$

If the rotations are replaced with the identity matrix, the pure translation case is estimated instead.

If the 3D points $\mathbf{X}$ also are known, as in paper C, the cost function can be defined on these instead, resulting in estimation of the full 6 degrees of freedom camera motion imaging an arbitrary scene. If $\mathbf{X}_1$ and $\mathbf{X}_2$ are two corresponding 3D points reconstructed from two different images and depth maps, they can be transformed to the position $\mathbf{X}_0$. This is the position where the reconstructed point should have been, if it was imaged at the same time as the first row in the first image:

$$\mathbf{X}_0 = \mathbf{R}(N_1)\mathbf{X}_1 + \mathbf{d}(N_1) \tag{4.14}$$

$$\mathbf{X}_0 = \mathbf{R}(N_2)\mathbf{X}_2 + \mathbf{d}(N_2). \tag{4.15}$$

By assuming that the scene is static, the difference between these points can be used to estimate the motion, resulting in the minimisation of:

$$J = \sum_{k=1}^{K} ||\mathbf{R}(N_{1,k})\mathbf{X}_{1,k} + \mathbf{d}(N_{1,k}) - \mathbf{R}(N_{2,k})\mathbf{X}_{2,k} - \mathbf{d}(N_{2,k})||^2, \tag{4.16}$$

where $N_{1,k}$ and $N_{2,k}$ are the rows where the kth 3D point is observed in the first and second image respectively.

4.3 Image rectification

In this thesis image rectification is the process of resampling the input image to a version which looks more rigid. When the camera motion has been estimated, i.e. its pose at the time instances of the knots (which corresponds to a certain image row) the poses for all the image rows can be acquired through interpolation. By using a regular grid on the input image, each row can be transformed by a different homography to create the forward mapping. The coordinate system to be transformed to can be chosen as a specific row, e.g. the one corresponding to the first or middle row of the image. This means that this *reference row* will be exactly the same in the input image and the rectified image. When using a pure rotation as motion model the rectification equation becomes:

$$\mathbf{x}' = \mathbf{K}\mathbf{R}_{\text{ref}}\mathbf{R}^T(N)\mathbf{K}^{-1}\mathbf{x}, \tag{4.17}$$

where $\mathbf{x}$ is the input image coordinate, $\mathbf{x}'$ its rectified position, $\mathbf{R}^T$ the camera's orientation the time instance the pixel was imaged and $\mathbf{R}_{\text{ref}}$ the rotation for chosen reference row.

The forward interpolation of the image can be done in different ways. Delaunay triangulation of the transformed grid, with the input pixels at each vertex together with interpolation of the Barycentric coordinates within each triangle will create a filled output image. This method (griddata in Matlab) is however very slow, and an alternative method is to "smear" each input pixel into a region (e.g. 3×3 closest output grid locations). The output RGB values are updated as (wr, wg, wb, w) together with a weight w, that depends on the grid location $\mathbf{u}$, according to:

$$w(\mathbf{u}) = \exp(-.5||\mathbf{u} - \tilde{\mathbf{x}}'||^2/\sigma^2) \tag{4.18}$$

where σ is a smoothing parameter. After looping over all pixels they are normalised by w, creating an output RGB image. If the camera motion is very fast, a local 3×3 region may not be enough to fill all output pixels and a larger region has to be used. This increases the computation time and a faster method without any risk of holes is to do a mesh warping on a graphics processing unit (GPU). A mesh can be placed on the input image and the GPU transforms each row to their rectified position. Values between rows are automatically interpolated (in hardware) so there is no risk of holes.

If equation 4.17 is reversed, the equation for the inverse mapping becomes:

$$\mathbf{x} = \mathbf{K}\mathbf{R}(N)\mathbf{R}_{\text{ref}}^T\mathbf{K}^{-1}\mathbf{x}'\,. \tag{4.19}$$

It is not possible to use this inverse interpolation correctly, since different pixels within a row should be transformed with different homographies, see figure 4.2. The pixels within a row in the input image do however share the same homography and can be used to correctly transform the image.

If the depth is known, the 3D points can be rectified by:

$$\mathbf{X}' = \mathbf{R}_{\text{ref}}(\mathbf{R}(N)\mathbf{X} + \mathbf{d}(N)) + \mathbf{d}_{\text{ref}}, \tag{4.20}$$

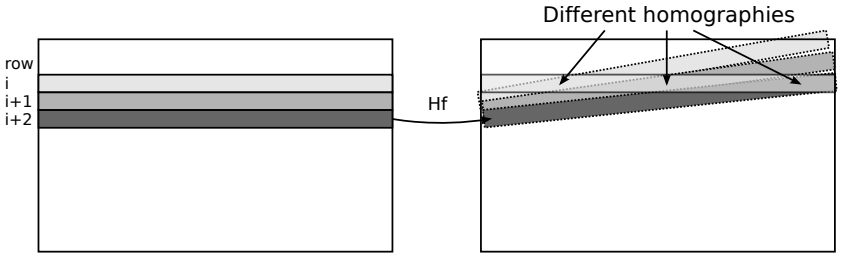

Figure 4.2: Left: Distorted input image. Right: Rectified output image.

where $\mathbf{X}$ is the original distorted 3D points and $\mathbf{X}'$ is the rectified version. Also, if the depth map and video frame are to be rectified, the same procedure as above can be used by projecting the 3D points onto the image plane and do forward interpolation.

4.4 Stabilisation

The rectification technique described in section 4.3 allows for an efficient implementation of video stabilisation. When an image is rectified, all the rows are transformed to a common coordinate system corresponding to the reference row. Instead of transforming each image to e.g. the middle row, one can do a temporal smoothing of all reference rows in the image sequence and use the smoothed versions instead.

Smoothing of rotations can be achieved by matrix averaging:

$$\tilde{\mathbf{R}}_k = \sum_{l=-n}^{n} w_l \mathbf{R}_{k+l} \tag{4.21}$$

where the temporal window is $2n+1$ and w are weights for the input rotations $\mathbf{R}_k$. The output of (4.21) is not guaranteed to be a rotation matrix, but this can be enforced by constraining it to be orthogonal [5]:

$$\begin{gathered} \hat{\mathbf{R}}_k = \mathbf{U}\mathbf{S}\mathbf{V}^T, \text{ where} \\ \mathbf{U}\mathbf{D}\mathbf{V}^T = \text{svd}(\tilde{\mathbf{R}}_k), \text{ and } \mathbf{S} = \text{diag}(1, 1, |\mathbf{U}||\mathbf{V}|). \end{gathered} \tag{4.22}$$

The motion estimation is done during a short frame interval, and since all optimisations have different origins they have to be transformed to a common coordinate system. The pure rotation model will probably not hold for a long video sequence. The stabilisation will be a restriction on the orientation, and there will still be some translation left, but not so much to be disturbing.

Chapter 5

Evaluation

This chapter describes the generated ground-truth dataset, and the methods used for evaluation of the algorithms.

5.1 Ground truth generation

In order to do a controlled evaluation, a synthetic dataset was developed for paper A and extended in paper B. The Autodesk Maya software was used to generate different camera motions in a 3D scene. Rolling-shutter frames were simulated by combing 480 global-shutter frames. One row in each global-shutter frame was used to create a rolling-shutter frame, starting at the top row and sequentially moving down to the bottom row. Figure 5.1 shows different kinds of motions in the scene.

The ground-truth for rolling-shutter rectification is the global-shutter frame. Which global-shutter frame to be used depends on which time instance (i.e. which row) the distorted image is reconstructed to. Global-shutter frames corresponding to the first, middle and last row have been generated. Depending on the motion, some parts of the ground-truth frame (borders and occlusions) are not visible in the rolling-shutter frame. Visibility masks have been generated that indicate which pixels in the ground-truth frames can be reconstructed from the corresponding rolling-shutter frame.

5.2 Evaluation measures

In paper A we compared our rectification to the ground-truth by calculating the average Euclidean distance to the colour pixel values in the ground truth images, within the valid mask. Pixels that deviate more than a certain threshold are counted as incorrect. This measure is however more sensitive in high-contrast regions, than in regions with low contrast. In paper B, we therefore used a variance-

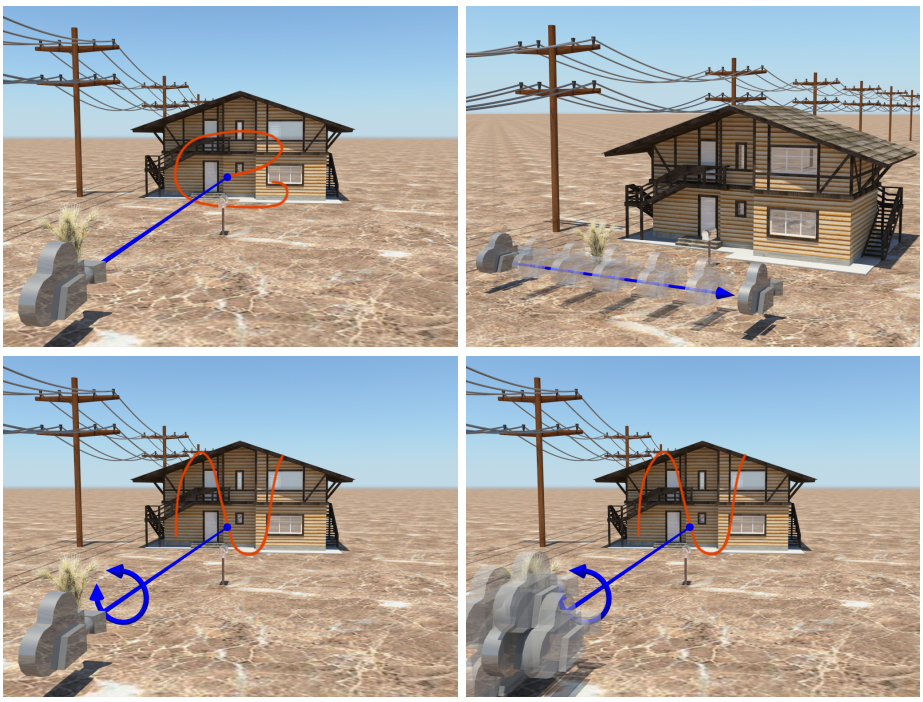

Figure 5.1: The four categories of synthetic sequences. Left to right, top to bottom: #1 rotation only, #2 translation only, # full 3DOF rotation. and #4 3DOF rotation and translation.

normalised error measure:

$$\epsilon(\mathbf{I}_{\text{rec}}) = \sum_{k=1}^{3} \frac{(\mu_k - I_{\text{rec},k})^2}{\sigma_k^2 + \varepsilon\mu_k^2} \,. \tag{5.1}$$

Here μ_k and σ_k are the means and standard deviations of each colour band in a small neighbourhood of the ground truth image pixel (we use a 3×3 region), and ε is a small value that controls the amount of regularisation. This measure also has the benefit of being less sensitive to sub-pixel rectification errors.

Video stabilisation is difficult to evaluate since we both want to reduce the image plane motions and maintain a correct geometry. When no ground-truth is available, one can evaluate image plane motion by comparing consecutive frames in a video with a certain motion. A video from when a person walking forward and holding the camera, will be shaky, but consecutive frames will be very similar if the stabilisation algorithm is good and is thus used as an evaluation measure.

When evaluating the rectification of 3D point clouds, a practical method is to measure geometrical properties of a known object, e.g. comparing the angles between the visible sides of a box, before and after rectification. Ground-truth angles are obtained by imaging the box when the sensor was stationary, see figure

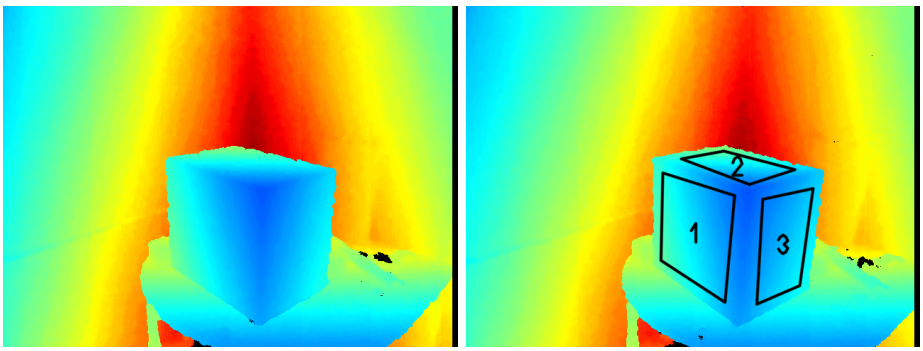

Figure 5.2: Left: Depth frame from a static sensor. Right: Manually marked planes on frame captured during sensor rotation.

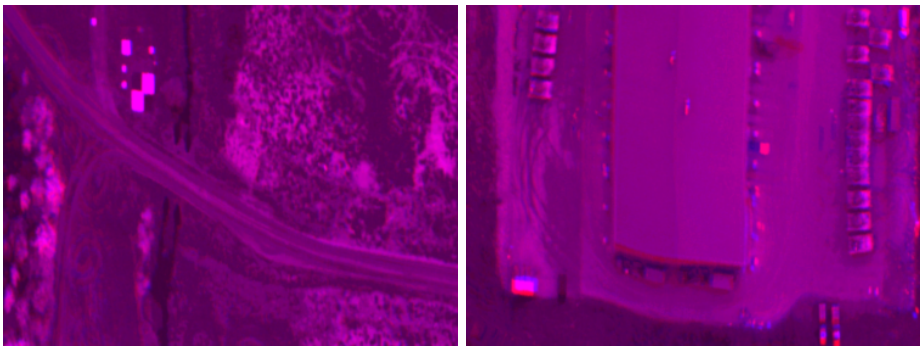

Figure 5.3: Result of co-alignment of push-broom strips. Alignment errors show up as red or blue areas. Correctly aligned pixel get magenta colour.

5.2. The plane angles can be estimated by finding the cube normals using RANSAC [3] and compute the angle between two normals using the formula:

$$\Theta_{k,l} = \sin^{-1}(\|\hat{\mathbf{n}}_k \times \hat{\mathbf{n}}_l\|)\,, \tag{5.2}$$

where $\hat{\mathbf{n}}_k$ and $\hat{\mathbf{n}}_l$ are normal vectors for the two planes.

In paper D and E push-broom data were considered, without any ground-truth. Visual inspection was used to evaluate the registration quality, as it is quite easy to observe, see figure 5.3.

Chapter 6

Concluding remarks

This chapter summarises the main results and discusses possible areas of future work.

6.1 Results

The methods presented in this thesis can be used to increase the usability of rolling-shutter cameras, both for researchers and end users. The main contributions are the development of the three dimensional models for rolling-shutter distortion correction. Paper A was the first paper describing this and gave superior results for hand-held camera motions compared to image-based methods. We also introduced the first rolling-shutter dataset which enables other researchers to evaluate their algorithms. Paper B introduced an efficient video stabilisation method in combination with the image rectification. A new GPU-based forward interpolation was also introduced and the paper extended the motion model to cope with faster motions.

When the Kinect sensor is used on mobile platforms it has to be moved slowly, or in a move-stop-look image acquisition so that the rolling-shutter artifacts are kept at a minimum. With our technique from paper C the data is rectified, and the sensor can be moved in an arbitrary manner.

Paper D and E introduced methods for co-aligning push-broom strips with similar techniques as for the rolling-shutter case, using image only data (paper D) and image data combined with gyroscope measurements (paper E).

6.2 Future work

The image-based motion estimation assumes that the scene is stationary. During evaluation it has been shown that it is robust to some object motion in the video, but if a large part of the optical flow originates from fast moving objects, a motion segmentation (and local rectification) may be required.

It would also be interesting to improve the quality and the temporal resolution

of the motion estimation. Possible ways may be to use higher order splines, use a more dense optical flow, variable knot positions, to model lens distortion and optimisation over a whole sequence. This may enable the algorithm to cope with even faster camera motions, such as when it is attached to a vibrating engine.

Another interesting future work would be an auto-calibration step, since it is quite cumbersome to manually calibrate each different camera model, and also to combine the rectification with some other application such as panorama stitching, augmented reality and so on.

In [6] a video rectification and stabilisation method based on accelerometer and gyroscope measurements was shown to be successful. Future work will be to also try to rectify 3D point clouds generated by the Kinect using these kind of sensors.

The co-alignment of push-broom strips is currently not good enough for an automatic change-detection. A more advanced motion model and possible estimation or incorporation of a height map may be required.

Bibliography

[1] Simon Baker, Daniel Scharstein, J. P. Lewis, Stefan Roth, Michael J. Black, and Richard Szeliski. A database and evaluation methodology for optical flow. In *IEEE ICCV*, Rio de Janeiro, Brazil, 2007.

[2] Michael Bosse and Robert Zlot. Continuous 3d scan-matching with a spinning 2d laser. In *ICRA09*, Kobe, Japan, May 2009.

[3] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24:381–395, June 1981.

[4] Per-Erik Forssén and Erik Ringaby. Rectifying rolling shutter video from hand-held devices. In *CVPR'10*, 2010.

[5] Claus Gramkow. On averaging rotations. *International Journal of Computer Vision*, 42(1/2):7–16, 2001.

[6] Gustav Hanning, Nicklas Forslöw, Per-Erik Forssén, Erik Ringaby, David Törnqvist, and Jonas Callmer. Stabilizing cell phone video using inertial measurement sensors. In *The Second IEEE International Workshop on Mobile Vision*, Barcelona, Spain, November 2011. IEEE.

[7] Johan Hedborg, Erik Ringaby, Per-Erik Forssén, and Michael Felsberg. Structure and motion estimation from rolling shutter video. In *IWMV workshop at ICCV'11*, 2011.

[8] B.D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI'81*, pages 674–679, 1981.

[9] Erik Ringaby, Jörgen Ahlberg, Per-Erik Forssén, and Niclas Wadströmer. Co-alignment of aerial push-broom strips using trajectory smoothness constraints. In *Proceedings SSBA'10 Symposium on Image Analysis*, pages 63–66, March 2010.

[10] Erik Ringaby, Jörgen Ahlberg, Niclas Wadströmer, and Per-Erik Forssén. Co-aligning aerial hyperspectral push-broom strips for change detection. In *Proceedings of SPIE Security+Defence*, volume 7835, Tolouse, France, September 2010. SPIE, SPIE Digital Library.

[11] Erik Ringaby and Per-Erik Forssén. Scan rectification for structured light range sensors with rolling shutters. In *IEEE International Conference on Computer Vision*, Barcelona, Spain, November 2011. IEEE, IEEE Computer Society.

[12] Erik Ringaby and Per-Erik Forssén. Efficient video rectification and stabilisation for cell-phones. *International Journal of Computer Vision*, 96(3):335–352, 2012. http://dx.doi.org/10.1007/s11263-011-0465-8.

[13] Jianbo Shi and Carlo Tomasi. Good features to track. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR'94*, Seattle, June 1994.

[14] Ken Shoemake. Animating rotation with quaternion curves. In *Int. Conf. on CGIT*, pages 245–254, 1985.

[15] Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.

[16] Assaf Zomet, Doron Feldman, Shmuel Peleg, Daphna Weinshall, and Ieee Computer Society. Mosaicing new views: The crossed-slits projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25:741–754, 2003.