Efficient Optimization Techniques for Localization and Registration of Images

Svärm, Linus

2015

Efficient Optimization Techniques for Localization and Registration of Images

Linus Svärm

Lund University

Faculty of Engineering
Centre for Mathematical Sciences
Mathematics

Mathematics
Centre for Mathematical Sciences
Lund University
Box 118
SE-221 00 Lund
Sweden

http://www.maths.lth.se/

Doctoral Theses in Mathematical Sciences 2015:1
ISSN 1404-0034

ISBN 978-91-7623-243-9
LUTFMA-1052-2015

Printed in Sweden by MediaTryck, Lund 2015

Preface

The contents of the thesis is based on the following papers. To comply with the rules for a doctoral thesis, my own contributions are also outlined.

Main papers

- Linus Svärm, Olof Enqvist, Fredrik Kahl, Magnus Oskarsson, "Improving Robustness for Inter-Subject Medical Image Registration using a Feature-Based Approach," *In submission.*

  Olof and I developed the methods. I wrote most of the code, carried out most of the experiments and wrote a small part of the paper.

- Linus Svärm, Olof Enqvist, Magnus Oskarsson, Fredrik Kahl, "Accurate Localization and Pose Estimation for Large 3D Models," *Conference on Computer Vision and Pattern Recognition* (CVPR 2014), Columbus, USA, 2014.

  My supervisors came up with the initial idea that I developed further into a method. I wrote most of the code, carried out the experiments and wrote some parts of the paper.

- Linus Svärm, Olof Enqvist, Fredrik Kahl, Magnus Oskarsson, "City-Scale Localization for Cameras with Known Vertical Direction," *Manuscript to be submitted.*

  My contribution was similar to that of the paper *Accurate Localization and Pose Estimation for Large 3D Models*.

- Erik Ask, Olof Enqvist, Linus Svärm, Fredrik Kahl, Giuseppe Lippolis, "Tractable and Reliable Registration of 2D Point Sets," *European Conference on Computer Vision* (ECCV 2014), Zürich, Switzerland, 2014.

  I started working with this paper relatively late, when most of the theoretical work had already been done. I contributed mainly to the experiments and only in a minor way to the text.

- Linus Svärm and Magnus Oskarsson, "Structure from Motion Estimation with Positional Cues," *18th Scandinavian Conference on Image Analysis* (SCIA 2013), Espoo, Finland, 2013.

  I developed the method, wrote most of the code and most of the paper, and carried out the experiments.

- Linus Svärm and Petter Strandmark, "Shift-map Image Registration," *International Conference on Pattern Recognition* (ICPR 2010), Istanbul, Turkey, 2010.

  Petter and I developed the method together. I wrote most of the code, performed all of the experiments and wrote some minor parts of the paper.

Subsidiary papers

- Sebastian Haner, Linus Svärm, Erik Ask and Anders Heyden, "Joint Under and Over Water Calibration of a Swimmer Tracking System," *4th International Conference on Pattern Recognition Applications and Methods* (ICPRAM 2015), Lisbon, Portugal, 2015.

  I contributed to the idea of the method, wrote minor parts of the code and of the paper.

- Linus Svärm, Zhayida Simayijiang, Olof Enqvist and Carl Olsson, "Point Track Creation in Unordered Image Collections Using Gomory-Hu Trees," *International Conference on Pattern Recognition* (ICPR 2012), Tsukuba, Japan, 2012.

  Olof and I developed the method together. I implemented the algorithms, carried out all of the experiments and wrote a small part of the paper.

Acknowledgements

I would like to express my gratitude to my supervisor Magnus Oskarsson, and to co-supervisors Fredrik Kahl and Olof Enqvist, for a lot of inspiration, guidance and interesting discussions. I would especially like to express my appreciation to Olof Enqvist without whom both this thesis, and my production these years, would have been a lot worse. I would also like to thank my co-authors and colleagues for fun collaborations and an inspiring workplace.

For everything else, I would like to thank my friends and my family.

This work has been financed by the Swedish Foundation for Strategic Research, within the programmes VINST, grant no. RIT08-0043 and ENGROSS, the Swedish e-science program: eSSENCE and the Swedish strategic research support initiative ELLIIT.

Contents

Chapter 1

Introduction

With some effort, it is possible to identify two main topics for the work in this thesis. The first topic is image-based localization, where the aim is to answer the question: Where was this photo taken? To be able to answer, two things are needed. First, you need a model of the world, or at least those parts of it, which are relevant for the application at hand, and secondly you need a method to relate the new image to the model. This thesis presents new algorithms for both parts.

The second part of the thesis is concerned with image registration. The goal of image registration is to find a transformation between two images depicting the same, or similar, objects. The area of focus is registration of two- and three-dimensional medical images.

A main concern during the development of several of the methods presented in the thesis has been the ability to produce reliable results in the presence of high levels of noise and large amounts of outliers in the data. In order to tackle these kinds of problems, recent theoretical advances in optimization have been used and extended.

1.1 Organization of the thesis

To put the thesis in perspective, Chapter 2 goes through some relevant theory and related work. The rest of the thesis is divided into two parts, reflecting the two main topics.

Part I - Localization. The focus of the first part of the thesis is image-based localization, i.e., the problem of localizing a novel image with respect to a model of the scene. The first step to build a localization system, is to construct a model of the geometry and appearance of the scene.

In Chapter 3, a method for building such models is presented that utilizes both image information and positional cues, e.g., GPS-measurements. This method was previously published in [105]. The next chapter, Chapter 4, turns to the localization problem itself. More precisely, it presents a method for image-based localization that also utilizes gravitational measurements. The main contributions are a fast approximate outlier rejection scheme, that enables us to handle large datasets with very large amounts of outliers, and an optimal algorithm for inlier optimization, that runs in polynomial time. This chapter is based on [104]. Chapter 5 discusses some possible extensions to these methods for the case of known camera height.

Part II - Registration. The second part of the thesis is concerned with image registration and starts with a short introduction to the problem of image registration. Next, Chapter 7 presents two methods for registration of 2D-point sets. This work is based on [5]. Chapter 8 considers the registration problem for three-dimensional medical images. This is based on material in submission. The thesis is concluded with a chapter on nonrigid 2D image registration. This is based on [106].

Chapter 2
Background

This chapter introduces some relevant theory and related work to make the rest of the thesis easier to understand and put it in perspective. Sections 2.1-2.4 concerns standard concepts common to computer vision and image analysis while Sections 2.5-2.7 introduce some results from robust model estimation, polynomial equation solving and convex optimization.

2.1 Images, feature points and matching

What is an image? In the computer vision literature, the concept is a floating one and trying to impose a strict definition here is likely to do more harm than good. Sometimes it will be sufficient to view an image as an array or possibly a function from index pairs in $\mathbb{N}^2$ to intensities in $\mathbb{R}$. But working on different resolutions or detecting structures with sub-pixel accuracy, we need the notion of a continuous image. In this work we will move freely between the different notions.

Many computer vision problems are difficult both to formulate and solve if working directly with the image as an array of intensity values. Hence, it is common to reduce the image information into something more abstract. A strategy that is common in computer vision, and in this thesis, is to use something called *point features*. A point feature is often made up of a position in the image, along with a description of the appearance of its neighborhood.

The most frequent use of point features is to identify similar structures in different images. The aim can be to find group similar images, or to find the same physical point in a scene photographed from different angles.

Feature detection. The first step of feature extraction is detection. The purpose of this step is to determine a set of discrete points in the image

for which to extract descriptor vectors. An important property of feature detectors is repeatability, i.e., that the same object or detail in different images is detected. This means that we want features to be well localized, since otherwise we will not be able to pinpoint the same position in both images. We also want to estimate the scale of detected feature, to be able to match objects despite viewing them at difference distance or scale. In many applications, it is also desirable to estimate some kind of characteristic feature orientation, that can be used to achieve rotation invariance.

The Difference-of-Gaussians detector, which is one of the most prevailing ones, has these properties. As the name suggests, interest points are obtained from scale-space extrema of a difference-of-Gaussians operator, as originally proposed in [15]. This produces a set of interest points in scale space. The detected scale is used to achieve scale invariance in feature matching.

Another common feature detector used in this thesis is called Maximally Stable Extremal Regions (MSER), see [76]. This detector was designed to match features under large perspective distortions. A maximally stable extremal region is a stable connected component of level sets in the image. From these regions, it is possible to get scale and orientation estimates of detected features.

Feature description. Now we want to compute descriptors for each feature point. Apart from being highly distinctive, we also often want a descriptor to be invariant to changes in illumination and 3D viewpoint. To facilitate matching of features, it is common to represent their appearance by a vector in a high dimensional space, and then measure similarity by using some distance metric in that space.

One such descriptor is called scale-invariant feature transform (SIFT) [72]. In short, SIFT works by placing a 4-by-4 grid above the feature, covering an area slightly larger than the blob that the detector found. If rotation invariance is desired, the grid is aligned with an estimated dominant direction. In each grid block, an 8-bin histogram of orientations is populated by image gradient measurements. Finally, the bin values from all histograms are stacked together, resulting in a $4 \cdot 4 \cdot 8 = 128$ dimensional feature vector.

Speeded Up Robust Features (SURF) [7] is another feature descriptor. It has some similarities to SIFT, but codes gradient information in a

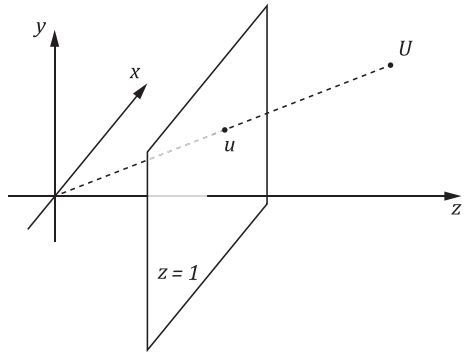


Figure 2.1: The coordinate system of a (calibrated) pinhole camera.

different way.

Feature matching. To measure similarity of features, a metric is used in feature space; often simply the Euclidean distance. When trying to find matching features between two images, for each point in one image, we find the point in the other image with smallest distance and pose this as a tentative match, or *correspondence*.

This strategy leads to a lot of wrong matchings. To remedy this, Lowe [73] proposed a simple criterion to filter out probable miss-matches: Reject a match if the ratio between the distance to the closest point, and the distance to the next closes point, is greater than some threshold (Lowe uses 0.8). This strategy has proved very effective and is widely used. Still, a lot of miss-matches are not caught by this criterion, and will need to be handled further down the pipeline.

2.2 Camera modelling

In this thesis we will use the pinhole-camera model. One important assumption of this model is that all light rays pass through one common point, the focal point of the camera. This point will serve as the camera centre. The image is generated by light from the scene being projected onto a plane in the camera we call the image plane.

Let a point, U, in the scene have the coordinates $(U_x, U_y, U_z) \in \mathbb{R}^3$.

The camera is placed at origin, looking down the positive z-axis. In real cameras the image plane usually lies behind the focal point. To get an upright image, we instead place the image plane in front of the camera, in the plane $z = 1$. The projection of point U in this image plane is given by the intersection of the image plane, and the line passing through U and the camera centre. This line can be described by the equation

$$v = \gamma U. \tag{2.1}$$

The intersection with the image plane is given by the condition $v_z = 1$. The final relation between the 3D-point U and the image point u is usually written as

$$\lambda u = U, \tag{2.2}$$

where λ is chosen such that $u_z = 1$.

In the general case, the 3D-points are not given in the coordinate system of the camera, meaning that their coordinates first need to be transformed to the camera coordinate system before the projection is performed. If the camera is positioned at a point C, with an orientation described by the rotation matrix R, the camera equation stated in matrix form is

$$\lambda u = R(U - C). \tag{2.3}$$

All u satisfying this equation with $\lambda > 0$ are valid representations for the image point. Instead of placing an image plane at a given z-coordinate, it is also common to choose u such that $\|u\|_2 = 1$. This yields an image sphere, rather than an image plane.

2.2.1 Camera calibration

So far we have been studying what is called a calibrated camera. For real cameras, the image is usually given in pixels with a coordinate system originating in one corner of the image, and with a focal length (distance between image plane and focal point) different from 1. The change of coordinate system between pixel coordinates and calibrated coordinates is a transformation $x \to Kx$, where K is called the calibration matrix. In its

simplest form it contains only scaling and translation and can be written as

$$K = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.4}$$

where f is the focal length of the camera and (p_x, p_y) is the principal point (the point on the image plane intersected by the optical axis of the camera). For some cameras, the above calibration matrix is not enough. If a camera has non-square pixels (*aspect ratio* $\alpha \neq 1$), or if the coordinate system is not orthogonal, with skew s, the calibration matrix is given by

$$K = \begin{bmatrix} f & s & p_x \\ 0 & \alpha f & p_y \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.5}$$

We can now write the camera equation for an uncalibrated camera as

$$\lambda u = KR(U - C). \tag{2.6}$$

If we instead use homogeneous coordinates for the 3D-points, $\hat{U} = \begin{bmatrix} U & 1 \end{bmatrix}^T$, we can rewrite the camera equation more compactly as

$$\lambda u = P\hat{U}, \tag{2.7}$$

where $P = \begin{bmatrix} KR & -KRC \end{bmatrix}$.

The pinhole-camera model is not entirely adequate, not even for many ordinary cameras. Modern cameras use lens systems to allow more light to reach the image plane. These lenses give rise to distortions that we are unable to model in a satisfactory way using linear models. However, often the only distortion that we cannot neglect is so called radial distortion [44]; distortion that only depends on the pixels' distance from the principal point. Often however, also radial distortion is weak enough to be neglected without causing any problems.

With adjustments for radial distortion, the pinhole camera model is still accurate enough for most computer vision tasks. For the work in this thesis it is enough to know that with known calibration matrix and known radial distortion, we can transform the images to fit the calibrated camera model (2.3) sufficiently well.

2.3 Scene estimation from images

A classical problem in computer vision is the problem of estimating the structure of a scene from a set of images. In this thesis, we consider the scene structure to be a set of points with appearance and position in $\mathbb{R}^3$. It may seem limiting compared to dense 3D models with textured surfaces, but sparse models are important both in themselves and as a first step in building dense models.

Typically, the sparse 3D points comes from feature detectors such as SIFT, which is described shortly in Section 2.1. A central notion is that of point correspondences; we say that image points in different images correspond if they are projections of the same 3D point, and that an image point corresponds to a 3D point if it is a projection of this point.

In the general case, no information is assumed about the cameras. We assume that we have $m \cdot n$ feature points u_{ij} from n unknown cameras P_i which are projections of m unknown scene points U_j. The so-called structure-from-motion problem is to estimate the unknown cameras P_i and scene points U_j, such that

$$\lambda_{ij} u_{ij} = P_i U_j \tag{2.8}$$

is satisfied for all i and j.

The most common way in which the structure-from-motion problem has been solved is by solving a sequence of subproblems. A few of the major subproblems are (see [44] for details)

- Given corresponding points in two images, estimate the relative motion of the cameras.
- Given that a point is seen in at least two known cameras, *triangulation* is the process of estimating the 3D-coordinates for this point.
- If a camera sees several points with known 3D-coordinates, the problem of estimating the camera's position and orientation relative to the points is known as the *camera pose problem*.

A common approach to SfM is the following strategy (with a much simplified description). Start by estimating the relative motion between two cameras, and triangulate all corresponding image points to get an initial set of scene points. Now iterate the three following steps:

1. Find a new image with many correspondences to the estimated scene points, and estimate the pose of this camera with respect to the scene.

2. Triangulate more scene points made possible by the added camera.

3. Update the entire solution using local refinement. More on this below.

This strategy is called *sequential structure from motion*. It has been used to create impressive results. For example, the software BUNDLER [96] has been used to create reconstructions in the order of 10^5 images.

There are also non-sequential approaches to structure from motion [75, 114]. Typically these start by estimating the orientations of all cameras. The remaining problem can be posed as a convex optimization problem [43, 51, 54].

2.3.1 Local refinement

For many geometric vision problems it is common to refine the solution using local optimization methods. The rationale for this is that we are often lacking tractable methods for finding the global optimum. Also, since we already have a solution that hopefully is not too far away from the optimal solution, local refinement methods might help us to find the solution, or at least a better one than we already have.

For reasons that will be discussed later, it is widely accepted that it is a good idea to try to find a solution that minimizes the sum of squared reprojection errors, i.e., the difference between measured image points and the reprojections of the corresponding 3D-points. Thus, the minimization is performed using non-linear least-squares algorithms. In the computer vision community, this is often called *bundle adjustment*, [109]. The most popular for geometric vision problems seems to be the Levenberg–Marquardt algorithm, [74]. One reason for this is the damping scheme used, resulting in an ability to quickly converge from a wide range of initial guesses.

2.4 Outliers and noise

In the last section we assumed that we were given image point correspondences. In practice these correspondences comes from matched features and many of them will be wrong, i.e., the corresponding points do not represent the same physical point, or they do not fit the model that we are using to explain the data. We call such correspondences *outliers*.

There is also more moderate noise that needs to be considered. Assume that we have n data points, or measurements, x_i. When fitting a model to these points, each measurement will cause some error (that can be zero), depending on how well the measurements fits the model. We let *residual functions* $r_i(\theta) : D \to \mathbb{R}_+$, describe how these fitting errors depend on the model parameters $\theta \in D$.

We wish to find model parameters that minimize some function of these residuals. How we wish to weight the different residual errors depends on the problem and on assumptions about the noise we have. We call these weighting functions *loss functions*.

Problem 2.1. *Given a set S of residual functions, and a loss function ℓ, we wish to find a model θ such that*

$$\sum_{r_i \in S} \ell(r_i(\theta)) \tag{2.9}$$

is minimized.

In terms of residual functions, we define outliers as

Definition 2.2. *Given a model θ, and a threshold ϵ, we define a residual r_i to be an inlier if $r_i(\theta) \leq \epsilon$, and to be an outlier if $r_i(\theta) > \epsilon$.*

Remark 2.3 When we talk about outliers or inliers without specifying a model θ, we implicitly value the measurements with regard to some optimal model (which may be unknown to us).

The simplest example of a robust loss function is the one that count the number of outliers, also called the zero-one loss function. It is defined as

$$\ell(r) = \begin{cases} 0 & \text{if } r \leq \epsilon, \\ 1 & \text{if } r > \epsilon. \end{cases}$$

If we have no outliers, and the measurement errors follow the same Gaussian distribution, then the maximum likelihood solution is given by minimizing the L_2-norm of the residual functions. That is, the loss function we use in this case is the function $\ell(r) = r^2$.

Let us look at an example from the previous section.

Example 2.4.1. *Noise comes mainly from imperfect feature point measurements, giving us a small error in the position of the image points. In the presence of such measurement errors, we can no longer expect the reprojection of a scene point U_j to align perfectly with its corresponding image point measurement u_{ij}. Assuming a calibrated camera, we instead get a reprojection error*

$$r_{ij} = \left\| \frac{(R_i^{(1)}(U_j - C_i), R_i^{(2)}(U_j - C_i))}{R_i^{(3)}(U_j - C_i)} - u_{ij} \right\|, \tag{2.10}$$

where C_i is the camera centre, and $R_i^{(k)}$ is the kth row of the camera rotation matrix. If we collect all camera- and point-parameters in θ, the structure-from-motion problem with L_2-norm loss can be stated as

$$\underset{\theta}{\text{minimize}} \sum r_{ij}^2(\theta). \tag{2.11}$$

By using the L_2-norm loss function we are putting large emphasis on points with large errors. This means that we are trying to find a solution that prevents the error of these points from getting even larger, at the expense of all other points. The result is that we are very sensitive to outliers.

For many geometric-vision problems, it is a common and reasonable assumption that there exist *correct* but noisy point correspondences as well as complete mismatches or outliers [9]. The errors in the positioning of correct correspondences follow approximately a normal distribution, whereas the residuals of outliers are uniformly distributed. To find a maximum likelihood estimate given this distribution, a loss of the following type

$$\ell(r) = -\log\left(c_1 + \exp\left(-r^2/c_2\right)\right) \tag{2.12}$$

should be minimized, where r is the residual error for one correspondence and the constants depend on the amount of inlier noise as well as on the rate of outliers; see Fig. 2.2.

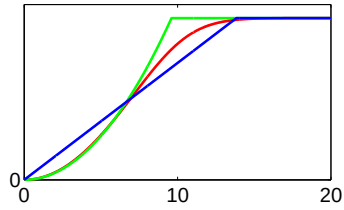


Figure 2.2: The robust loss function (red) suggested in [9], the truncated L_2-error (green) and the truncated L_1-error (blue).

An approximation which is commonly used is obtained by truncating the squared error,

$$\ell(r) = \begin{cases} r^2 & \text{if } r \leq \epsilon, \\ \epsilon^2 & \text{otherwise.} \end{cases}$$

However, the quality of this approximation depends on the rate of outliers in data. At higher rates the loss function levels out much more slowly. In this case a truncated L_1-loss, given by

$$\ell(r) = \begin{cases} r & \text{if } r \leq \epsilon, \\ \epsilon & \text{otherwise,} \end{cases}$$

can be a more appropriate choice, see Figure 2.2. Another advantage is that the truncated L_1 loss is not as sensitive to the choice of ϵ. The truncated L_1-norm does not produce maximum likelihood estimates for any particular noise distribution. Rather than that, it is motivated by the fact that it has been shown to work well over a wide range of distributions and that it is robust to arbitrary changes in a large portion of the data. This is exactly the meaning of the word robust in the field of robust statistics.

2.4.1 RANSAC

A popular, and often good enough, way of handling the presence of outliers is a technique called Random Sample Consensus (RANSAC), [33].

The guiding idea is the following: If a model is estimated from measurements where outliers are present, the result will almost certainly be poor. Further, if we use only a small subset of the measurements, the risk

of it containing an outlier, and thus getting a poor model, is reduced. The outline of the algorithm is as follows:

1. Select at random a small subset of measurements and solve the problem using only these.

2. Count the number of measurements which are inliers with regard to the model estimated in step 1. These form the so called *consensus set*.

3. Iterate step 1 and 2 a number of times. Store the model giving the largest consensus set.

Since using fewer measurements decreases the risk of selecting an outlier, we want to use as few as possible in each iteration. From a combinatorial viewpoint, it is advantageous to use so called *minimal solvers*, i.e., solvers that use the smallest possible number of measurements for model estimation. For example, a minimal solver for calculating the relative pose between two fully calibrated cameras requires five correspondences, [99].

Note that although we only talked about counting the number of inliers above—corresponding to the zero-one loss—it is possible to use RANSAC with an arbitrary loss functions. Instead of counting the inliers, just calculate the loss for each hypothetical model, and save the one with smallest loss.

Given an estimated rate of outliers, it is simple to calculate how many iterations are required to find an outlier-free subset with some probability. In practice however, because of noise, one is often required to find and test several outlier-free subsets.

Numerous versions of the original RANSAC algorithm have been published, with different advantages and weaknesses. See [103] for overview and performance comparisons.

2.5 Robust model estimation

The RANSAC algorithm presented in the previous section normally produces high-quality solutions, but theoretically, there is no guarantee that it will find the optimal solution to Problem 2.1 for any of the loss functions that we have discussed. In recent years methods capable of doing this have

been introduced, at least for the zero-one loss and the truncated L_2 loss. As these methods will be used in the thesis, this section goes through some theory and algorithms. For details, and proofs of results presented here, see [30].

The notation is the same as in the previous section. We want to estimate model parameters θ living in some d-dimensional manifold D, minimizing

$$\sum_{r_i \in S} \ell(r_i(\theta)), \tag{2.13}$$

where ℓ is the truncated L_2 loss. The following observation will be useful. Let θ^* be the optimal solution minimizing

$$\sum_{r_i \in S} \min(\epsilon^2, r_i^2(\theta)), \tag{2.14}$$

and let I^* be the index set for all inlier residuals to this solution. Then θ^* will also be the optimal solution to

$$\underset{\theta}{\text{minimize}} \sum_{i \in I^*} r_i^2(\theta). \tag{2.15}$$

To see this, assume to the contrary that there exists a θ' such that

$$\sum_{r_i \in I^*} r_i^2(\theta^*) > \sum_{r_i \in I^*} r_i^2(\theta'). \tag{2.16}$$

Let O^* denote the residuals not in I^*. If we add $\epsilon^2 |O^*|$ to (2.16), the left-hand side becomes the total loss at θ^*,

$$\sum_{r_i \in S} \min(\epsilon^2, r_i^2(\theta^*)) > \sum_{r_i \in I^*} r_i^2(\theta') + \epsilon^2 |O^*| \tag{2.17}$$

$$\geq \sum_{r_i \in S} \min(\epsilon^2, r_i^2(\theta')), \tag{2.18}$$

which is a contradiction. This is also true for the truncated L_1 loss function.

This observation immediately suggests a strategy for finding the optimal solution to (2.14), or the corresponding problem with truncated L_1 loss, see Algorithm 1. If we can find a way of enumerating all inlier sets, the strategy means that we can solve both problems using the truncated norms, as well as minimizing the number of outliers.

Algorithm 1 Strategy for minimizing truncated L_2

For each possible set of inliers I,
Solve $\theta^* = \underset{\theta}{\operatorname{argmin}} \sum_{i \in I} r_i^2(\theta)$
Evaluate θ^* on (2.14) and update the best model.

2.5.1 Enumerating all inlier sets

Algorithm 1 requires us to be able to enumerate all possible inlier sets. But how do we find them? And how many are there? To investigate, we start with the following observation.

By Definition 2.2, each parameter vector, θ, induces a partition of the residuals, into inliers, I, and outliers, O. Conversely, we see that each residual, $r_i(\theta)$, partitions the parameter space into two sets; one set with all θ such that $r_i(\theta)$ is an inlier, and one set with those making it an outlier. The following definition will be useful.

Definition 2.4. *Given a partition of the set of residuals into the subsets I and O, we let $D(I, O)$ denote the set of parameter vectors $\theta \in D$ such that the residuals in I are inliers and the residuals in O are outliers.*

We will only be interested in *feasible* partitions. That is, partitions where $D(I, O)$ is non-empty. Henceforth we will always assume that the partitions we work with, and search for, are feasible partitions.

An example of how one residual partitions the parameter space into two such sets is shown in the left of Figure 2.3. To the right in the same figure, an example is shown of how the complete set of residuals together partition the parameter space into many regions. Since all parameter-values θ in one such region induce the same inlier-outlier partition, it is sufficient to find one parameter value representing this region.

Assume that we have a partition (I, O). To find one representative $\theta \in D(I, O)$, we formulate and solve a dummy problem, see Definition 2.5. Both the constructed problem and its goal function, f, merely serves as an analytical tool when deriving an algorithm.

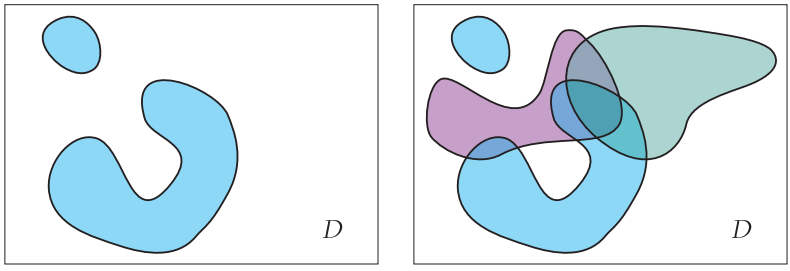


Figure 2.3: *Left:* Each residual divides the parameter space into two parts. One consisting of all points θ that makes the residual an inlier, and the complement making it an outlier. *Right:* All residuals together divides the parameter space into a complex map. To be sure to find all feasible inlier-outlier partitions, we want to find one θ for each of these areas.

Definition 2.5. *Given a differentiable function, f, and a partition (I, O) of the residuals in S, we define* DUMMY*(I,O) as the optimization problem:*

$$\underset{\theta}{\text{minimize}}\ f(\theta) \quad s.t. \quad \theta \in \bar{D}(I, O), \tag{2.19}$$

where $\bar{D}(I, O)$ is the closure $D(I, O)$.

The following technical condition will be useful.

Condition 2.5.1. *(a) The domain D is a d-dimensional differentiable manifold embedded in $\mathbb{R}^m$, with $m-d$ polynomial constraints, $h_j(\theta) = 0$.*

(b) The residual constraints $r_i(\theta) \leq \epsilon$ can be written as $g_i(\theta) \leq 0$, where g_i are polynomials in $\mathbb{R}^m$.

(c) The goal function, f, is a polynomial such that $f \to \infty$ when $|\theta| \to \infty$.

Henceforth, we will work with the polynomials g_i, instead of the residuals r_i. We do this since the residuals not necessarily needs to be polynomials. To simplify notation we also introduce

$$s_i = \begin{cases} 1 & \text{if } i \in \text{ind}(I), \\ -1 & \text{otherwise,} \end{cases}$$

where ind(I) is the index set for the set of residuals I. Now we can rewrite DUMMY(I, O) as

$$\begin{array}{ll} \underset{\theta}{\text{minimize}}\ f(\theta) \quad s.t. & \\ s_i g_i(\theta) \leq 0 & i = 1, \ldots, n \\ h_j(\theta) = 0 & j = 1, \ldots, m - d. \end{array} \tag{2.20}$$

2.5.2 Critical points

As we will see, the points we are looking for in parameter space are exactly those we call *critical* points.

Definition 2.6. *A parameter vector $\theta \in D$ is* critical *to a set of residuals $B \subset S$ if*

$$g_i(\theta) = 0 \quad \forall g_i \in B \tag{2.21}$$

and

$$\{\nabla g_i(\theta) : g_i \in B\} \cup \{\nabla f(\theta), \nabla h_1(\theta), \ldots\} \tag{2.22}$$

is linearly dependent and that there is no proper subset of B with this property.

Lemma 2.7. *If $\theta \in D$ is critical to a set of residuals, B, then $|B| \leq d$, where d is the dimension of D.*

We will also need so-called FJ-points. An FJ-point of an optimization problem is a point satisfying the Fritz John conditions for local optimality. For the optimization problem 2.20, a feasible point θ is an FJ-point if there exists a non-trivial solution, $\mu_i \geq 0$ and $\mu_i g_i(\theta) = 0$ for all i, to

$$\mu_0 \nabla f(\theta) + \sum_i \mu_i s_i \nabla g_i(\theta) + \sum_j \lambda_j h_j(\theta) = 0. \tag{2.23}$$

Theorem 2.8. *Let (I, O) be a feasible partition av the residuals, and assume that Conditions 2.5.1 are satisfied. Then we have:* (i) DUMMY(I, O) *has at least one FJ-point, and* (ii) *this point is critical to a set of residuals B of size $\leq d$.*

For a proof of this theorem, see [30].

2.5.3 Finding the critical points

Theorem 2.8 gives us a link between possible partitions and critical points. The strategy for finding all partitions will be to first find all critical points, then for each of these points find the unique partition induced by the point.

The theorem also gives us a way of finding all critical points. Since each point in which we are interested is critical to a set of residuals, B, where $|B| \leq d$, we can simply go through all subsets of the residuals $\leq d$. The number of such subsets are

$$\sum_{k=0}^{d} \binom{n}{k} = \mathcal{O}(n^d), \tag{2.24}$$

where d is the dimension of the parameter manifold.

If $|B| = d$, the critical points can be found by solving the following system of polynomial equations

$$\begin{aligned} g_i(\theta) &= 0 \quad \forall g_i \in B \\ h_j(\theta) &= 0 \quad j = 1, \ldots, m-d. \end{aligned} \tag{2.25}$$

From Bezout's theorem we know that the number of solutions is either infinite, or bounded by the product of the polynomials' degrees [37]. In our applications the measurements are typically affected by random, real-valued noise making the probability of degeneracy practically negligible.

If $|B| < d$, we are still able to find the critical points by adding the constraint that

$$\{\nabla g_i(\theta) : g_i \in B\} \cup \{\nabla f(\theta), \nabla h_1(\theta), \ldots\} \tag{2.26}$$

is linearly dependent. These constraints can be stated using determinants, resulting in polynomial equations. By using a suitable choice of goal function f, we will get a finite set of equations. We make this a condition.

Condition 2.5.2. *For each subset $\leq d$ of residuals, the number of critical points are finite.*

2.5.4 Enumerating feasible partitions

Theorem 2.8 links feasible partitions of the residuals to critical points. We have seen how we can find all critical points. Now we would like to use

these points to find all feasible partitions. To be able to keep the exposition simple, we will here ignore (rare) degenerate cases. These are also solvable, but will not be covered here, see Appendix B in [30]. To make sure that we have no degenerate cases we add the following conditions.

Condition 2.5.3. *If θ is a critical point to the set of residuals, B: (a) then there are no active residuals outside of B. (b) then the gradients to the active constraint are linearly independent.*

This condition is essentially used to ensure that constraints are in some sense random, that we have no degeneracies.

Previously we have seen that for a partition (I, O), there exists an FJ-point which is critical to a subset, B, of the residuals, where $|B| \leq d$. Now we wish to solve the following problem. Given a θ^* critical to a set B, find all partitions (I, O) such that θ^* is an FJ-point to $\text{DUMMY}(I, O)$. As we will see, this will only by one unique partition if Condition 2.5.3 is satisfied.

Step 1. For θ^* to be an FJ-point, it must be feasible, that is $\theta^* \in \bar{D}(I, O)$. If Condition 2.5.3 is satisfied, we know that $g_i(\theta^*) \neq 0$ for residuals which are not in B. This means that we can check if each residual i not in B is an inlier or outlier by checking if $g_i(\theta^*) < 0$ or if $g_i(\theta^*) > 0$.

Step 2. Since θ^* is critical, we know that

$$\gamma_0 \nabla f(\theta^*) + \sum_{i \in \text{ind}(B)} \gamma_i \nabla g_i(\theta^*) + \sum_{j}^{m-d} \lambda_j \nabla h_j(\theta^*) = 0, \tag{2.27}$$

where both γ_i and λ_j may be negative. According to Condition 2.5.3(b), the gradients are linearly independent. This means that γ_0 cannot be zero. After rescaling, we can write the equation as

$$\nabla f(\theta^*) + \sum_{i \in \text{ind}(B)} \gamma_i \nabla g_i(\theta^*) + \sum_{j}^{m-d} \lambda_j \nabla h_j(\theta^*) = 0. \tag{2.28}$$

From Definition 2.6 we know that all but one the ∇g_i's create a linearly independent set. This means that all γ_i's are non-zero, and all coefficients are uniquely determined. Thus, we can calculate all gradients and then determine the unique solution for the γ_i's and λ_i's.

Step 3. Equation (2.28) is very similar to the criteria for an FJ-point. The sign of γ_i tells us if residual i is an inlier (γ_i is positive) or an outlier (γ_i is negative). If we for $\gamma_i < 0$ let $s_i = -1$, and for $\gamma_i > 0$ let $s_i = 1$, equation (2.28) can be written as

$$\nabla f(\theta^*) + \sum_{i \in \text{ind}(B)} |\gamma_i| s_i \nabla g_i(\theta^*) + \sum_{j}^{m-d} \lambda_j \nabla h_j(\theta^*) = 0. \tag{2.29}$$

Comparing with the criteria for an FJ-point, (2.23), we see that θ^* is an FJ-point to DUMMY(I, O). Algorithm 2 describes how all feasible partitions can be enumerated.

Algorithm 2 Enumerating all feasible partitions

For each subset $B \subset S$ of residuals, with $|B| \leq d$,
  Compute all critical points to B.
  For each critical point θ^*,
    Set $I = \emptyset$.
    For each $g_i \notin B$,
      If $g_i(\theta^*) < 0$, add r_i to I.
    Compute the γ_i's of (2.28).
    For each $\boldsymbol{g}_i \in B$,
      If $\gamma_i > 0$, add r_i to I.
    Set $O = S \setminus I$, and store partition (I, O).

Theorem 2.9. *If Conditions 2.5.1-2.5.3 hold, Algorithm 2 finds all partitions in $\mathcal{O}(n^{d+1})$-time, where n is the number of measurements.*

Proof. There are $\mathcal{O}(n^d)$ subsets $B \subset S$, with $|B| \leq d$. For a given problem, the number of critical points to each B is independent on n, thus they are $\mathcal{O}(1)$. For each critical point θ^* we need to calculate $g_i(\theta^*)$ for $i = 1, \ldots, n$. This results in a complexity $\mathcal{O}(n^{d+1})$. □

2.5.5 Minimizing the number of outliers

Using Algorithm 2 for finding the number of partitions automatically allows us to find the solutions which minimizes the number of outliers.

However, if our goal is to minimize the number of outliers there is a simpler strategy that we can use, see Algorithm 3.

Algorithm 3 Enumerating all feasible partitions

For each subset $B \subset S$ of residuals, with $|B| \leq d$,
  Compute all critical points to B.
  For each critical point θ^*,
    Count the outliers $g_i(\theta^*) > 0$.
    If this is the smallest number so far, store θ^*.

Theorem 2.10. *If Conditions 2.5.1-2.5.3 are satisfied, Algorithm 3 finds the solution θ^*, minimizing the number of outliers, in $\mathcal{O}(n^{d+1})$-time.*

2.5.6 Model estimation under truncated L_1- and L_2-norms

As we have seen earlier, both truncated L_1 and truncated L_2-norm can be minimized if we have a way of finding all feasible partitions of the residuals. Now we know how to accomplish this. As we also have seen, the parameter vector θ^* minimizing truncated L_2-norm for the problem is the same solution that minimizes ordinary L_2-norm for the inlier-set to the optimal solution. Thus, if we can minimize ordinary L_2, or L_1, we can now solve their truncated counterpart using Algorithm 4.

Algorithm 4 Minimizing truncated-L_2 (or L_1)

Compute all possible inlier-outlier partitions, using Algorithm 2.
For each inlier set,
  Compute the optimal L_2-solution.
  If this is the best solution so far, store it.

Remark. The method presented here is, of course, not suitable for all kinds of problems. One requirement is that we are able to formulate problem and conditions as polynomials. Another one is that the method only is practically useful for lower-dimensional problems (typically 2,3 and 4).

2.6 Solving systems of polynomial equations

Often, it is possible to formulate minimal solvers as systems of polynomial equations. In this section, we present a technique for solving such systems using the so-called action matrix. The action matrix may be seen as a multivariate extension of the companion matrix, which is used to find the roots of univariate polynomials, see [23].

We begin by showing how the companion matrix allows us to use tools from linear algebra and matrix theory to find the roots of polynomials of one variable. Consider the polynomial of one variable

$$h(x) = x^n + c_{n-1}x^{n-1} + c_{n-2}x^{n-2} + \cdots + c_1 x + c_0, \tag{2.30}$$

of degree n, with coefficients c_i, $i = 0 \ldots n-1$. To find the roots, we use the observation that

$$x x^{k-1} = x^k, \tag{2.31}$$

and that for $h(x) = 0$ we have that

$$x x^{n-1} = x^n = -c_{n-1}x^{n-1} - c_{n-2}x^{n-2} - \cdots - c_1 x - c_0. \tag{2.32}$$

Introducing the vector $\mathbf{b} = [x^{n-1} x^{n-2} \cdots x 1]^T$, we can write these relations in matrix form as

$$\underbrace{\begin{pmatrix} -c_{n-1} & -c_{n-2} & \cdots & -c_1 & -c_0 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \\ 0 & 0 & \cdots & 1 & 0 \end{pmatrix}}_{\mathbf{c}} \underbrace{\begin{pmatrix} x^{n-1} \\ x^{n-2} \\ \vdots \\ x \\ 1 \end{pmatrix}}_{\mathbf{b}} = \underbrace{\begin{pmatrix} x^n \\ x^{n-1} \\ \vdots \\ x^2 \\ x \end{pmatrix}}_{x\mathbf{b}}. \tag{2.33}$$

This equation is valid for all solutions to our original problem. Thus, for such an x we have that $xb = \mathbf{c}b$, i.e., x is an eigenvalue to $\mathbf{c}$. The matrix $\mathbf{c}$ is called the companion matrix. One way to find the roots of a univariate polynomial is thus to calculate the eigenvalues of its corresponding companion matrix.

2.6.1 Generalization to system of multivariate polynomials

We will now show how the method can be generalized to systems of multivariate polynomials, which was first presented by Lazard in 1981 [63]. First, we need some definitions and results from algebraic geometry.

Definition 2.11. *A monomial in* $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ *is a product of the form*

$$x_1^{\alpha_1} x_2^{\alpha_2} \ldots x_n^{\alpha_n}, \tag{2.34}$$

where α_i *are non-negative integers.*

We will use the notation

$$\mathbf{x}^{\alpha} = x_1^{\alpha_1} x_2^{\alpha_2} \ldots x_n^{\alpha_n}, \tag{2.35}$$

with $\alpha = (\alpha_1, \ldots, \alpha_n)$. The *total degree* of a monomial is the sum

$$|\alpha| = \alpha_1 + \alpha_2 + \cdots + \alpha_n. \tag{2.36}$$

Definition 2.12. *A polynomial* f *in* $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ *is a finite sum of the form*

$$f(\mathbf{x}) = \sum_{\alpha} c_\alpha \mathbf{x}^{\alpha}, c_\alpha \in \mathbb{C} \tag{2.37}$$

The set of all such polynomials is denoted by $\mathbb{C}(\mathbf{x})$. Solving a system of multivariate polynomials can now be formulated as the problem of finding the solution set to a set of polynomials $f_i(\mathbf{x}) \in \mathbb{C}(\mathbf{x})$, in the variables $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, such that

$$\begin{aligned} f_1(\mathbf{x}) &= 0 \\ &\vdots \\ f_m(\mathbf{x}) &= 0. \end{aligned} \tag{2.38}$$

The zero set (solution set) of a system of multivariate polynomials defines an *affine variety* V. We are only interested in problems with finite (but positive) number of solutions, i.e., finite varieties. A system of polynomial equations, as in (2.38), generates an *ideal* I, which is defined as

Definition 2.13. *The generated ideal I of a set of polynomials is the set*

$$I = \left\{ \sum_{i=1}^{m} h_i(\mathbf{x}) f_i(\mathbf{x}) : \forall h_1, \ldots, h_m \in \mathbb{C}(\mathbf{x}) \right\}. \tag{2.39}$$

An ideal is *radical* if it is identical to the set of all polynomials vanishing on V. Two polynomials f and g are said to be equivalent with respect to I iff $f - g \in I$; denoted as $f \sim g$. Thus, equivalent polynomials have identical values on the variety.

Example: Consider the system

$$\begin{aligned} x - y &= 0 \\ xy - 1 &= 0. \end{aligned} \tag{2.40}$$

Let I be the ideal generated by the equations above. It is trivial to see that $x \sim y$ and that $xy \sim 1$ with respect to I. If we write these equivalence relations as

$$\underbrace{\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}}_{\mathbf{c}} \underbrace{\begin{pmatrix} 1 \\ x \end{pmatrix}}_{\mathbf{b}} = \underbrace{\begin{pmatrix} y \\ yx \end{pmatrix}}_{y\mathbf{b}}, \tag{2.41}$$

we see that the system of polynomials, analogously to the use of the companion matrix above, looks like an eigenvalue problem. In this case, the relationship shows that values of the y-variable corresponding to solutions of the original problem are eigenvalues of the matrix $\mathbf{c}$, and that the basis vector $\mathbf{b}$ evaluated at these solutions gives the corresponding eigenvectors. Thus, we will be able to find the solutions to the original system among the eigenvalues and eigenvectors of matrix $\mathbf{c}$.

The basic idea of the method presented here, as in the example above, is to write the system of equations as an eigenvalue problem. If we, by working with polynomials equivalent to the polynomials of the original problem, are able to rewrite the problem on the form

$$\mathbf{m}_a \mathbf{B} = a(\mathbf{x})\mathbf{B}, \tag{2.42}$$

where $\mathbf{m}_a$ is a matrix of coefficients, $\mathbf{B}$ a basis vector with monomials and $a(\mathbf{x})$ a polynomial, then we have found such an eigenvalue problem. Here

$\mathbf{m}_a$ is the so called *action matrix*, describing the action of the polynomial $a(\mathbf{x})$ (called the *action monomial*) on the monomials in the basis vector $\mathbf{B}$.

In the general case, it is not as easy as in the example above. Both the action matrix and the basis vector need to be constructed in some way. One approach is to use Buchberger's algorithm [14] to compute a gröbner basis for the ideal generated by the system of equations. The gröbner basis is then used to construct the action matrix. However, apart from being very slow, because of limitations in computers it often fails completely when required to use floating point math.

In this thesis, we will use a method called *single elimination with basis selection* [32, 16, 17]. The idea of this method is to increase the number of equations of the original system of equations, and then reduce the system with tools from linear algebra and matrix theory.

The first step is to expand the set of equations to the original problem. The equations that we add should evaluate to zero on the variety, so that the solution set remains the same. In the end, we will be using methods from linear algebra for calculating the solutions, meaning that we are only interested in linearly independent equations. It is possible to add any equations as long as they evaluate to zero on the variety. A common approach, however, is to generate new equations by multiplying the original ones by different monomials. The number of equations needed depends on the problem at hand, and on the equations added. For practical reasons, equations are normally added by multiplying with all monomials up to a certain degree, then check if we have enough equations, and otherwise continue by adding more. We write the expanded set of equations as

$$\mathbf{CX} = \mathbf{0}, \tag{2.43}$$

where $\mathbf{C}$ is a matrix of coefficients and $\mathbf{X}$ a vector collecting all the monomials among these equations. Written on this form, we can use tools from numerical linear algebra for eliminating terms and monomials.

2.6.2 Reduction and choice of basis

In [16] a method was introduced for choosing a suitable basis while performing this elimination. Given an expanded system of equations, as in (2.43), we start by searching for monomial, that might be suitable to have in the basis. Finding an action matrix is the same as expressing $a(x)x^\alpha$

as a linear combination of base vectors. We need to do this for all elements x^α in the basis. Thus, a first condition is that $a(x)x^\alpha$ is present among the monomials in (2.43). *Permissible* monomials are monomials with this property, whereas *reducible* monomials are monomials $a(x)x^\alpha$, with x^α permissible. By reducing these monomials we can acquire the action matrix. The remaining set of monomials will not contribute with any information and we try to get rid of them as soon as possible. In accordance with [16], we call these *excessive* monomials. This means that we have divided the monomials in $\mathbf{X}$, into three subsets

$$\mathcal{M} = \mathcal{E} \cup \mathcal{R} \cup \mathcal{P}, \tag{2.44}$$

where $\mathcal{E}, \mathcal{R}$ and $\mathcal{P}$ denote the excessive, reducible and permissible sets respectively.

Now rewrite (2.43) as

$$\begin{bmatrix} \mathbf{C}_\mathcal{E} & \mathbf{C}_\mathcal{R} & \mathbf{C}_\mathcal{P} \end{bmatrix} \begin{bmatrix} \mathbf{X}_\mathcal{E} \\ \mathbf{X}_\mathcal{R} \\ \mathbf{X}_\mathcal{P} \end{bmatrix} = \mathbf{0}. \tag{2.45}$$

Our goal is to select the basis from the permissible set $\mathcal{P}$, and by performing elimination (reduction) on the reducible set $\mathcal{R}$, end up with a system describing the action of the action monomial on this basis. First, we eliminate the excessive monomials $\mathcal{E}$, by for example QR-factorization, giving us the system

$$\begin{bmatrix} \mathbf{U}_{\mathcal{E}_1} & \mathbf{C}_{\mathcal{R}_1} & \mathbf{C}_{\mathcal{P}_1} \\ \mathbf{0} & \mathbf{U}_{\mathcal{R}_2} & \mathbf{C}_{\mathcal{P}_2} \\ \mathbf{0} & \mathbf{0} & \mathbf{C}_{\mathcal{P}_3} \end{bmatrix} \begin{bmatrix} \mathbf{X}_\mathcal{E} \\ \mathbf{X}_\mathcal{R} \\ \mathbf{X}_\mathcal{P} \end{bmatrix} = \mathbf{0}, \tag{2.46}$$

where $\mathbf{U}_{\mathcal{E}_1}$ and $\mathbf{U}_{\mathcal{R}_2}$ are upper-triangular matrices. Since the set $\mathcal{E}$ is not contributing any information, we can remove the top row so that we get

$$\begin{bmatrix} \mathbf{U}_{\mathcal{R}_2} & \mathbf{C}_{\mathcal{P}_2} \\ \mathbf{0} & \mathbf{C}_{\mathcal{P}_3} \end{bmatrix} \begin{bmatrix} \mathbf{X}_\mathcal{R} \\ \mathbf{X}_\mathcal{P} \end{bmatrix} = \mathbf{0}. \tag{2.47}$$

In the last step we also make sure that $\mathbf{C}_{\mathcal{P}_3}$ becomes upper-triangular and select a basis from $\mathcal{P}$. We accomplish this by performing QR-decomposition with column pivoting of the bottom row. This causes a reordering of the

elements in $\mathbf{X}_\mathcal{P}$. We select the $|V|$ last elements in $\mathbf{X}_\mathcal{P}$ as our basis $\mathcal{B}$. After splitting $\mathbf{X}_\mathcal{P}$ into $[\mathbf{X}_{\mathcal{P}'}\mathbf{X}_\mathcal{B}]$ we get

$$\begin{bmatrix} \mathbf{U}_{\mathcal{R}_2} & \mathbf{C}_{\mathcal{P}'} & \mathbf{C}_{\mathcal{B}_1} \\ \mathbf{0} & \mathbf{U}_{\mathcal{P}'} & \mathbf{C}_{\mathcal{B}_2} \end{bmatrix} \begin{bmatrix} \mathbf{X}_\mathcal{R} \\ \mathbf{X}_{\mathcal{P}'} \\ \mathbf{X}_\mathcal{B} \end{bmatrix} = \mathbf{0}. \tag{2.48}$$

If we rearrange the equations as

$$\begin{bmatrix} \mathbf{X}_\mathcal{R} \\ \mathbf{X}_{\mathcal{P}'} \end{bmatrix} = - \begin{bmatrix} \mathbf{U}_{\mathcal{R}_2} & \mathbf{C}_{\mathcal{P}'} \\ \mathbf{0} & \mathbf{U}_{\mathcal{P}'} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{C}_{\mathcal{B}_1} \\ \mathbf{C}_{\mathcal{B}_2} \end{bmatrix} \mathbf{X}_\mathcal{B}, \tag{2.49}$$

we are able to see how $T_{a(\mathbf{x})}$ acts on the basis. This gives us a way of building the action matrix $\mathbf{m}_a$ — for each monomial $\mathbf{x}^\alpha \in \mathcal{B}$ we have that either $a(\mathbf{x})\mathbf{x}^\alpha \in \mathcal{B}$, in which case the corresponding element in $\mathbf{m}_a$ can be seen trivially, or we have that $a(\mathbf{x})\mathbf{x}^\alpha \in \mathcal{R} \cup \mathcal{P}'$ and it can be retrieved from (2.49).

The method presented here is the column-pivoting method presented in [17]. For a more detailed exposition we refer to that paper.

Will this method always work? The short answer is no. It may happen that it is impossible to perform one, or several, of the steps above. The monomial basis we selected might be too small to express the reduced polynomials. In that case we can try generating more equations and using a larger basis. Unfortunately, at some point numerical problems, might keep us from further increasing the size.

Another problem that might occur is eigenvalues with multiplicity. We know that the basis vector exists evaluated among the eigenvectors. In the case of distinct eigenvalues, everything is known up to unknown scaling. When multiplicity increases above one, eigenspaces with higher dimension than one might make it difficult to find the solution. Another thing one might try is using a different action monomial to get another eigenvalue-problem.

When we have found something that works for one problem instance, we will not have to redo all the work again for the same problem. The structure of the action matrix and the choice of basis only depends on the structure of the polynomials, not on their coefficients. This also means that for a specific problem, it is possible to create very fast solvers by hardcoding the right operations in C code. This can even be automated [58].

Example: Solving a system using the action-matrix method.
We will try to solve the system

$$\begin{aligned} x^2 + 3x + y + 1 &= 0 \\ x + y + 9 &= 0 \end{aligned} . \tag{2.50}$$

We generate one more equation by multiplying the lower equation with x, giving us

$$x^2 + xy + 9x = 0 . \tag{2.51}$$

We now have the monomials x^2, xy, x, y and 1. If we choose y as action monomial, 1 and y become *permissible*, xy and y *reducible* and x^2 *excessive*. We sort the monomials as above and write the system on matrix form

$$\begin{pmatrix} 1 & 0 & 1 & 3 & 1 \\ 0 & 0 & 1 & 1 & 9 \\ 1 & 1 & 0 & 9 & 0 \end{pmatrix} \begin{pmatrix} x^2 \\ xy \\ y \\ x \\ 1 \end{pmatrix} = \mathbf{0}. \tag{2.52}$$

By using QR methods, we can eliminate the excessive monomials from the other equations. We get

$$\begin{pmatrix} 1 & 0 & 1 & 3 & 1 \\ 0 & 1 & -1 & 6 & -1 \\ 0 & 0 & 1 & 1 & 9 \end{pmatrix} \begin{pmatrix} x^2 \\ xy \\ y \\ x \\ 1 \end{pmatrix} = \mathbf{0}. \tag{2.53}$$

Now discard the excessive monomials,

$$\begin{pmatrix} 1 & -1 & 6 & -1 \\ 0 & 1 & 1 & 9 \end{pmatrix} \begin{pmatrix} xy \\ y \\ x \\ 1 \end{pmatrix} = \mathbf{0}, \tag{2.54}$$

and let $\mathcal{B} = \{x, 1\}$. This gives us

$$\begin{pmatrix} 1 & -1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} xy \\ y \end{pmatrix} = - \begin{pmatrix} 6 & -1 \\ 1 & 9 \end{pmatrix} \begin{pmatrix} x \\ 1 \end{pmatrix} . \tag{2.55}$$

Solving this system gives us

$$\begin{pmatrix} xy \\ y \end{pmatrix} = \begin{pmatrix} -7 & -8 \\ -1 & -9 \end{pmatrix} \begin{pmatrix} x \\ 1 \end{pmatrix}. \tag{2.56}$$

From this we can directly observe how the action monomial acts on the basis, and thus the action matrix

$$y\mathbf{B} = y \begin{pmatrix} x \\ 1 \end{pmatrix} = \begin{pmatrix} -7 & -8 \\ -1 & -9 \end{pmatrix} \mathbf{B} = \mathbf{m}_a \mathbf{B}. \tag{2.57}$$

Eigenvalues to this matrix are -5 and -11. The eigenvector corresponding to -5 is $(0.970, -0.243)^T$. As stated above, the eigenvectors give the basis vector evaluated at a solution, but with an unknown scaling. Hence

$$\begin{pmatrix} x \\ 1 \end{pmatrix} = \lambda \begin{pmatrix} 0.970 \\ -0.243 \end{pmatrix} = \begin{pmatrix} -4 \\ 1 \end{pmatrix}. \tag{2.58}$$

The first solution is thus given by $x = -1$, $y = -5$. From the second eigenvalue we get the solution $x = 2$, $y = -11$.

2.7 Convexity and L_∞-optimization

In this section, some properties of convex sets and functions that will be needed in the thesis are presented.

Definition 2.14. *A set S in a real vector space is* convex *if for all $x, y \in S$*

$$\lambda x + (1 - \lambda)y \in S, \quad \textit{for all } \lambda \in [0, 1]. \tag{2.59}$$

A point on the form

$$x = \sum_i^n \lambda_i x_i, \tag{2.60}$$

where $\sum_i^n \lambda_i = 1$ and $\lambda_i \geq 0$ is called a convex combination of the points $x_1, \ldots, x_n$. A set is convex iff it contains each convex combination of its points.

A special set that we will work with is the *second order cone*. It is a set in $\mathbb{R}^{n+1}$ that we can write on the form

$$\{(x,t) \in \mathbb{R}^{n+1} : \|x\|_2 \leq t\}. \tag{2.61}$$

From properties of norms it follows that the second order cone is a convex set in $\mathbb{R}^{n+1}$. Convexity is preserved under affine mappings. When applied to the second order cone, we get convex sets of the type

$$\{z : \|Az + b\|_2 \leq c^T z + d\}. \tag{2.62}$$

Convexity is also preserved under intersection. Thus a set that is given by several of the constraints above is also convex.

Definition 2.15. *A function $f : S \mapsto \mathbb{R}$ is* convex *if the epigraph of f,*

$$\text{epi} f = \{(x,y) : x \in S, y \geq f(x)\}, \tag{2.63}$$

is a convex set.

A convex optimization problem is one where the objective is to minimize a convex function on a convex set. Since the intersection of convex sets is convex, convex optimization problems can be written as

$$\begin{aligned} \text{minimize} \quad & f(x) && (2.64)\\ \text{subject to} \quad & f_i(x) \leq 0, \quad i = 1, \ldots, m, && (2.65)\\ & a_i^T x = b_i, \quad i = 1, \ldots, p, && (2.66) \end{aligned}$$

where $x \in \mathbb{R}^n$ and both f and f_i are convex functions from $\mathbb{R}^n$ to $\mathbb{R}$, and the equality constraints are affine. These problems can be solved in polynomial time by for example interior point methods, [10].

Definition 2.16. *A function f with the property that its sublevel sets*

$$\{x \in \text{domain}(f) : f(x) \leq \alpha\} \tag{2.67}$$

are convex for all $\alpha \in \mathbb{R}$ is called quasiconvex.

Convex functions are quasiconvex, but the opposite is not always true; all quasiconvex functions are not convex. For a given threshold, α, the problem of determining if there exists an x such that $f(x) < \alpha$, is a convex

Algorithm 5 Bisection to minimize quasiconvex functions

Let l and u be given bounds such that $l < \min_x f(x) < u$.
Iterate:
$\alpha = (l + u)/2$.
If feasible solution x exists for $f(x) \leq \alpha$,
$u = \alpha$.
Otherwise,
$l = \alpha$.

feasibility problem. These can also be solved using interior point methods. This means that quasiconvex functions can be minimized using *bisection*. See Algorithm 5 which converges to the minimum for such functions.

For a particular class of convex optimization problems, the constraints are second order cones, of the form (2.62). These problems are called *second order cone programs* (SOCP). There are publicly available software packages for solving a SOCP, e.g., SeDuMi [102].

Part I

Localization

Chapter 3

3D reconstruction with position measurements

This chapter presents a system for structure-from-motion estimation using additional positioning data, such as GPS measurements. The system incorporates the additional data throughout the pipeline; in the outlier-detection step, the initial reconstruction and the final bundle adjustment. The initial reconstruction is based on a novel objective function which is solved using convex optimization. This reconstruction is refined based on a novel near L_2 minimization of the reprojection error using convex optimization methods. Results are presented on synthetic and real data, showing improvements with respect to robustness, accuracy and speed. This chapter is based on [105].

3.1 Introduction

Today cameras are ubiquitous and image data is readily available. In addition to pure image data there is, in many cases, more information available such as GPS data, user geo-tagged images or depth information of the scene. For instance many smartphones tag images with GPS-information. The additional information is often complementary in nature to the pure image information. It would be reasonable to use this information in conjunction with the image data in order to simplify the estimation problem, as well as making it more robust and faster. We will see in the experiments that adding additional constraints on positions, improves convergence of the interior point method used for solving the convex problems. A challenge in this context is to find a suitable objective function, that fuses information from the different sensors while taking into account the kinds

of noise that they are associated with. This is one concern of this chapter.

Much of the work that has been done incorporating both visual and positional data are set in a real-time framework, in a SLAM setting see e.g., [81]. Incorporating positional information in systems such as these is usually done in the final bundle adjustment. A number of contributions exist that incorporate, e.g., GPS information into the final optimization, see e.g., [64, 87, 47, 100, 71] or odometric data [77, 57, 101]. If the initial estimates are not good enough this could lead to problems with local minima, which we show in the experimental section. We have found that incorporating additional known positional cues during the whole estimation process can make a large difference on the final reconstruction.

In this chapter, we will focus on a batch setting, where we try to solve the whole SfM problem with all the data available. We will show how additional positional data can be incorporated in the complete structure from motion estimation framework, and in this way both make the solutions more robust and more accurate as well as in some cases even speed up convergence. In order to do this we formulate a number of error measures that incorporate both image and positional data. We show, using methods from convex optimization, how these minimization problems can be solved in an efficient way. We have focused on estimating the translation of the cameras and 3D points and assume that the orientation of the cameras is known. There are a number of efficient methods for estimating consistent rotations between cameras, see [38, 112, 75], and any of these can be used in conjunction with our system.

The main contributions can be summarized as: i) a system for structure-and-motion estimation based on image data and additional positional data. The positional cues can be used in all steps of the estimation. ii) an approximate L_2-norm formulation of the reprojection error. The goal function can be solved globally optimal using a novel method based on convex optimization.

The proposed methods rely heavily on previous work on using convex optimization for structure from motion. Hence, we start by going through the basics as introduced by [43, 51, 53]. Then, in sections 3.3-3.5, we look at the proposed system for structure from motion. This is then tested on both real and synthetic data in Section 3.6.

3.2 Convex structure from motion

Given a set of image points, seen in a set of images, we would like to recover the scene structure and the relative motion between the cameras. The noise depends mainly on uncertainty in the detected image feature points. Thus, it makes sense to try to minimize the deviation of the reprojected 3D points, from the measured image points. Let u_{ij} be image point ij, U_j the corresponding estimated 3D point, and R_i and C_i orientation and center of camera i, respectively. Then the reprojection error can be written as

$$r_{ij}(\theta) = \left\| \frac{\left(R_i^{(1)}(U_j - C_i), R_i^{(2)}(U_j - C_i) \right)}{R_i^{(3)}(U_j - C_i)} - u_{ij} \right\|, \tag{3.1}$$

where $R_i^{(1)}$ denotes the 1st row of R_i and θ represents the collection of parameters. As discussed in Section 2.4, if the measurement errors are assumed to follow the normal distribution, the maximum likelihood solution is given by minimizing the L_2-norm of the reprojection errors, given by

$$\underset{\theta}{\text{minimize}} \sum_{i,j} r_{ij}^2(\theta). \tag{3.2}$$

Unfortunately, this is in general a very difficult function to minimize. Only local methods exist, which cannot guarantee that the solution is found. One way of handling this is to instead minimize the L_∞-norm of the errors, see [43],

$$\underset{\theta}{\text{minimize}} \ \max_{i,j} \ r_{ij}(\theta). \tag{3.3}$$

Using the L_∞-norm instead of the L_2-norm, makes it possible to find the global minimum. To see why, we start by rewriting the problem as

$$\underset{\theta,\epsilon}{\text{minimize}} \quad \epsilon \quad \text{s.t.} \tag{3.4}$$

$$r_{ij}(\theta) \leq \epsilon \quad \forall \, i, j. \tag{3.5}$$

Here ϵ is minimized and since $\epsilon \geq r_{ij}(\theta)$ for all i and j, ϵ has to take the same value as the largest residual $\max_{i,j} r_{ij}(\theta)$. Therefore the two formulations are equivalent.

Now, consider again the reprojection error. To simplify notation, let θ be a vector with all U and C and introduce a vector c_{ij} such that

$$c_{ij}^T\theta = R_i^{(3)}(U_j - C_i) \tag{3.6}$$

i.e., the depth of point j relative to camera i. We can rewrite the reprojection error (3.1) as

$$r_{ij}(\theta) = \frac{\left\| \left(R_i^{(1)}(U_j - C_i), R_i^{(2)}(U_j - C_i) \right) - u_{ij} c_{ij}^T \theta \right\|}{c_{ij}^T \theta}. \tag{3.7}$$

If we also introduce a_{ij} and b_{ij} such that

$$a_{ij}^T\theta = R_i^{(1)}(U_j - C_i) - u_{ij}^{(1)} c_{ij}^T\theta \tag{3.8}$$

and

$$b_{ij}^T\theta = R_i^{(2)}(U_j - C_i) - u_{ij}^{(2)} c_{ij}^T\theta, \tag{3.9}$$

we get a much simpler form

$$r_{ij}(\theta) = \frac{\left\| \left(a_{ij}^T\theta, b_{ij}^T\theta \right) \right\|}{c_{ij}^T\theta}. \tag{3.10}$$

Since the depth is positive, we can use (3.10) to rewrite the constraints in problem (3.4), giving us

$$\begin{aligned} &\underset{\theta,\epsilon}{\text{minimize}} \quad \epsilon \quad \text{s.t.} \\ &\left\| \left(a_{ij}^T\theta, b_{ij}^T\theta \right) \right\| \leq \epsilon\, c_{ij}^T\theta \quad \forall\, i, j. \end{aligned} \tag{3.11}$$

When ϵ and all a_{ij}, b_{ij} and c_{ij} are known, we know from Section 2.7 that these are second-order cones. Further, we know that the intersection of all convex constraints is convex. Hence, this problem can be minimized using the bisection method presented in Algorithm 5 in the same section.

From now on, we assume that the rotational part of each camera is determined in advance, as is done in many recent approaches, e.g., [75, 114]. Methods for robust estimation of camera rotations are presented in [38, 42, 112, 75].

3.3 Robust initial reconstruction

As we saw in the previous section, the structure-and-motion problem can be solved optimally in L_∞-norm by solving a sequence of convex feasibility problems. However, this is under the assumption that we have correct matchings between corresponding feature points in different images. This is rarely the case in real world scenarios. Incorrect point matches, i.e., outliers in the data, will prevent us from minimizing ϵ enough to get good reconstructions. We will here present a minimization scheme that can be used to remove outliers in a way similar to [95]. We start by fixing ϵ, making it a threshold for inliers. As suggested in [25], the cone constraints can be made more flexible using auxiliary variables, s_{ij}. These allow the reprojection errors to become larger than the prescribed threshold ϵ. A really robust formulation is to minimize the number of non-zero s_{ij}, but this is a very difficult problem, so following [25] we settle for the L_1 relaxation and minimize the sum of all s_{ij} under the constraint that all $s_{ij} \geq 0$. The optimization problem becomes

$$\begin{aligned} \underset{\theta, s_{ij}}{\text{minimize}} \quad & \textstyle\sum_{i,j} s_{ij} \quad \text{s.t.} \\ & s_{ij} \geq 0 \\ & \left\| \left(a_{ij}^T \theta, b_{ij}^T \theta \right) \right\| \leq \epsilon \, c_{ij}^T \theta + s_{ij} \quad \forall \, i, j. \end{aligned} \tag{3.12}$$

In general the global scale can never be recovered in structure from motion. A problem with this formulation is that is has a bias towards smaller reconstructions (as this will make the s_{ij} smaller). In order to avoid the trivial solution C = 0 and $U = 0$, for all cameras and points, we need to regulate the scale. There are several ways of doing this, e.g., let $C_1 = \mathbf{0}$ and scale the translation vector of C_2 to unit length, see [25, 94]. But this is not suitable for our case since the slack variables make it possible to ignore any equations including these cameras and place all other cameras at the origin. Instead we fix the scale by enforcing all depths to be larger than, or equal to, 1. This is more robust as it affects all cameras and points. Note that as our formulation has a bias towards smaller reconstructions, there is no risk that the scale will increase towards infinity.

Having solved (3.12), all outliers can be purged from our problem by removing all image points u_{ij} for which $s_{ij} > 0$. Thus, solving one convex optimization problem, we get a solution with maximum reprojection error smaller than ϵ.

3.3.1 Incorporating position measurements

Probably the most readily available measurements, besides the image itself, is GPS-data. Thus we decided to use such information in our framework. Again the scale ambiguity produces a small problem. One could use GPS data to disambiguate the scale, but using a robust formulation this can be risky. Depending on the amount of GPS data available, it might still be beneficial to set all points and free cameras at the origin. To get a general approach that works well with anything from a few GPS measurements to full coverage, we stick to the scale defined by requiring all depths to be larger than 1 and introduce an unknown scale factor on the GPS measurements. With $\hat{C}_j$ denoting a position measurement for camera j, the GPS error is

$$\frac{\left\| \varsigma \hat{C}_j - C_j \right\|}{\varsigma}, \tag{3.13}$$

where ς is the unknown scaling factor.

By taking camera measurements into consideration in the initial outlier removal step, the set of feasible solutions shrinks, hence, reducing the risk that an outlier fits into the solution. Although GPS measurements can be rather noisy it is uncommon with outlier measurements, so we can use hard constraints on the form

$$\left\| \varsigma \hat{C}_i - C_i \right\| \leq \varsigma \omega, \tag{3.14}$$

where ω is some predefined error threshold.

Structure position estimates

We will not go through the details here, but it is more or less straightforward to incorporate any additional positional cues on scene structure points that are available. This could be beneficial in many settings. Such data could, e.g., be depth measurements from time-of-flight cameras or other types of depth sensors such as calibrated stereo-cameras.

3.4 Approximate least squares

As we have seen, using a bisection algorithm, we can find the optimal solution to the L_∞ problem. However, as discussed in Section 3.2, what we

really want to minimize is more similar to the L_2-norm of the reprojection errors. Ideally we would like to formulate this as a convex optimization problem. This is asking too much. However we can formulate an approximation to the L_2-norm as a second-order cone program (SOCP). We would like to solve

$$\underset{\theta}{\text{minimize}} \sum_{i,j} r_{ij}^2(\theta). \tag{3.15}$$

We start by looking at the squared reprojection error for one point u (dropping the indices to improve readability),

$$r^2(\theta) = \frac{\left(a^T\theta\right)^2 + \left(b^T\theta\right)^2}{(c^T\theta)^2}. \tag{3.16}$$

This function is non-convex and hard to optimize. Hence, we replace the denominator, with $\hat{\lambda}c^T\theta$, where $\hat{\lambda}$ is the approximative depth that we obtained from the robust reconstruction scheme proposed in the previous section. The resulting function,

$$r^2(\theta) = \frac{\left(a^T\theta\right)^2 + \left(b^T\theta\right)^2}{\hat{\lambda}\, c^T\theta}, \tag{3.17}$$

is convex, as proven by the following lemma.

Lemma 3.17. *A set*

$$\{(x,t) : \frac{(a^Tx)^2 + (b^Tx)^2}{c^Tx} < t \text{ and } c^Tx > 0\} \tag{3.18}$$

where a, b *and* c *are constant vectors, is equal to the second order cone*

$$\left\{(x,t) : t + c^Tx > \left\|\left(2a^Tx, 2b^Tx, t - c^Tx\right)\right\|_2\right\}. \tag{3.19}$$

Proof. We start from

$$t + c^Tx > \left\|\left(2a^Tx, 2b^Tx, t - c^Tx\right)\right\| \tag{3.20}$$

and show that it is equivalent to the inequalities defining the first set. First note that

$$\text{LHS} = t + c^Tx > \text{RHS} > t - c^Tx. \tag{3.21}$$

After reordering, we get $c^T x > 0$. Now, if both sides of (3.20) are positive we change nothing by taking squares. This yields

$$(t + c^T x)^2 > 4(a^T x)^2 + 4(b^T x)^2 + (t - c^T x)^2, \tag{3.22}$$

which is equivalent to

$$4tc^T x > 4(a^T x)^2 + 4(b^T x)^2. \tag{3.23}$$

As we know that $c^T x > 0$ we can divide both sides,

$$t > \frac{(a^T x)^2 + (b^T x)^2}{c^T x}. \tag{3.24}$$

It remains to show that the assumption, $t + c^T x > 0$ that we made taking squares is still satisfied. However, $c^T x > 0$ and (3.24) implies $t > 0$, so this is satisfied. Hence we have shown that the two sets are the same. □

Now let us see how we can use this result. Using the approximate squared residual, $g(\theta)$, we formulate the following approximate least squares problem.

$$\underset{\theta, t_{ij}}{\text{minimize}} \quad \sum_{i,j} t_{ij} \tag{3.25}$$

$$\text{subject to} \quad t_{ij} \geq g_{ij}(\theta). \tag{3.26}$$

Using Lemma 3.17, we can rewrite a constraint $g(\theta) \leq t$ as a second-order cone constraint.[1] This results in the second-order cone program,

$$\underset{\theta, t_{ij}}{\text{minimize}} \quad \sum_{i,j} t_{ij} \quad \text{s.t.} \tag{3.27}$$

$$\left\| \left(2a_{ij}^T \theta, 2b_{ij}^T \theta, t - \hat{\lambda} c_{ij}^T \theta \right) \right\| \leq t_{ij} + \hat{\lambda} c_{ij}^T \theta \quad \forall\, i, j. \tag{3.28}$$

Just as for (3.12), we have bias towards small reconstructions. Here, it is a result of using the depth approximations. Hence we use the same constraints enforcing all depths to be larger than, or equal to, 1.

To conclude our work so far, we summarize the preceeding sections in Algorithm 6 for computing structure and motion using an approximate L_2-norm.

[1] The constraints are not equivalent on the boundary, but this is of no practical importance.

Algorithm 6 Approximate L_2 SfM

1. Perform outlier removal as described in Section 3.3, by solving (3.12).
2. Use the obtained depth estimates, $\hat{\lambda}_{ij}$, to set up (3.27). Solve.

3.4.1 Incorporating position measurements

Recall that $\hat{C}_j$ is the position measurement for camera j. If the measurement error is assumed to follow a normal distribution, the maximum likelihood solution is to minimize the L_2-norm of measurement errors. From (3.13) we see that a squared GPS residual has the form

$$\frac{\left\| \varsigma \hat{C}_j - C_j \right\|^2}{\varsigma^2}. \tag{3.29}$$

Since this is the same form as the squared reprojection errors we can use the same idea to get a convex formulation. First we replace the denominator with $\hat{\varsigma}\varsigma$, where $\hat{\varsigma}$ is the approximate scale factor obtained from the initial robust reconstruction step. Then we use Lemma 3.17 and obtain

$$\underset{\theta, q_i, \varsigma}{\text{minimize}} \quad \sum_i q_i \quad \text{s.t.} \tag{3.30}$$

$$\left\| \begin{pmatrix} 2(\varsigma \hat{C}_i - C_i) \\ q_i - \hat{\varsigma}\varsigma \end{pmatrix} \right\| \leq q_i + \hat{\varsigma}\varsigma. \tag{3.31}$$

Note that without the depth-constraints presented earlier, this formulation does not make sense since an optimal solution is to set all variables to zero.

Under the assumption that the different types of errors are independent and Gaussian, the maximum likelihood estimate is given by scaling the squared residuals by the inverse of their variance [2]. This will allow larger residuals for uncertain sensors. The modified objective function for the second optimization step becomes

$$\underset{\theta, t_{ij}, q_i, \varsigma}{\text{minimize}} \quad \frac{1}{\sigma_r^2} \sum_{i,j} t_{ij} + \frac{1}{\sigma_{pos}^2} \sum_i q_i, \tag{3.32}$$

subject to the constraints in (3.27) and the constraint (3.31).

3.5 The full framework

To sum up, the proposed reconstruction pipeline is outlined in Algorithm 7.

Algorithm 7 SfM with Positional Cues

1. Outlier removal and depth estimation. Solve

$$\underset{\theta, s_{ij}, \varsigma}{\text{minimize}} \quad \sum_{i,j} s_{ij} \tag{3.33}$$

   subject to the constraints in (3.12), (3.14), and $\lambda_{ij} \geq 1$ for all ij.

2. Use the solution obtained during step 1 to get depth estimates $\hat{\lambda}_{ij}$. Solve (3.32) subject to the constraints in (3.27), the constraints (3.31) and (3.14), and $\lambda_{ij} \geq 1$ for all i, j.

3. Refine solution using L_2-norm bundle adjustment.

3.6 Experiments

In this section we test our system in a number of experiments using both synthetic and real data. The software SeDuMi is used to solve all convex problems.

3.6.1 Experiments on simulated data

In order to be able to compare with ground truth camera positions, we have performed experiments on synthetic data.

Here we present two such scenarios. In the first one, an imaginary street was placed along a circular arc. Along the sides of the street, 3D points were put on facades. Equidistant cameras were placed along the street, seeing some of the points. Each point was registered on the camera's image plane with Gaussian distributed error (standard deviation 0.04). Further, each camera was annotated with position measurements, also with

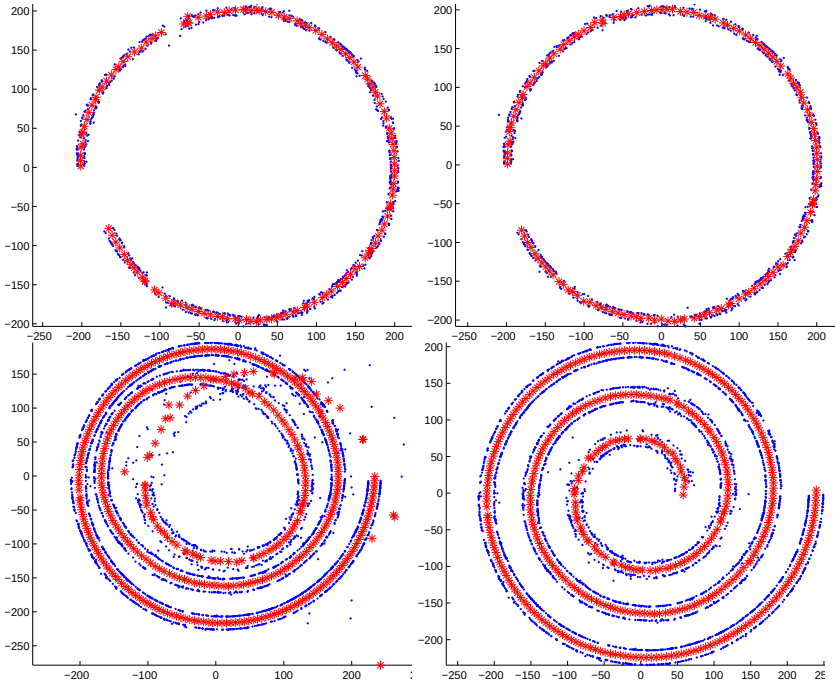


Figure 3.1: Comparison between SfM estimation for synthetic city streets. Without using GPS cues (left circle and spiral) and using GPS cues (right circle and spiral). Red dots are camera positions, blue dots are the estimated structure.

	Circle		Spiral	
	no gps	gps	no gps	gps
RMS	6.06	2.35	4.33	1.54
σ_U	3.32	1.13	1.57	0.92
σ_C	5.78	1.01	34.6	0.71

Table 3.1: Results from the synthetic experiments. The standard deviation of the errors in the estimated 3D points and camera positions are given by σ_U and σ_C respectively. All measures are given in meters.

Gaussian errors to simulate GPS-data. In [64], the typical standard deviation of a consumer GPS-device is given as 2.34 m. This is what we use in our synthetic experiments. Finally, 10 percent of all point correspondences between cameras were mismatched, to simulate gross outliers. The second scenario was constructed in the same way, but with a different geometry with uniformly distributed GPS error (0 to 4 m) with all cameras in a spiral pattern.

For each scenario, results with and without GPS data can be seen in Figure 3.1 and Table 3.1. Looking at the table, we see that using GPS information, the standard deviation of the camera position error, i.e., distance between estimated camera position and ground truth, had a standard deviation σ_C of merely 1.01 and 0.71 for the two scenarios. We also get smaller errors in the estimated 3D points (standard deviation σ_U). This despite the simulated GPS-errors being much larger.

3.6.2 Experiments on real data

We have conducted experiments on a number of street-view images with additional positional data available for each frame. These images are 360 degree panoramas that were rotationally registered in a common frame during acquisition. This means that the orientations of all cameras are known. We start by extracting SIFT keypoints and match these pairwise between images using RANSAC. This is the data that is fed into our system. We run the initial L_∞ optimization to root out outliers in the data. We then run our near L_2 optimizer and lastly we do a final bundle adjustment. The additional positional data is used throughout this process. In Figure 3.3,

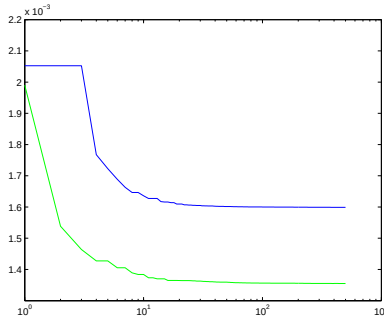


Figure 3.2: Here the L_2 reprojection error is shown as a function of the number of iterations. The top blue curve shows the convergence without GPS information and the bottom green curve shows the convergence using the GPS information.

two such reconstructions are shown. In these plots the corresponding reconstructions when the positional cues are not used are also shown. As can be readily seen there are a number of problems with the reconstructions without the positional cues that are remedied using them. Run-times and number of points and cameras for the two setups are shown in Table 3.2.

In Figure 3.2, the convergence of the optimization is compared with and without using the positional cues. Here the top curve shows the behavior without the positional cues. As can be seen, the bottom green curve converges both faster and to a lower minimum. This is rather surprising

Table 3.2: Runtimes (in seconds) and number of points and cameras for the different experiments, and the different steps in Algorithm 7.

	Nbr of cams	Nbr of points	Step 1	Step 2	BA	Step 1 +GPS	Step 2 +GPS	BA +GPS
Circle	146	1082	8.9	2.9	7.7	7.7	3.4	5.4
Spiral	405	3422	25	8.8	32	14	9.3	34
City 2	332	21422	678	393	601	647	374	513
City 1	1330	79496	250	163	238	250	131	270

since it is the L_2-norm of the reprojection errors that is shown. Incorporating the positional cues in the optimization will add terms to goal function. That we still reach a lower minimum means that we have found a better optimum and the blue curve has found a local optimum.

3.7 Concluding discussion

During the work with this system a few observations have been made.

i) This structure-from-motion system seems to work well for many problems.

ii) Incorporating positional measurements does not only improve robustness, but it seems that it sometimes also improves convergence for the final bundle-adjustment step — at least for moderately sized problems — resulting in good total execution times. This is also true for the approximate L_2 formulation, where the result sometimes is good enough to skip bundle adjustment entirely.

iii) One drawback of the system is that it is unable to handle large amounts of outliers. However, using the common steps for filtering out poor feature matches and for creating point tracks, this will rarely cause any problems. But there can be other situations where a need for robustness to outliers is greater.

Because of poor scaling to very large structure-from-motion problems, the next step would be to use this system in conjunction with some way of merging reconstructions. This would enable solving very large-scale problems. Being able to incorporate positional constraints on points and cameras could be a big advantage in such a setting.

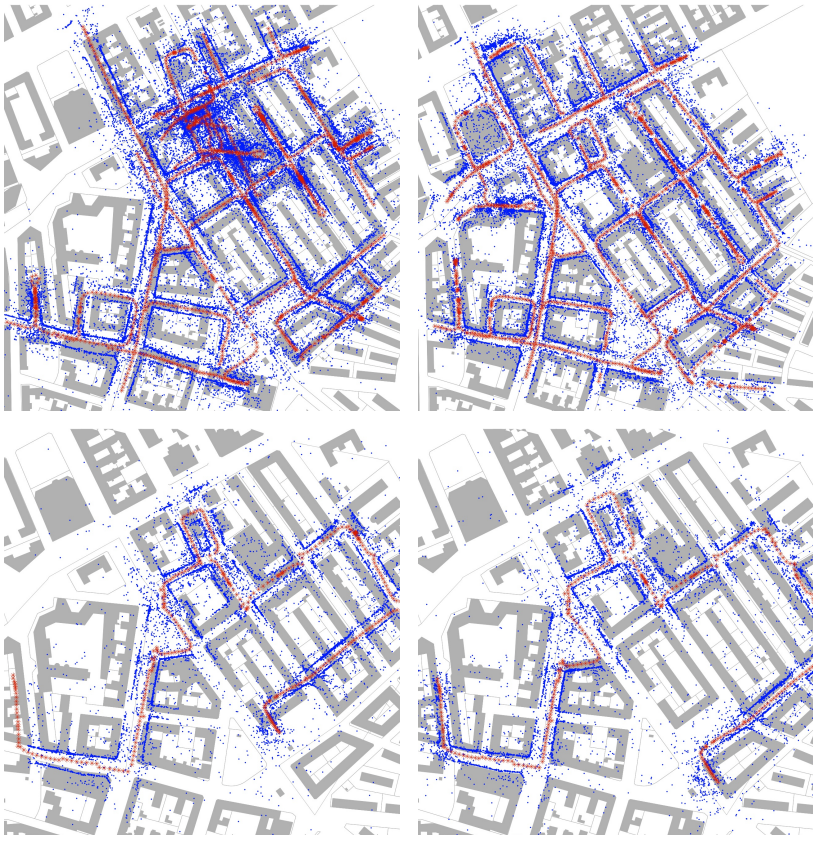


Figure 3.3: Comparison between SfM estimations for a city section. Without using GPS cues (top left) and using GPS cues (top right). Both reconstructions are registered to a GIS model of the city section. Without using GPS information during step 1 and 2, the solution gets stuck in a local minimum. The second city section without using GPS cues (bottom left) and using GPS cues (bottom right). The solution without GPS looks decent, but suffers from drift.

Chapter 4

Localization with orientation measurements

This chapter is concerned with the problem of localizing a novel image in a large 3D model, given that the gravitational vector is known. In principle, this is just an instance of camera pose estimation, but the scale of the problem introduces some interesting challenges. Most importantly, it makes the correspondence problem very difficult so there will often be a significant number of outliers to handle.

To tackle this problem, we use recent theoretical as well as technical advances. Many modern cameras and phones have gravitational sensors that allow us to reduce the search space. Further, there are new techniques to efficiently and reliably deal with extreme rates of outliers. We extend these methods to camera pose estimation by using accurate approximations and fast polynomial solvers. This chapter is based on [104].

4.1 Introduction

A classic problem in computer vision is estimating the orientation and position of a camera, given positions of a number of points in 3D and their projections in the camera image. The so-called *pose estimation problem* has been solved in many contexts and for many camera models, see [40].

Another problem that has attracted increasing attention over the past years is the *localization problem*, i.e., estimating the position (and sometimes the orientation) of a viewer or a camera given image data. A number of approaches have been suggested for solving this problem. Many have adopted an image retrieval approach, where a query image is matched to a database of images using visual features. Sometimes this is combined with

a geometric verification step, but in many cases the underlying geometry is largely ignored, see [45, 93, 67, 48].

The approach that we pursue in this chapter is viewing the localization problem as a pose estimation problem by matching an image to a large 3D model of the environment. In such an approach one crucial step is the robust matching of image features to features in the 3D model. The ability to handle massive amounts of outliers in the data is absolutely paramount.

For many practical applications, using e.g., vehicle-mounted cameras or devices such as smart phones with accelerometers, we can assume that the direction of the gravitational vector is known. This simplifies the problem by reducing the search space and we show that this enables tractable, efficient, robust and accurate algorithms for localization. A key observation is that the problem can be recast as a particular type of registration problem. We use this formulation and present a number of algorithms for performing outlier removal and pose estimation in low-order polynomial time.

We investigate two settings; When the height is restricted to some (not necessarily small) interval and then in Chapter 5 when the height of the camera relative some ground plane is known.

4.1.1 Related work

A number of solutions have been proposed for solving for the localization problem as a camera pose problem via 2D-to-3D matching, see [68, 19, 69, 91, 92]. The main focus has been to develop sophisticated heuristics for finding reliable matching schemes and avoiding the generation of false correspondences. We take a radically different standpoint: We instead allow the matching scheme to generate a lot of correspondences – correct or incorrect – in order to make sure that we do not miss any good correspondences. The focus of our approach is on the ability to handle large amount of outliers in a reliable and tractable manner.

Many approaches for robust estimation based on the RANSAC framework have been proposed over the years; see e.g., [20]. Although this works well in many cases, there are three main issues with these approaches that have to be addressed. Firstly, there is no guarantee that they will obtain a reasonable solution even if there exists one. Secondly it can be hard to determine if there is no solution at all. And finally, the number of iter-

ations required to find a solution with high probability tends to make the approach impracticable for the rates of outliers that we consider.

Another approach for handling outliers in a robust way is the L_∞-framework, see [52, 54, 95] including recent extensions [83, 111]. Many of these approaches work well for large scale problems, but break down with large rates of outliers.

Solving computer vision problems using IMU or accelerometer data in addition to visual data has been proposed in a number of previous papers. Some use it together with RANSAC, [36, 59], while others use it to bootstrap the filtering process in SLAM type approaches, see [79, 84, 98].

The most similar works to ours include [13, 65, 66, 29, 30] where the aim is to develop algorithms that provably optimizes a robust error norm. In most cases this simply means minimizing the number of outliers but some [4] also consider the truncated L_2-norm. Several of these approaches are based on branch-and-bound which has exponential-time complexity. To our knowledge, none of the above approaches are able to solve the pose problem with a provably optimal algorithm based on a robust error criterion that runs in polynomial-time.

4.2 Problem formulation

Assume that we have a large 3D model of, e.g., a city, where the term 3D model refers to a set of 3D points each equipped with a descriptor vector describing its local appearance. Given a new image from the same scene, the task at hand is to pinpoint the location and orientation that the camera had when capturing this image. This is normally referred to as camera pose estimation.

We assume that the camera calibration is known and that the camera has known orientation with respect to the 3D model. A typical case when this is true is when the camera is mounted on a vehicle or the camera is in a smart phone with accelerometers that measure the gravitational vector when stationary. Naturally this second case also requires that the 3D model was reconstructed using similar sensors, so the direction of gravity is known in the 3D model. Finally, we assume that the ground plane has been roughly located in the 3D model. One way to do this is by considering the height of the cameras used in the reconstruction. This last assumption is not required, but it will improve computation times significantly.

Consider a coordinate system with the camera at the origin and the z-axis points upwards. Let the 3-vector U denote a 3D-point and let u be a hypothetical correspondence in the image. The relative orientation between the camera and the point is known up to a rotation about the z-axis. In the noise-free case each correspondence should satisfy,

$$\lambda Su = U' = RU + t, \tag{4.1}$$

with

$$R = \begin{pmatrix} \bar{R} & 0 \\ 0 & 1 \end{pmatrix}, \tag{4.2}$$

where S is a known 3×3 rotation matrix and $\bar{R}$ is an unknown 2×2 rotation matrix.

Since finding accurate correspondences is difficult, we need to solve this problem in a robust way. A common approach is to simply optimize the number of consistent measurements, i.e., inliers. A consistent measurement is one with a reprojection error below some threshold. Measuring reprojection errors is normally the preferred choice, as this accurately models the limited precision of feature detection techniques.

Although this formulation leads to a challenging optimization problem, using recent advances in robust estimation it is possible to solve it in polynomial time with respect to the number of correspondences.

Let the 3D point be rotated and translated to the camera coordinate system. It is well-known that set of points in $\mathbb{R}^3$ that yields a reprojection error smaller than a threshold ϵ, forms a cone C in $\mathbb{R}^3$. A 3D point U is an inlier if

$$U' = RU + t \tag{4.3}$$

lies inside this cone C. Hence the camera pose problem can be viewed as a registration problem, namely that of registering a number of 3D points U_i to the corresponding cones C_i, see Figure 4.1.

4.3 Overview of the approach

In Section 2.5 it was shown how the number of outliers can be minimized in polynomial time. In order to do this, we need to define a goal function

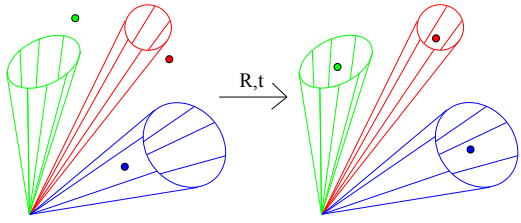


Figure 4.1: The registration problem for points lying on cones: Find a 3D translation and a planar rotation so that the 3D points lie on or within the cones.

on the parameter space and then construct a set of solvers. The details of how this is done is described in Section 4.5. The main theorem from [30] shows that one of the solution points generated in this way will be optimal with respect to the number of outliers. In this way we can minimize the number of outliers in $\mathcal{O}(n^4)$, or $\mathcal{O}(n^5)$, time for the problems with known and unknown height respectively.

Clearly, this approach is often far too slow to be practical. To overcome this, [4] proposes a simple and fast outlier rejection method to be used as a preprocessing step to the optimal estimation. The technique is specialized to the problems of stitching and 2D-2D registration, but in the next section we will see how to generalize it to our setting.

We propose the following localization pipeline. Start by matching features points between the image and the model. Then run the outlier rejection schemes to quickly eliminate a large amount of wrong matches. Finally use Algorithm 9 to find the best solution. In a number of experiments we show that this approach works for both very large models and for outlier rates up to more than 99%; see Section 4.6.

4.4 Fast outlier rejection

The purpose of this section is to present a fast method for rejecting outlier correspondences, while being certain not to alter the optimal solution. Pretend for a moment, that the height (z-coordinate) of the camera relative to

the 3D model is known. The unknown rotation and translation between camera and world coordinates is

$$R = \begin{pmatrix} \bar{R} & 0 \\ 0 & 1 \end{pmatrix}, \quad t = \begin{pmatrix} t_x \\ t_y \\ 0 \end{pmatrix}, \tag{4.4}$$

and as this transformation does not change the height of a given 3D point

$$U = \begin{pmatrix} v \\ h \end{pmatrix}, \tag{4.5}$$

it will always intersect the corresponding cone at $z = h$. Hence we can restrict the cone constraint to this plane and rather than a cone we get a conic section. This is normally, but not necessarily, an ellipse, see Figure 4.2. The correspondence is an inlier if

$$U' = RU + t \tag{4.6}$$

lies inside this conic section. The third coordinate of the 3D point was only interesting to determine the conic section. Having done that we can drop the third coordinate and hence we have an instance of 2D-2D-registration of points to conic sections. With this motivation, the next section is concerned with fast outlier rejection for the case of 2D-2D registration, and then, in Section 4.4.2, we will see how to use the same ideas for the case of unknown height.

4.4.1 Basic algorithm for outlier rejection

The following problem will play an important role in our outlier rejection scheme.

Problem 4.18. *Given 2D-points v_i and corresponding regions $C_i \subset \mathbb{R}^2$, find a rotation R and translation t such that*

$$Rv_i + t \in C_i, \tag{4.7}$$

for as many of the pairs (v_i, C_i) as possible.

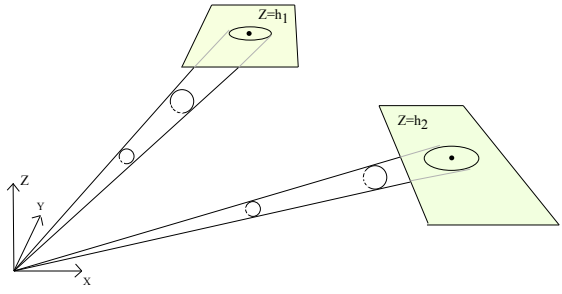


Figure 4.2: The reprojection errors for two example points are propagated as cones in 3D. The points in 3D are located at height h_1 and h_2 respectively. The error cones intersected with planes parallel to $z = 0$ give rise to error conics.

We call a point v_i an outlier with respect to a transformation if it does not place it inside region C_i.

The basis for our outlier rejection scheme is a bounding function of the following kind: *If correspondence i is an inlier, then there are no more than B_i inliers.* We will soon see how to achieve such a bound and also how to produce a lower bound, L, on the number of inliers. If $B_i < L$ then correspondence i can be permanently removed from the problem. First recall the definition of Minkowski addition from geometry, as stated in Definition 1. The Minkowski difference is defined in an analogue way. As an example, Figure 4.3 illustrates the Minkowski difference of an ellipse and a parabola.

Definition 1. The Minkowski sum of two sets of position vectors A and B is the set

$$\{a + b : a \in A, b \in B\}. \tag{4.8}$$

For technical reasons, we select a central point, c_i, from each C_i. For bounded C_i a natural choice is the centre of mass. Let $\bar{C}_i = \{x : x + c_i \in C_i\}$, i.e., the set C_i translated to the origin.

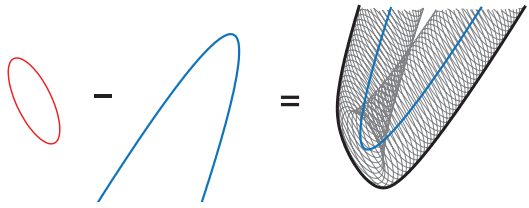


Figure 4.3: Illustration of the Minkowski difference of an ellipse and a parabola. The black outline shows the difference of the two regions.

Problem 4.19 (K)**.** *Find a rotation R, such that*

$$R(v_i - v_K) - (c_i - c_K) \in (\bar{C}_i - \bar{C}_K) \tag{4.9}$$

for as many of the points v_i as possible.

Theorem 4.4.1 gives a useful connection between Problem 4.19 and Problem 4.18.

Theorem 4.4.1 If v_K is an inlier to Problem 4.18, then Problem 4.19 has at least as many inliers as Problem 4.18.

Proof. Let R, t be the solution to Problem 4.18. Then

$$Rv_i + t \in C_i \tag{4.10}$$

for $i = 1, \ldots, k$. Using the same R in Problem 4.19, we get

$$R(v_i - v_K) - (c_i - c_K) = (Rv_i + t - c_i) - (Rv_K + t - c_K). \tag{4.11}$$

Since $(Rv_i + t) \in C_i$, by definition $(Rv_i + t - c_i) \in \bar{C}_i$. Similarly, $(Rv_K + t - c_K) \in \bar{C}_K$ and hence the difference on the right hand side in (4.11) lies in $\bar{C}_i - \bar{C}_K$. □

This theorem means that we can use Problem 4.19 to get bounds on the number of inliers to Problem 4.18. The advantage of this is that Problem 4.19 only has one unknown parameter and can be efficiently solved in the following way.

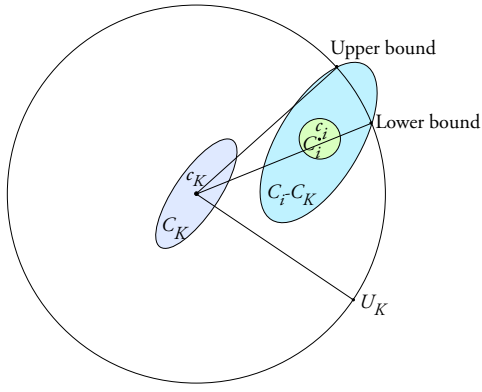


Figure 4.4: Propagating the errors from point U_K to U_i can be done using the Minkowski sum of the error conics. The bounds on the rotation for point U_i is also shown.

We seek a rotation angle that maximizes the number of points enclosed in their corresponding sets. Each point-to-set correspondence will be consistent with any θ in some set of intervals I_i. Naturally, the method for computing these intervals depends on the nature of the sets $\bar{C}_i - \bar{C}_K$, but they are normally easy to compute. See Figure 4.4 for a depiction of the bounds for one point U_i. By sorting all interval boundaries and going through the sorted list, we can find an optimal choice of θ. The computational cost of this is $\mathcal{O}(n \log n)$. See Algorithm 8. For the angle with maximally many consistent points, we also reproject all points yielding a lower bound L on the number of inliers. Assuming that we repeat it for each correspondence we get a complexity of $\mathcal{O}(n^2 \log n)$.

4.4.2 Outlier rejection for localization

We return to the original problem where the camera height is not known exactly. Let's say that there is an uncertainty of $\pm\Delta$. This means that the relative height of a point is limited to an interval $[h - \Delta, h + \Delta]$. For this 3D point to be an inlier we need to register it to the cone cut off at these two levels. As we are looking for lower bounds on the number of outliers we are free to consider a relaxation that has at least as many inliers.

Algorithm 8 Fast Outlier Rejection

Given a lower bound, L, on the number of inliers, compute an upper bound, B_K, assuming that correspondence K is an inlier. If $B_K < L$, remove correspondence K.

For each $i \neq K$
  Compute $M_i \supset C_i - C_K$.
  Find the intervals of angles such that $RU_i + t - c_i \in M_i$.
Sort the set of interval boundaries.
For each interval boundary $\theta^{(i)}$
  Let $b^{(i)}$ be the number of intervals containing $\theta^{(i)}$.
If $\max b^{(i)} + 1 < L$, then remove correspondence K.

Figure 4.5, shows the standard case. The conic sections at $h \pm \Delta$ are ellipses. For this point to be an inlier we want to register U to the cut-off cone. A weaker constraint is that the projection of U in the ground plane should be registered to the projection of the cut-off cone.

In the elliptical case, the projected shape is the convex hull of the conic section at height $h - \Delta$ and the conic section at height $h + \Delta$. Note that we could use these convex hulls in the outlier-rejection step since they are made up of linear and quadratic curves, but to keep geometry simple we instead use enclosing quadrilaterals; see Figure 4.5. Again this is possible since we are looking for lower bounds on the number of outliers.

After projection, we have transfered the problem to a 2D registration problem, so we can use the method from Section 4.4.1 to remove outliers.

Technical details. The uncertainty, Δ, in camera height is important. Naturally a smaller uncertainty will allow us to remove more outliers. Hence, if the actual uncertainty is large we construct k subintervals and perform the outlier rejection step in each of them. Correspondences which are rejected for all intervals can be permanently removed.

Since we will be able to remove more outliers if we have a higher lower bound L on the number of inliers, we repeat stepping through all subintervals twice; in the first iteration we hope to find a decent lower bound, and in the second iteration we hope to prune more outliers. With k height intervals we get a total complexity of $\mathcal{O}(kn^2 \log n)$.

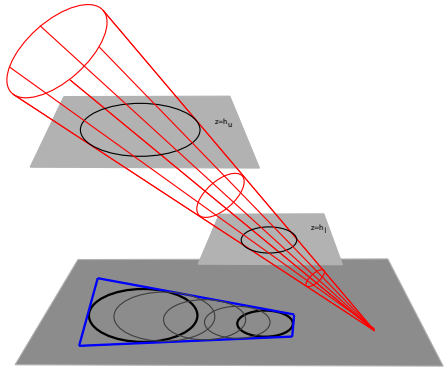


Figure 4.5: Cutting a cone with a number of planes between two heights $z = h_l$ and $z = h_u$ and projecting them onto the ground plane results in a shape (black) that can be approximated with a quadrilateral (blue).

4.5 Searching for critical points

As described in Section 4.3 we can find the optimal number of inliers by extracting all the critical points to a constructed optimization problem. First we decide on a goal function f on the parameter space. Normally a linear goal function will yield the simplest equations. We need to solve the following problems:

- Given four residuals compute all points satisfying $r_i = \epsilon$ for $i = 1, 2, 3, 4$.

- Given three residuals, compute all points such that $r_i = \epsilon$, $i = 1, 2, 3$ and the set of gradients is linearly dependent.

- Given two residuals, compute all points such that $r_i = \epsilon$, $i = 1, 2$ and the set of gradients is linearly dependent.

- Given one residual, compute all points such that $r_1 = \epsilon$, and the set of gradients is linearly dependent.

We have developed specialized solvers for all the cases. One of the solution points generated in this way will be optimal with respect to the number of outliers.

The residual constraint for a point U_i can be formulated as

$$U_i'^T C_i U_i' = 0, \quad U_i' = RU + t. \tag{4.12}$$

For our application, each of these problems can be formulated as the solution to a system of polynomial equations. We will briefly describe how we construct the first two solvers.

4.5.1 The 4-Point Solver

The parameter space can be embedded in $\mathbb{R}^5$ by setting

$$t = \begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix} \tag{4.13}$$

and

$$\bar{R} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} = \begin{pmatrix} a & -b \\ b & a \end{pmatrix} \tag{4.14}$$

and adding the embedding constraint $a^2 + b^2 = 1$. The first four equations from (4.12) are in general full second degree polynomials in the five variables (a, b, t_x, t_y, t_z). We can simplify this somewhat by fixating the coordinate system so that the first point is at the origin. This leads to that the first equation doesn't involve the rotation parameters. Together with the embedding constraint this yields a system of five quadratic equations. In order to construct the action matrix for the system we multiply our five equations with all monomials of up to total degree equal to three. This gives a large system of equations, involving 280 polynomials of degree five. We can write these as

$$A_{280\times 252} b_{252\times 1} = \mathbf{0}, \tag{4.15}$$

where A only contains entries based on measured image data, and b involves the unknown monomials (which are of total degree up to five). We

can construct the action matrix from A using QR factorization and the solution is then found from an Eigenvalue decomposition of the action matrix. This typically yields 28 solutions, but rarely more than 8 real-valued ones. We have implemented a solver where the most time consuming step is doing the QR factorization of the 280×252 matrix A. On a desktop computer the running time for this type of solver is in the order of a few milliseconds.

4.5.2 The 3-point solver

Although the technique from [30] is based on introducing a dummy goal function, this function is actually never used in the 4-solver. This is not the case for the 3-solver. To get as simple equations as possible we use a linear goal function, $f = a$, so that $\nabla f = [1\,0\,0\,0\,0]$. This should be linearly dependent with the gradients of the two registration constraints (4.12), and the gradient of the embedding constraint. This constraint is given by the determinant of the following matrix,

$$D' = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 2a & 2b & 0 & 0 & 0 \\ d'_{31} & d'_{32} & d'_{33} & d'_{34} & d'_{35} \\ d'_{41} & d'_{42} & d'_{43} & d'_{44} & d'_{45} \\ d'_{51} & d'_{52} & d'_{53} & d'_{54} & d'_{55} \end{bmatrix}, \tag{4.16}$$

where again $d'_{31}, \ldots, d'_{55}$ are linear polynomials in the unknowns. The determinant is equal to $2b \cdot q$ where q is a third degree polynomial in the five variables. Combining this equation with the three registration constraints (4.12) and the embedding constraint we end up with a set of equations that in general give 40 solutions. By multiplying this set of equation with a number of monomials we can construct the action matrix. Again the most time consuming step in the solver is doing a QR factorization, in this case of a 1260×1278 matrix.

4.5.3 Computational complexity

Algorithm 9 shows the steps of the outlier minimization algorithm. As the number of sets of ≤ 4 residuals is $\mathcal{O}(n^4)$ we can only do exhaustive sampling of these sets for low number of correspondences. This might seem

very restrictive, but first note that as the fast outlier rejection method normally removes all but a few of the outliers, it is the number of inliers that will be relevant with respect to efficiency. Moreover, in cases with hundreds of inliers, we will get a good pose estimation even without computing the globally optimal one. If speed is prioritized over optimality, another choice is to use ordinary minimal solvers after the rejection step.

For the experiments, we will sometimes exhaustively search all the subsets and sometimes stop when we have detected a good solution (i.e., with enough number of inliers) or reached a maximum number of iterations.

Algorithm 9 Outlier minimization

Given a set of image points u_i and 3D points U_i estimate the pose that minimizes the number of outliers.

Transform the problem to a point-to-cone registration.
For each subset of correspondences of size ≤ 4:
  Use the relevant solver to estimate the critical points.
  For each critical point (R, t):
    Count the inliers and update the best solution.

4.6 Experiments

We have conducted a number of experiments, both on synthetic and real data, to test robustness, speed and accuracy of the proposed methods.

4.6.1 City-Scale Localization

To evaluate the unknown-height method on challenging real-world data, we have performed a localization experiment on the Dubrovnik dataset [68]. It consists of a 3D model with approximately 2 million points reconstructed from 6000 images. Naturally, each point is also equipped with SIFT descriptors. In addition to the 3D model, the dataset also provides 800 test images with computed estimates of camera positions and orientations. As these estimates are also based on vision algorithms, they are not exactly ground-truth. In fact they are known to contain some outliers.

When building such a 3D model it is possible to also get a rough estimate of the ground plane and, based on the estimated matchings, get an interval of possible heights for Algorithm 8. As the ground plane is not available for the Dubrovnik dataset we synthesize this information by picking a ± 5 meters interval around the provided estimated height. Note that the length of this interval will mainly affect the running time and not the accuracy of the final result.

A similar problem is that the dataset contains no orientation measurements. Again we synthesize this information using the provided estimated camera orientations and adding a random rotation distributed uniformly on $[0, 1^\circ]$; see Section 4.6.5 for a motivation.

For each image, correspondences to the 3D model were established using standard SIFT matching (with matching ratio 0.9). Then the image was localized by running Algorithm 8 followed by Algorithm 9 with a maximum of 1000 iterations. As discussed in Section 4.5.3, we stop early if a reasonable amount of inliers is found.

Table 4.1 shows a comparison to other methods. In accordance with [68], an image is considered correctly localized if at least 12 correct inliers are found. For the two images where this was not the case, we found 8 and 9 inliers respectively and the errors were small. So essentially these two cases were not failures. Moreover, as our algorithm is optimal for a given bound on the errors, in our case 6 pixels, we can say that for these two images there does not exist a solution with 12 inliers. This is not a contradiction to [69] as we have a more restricted camera model. Also, since there are no correspondences provided in the dataset between the SIFT-points in the query images and the model points, it is somewhat difficult to compare performance between the different methods.

The median error of our method is significantly lower than for the other methods. This shows the advantage of using measurements from an orientation sensor – even if that sensor has an error of up to 1°.

Using a single threaded C-implementation, the median running time for Algorithm 8 was 5.06 seconds, containing 4766 point correspondences, so most problems for this dataset are large. For a more reasonably sized problem of 1000 correspondences, the running time was approximately 0.3 seconds. For the largest problem, with 17199 points, the execution time was 55.4 seconds. For Algorithm 9, we only had a Matlab-implementation. With this implementation, each iteration takes approximately 0.3

Method	# reg. images	Median error (m)	# error < 18.3 m	# error $> 400m$
Our	798	0.56	771	3
[92]	795.5	1.4	704	9
[91]	783.9	1.4	685	16
[91]	782.0	1.3	675	13
[68]	753	9.3	655	-
[19]	789	-	-	-
[69]	800	-	-	-

Table 4.1: Results on the Dubrovnik dataset; see [92].

seconds. In almost all cases for this experiment, the first step finds more than enough inliers for a very good solution, making it unnecessary to run the second step.

4.6.2 Shopping street experiment

To test if the gravitational sensor of mobile devices is accurate enough to use for localization, 101 query images were captured on a shopping street using an iPhone 4. From another set of 412 images, covering the same street, a 3D model was built using the method in [82]. The query images were localized in the 3D model using Algorithm 8 and 9. The gravitational vector was captured by the internal sensors in the phone.

The results are evaluated by counting the number of inliers, and by visually verifying the inlier correspondences as well as the position on the street. In all cases the computed camera pose was visually correct and in 100 of the images there were at least 12 inliers. In one case there were only 10 inliers but the pose was still correct.

The experiment also shows that very high rates of outliers can occur in practice. Due to a significant difference in lighting conditions, the feature matching was unusually difficult; see Figure 4.6. To get any correct matches the SIFT matching ratio was increased to 0.95. Naturally this produced a lot of erroneous correspondences; see Figure 4.7.

Figure 4.6: One of the SLR images used for building the model (left) and one of the iPhone test images (right). Note the illumination difference.

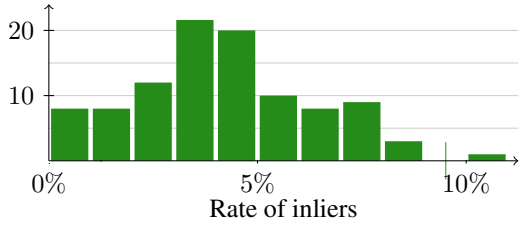


Figure 4.7: Rate of inliers for the 101 query images in the shopping street experiment.

4.6.3 Semi-synthetic experiment

Since we have no way of obtaining ground-truth positions for the shopping street experiment, we also constructed a semi-synthetic setup. The same 3D model and correspondences as in the shopping street experiment were used, but the image points were recomputed to have control over the noise. For each image, a subset of 10 correct image points was selected. Gaussian noise with standard deviation 0.005 was added to the calibrated points and the gravitational vector was corrupted with a uniform noise angle on $[0, 1°]$.

Both for 3-point RANSAC, 2-point (with known vertical direction [59]) RANSAC and Algorithm 9, exhaustive sampling of all the minimal subsets was performed. The localization errors in meters for the three methods are compared in Figure 4.8. In most cases the methods work well, but the RANSAC methods are more likely to produce large errors.

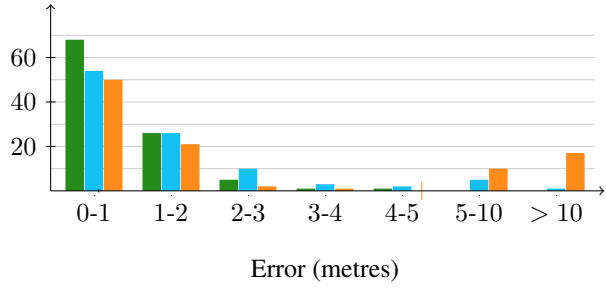


Figure 4.8: Histogram over the localization error of the proposed method (green) compared to exhaustive 2-point and 3-point RANSAC (blue and orange) for the semi-synthetic experiment.

4.6.4 Timing comparison

Another experiment was performed on a subset of the test images from the Dubrovnik dataset. Taking a number of images, all but 10 inliers were removed. Then Algorithm 8 was run with varying number of outliers. The execution times as a function of the number of outliers can be seen in Figure 4.9. As a comparison, we have run a 3-point pose solver (implemented

in C) and a 2-point (plus up-direction) pose solver (implemented in Matlab), in RANSAC loops. The number of RANSAC iterations was chosen such that the probability of getting at least one outlier-free minimal set was 0.99. Algorithm 8 is much faster than RANSAC for high rates of outliers. This is expected as our outlier removal runs in $\mathcal{O}(n^2 \log n)$ compared to 3-point RANSAC which increases as $\mathcal{O}(n^4)$ for this experiment. Comparing execution times to the 2-point RANSAC is unfair since the implementation used is very inefficient. Simulations of a C-implementation indicate that the solver is approximately as fast as Algorithm 8; being faster for low rates of outliers and slightly slower for higher rates. Naturally the polynomial solvers of Section 4.5 will also require some time. But as the number of inliers is small and all but a few outliers are removed by the rejection step, this is very negligible in this case.

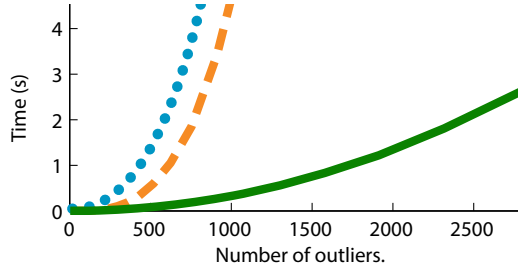


Figure 4.9: Execution times for a 3-point RANSAC loop (orange dashed line), a 2-point RANSAC loop (blue dotted line) and the proposed outlier removal step (green line). The x-axis shows the number of outliers. The number of inliers was fixed to 10.

4.6.5 A note on errors in orientation

Modern MEMS accelerometers are incorporated in many of today's handheld devices such as mobile phones and tablet computers. These accelerometers make it possible to measure the gravitational vector when the device is stationary. The sensitivity of such measurements has increased over the last years, and the typical accuracy is around 1mg which corresponds to an error less than 0.1°. However there is also a zero g level error offset which

is typically about 1°. This can in some cases be calibrated away, if the same device is used. If there is a slight motion during the capture of the image, the accelerometer will not correctly measure the gravitational vector, but this can be compensated for, using the gyroscopes that are present in most devices. These typical error values are from the ST Electronics sensor LIS331DLH used by iPhone 4, see [1] for details.

In our setup these errors can easily be incorporated by increasing the size of the error cones. If we decompose the errors in orientation, in tilt and roll angles, errors in tilt will be most significant. For the roll angle the impact of the error will increase for points farther from the centre of the image.

4.7 Conclusions

We have in this chapter presented a pose estimation framework that can handle large amount of outliers in the data. It assumes knowledge about the orientation of the camera relative to the ground plane. This information is readily available for many practical applications using e.g., cameras mounted on vehicles or hand held devices such as smart phones with gravitational sensors. The experiments show that using this information we improve both localization accuracy and robustness to outliers.

Chapter 5

Extensions for the planar case

In applications such as city-scale localization it is not very likely that the camera height is exactly known. There are, however, other applications where this is the case, e.g., a vehicle moving in a building. In this chapter we consider modifications to our localization approach to adapt it to this setting.

Naturally, the outlier rejection scheme from Section 4.4 can be used directly, although for the applications mentioned above it is less likely to have huge rates of outliers. Just as with unknown height, we can minimize the number of outliers by going through the set of certain critical points. The next section discusses how to find these critical points in the planar case. This chapter is based on [104].

5.1 Finding the critical points

First we decide on a goal function f on the parameter space. Normally a linear goal function will yield the simplest equations. We need to solve the following problems:

In the case when the height is known we need to register a number of points to conic sections as described in Section 4.4. We have a planar rotation and a 2D translation to estimate and hence we need three points. The three equations are of the form

$$v_i'^T C_i v_i' = 0, \quad v_i' = \begin{pmatrix} a & -b \\ b & a \end{pmatrix} v_i + \begin{pmatrix} t_x \\ t_y \end{pmatrix}. \tag{5.1}$$

These are in general full second degree polynomials in the four variables (a, b, t_x, t_y), i.e., we can write them as

$$C_{4\times 15} X = 0, \tag{5.2}$$

with

$$X^T = [a^2 \;\; ab \;\; at_x \;\; at_y \;\; b^2 \;\; bt_x \;\; bt_y \;\; t_x^2 \;\; t_x t_y \;\; t_y^2 \;\; a \;\; b \;\; t_x \;\; t_y \;\; 1]. \quad (5.3)$$

Together with the embedding constraint $a^2 + b^2 = 1$, this yields a system of four quadratic equations. By multiplying with monomials up to degree three we get a new system of equations from which the action matrix can be computed. This system then gives 16 solutions, but rarely more than 8 real-valued ones. We have implemented a fast solver where the most time-consuming step is doing a QR factorization of a 110×98 matrix.

For the two-point solver we again use a linear goal function, $f = a$, to get as simple equations as possible. In addition to the the constraints from the two points and the embedding constraint we get the final fourth constraint from the determinant of

$$M = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 2a & 2b & 0 & 0 \\ m_{31} & m_{32} & m_{33} & m_{34} \\ m_{41} & m_{42} & m_{43} & m_{44} \end{bmatrix}, \quad (5.4)$$

where again $m_{31}, \ldots, m_{44}$ are linear polynomials in the four unknowns. The determinant is equal to $2b \cdot p$ where p is a full second degree polynomial in the four variables. Combining this equation with the two registration constraints (5.1) and the embedding constraint we end up with exactly the same equation structure as for the 3-solver and hence the same solver can be used

5.2 Truncated L_2-norm

In [4], it is shown that robust estimation under truncated L_2-norm can be performed if we can produce the following requirements

- Solvers to compute all critical points.
- A solver for optimizing the ordinary L_2-norm for a given set of residual functions.

These ideas are applied to stitching and 2D-2D registration. We will show how to address the camera pose problem in a similar fashion. We have already discussed how to deal with the first requirement and proceed directly

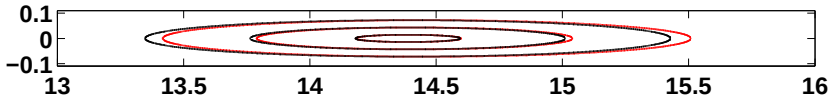


Figure 5.1: Contour plot for the exact reprojection errors (red) and the approximated version (black). The levels 1 pixel, 3 pixels and 5 pixels are shown for a 3D point 15 meters away and at 1 meters height.

to the second. Again we use the reformulation as a 2D-2D registration problem, but we now need to analyze this reformulation more closely.

Recall (4.1)

$$\lambda u = S \begin{pmatrix} U' \\ h \end{pmatrix} = \begin{pmatrix} RU + t \\ h \end{pmatrix}, \tag{5.5}$$

As we know h and the camera orientation with respect to the plane $z = h$, we can easily compute the unique point in this plane that would yield zero reprojection error. Let $\bar{c}$ denote this point. Ideally we would like to map U exactly to $\bar{c}$, which corresponds well to the registration problem. However, as no exact mapping for all points exists, we want to compute a solution that minimizes the L_2-norm of the reprojection errors—and this is more tricky. The mapping from the plane $z = h$ to reprojection errors is too complex to allow for a simple solution. But if we approximate this mapping, e.g., using its Taylor expansion, we get a more tractable problem. To clarify, consider an image point u_k and a plane $z = h$. Let $\bar{c}$ be the unique point in $z = h$ that yields zero reprojection error. We form a function $\rho(x, y)$ such that $\rho(U')$ is the reprojection error of a point $\begin{pmatrix} U' \\ h \end{pmatrix}$. Then the second-order Taylor expansion of ρ at $\bar{c}$ can be written

$$\rho(U') \approx \left(U' - \bar{c}\right)^T A \left(U' - \bar{c}\right). \tag{5.6}$$

Figure 5.1 shows the accuracy of this approximation for a difficult case.

5.2.1 Approximate-L_2 solver

The basis for being able to solve the truncated L_2-norm optimization above is a solver for standard L_2. With the reformulation and Taylor ap-

proximation above this problem can be viewed as minimizing

$$\ell(R,t) = \sum_i \left(U_i' - \bar{c}_i\right)^T A_i \left(U_i' - \bar{c}_i\right) \tag{5.7}$$

where A_i is a symmetric 2×2-matrix and

$$U_i' = RU_i + t. \tag{5.8}$$

To eliminate t, look for stationary points

$$\nabla_t \ell = \sum_i A_i \left(RU_i + t - \bar{c}_i\right) = 0. \tag{5.9}$$

This yields t as a linear function of R. By reinserting this into (5.7), we get the full loss as a quadratic function of $\cos\theta$ and $\sin\theta$. Straightforward analysis shows that the stationary points to this loss function can be found by solving a degree-four equation, and the minimum can then easily be found.

Algorithm 10 Truncated L_2

Given a set of image points u_i and 3D point U_i estimate a planar pose that minimizes the approximate truncated L_2-norm of the reprojection errors.

Transform the problem to a point-to-conic registration.
For each triplet of correspondences:
  Use the 3-solver to estimate R and t.
  Find the neighbouring inlier sets; see [4].
  For each neighbouring inlier set:
    Use the approximate L_2-solver to estimate R and t.
    Evaluate loss function and update the best solution.
For each pair of correspondences:
  Use the 2-solver to estimate R and t.
  Same steps as above.

5.3 Experiments

In order to test algorithm 10 we have conducted another synthetic experiment. In this setup we chose to just compare the results from algorithm 9

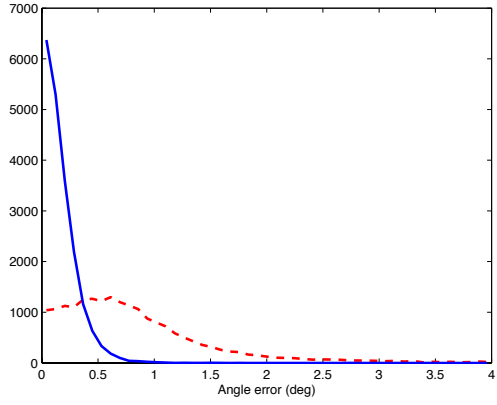


Figure 5.2: Histogram of the angular errors compared to ground truth. Red dashed lines are without running algorithm 10 and the blue lines are with.

(adapted for known height) with results from algorithm 10 without any outliers in the test data. The results can be seen in figure 5.2 and figure 5.3 where respectively histograms of the angle and position errors are shown. Red dashed lines are without running algorithm 10 and the blue lines are with. One can clearly see that both the angular and positional errors decrease significantly when using algorithm 10 .

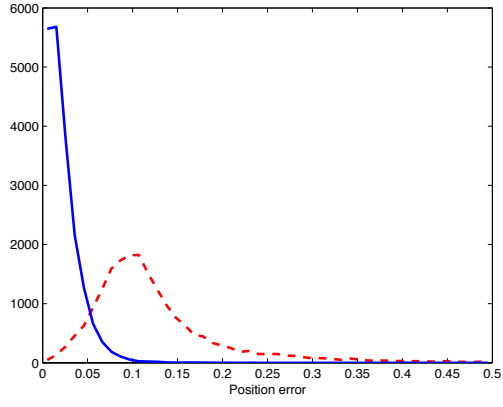


Figure 5.3: Histogram of the positional errors compared to ground truth. Red dashed lines are without running algorithm 10 and the blue lines are with.

Part II

Registration

Chapter 6
The registration problem

6.1 Image registration

Image registration is a classical problem in computer vision and it appears as a subroutine for many imaging tasks. For example, it is a prerequisite for shape analysis and modeling [22] and for automated analysis of multimodal microscopy images [61]. It is also an important component in image guided surgery where fiducial markers often are used for estimating the transformation [35, 26].

Image registration is the problem of transforming different sets, or types, of image data into a common coordinate system. The data can for example be several photographs, data from different imaging devices, from different viewpoints or of different objects. Registration is necessary in order to compare or integrate the data obtained from these measurements.

It is common to pick one image as the reference, and then apply geometric transformations to the other images so that they align with the reference. The transformation T_θ applied to a point x in the coordinate system of one image, produces a transformed point x',

$$x' = T_\theta(x), \tag{6.1}$$

for the coordinate system of the reference image, where θ is the parameter vector defining the transformation. If the point x corresponds to a point y in the reference image, the registration error or residual is simply the Euclidean distance

$$r(\theta) = ||y - T_\theta(x)||_2. \tag{6.2}$$

Image registration algorithms can be classified based on their transformation models. A broad category of transformation models are the rigid

transformations; including rotation, scaling and translation. One often includes affine transforms that incorporate shear and directional scaling in this group . Rigid transformations are global in nature, meaning that they are not able to model local geometric differences between images. The second category of transformations allow elastic, or nonrigid, transformations. These transformations are capable of locally warping the images for finer alignment to the reference image.

Image registration algorithms can also be classified into intensity-based and feature-based methods. Intensity-based methods compare intensity patterns in images using different metrics. Feature-based methods find correspondences between image features such as points or contours and use these to estimate a transformation.

The following chapters introduce new methods for image registration.

- In Chapter 7, two new feature-based registration methods for rigid transformations in the plane are presented, based on the L_1-norm and the truncated L_1-norm. They exhibit worst-case complexity $\mathcal{O}(n^3)$ and $\mathcal{O}(n^3 \log(n))$, respectively, where n is the number of correspondences.

- In Chapter 8, we make a case for feature-based registration of 3D medical images. Fast outlier rejection methods for rigid and affine transformations are presented that, together with random sampling, makes it possible to handle a large amount of outliers at reasonable computational cost.

- In Chapter 9, a method for 2D nonrigid registration is presented. The method is based on discrete multi-label optimization.

Naturally, the registration problem has been studied in depth. When choosing the method of preference, one is often faced with the dilemma of choosing between fidelity, robustness and efficiency. Using a simplified, mathematical model of the problem enables efficient computations, but sacrifices realism. While using a more realistic model incurs the computational cost of hard inference. As an example, consider the case of feature-based registration under the assumption that the measurement noise in the target image can be modeled by independently distributed Gaussian noise. This is in fact the standard Procrustes problem which can be solved in closed form. However, the model is not very realistic as there are typically

erroneous measurements - *outliers* - among the feature correspondences. This makes the registration estimates very unreliable. On the other hand, modeling outliers leads to a much more complicated optimization problem and solving this problem exactly is sometimes dismissed as infeasible. Heuristic methods based on random sampling and expectation maximization dominate the field. We show that one can achieve a method which is both efficient (in terms of speed) and reliable (with respect to outliers).

6.2 Related work

Closed form solutions to the standard Procrustes alignment problem have been known for a long time [46] and used in various settings, for instance, in surface alignment [8]. However, as the estimate is based on least-squares (minimum of L_2-errors), outliers will have a large influence and that makes the approach unreliable.

Much effort has been spent on finding good interest point descriptors [41, 18], but it is still difficult to avoid incorrect correspondences. Some approaches are conservative in the matching process and rely on getting no false matches [39]. This may work well for particular applications, but not in general. Robust estimators based on iterative methods have been proposed, e.g., [62], but they are sensitive to initialization.

2D affine registration. Already in [85], it is emphasized that robustness is a key issue and a multi-scale approach is proposed that integrates local measures to obtain an estimate of a rigid transformation. The method is applied to the problem of registering serial histological sections. In [78], a probabilistic method is developed that explicitly models outliers and which regards the registration problem as an inference problem. Inference is performed via expectation-maximization. The method in [34] proposes to use the Huber kernel as a residual function to make the registration less sensitive to outliers. Levenberg-Marquardt iterations are performed in order to minimize the loss function. In [88] deterministic annealing is proposed in order to optimize a robust loss function for the registration of autoradiograph slices. Yet another example is [28], where meta-heuristics is applied for the optimization step of the registration of angiograms. See also the registration survey [6]. All of these local optimization techniques are dependent on a good initial estimate and they are susceptible to local optima.

Hence, they cannot guarantee the quality of their solutions.

Another popular approach for dealing with outliers is hypothesize-and-test approaches based on RANSAC [33]. These methods are by nature random (which can be remedied by exhaustively examining all possible subsets). Still, the estimators have no guarantee of finding the optimal solution which makes these methods unreliable. This will be empirically demonstrated in our evaluation.

Several works have focused on optimal estimators based on branch-and-bound. One of the first algorithms was developed in [13] and it finds the rigid transformation that maximizes the number of inliers. In [31], a robust estimator based on a vertex cover formulation is proposed and in [66], a formulation based on integer programming is given. The methods are independent of initialization and converge to a global optimum. However, as they are based on branch-and-bound, the computational complexity of the algorithm is exponential. The most closely related work to the methods presented in Chapter 7 is [4], where a truncated L_2-norm algorithm is derived with complexity $\mathcal{O}(n^4)$. However, the runtime tends to be prohibitive (see experimental Section 7.4), making it a less tractable alternative.

3D affine registration. The most similar work to the methods presented in Chapter 8 is the feature-based method in [107]. They model both inliers and outliers in a statistical setting, and thereby can learn parameters. A drawback is that candidate transformations are generated based on a single correspondence which may severely limit the actual search space. Robustness has also been addressed in intensity-based methods, e.g., in [85], using the L_1-norm and [89], where another robust loss function is utilized. The downside is that the optimization relies on local refinement which is sensitive to initialization.

Nonrigid registration. The field of nonrigid registration is vast. For a review of nonrigid registration methods, see [24]. Another survey, focused on nonrigid medical image registration, is given in [97].

Chapter 7

Optimal 2D registration

This chapter introduces two new methods of registering 2D point sets over rigid transformations when the registration error is based on a robust loss function. In contrast to previous work, these methods are guaranteed to compute the optimal transformation, and at the same time, the worst-case running times are bounded by a low-degree polynomial in the number of correspondences. In practical terms, this means that there is no need to resort to ad-hoc procedures such as random sampling or local descent methods that cannot guarantee the quality of their solutions.

The methods is tested in several different settings, in particular, a thorough evaluation on two benchmarks of microscopic images used for histologic analysis of prostate cancer has been performed. Compared to the state-of-the-art, the results show that the methods are both tractable and reliable despite the presence of a significant amount of outliers.

This chapter is based on [5].

7.1 Introduction

We seek to develop robust registration procedures for combining information from different sources and modalities. The images may be degraded and have limited/varying fields of view. We present experimental results from two different applications.

In our first setting, we are dealing with images of the human brain and the goal is to study the perfusion of blood flow through small vessels, so-called capillaries in the white and gray matter regions of the brain. This is important for patients with hydrocephalus who are treated by placing a drainage tube (shunt) between the brain ventricles and the abdominal cavity to eliminate the high intracranial pressure. To capture the anatomy

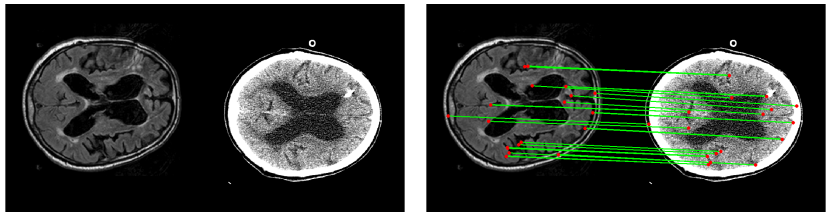


Figure 7.1: *Left:* Example slice from the MR-Flair set and the corresponding CT-slice. Note the big difference in appearance and noise level. *Right:* The correct point correspondences detected by our truncated L_1-algorithm using a threshold of 8 pixels.

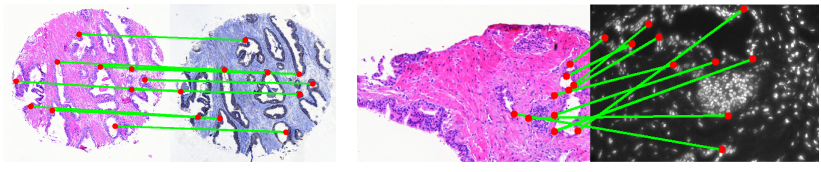


Figure 7.2: Examples from our two benchmarks with 10 manually marked correspondences. *Left:* Prostate tissue stained with H&E and p63/AMACR. *Right:* Prostate tissue stained H&E and TRF (fluorescent). The goal is to find a rigid transformation that aligns the two images using features from an automated method such as SIFT.

of the region of interest, MR-Flair images have been obtained. The perfusion data is obtained via contrast-enhanced CT images taken at one second apart during a two-minute session. To acquire good temporal resolution, only a couple of slices can be captured at each time instant. The challenge here is to register single slices from the CT image to the full 3D volume of the MR image. As the head of the patient is in an upright position, the mapping from one CT slice to the corresponding (but unknown) slice in the MR-Flair volume is well described by a rigid 2D transformation after having adjusted for known scale differences. See Fig. 7.1 for an example.

In our second setting, the objective is to perform histologic analysis of biopsies. Prostate cancer is the second most common cancer in men worldwide [49] and whose gold standard of diagnosis and prognosis is based

on histologic assessment of tumours in images stained with Hematoxylin and Eosin (H&E). Several automatic pattern recognition prototypes exist [80, 27]. In order to improve the accuracy in clinical practice, considerable research efforts have been directed to complement the analysis with additional types of stainings and imaging modalities [61]. One example is given in the left of Fig. 7.2 where two adjacent tissue sections have been stained with H&E and antibodies directed against p63/AMACR, respectively. Another example is given in the right of Fig. 7.2 with one H&E staining and one Time Resolved Fluorescence (TRF) image measuring the Androgen Receptor (AR) obtained from the same section. This type of images is quite challenging for any automated approach because reliable feature correspondences are hard to obtain and there are image degradations due to imperfect acquisition.

We develop two new robust methods for feature-based image registration based on the L_1-norm of the residual functions. As we saw in Section 2.4, from a statistical point of view, this model is well-suited for dealing with outliers. The methods are compared and extensively evaluated on one a set of CT/MR-Flair data, as well as two different benchmarks in prostate tissue samples. The focus of our evaluation is on two important properties that a satisfactory solution should possess, namely *tractability* and *reliability*. The first term refers to the computational complexity. We investigate both the performance in practice and derive theoretical complexity bounds as a function of the number of feature correspondences. The second one concerns the reliability of the estimate. We are interested in methods that produce provably optimal estimates under a robust loss function. If the registration fails, then it can be either due to lack of good correspondences or the algorithm's inability to find a good solution. In our approach, the latter source of error is removed from the process.

Two new registration methods are presented here, based on the L_1-norm and the truncated L_1-norm with worst-case complexity $\mathcal{O}(n^3)$ and $\mathcal{O}(n^3 \log(n))$, respectively, where n is the number of correspondences. Note that the algorithms we propose is restricted to rigid point set registration in the plane, and other settings are not considered here.

7.2 Fast optimization of the truncated L_1-norm

Given corresponding point coordinates in two images, $\mathbf{x}_i = (x_i, y_i)^T$ and $\mathbf{x}'_i = (x'_i, y'_i)^T$, $i = 1, \ldots, n$, consider the following problem

$$\min_{R,\mathbf{t}} \sum_{i=1}^{n} \ell(||R\mathbf{x}_i + \mathbf{t} - \mathbf{x}'_i||_1) \tag{7.1}$$

where R is a 2×2 rotation matrix and $\mathbf{t}$ a translation vector, parameterized as

$$R(\alpha) = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \text{ and } \mathbf{t} = \begin{bmatrix} t_1 \\ t_2 \end{bmatrix},$$

respectively and where ℓ is the loss function $\ell(r) = \min\{r, \epsilon\}$ for some given threshold ϵ, that is, the truncated L_1-norm.

The following observation allows us to simplify the problem.

Lemma 7.20. *For any fixed rotation R, consider the minimization of (7.1) over* $\mathbf{t}$

$$\min_{\mathbf{t}} \sum_{i=1}^{n} l\left(|x_i\cos\alpha - y_i\sin\alpha + t_1 - x'_i| + |x_i\sin\alpha + y_i\cos\alpha + t_2 - y'_i|\right). \tag{7.2}$$

Then there exist indices j and k in $\{1, \ldots, n\}$ such that

$$t_1^* = x'_j - x_j\cos\alpha + y_j\sin\alpha \text{ and } t_2^* = y'_k - x_k\sin\alpha - y_k\cos\alpha \tag{7.3}$$

is an optimal choice of $\mathbf{t}$.

In order to get a geometric intuition why the above lemma is true, consider the graph of the loss function in (7.2). Note that it is piecewise linear in $\mathbf{t}$ and a global minimum can be found by examining all break points, that is, points which are non-differentiable in all directions. There are two different causes for non-differentiability in our objective function. One is due to truncation and one is due to taking absolute values. Our proof shows that break points that are also local minima are given by (7.3). This means that break points caused by truncation need not be examined since all local minima are due to taking absolute values.

Proof. The optimal $\mathbf{t}^*$ to the truncated L_1-loss, denoted $L(\mathbf{t}^*)$, is also a global minimizer to the L_1-loss on the set of optimal inlier correspondences (those that have residuals less than ϵ). To see this, let $L_{inliers}(\mathbf{t}^*)$ be the optimal loss on the inliers and $L_{outliers}(\mathbf{t}^*)$ the loss on the outliers. Assume that there exists a different solution $\mathbf{t}$ with

$$L_{inliers}(\mathbf{t}) < L_{inliers}(\mathbf{t}^*). \tag{7.4}$$

Clearly, $L_{outliers}(\mathbf{t}) \leq L_{outliers}(\mathbf{t}^*)$ as this is already maximal. Hence

$$L(\mathbf{t}) = L_{inliers}(\mathbf{t}) + L_{outliers}(\mathbf{t}) < L_{inliers}(\mathbf{t}^*) + L_{outliers}(\mathbf{t}^*) = L(\mathbf{t}^*) \tag{7.5}$$

which is a contradiction. This shows that an optimal $\mathbf{t}^*$ is a local optimum to the L_1-loss on a subset of the residuals. The L_1-loss is given by

$$\sum_{i=1}^{n} |x_i \cos\alpha - y_i \sin\alpha + t_1 - x_i'| + |x_i \sin\alpha + y_i \cos\alpha + t_2 - y_i'|.$$

As no absolute value contains both t_1 and t_2 we can write this as a function of t_1 plus a function of t_2 and the minimization with respect to t_1 and t_2 can be analyzed separately. Consider the t_1-part. We have a piecewise linear function that tends to infinity as $|t_1|$ tends to infinity and thus a minimizer of this function is at a break point. The break points are due to the absolute values - there is a break point whenever one of the absolute values is exactly zero. Hence a minimizer exists for which at least one absolute value is zero, so $t_1^* = x_j' - x_j \cos\alpha + y_j \sin\alpha$ for some j as stated in the lemma. The same argument for t_2 proves the lemma.

□

This lemma shows that if the two indices j and k are given (for example, by exhaustively trying all possibilities), we can reduce the problem via substitution of $\mathbf{t}^*$ in (7.3) to a one-dimensional search over rotation angle α,

$$\begin{aligned} \min_{\alpha} \sum_{i=1}^{n} \ell(&|\delta x_{ij} \cos\alpha - \delta y_{ij} \sin\alpha - \delta x_{ij}'| + \\ &+ |\delta x_{ik} \sin\alpha + \delta y_{ik} \cos\alpha - \delta y_{ik}'|), \end{aligned} \tag{7.6}$$

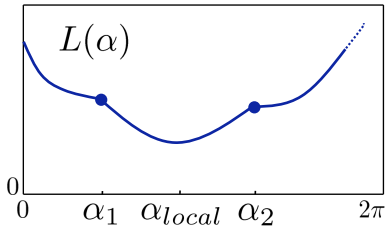


Figure 7.3: Sketch of the objective function in (7.6), denoted $L(\alpha)$, which is piecewise smooth.

where $\delta x_{ij} = x_i - x_j$, $\delta y_{ij} = y_i - y_j$, etc. Let us denote the resulting, piecewise smooth objective function in (7.6) by $L(\alpha)$, see Fig. 7.3 for an illustration. It has optimum either at a break point or at a stationary point. The break points are places where the derivative $L'(\alpha)$ is discontinuous and occur when an absolute value is exactly zero or the number in an input to ℓ is exactly ϵ. Hence the number of break points grows linearly with n. Given the break points $\alpha_1, \alpha_2, \ldots, \alpha_M$, consider an interval $[\alpha_i, \alpha_{i+1}]$ of $L(\alpha)$. It can be described by

$$L(\alpha) = w_1 \cos\alpha + w_2 \sin\alpha + w_3, \tag{7.7}$$

for some constants w_1, w_2 and w_3. By examining all intervals, we can compute the optimal rotation angle α^* using Algorithm 11.

7.2.1 Complexity

There are two important things to note here. First, that each time we compute w_1, w_2 and w_3 for $[\alpha_i, \alpha_{i+1}]$ in (7.7), we can take advantage of the constants from the previous interval $[\alpha_{i-1}, \alpha_i]$. Only the coordinates $\mathbf{x}_i$ and $\mathbf{x}'_i$ that gave rise to α_i are required for computing the update. Second, that there is only one local minimum to

$$w_1 \cos\alpha + w_2 \sin\alpha + w_3,$$

being

$$(\cos\alpha, \sin\alpha) = \pm(w_1, w_2)/\sqrt{w_1^2 + w_2^2}, \tag{7.8}$$

Algorithm 11 Finding the rotation angle

Set $L^* := \infty$.
Compute all break points of $L(\alpha)$ for $\alpha \in [0, 2\pi)$.
Sort the break points α_1, α_2, ..., α_M.
for $i = 1, \ldots, M$
    Compute $L(\alpha_i)$ and compare with L^*.
    Compute w_1, w_2 and w_3 of (7.7) for $[\alpha_i, \alpha_{i+1}]$.
    Compute local minimum α_{local} of (7.7).
    if $\alpha_{local} \in [\alpha_i, \alpha_{i+1}]$,
        compute $L(\alpha_{local})$
        compare with L^*.

given by the minus sign. Hence each step in the for-loop of Algorithm 11 is $\mathcal{O}(1)$ so the computationally heaviest step is the sorting. Given the indices j and k, we can find an optimal α^* in $\mathcal{O}(n \log n)$. If we consider all possible index pairs j and k exhaustively, the total complexity is $\mathcal{O}(n^3 \log n)$. Note that the most complex arithmetic operations in the algorithm consists of computing square roots.

7.2.2 Fast outlier rejection

To increase the speed even more we propose a fast outlier rejection step as preprocessing, inspired by the work in [4]. For this we need a variant of Algorithm 11 that works with the zero-one loss (denoted by L_0), that is, counting the number of outliers rather than truncated L_1-norm. First note that the zero-one loss has the same break points as truncated L_1 and that the loss function only changes values at these break points. There, it either increases with one or decreases with one. Algorithm 12 lists the details.

We will use this algorithm together with the following observation.

> Assume that for the optimal transformation $(R^*, \mathbf{t}^*)$, correspondence k is an inlier and there are N outliers, i.e., residuals larger than ϵ. If we change the translation to $\mathbf{t}$ so that $r_k(R^*, \mathbf{t}) = 0$, then, since $||\mathbf{t} - \mathbf{t}^*|| \leq \epsilon$, the error on inliers has increased with at most ϵ so there are at most N residuals larger than 2ϵ.

Algorithm 12 Upper bound on inliers

Initialize best loss, $L_0^* = \infty$.
Compute all break points of $L_0(\alpha)$ for $\alpha \in [0, 2\pi)$.
Sort the break points $\alpha_1, \alpha_2, \ldots, \alpha_M$.
Compute $L_0(\alpha_1)$ and update L_0^*.
for $i = 2, \ldots, M$
    Depending on the type of α_i
    Set $L(\alpha_i) = L(\alpha_{i-1}) \pm 1$ and update L_0^*.

This means that we can use Algorithm 12 with threshold 2ϵ to produce a bound of the following kind: *If correspondence k is an inlier, then there are at least N outliers.* This also yields a bound on the truncated L_1 loss, as if N residuals are $> \epsilon$, then the truncated L_1 loss is at least $N\epsilon$. If this is a higher loss than one we have already found, we can discard correspondence k from further consideration.

Algorithm 13 Fast outlier rejection

Given an upper bound L_c on the optimal loss.
for $i = 1, \ldots, n$
    Set $\mathbf{t} = \mathbf{x}_i' - \mathbf{x}_i$
    Use Algorithm 12 with threshold 2ϵ to compute L_0^*
    (The output L_0^* is a bound on the number of outliers)
    if $L_0^*\epsilon > L_c$,
        discard correspondence i

A value for L_c can be found by running Algorithm 13 using ϵ in place of 2ϵ and simply storing the best loss function value rather than discarding points. As the dominating cost inside the loop is the sorting in Algorithm 12 running this scheme to remove outliers costs only $\mathcal{O}(n^2 \log n)$ and can be used as a preprocessing step while keeping guaranteed optimality.

7.3 Fast optimization of the L_1-norm

Optimizing the L_1-norm is a simpler problem compared to the truncated case. In fact, one can set $\epsilon := \infty$ and use the same algorithm, but we can do better. Lemma 7.20 still applies, so we can eliminate the translation and only consider the rotation problem, which simplifies to

$$\min_{\alpha} \sum_{i=1}^{n} |\delta x_{ij} \cos\alpha - \delta y_{ij} \sin\alpha - \delta x'_{ij}| + \\ + |\delta x_{ik} \sin\alpha + \delta y_{ik} \cos\alpha - \delta y'_{ik}|. \quad (7.9)$$

An important difference here is that we can compute the break points for the first term and the second term *independently*. This means that we can precompute and sort all the break points for $j, k = 1, \ldots, n$ in $\mathcal{O}(n^2 \log(n))$ and then use the for-loop of Algorithm 11 to find the optimal α^*. Now, the heaviest part is no longer the sorting. The total time complexity is $\mathcal{O}(n^3)$ since the for-loop is $\mathcal{O}(n)$ and exhaustively trying all combinations of j and k is $\mathcal{O}(n^2)$.

7.4 Experiments

The proposed methods have been evaluated on two challenging registration tasks.

7.4.1 Registering CT to MR-Flair

Our first experiment is a demonstration of the applicability of the method. For more quantitative results, see Sections 7.4.2 and 7.4.3. The dataset consists of 44 image slices captured using the MR-Flair methodology and 4 image slices from a CT-scan of one single subject; see Figure 7.1. To correlate the information provided by the different modalities, one would like to register each of the CT slices to the MR-Flair volume. As the CT slices are roughly aligned with the slices of the MR-Flair volume, we can use standard 2D SIFT to obtain correspondences. To improve the matching performance all descriptors were extracted at a fixed scale (12 pixels) instead of using the estimated scale from the Difference-of-Gaussians detector. The motivation is that in very noisy images the scale estimation

tends to be uncertain. The calculated SIFT points were matched using Lowe's ratio criterion with a threshold at 0.8.

The proposed algorithms were compared to the algorithm for truncated L_2-norm from [4] as well as standard L_2-minimization and RANSAC followed by L_2-minimization on the inlier set. Their performance was evaluated using 10 manually selected correspondences by an expert. We have defined a failure case as one with is an angular error larger than 7.5° or translation error larger than 20 pixels. Experimental results for different outlier thresholds are given in Table 7.1. Due to the high noise level, good ground truth is difficult to obtain even for an expert. One image in particular resulted in significantly larger differences between the hand selected points and the computed solutions. In light of the consistent results from all methods on this pair it is our belief that the hand marked correspondences are inaccurate. A further indication is that the root mean squares error is as high as 9 pixels on the ground truth set using an L_2-optimal solver. This one image is the only failure case for the truncated methods, and it appears as a failure case for all the methods tested. The frequently used intensity-based method called NIFTYREG [85] using mutual information was also tested, but without any reasonable registration results at all. Note that this method was developed to cope with outlier structures by using robust estimation techniques.

	RANSAC									Truncated norms					
ϵ	100 iter.			500 iter.			1000 iter.			L_1			L_2		
5	5.0°	41p	75%	6.5°	22p	50%	7.2°	21p	50%	5.6°	12p	25%	5.7°	12p	25%
8	6.6°	18p	50%	3.9°	21p	50%	6.7°	15p	50%	5.4°	11p	25%	5.9°	13p	25%
12	5.5°	13p	25%	6.8°	19p	50%	6.0°	19p	50%	5.9°	9.7p	25%	5.8°	15p	25%
∞	-			-			-			12°	54p	50%	22°	142p	100%

Table 7.1: The results for the brain-image experiments. A threshold level of ∞ means that no threshold is used. See Table 7.2 for explanation.

7.4.2 Registering histology sections

The second set of experiments is concerned with the registration of histology sections of prostate tissue. We used one dataset with 88 image pairs of

adjacent slides of prostate tissue, stained using H&E and p63/AMACR, respectively. Another dataset consists of 103 images of H&E stained slides, in which sub-parts are also analyzed with TRF. Examples can be seen in Fig. 7.2. The size of the stained images are on the order of 1100x1100, while the TRF images are 368x546.

As in the previous experiment we used SIFT features. Matching was restricted to the same scale octave and we used Lowe's ratio criterion with a threshold at 0.9 to discard poor matches. This yielded 800-1500 matches for the first dataset, and, due to TRF images being smaller, 40-500 matches in the second dataset. The inlier rate varies from 1% to 40% with a 10% average for the H&E-p63/AMACR set and from 4% to 54% for the H&E-TRF set with a 28% average.

For each problem instance, 10 correspondences were manually picked by an expert and used to compute an optimal transformation under the L_2-loss. Reported results are compared to the rotation and the translation of this estimate, as in the brain experiments. We have also selected two failure criteria based on these comparisons. The first being that the rotation error is larger than 5°, the second that the translation error is larger than 25 pixels. The percentage of results that fail according to these criteria are presented.

	RANSAC									Truncated norms					
ϵ	100 iter.			500 iter.			1000 iter.			L_1			L_2		
1p	11°	204p	48%	8.4°	221p	27%	8.1°	105p	28%	2.7°	61p	11%	2.5°	58p	8%
5p	14°	280p	42%	2.9°	53p	9%	1.9°	28p	5%	1.2°	7p	3%	0.43°	6.4p	2%
10p	7.8°	158p	28%	1.2°	42p	6%	2.2°	43p	6%	0.29°	4.8p	1%	0.28°	4.6p	1%
20p	4.0°	78p	18%	2.4°	34p	8%	0.9°	23p	3%	0.27°	4.0p	0%	0.26°	3.9p	0%
∞	-			-			-			2.4°	6.5p	5%	6.5°	94p	69%

Table 7.2: The results for the H&E - p63/AMACR benchmark. In the left column, the inlier threshold ϵ is varied. Then, for each of the methods (RANSAC with varying number of iterations, and the truncated L_1- and L_2-norms), three numbers are reported: average rotation error (degrees), average translation error (pixels) and failure rate. A failure case is one with error in rotation larger than 5° or in translation larger than 25 pixels. When $\epsilon = \infty$, no truncation takes place.

The experimental results on H&E-p63/AMACR are shown in Table 7.2. The most accurate results are obtained by the truncated L_2-method. Truncated L_1-norm performs poorly on the lowest threshold, but at more reasonable levels for this task, performance is similar to the truncated L_2. None of the methods based on RANSAC succeeds on all examples, although the accuracy is good at higher thresholds with 1000 iterations. We also note that regular L_1-norm (marked ∞) succeeds much more frequently than L_2-norm and with better accuracy than a majority of the RANSAC variants—on a dataset with only 10% inliers on average.

	RANSAC									Truncated norms					
ϵ	100 iter.			500 iter.			1000 iter.			L_1			L_2		
1p	2.5°	31p	6%	0.40°	6.2p	1%	0.31°	2.7p	0%	0.34°	2.9p	0%	0.31°	2.6p	0%
2p	2.3°	31p	5%	0.29°	2.6p	0%	0.27°	2.7p	1%	0.29°	2.6p	0%	0.28°	2.6p	0%
3p	1.8°	23p	4%	0.29°	2.7p	0%	0.29°	2.7p	0%	0.28°	2.6p	0%	0.28°	2.6p	0%
4p	0.66°	8.5p	3%	0.28°	2.6p	0%	0.28°	2.6p	0%	0.28°	2.6p	0%	0.27°	2.6p	0%
5p	1.1°	7.4p	2%	0.26°	2.5p	0%	0.26°	2.5p	0%	0.27°	2.5p	0%	0.26°	2.6p	0%
10p	0.76°	7.8p	1%	0.27°	2.4p	0%	0.27°	2.4p	0%	0.26°	2.5p	0%	0.26°	2.4p	0%
∞	-			-			-			16°	173p	57%	34°	341p	100%

Table 7.3: The results for the H&E - TRF benchmark. See Table 7.2 for explanation.

Results from the benchmark experiment on H&E-TRF registration are shown in Table 7.3. This dataset has significantly fewer matches per image pair and higher inlier ratios, making it more suitable for RANSAC. With 1000 iterations, RANSAC performs on par with truncated L_1-norm and truncated L_2-norm, but with fewer iterations there are still some failures. The poor results for regular L_1-norm and L_2-norm show that for this task, aligning a sub-image to a larger image, using truncated norms is essential.

7.4.3 Speed

The theoretical worst time complexities are stated in Table 7.4. In practice RANSAC is not run exhaustively but with a fixed number of k iterations, giving a complexity of $\mathcal{O}(nk)$. For average-size problems (280 matches) and $k = 1000$, RANSAC required 73 ms. The fastest (but worst-

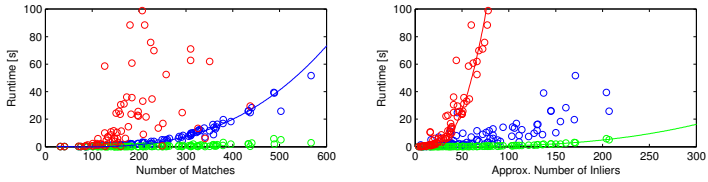


Figure 7.4: *Left:* Runtime as a function of number of matches is graphed for truncated L_1-norm (green), truncated L_2-norm (red) and regular L_1-norm. The L_1-method follows closely a $\mathcal{O}(n^3)$-curve (blue). *Right:* Runtime as a function of number of inlier matches is graphed. The truncated methods are more correlated to the number of inliers, see the $\mathcal{O}(n^4)$-curve (red) and the $\mathcal{O}(n^3 \log(n))$-curve (green), respectively. ($\epsilon = 10$ pixels.)

performing) method is the closed-form L_2-method with a typical runtime of 0.2 ms. For the remaining methods timing plots are shown in Fig. 7.4. Because of the fast outlier rejection scheme discussed in Section 7.2.2, runtimes of truncated L_1-norm and L_2-norm depend mainly on the size of the inlier sets. The full L_1-method has no such advantage. These numbers clearly show the advantage in runtime for the truncated L_1-method over both the regular L_1-norm and the truncated L_2-norm. However, on datasets consisting of a majority of inliers, the lower complexity of the L_1-norm would give faster runtimes as all operations are identical apart from the sorting strategies. The timing statistics is from experiments on H&E-TRF, though the same analysis holds for H&E-p63/AMACR.

7.5 Discussion

So what is the right way to attack feature-based image registration in presence of outliers? The literature provides us with a vast amount of choices, but many of these are based on local optimization and require a reasonable starting solution, which means that the outlier problem is already more-or-less solved. To handle really difficult outlier problems, RANSAC-type algorithms are the standard against which others are measured. However, as our experiments show, they are sub optimal both in terms of accuracy and with respect to the risk of failure. Some of the failures could be avoided by increasing the number of iterations - even up to exhaustively searching

Algorithm	*complexity*	*tractability*	*reliability*	*reference*
RANSAC	$\mathcal{O}(n^3)$	high	medium	[33]
Truncated L_1-norm	$\mathcal{O}(n^3 \log(n))$	high	high	this paper
L_1-norm	$\mathcal{O}(n^3)$	high	medium	this paper
Truncated L_2-norm	$\mathcal{O}(n^4)$	medium	high	[4]
L_2-norm	$\mathcal{O}(n)$	high	low	[46]

Table 7.4: Characteristics of the algorithms presented or discussed in the paper. Note that the stated complexity for RANSAC is for exhaustive selection of all minimal subsets which can be thought of as a worst time complexity bound.

all the minimal subsets. But, that will increase the complexity to $\mathcal{O}(n^3)$, being practically the same as the algorithms proposed here (Table 7.4). More importantly, even then there is no guarantee as to the solution quality (Fig. 7.5). Hence, we would only recommend RANSAC when the amount of outliers is known to be low and the available runtime is very limited.

This contrasts sharply to the typical setting for medical image registration where the process is performed offline. With different image modalities, the rates of outliers are usually high. In these cases the increased reliability of optimizing a truncated norm is valuable and the L_1-based methods, although slower than RANSAC, should be efficient enough for most applications. Our experiments indicate only a small gain in accuracy

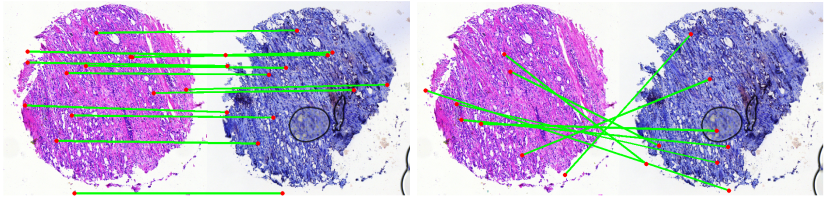


Figure 7.5: *Left:* 13 inliers among 1179 hypothetical SIFT matches of the truncated L_1-method (*success*). *Right:* 8 inliers of RANSAC with 1000 iterations (*failure*). This was the hardest case to register among all pairs. ($\epsilon = 20$ pixels.)

for the truncated L_2-norm, so using the truncated L_1-norm would be the general recommendation.

In many applications, the actual improvement in terms of accuracy and failure rates of these methods might not be huge. This is compensated by the value of removing a possible error source and not having to tune the parameters of the algorithm. We believe that the choice between a tractable, reliable algorithm with guaranteed high-quality solutions and a fast algorithm with no guarantees whatsoever should be an easy one.

Chapter 8

Reliable 3D registration

In this chapter, new feature-based methods for rigid and affine image registration are presented. These are compared to state-of-the-art intensity-based techniques as well as existing feature-based methods. On challenging datasets of brain MR and whole-body CT images, improvements in terms of speed, robustness to outlier structures and dependence on initialization is shown.

8.1 Introduction

Image registration is at the core of many applications in medical imaging. It serves as a tool for performing motion correction, detecting anatomical changes and for fusing information from different modalities. It has also become an important component in atlas-based segmentation methods, whose success in, e.g., neuroimaging is largely due to the ability to accurately register brain MRI:s. However, a number of issues need to be addressed in order to improve the applicability in other domains where the biological inter-variation is more significant. Specifically, the following challenges are address here: (i) Efficiency. In atlas-based segmentation, one needs to register each atlas image to the target and hence, the approach risks being time-consuming, or alternatively, one is forced to reduce the size of the atlas. (ii) Robustness to outlier structures. In many scenarios, one would like the registration method to ignore certain regions as they are not present in both images, for example, due to lesions, movements or varying field of view. (iii) Reliability. Many registration methods are dependent on a good initialization, that is, they are dependent on a close-by starting point in order to converge to a good solution, otherwise they risk getting trapped in a poor, locally optimal registration.

Unlike many popular registration methods which are intensity-based,

we will follow a feature-based approach. The main contributions is to show that we can improve state-of-the-art in terms of the above-mentioned challenges via a feature-based approach. In terms of methodology, we present new techniques for fast outlier rejection, which in combination with random sampling, strategies such as RANSAC [33], makes it possible to handle a huge amount of outliers at a reasonable computational cost. We handle the problem of inter-subject variability, but do not consider registration across different modalities. We are also restricting our efforts to low-dimensional transformations (rigid and affine) and we do not focus on sub-pixel accuracy. If this is required, one can refine our solution using standard nonrigid registration methods. We have tested our method on several 3D datasets.

In computer vision, random sampling techniques, e.g., RANSAC is often used for robust estimation of low-dimensional models. We will improve on this technique and adapt it to registration by incorporating an outlier rejection scheme, which is guaranteed not to remove any potential inliers, and at the same time, being computationally very efficient. This makes our approach very robust to high rates of outliers. This means that we can optimize our interest point detector so that it finds as many correct correspondences as possible, and not as in standard methodology like SIFT, where the ratio of inliers and outliers is considered.

There are several standard registration toolkits publicly available and we will compare our approach to several of them, including IRTK [90], NIFTYREG [85] and the feature-based method in [107]. We also include the robust approach in [89], which has proved more robust than the registration methods FLIRT [50] and SPM [21].

8.2 Features and matching

This section describes the pipeline we use for feature extraction and matching.

In 2D image analysis, rotation-invariant feature descriptors like SIFT [73] and SURF [7] are standard 2D tools, both of which have been adapted to 3D [3, 39]. The path we choose for 3D features is similar, though not identical to either of SIFT and SURF. Revisit Section 2.1 for a description of feature detectors and descriptors. The features we use work as follows.

(i) Interest point detection. Like SIFT we obtain interest points from

scale-space extrema of a difference-of-Gaussians operator. This produces a set of interest points in scale space. The detected scale will be used to achieve scale invariance in feature matching. The Gaussian scale pyramid used to compute difference-of-Gaussians is stored and reused in the descriptor computation.

(ii) Orientation assignment. We achieve rotational invariance by assigning a dominant orientation to each interest point which is common practice and this is standard procedure in feature-based techniques. In 2D, it is sufficient to find the dominant direction and align the feature histograms accordingly. In 3D, however, we need to assign a full 3D orientation to each interest point. To do so, we follow [3] and first find a dominant direction by considering gradients around the feature point. Then, all gradients are projected on a plane orthogonal to the dominant direction to define a second orthogonal direction. In cases with known (or negligible) orientation differences between source and target, one can assign the canonical orientation to all interest points. This improves the discriminative power of the descriptor and allows for more efficient computations.

(iii) Descriptor computation. A patch is defined around the interest point concerned. The patch is aligned to the orientation assignment and its size is proportional to the interest point scale. The patch is divided into $4 \times 4 \times 4$ regions and in each region, gradients are computed. These are used to compute the SURF descriptor being a list of 6 values for each region. These $64 \times 6 = 384$ values form the descriptor vector that can be used to recognize similar interest points.

(iv) Feature matching. There will always be outliers among the matches. In computer vision, random sampling techniques, e.g., RANSAC is often used for robust estimation of low-dimensional models. We will improve on this technique and adapt it to registration by incorporating an outlier rejection scheme, which is guaranteed not to remove any potential inliers, and at the same time, being computationally very efficient. This makes our approach very robust to outliers - we can handle extreme rates of outliers.

8.3 Transformation estimation

Our main contribution is a technique to deal with large rates of incorrect matches—outliers. Given n feature points $x_j \in \mathbb{R}^3$ in the source image

and corresponding $y_j \in \mathbb{R}^3$ in the target image, we seek a transformation T that minimizes the truncated L_2 loss of the residuals

$$\ell(T) = \sum_{j=1}^{n} \min(r_j^2(T), \epsilon^2), \tag{8.1}$$

where $r_j(T) = |T(x_j) - y_j|$ is the Euclidean error in the target image. The function in (8.1) assigns a quadratic loss to inlier correspondences with a residual lower than ϵ (which should relate to the noise level) and a constant loss to outlier correspondences with a residual higher than ϵ. We will consider transformations on the form $T(x) = Mx + t$. If M is a rotation matrix, we get a rigid transformation and if it is a positive definite matrix, we have a general affine transformation.

Our approach is a generalization of [4] which was developed for multiview geometry. The first step is an outlier rejection algorithm with running time that is *independent* of the rate of outliers. In practice, it will remove most of the outliers while *guaranteed* not to remove any inlier correspondences. In the second step, we use RANSAC [33] to get rid of a few remaining outliers and estimate an accurate transformation.

8.3.1 Outlier rejection

We propose two outlier rejection schemes for different settings. Our starting point is Problem 8.21.

First, we will consider maximizing the number of inliers, and then adapt it to the truncated L_2 loss in (8.1).

Problem 8.21. *Find a transformation given by (M, t) such that $|Mx_j + t - y_j| \leq \epsilon$, for as many j as possible.*

The algorithm loops through all n correspondences and performs an outlier test for each correspondence. If the test is positive, then the correspondence can be removed permanently since it is guaranteed not to be part of the optimal solution. Let L be the number of inliers of the best solution found so far. We compute a bound of the following type. If correspondence K, (x_K, y_K), is an inlier to the (unknown) optimal transformation, then there are no more than U_K inliers. Hence, if $U_K < L$ we get a contradiction and correspondence K must be an outlier. Essential is

of course to be able to quickly compute the upper bound U_K under the assumption that correspondence K is an inlier. This issue will be the focus for the remainder of this section.

The first step is reducing the original problem to a simpler one by eliminating the translation t.

Problem 8.22. *Given K, let $\tilde{x}_j = x_j - x_K$, $\tilde{y}_j = y_j - y_K$ and find M, such that $|M\tilde{x}_j - \tilde{y}_j| \leq 2\epsilon$ for as many j as possible.*

Proposition 8.3.1 If correspondence K is an inlier to Problem 8.21, then Problem 8.22 has at least as many inliers as Problem 8.21.

Proof. Let (M, t) be the solution to Problem 8.21. For any inlier j, we have $|Mx_j + t - y_j| \leq \epsilon$, and using the same M in Problem 8.22, we get

$$\begin{aligned} &|M\tilde{x}_j - \tilde{y}_j| = |M(x_j - x_K) - (y_j - y_K)| \\ &\leq |Mx_j + t - y_j| + |Mx_K + t - y_K| \leq 2\epsilon. \end{aligned}$$

□

Hence, solving Problem 8.22 produces an upper bound U_K. As a by-product, we get candidate solutions that we can use to continuously improve the best solution found so far, which will make the outlier test more efficient as the L increases.

8.3.2 Outlier rejection for rigid registration

For rigid transformations ($M = R$), we also make use of the dominant directions assigned to each feature point by the descriptor. Let u_j be the dominant direction of point j in the source image and v_j the corresponding direction in the target. As these are used to align the descriptors it is unlikely for a correct match not to satisfy the following angular error bound

$$\rho_j(R) = \angle(Ru_j, v_j) \leq \tau, \tag{8.2}$$

for a moderate threshold τ. We will further simplify our problem to a 1D-rotation problem (rotation around one axis).

Problem 8.23. *Find a rotation R with $Ru_K = v_K$ such that the following constraints are satisfied for as many j as possible.*

$$||\tilde{x}_j| - |\tilde{y}_j|| \leq 2\epsilon, \tag{8.3}$$

$$\angle(Ru_j, v_j) \leq 2\tau, \tag{8.4}$$

$$\angle(R\tilde{x}_j, \tilde{y}_j) \leq \tau + \alpha, \tag{8.5}$$

where α is obtained from $|\tilde{x}_j|^2 + |\tilde{y}_j|^2 - 2|\tilde{x}_j||\tilde{y}_j| \cos\alpha = 4\epsilon^2$.

Proposition 8.3.2 If correspondence K is an inlier to the optimal transformation of Problem 8.21 that also satisfies (8.2), then Problem 8.23 has at least as many inliers as Problem 8.21.

Proof. Let R_0 be the optimal rotation to Problem 8.22 and consider the triangle with sides $A = R_0\tilde{x}_j$, $B = \tilde{y}_j$ and $C = A - B$. Then,

$$|C| = |R_0\tilde{x}_j - \tilde{y}_j| \leq 2\epsilon, \tag{8.6}$$

and with the triangle inequality on ABC, we get (8.3). Using the rule of cosines on the same triangle we get $\angle(R_0\tilde{x}_j, \tilde{y}_j) \leq \alpha$ with α as above. To find a suitable R, we use the fact that

$$\angle(Ru_K, v_K) = \rho_K(R, t) \leq \tau. \tag{8.7}$$

Let R_Δ be a rotation with rotation angle τ mapping R_0u_K to v_K and set $R = R_\Delta R_0$. Then $\angle(R\tilde{x}_j, \tilde{y}_j) \leq |R_\Delta| + \angle(R_0\tilde{x}_j, \tilde{y}_j) \leq \tau + \alpha$ and using the triangular inequality for rotations

$$\angle(Ru_j, v_j) \leq |R_\Delta| + \angle(R_0u_j, v_j) \leq 2\tau,$$

This shows that R is a solution to Problem 8.23 with at least as many inliers as the optimal solution to Problem 8.22. Applying Proposition 8.3.1 completes the proof. □

Algorithm 14 shows how this result can be used to reject outliers. After the change of coordinates in Step 1, any rotation satisfying $Ru_K = v_K$ will be a rotation about the first coordinate axis. Hence the subsequent task is to find a rotation angle, ϕ.

Computationally, the most expensive part is Step 4 which includes sorting. Hence the complexity of Algorithm 14 is $\mathcal{O}(n \log n)$. If we repeat this for every K we get a total cost of $\mathcal{O}(n^2 \log n)$ for our outlier rejection scheme.

Algorithm 14 Upper bound U_K for correspondence (x_K, y_K)

1. Change coordinates s.t. $\tilde{u}_K = \tilde{v}_K = (1, 0, 0)^T$
2. Remove any correspondence that violates (8.3).
3. For each remaining correspondence $(\tilde{x}_j, \tilde{y}_j)$:
   3a. Compute interval I_a of rotation angles satisfying (8.4).
   3b. Compute interval I_e of rotation angles satisfying (8.5).
   3c. Intersect I_a with I_e. Store the resulting 0-2 intervals.
4. Compute an angle ϕ inside as many intervals as possible.

Output the maximum number of intersecting intervals, U_K.

8.3.3 Outlier rejection for affine registration

The method from the previous section can be used to perform outlier rejection whenever the deformation is known to be small, but some important applications require a more significant scaling to register the images, for example, registering a whole-body scan of a tall person to one of a short person. To handle such cases efficiently, we propose an outlier rejection scheme based on the assumption that the rotation between images is small. This is true for most medical 3D images, such as CT, MR or PET. The class of transformations that we will use in this case is

$$T(x) = sx + t, \tag{8.8}$$

where s is a positive scale factor. Now consider Problem 8.22 for this transformation and some index K. For each j we can compute an interval constraint on s for correspondence j to be an inlier. We find the interval by the following geometric reasoning. As s varies, the point $s\tilde{x}_j$ moves along a line segment starting at the origin. We are interested in points such that $|s\tilde{x}_j - \tilde{y}_j| \leq 2\epsilon$. It is easy to see that this is true for an interval of s and we can find the interval boundaries by using simple linear algebra

$$4\epsilon^2 = |s\tilde{x}_j - \tilde{y}_j|^2 = s^2|\tilde{x}_j|^2 + |\tilde{y}_j|^2 - 2s\,\tilde{x}_j^\tau \tilde{y}_j, \tag{8.9}$$

and solving the obtained quadratic equation. As in rigid registration, we seek an s^* that intersects as many of these intervals as possible. This can be found by sorting all the interval boundaries and going through the sorted list once. The number of intervals intersecting at s^*, U_K is an upper

bound on the form: *If correspondence K is an inlier, then there are no more than U_K inliers.* The complexity of running this scheme for all of the n correspondences is again $\mathcal{O}(n^2 \log n)$.

8.3.4 Outlier rejection for truncated least squares

For the truncated L_2 loss in (8.1), we can use these methods in the following way: Assume that correspondence K is an inlier and compute a bound, U_K, on the number of inliers. The truncated L_2 loss cannot be smaller than $(n - U_K)\epsilon^2$. If the best solution found so far has a lower loss, then we get a contradiction and correspondence K can be rejected.

8.4 Experimental results

This section presents experimental results for three different set-ups. The performance is compared to five other methods, namely IRTK, NIFTYREG, Feature-Based Alignment (FBA) from [107] (not applicable for affine registration) and the robust method in [89], hereafter called ROBUST. These methods were used with default settings including multi-resolution initializations, except ROBUST for which the outlier sensitivity was decreased for better results. Note that in [89], ROBUST was shown to perform better than FLIRT [50] and SPM [21]. Thus, these have been omitted from our experimental evaluations.

8.4.1 Rigid registration of brain MR images

In the first experiment, we quantify the robustness to outlier structures in the images using 30 T1-weighted brain MR scans from `brain-development.org` with resolution of 180^3. We use the same setup as in [89] for testing robustness. For each image, a random rigid transformation is applied to reflect possible head movements in a scanner: a random 50 voxel translation and a rotation about a random axis. We add Gaussian noise ($\sigma = 10$) and create outlier boxes of size 20^3 by copying a box from another image, see left of Fig. 8.1. We evaluate the dependence on rotation angle with 40 outlier boxes in each image and then the sensitivity to varying the number of outlier boxes with a fixed amount of rotation (30 degrees). Our framework for rigid registration was used with an outlier threshold $\epsilon = 5$ voxels for the Euclidean error and $\tau = \pi/5$ for the angular error.

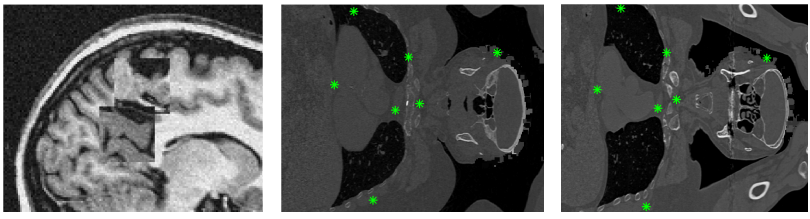


Figure 8.1: (Left) Test image from the Brain MR dataset. (Middle) Close-up of a whole-body CT image with a few of the detected feature points. (Right) Corresponding feature points automatically matched in a CT scan of another subject.

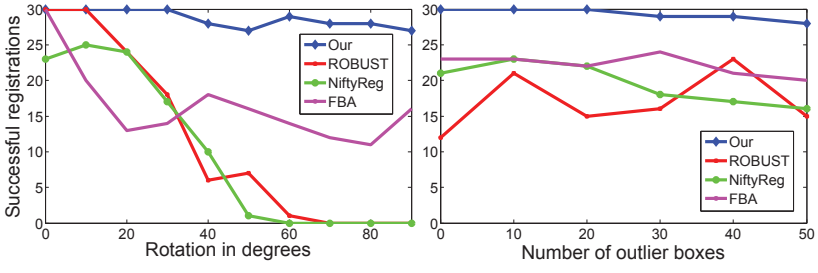


Figure 8.2: Number of successful registrations (out of 30) as a function of, (left) varying amount of rotation and 40 outlier boxes and (right) varying number of outlier boxes and 30 degrees rotation. See text for details.

Results are shown in Fig. 8.2. We plot the number of successful registrations for each method, i.e., registrations with a rotation error less than 1 degree and a translational error less than 1 voxel. In both settings, our method significantly outperforms the competitors. IRTK, which is not designed to deal with outlier patches, failed consistently and we chose to leave it out of the comparison.

8.4.2 Affine registration of organ CT images

In this experiment we used 10 whole-body CT images (resolution $512 \times 512 \times 800$) of different subjects to evaluate the robustness to initialization.

There are 15 organs, e.g., kidneys, spleen, liver, that have been manually delineated. We used cropped organ images from one subject in order to register them to the whole-body CT of a different subject using IRTK, ROBUST, NIFTYREG and our method with the outlier threshold ϵ at 30 voxels. The performance is measured by the distance in voxels between organ centres in the target and warped source images.

Table 8.1 reports the proportion of successfully registered organs. IRTK did not perform well with cropped images (as expected since it is dependent on a reasonable initialization), so we left it out of the comparison. Note that we are primarily interested in robustness (and not accuracy) and therefore we concentrate on rates of successful registrations. It is evident that ROBUST is dependent on a good initialization while NIFTYREG and our method are less so.

	Proposed	ROBUST	NIFTYREG
$\leq$ 20 voxels	71%	0%	25%
$\leq$ 50 voxels	97%	0%	90%
$\leq$ 100 voxels	100%	11%	100%

Table 8.1: Results for the multi-organ registration dataset. Reported residuals are between organ mass centres computed from manual segmentations.

8.4.3 Affine registration of whole-body CT

In this experiment we test the performance of pairwise registering whole-body CT images using the same data, settings and algorithms as in the previous experiment. We ran a number of inter-subject registrations and measured how many of the organs that were successfully registered as defined in the previous experiment. The results are given in Table 8.2. Even though the CT scans are roughly aligned from the beginning, the competing methods are less successful than ours.

8.4.4 Execution times

Average execution times for the different methods and experiments are given in Table 8.3. Even for registering a single image pair, our approach is

	Proposed	ROBUST	NIFTYREG	IRTK
$\leq$ 20 voxels	46%	18%	12%	16%
$\leq$ 50 voxels	92%	59%	69%	51%
$\leq$ 100 voxels	100%	95%	96%	72%

Table 8.2: Results for pairwise, whole-body registration. The percentage of successfully registered organs (at different thresholds) is given. In total, there are 15 segmented organs.

more efficient, but the real advantage appears in a multi-atlas setting. The most expensive step in our algorithm (and similarly for FBA) is computing features for the two images. In an atlas setting, the features can be precomputed for the atlas. As an example, the time to register one whole-body CT image to a multi-atlas of size N is $20 + 1.4N$ seconds. With $N = 50$, our method would take 1.5 minutes, IRTK over an hour, NIFTYREG about 7 hours and ROBUST 22 hours on the same computer.

	Proposed	ROBUST	NIFTYREG	IRTK	FBA
Brain	4 s	214 s	26 s	-	10 s
Organ	21 s	1614 s	527 s	-	-
Whole-body	39 s	5356 s	1448 s	222 s	-

Table 8.3: Mean execution times for one registration.

8.5 Concluding discussion

Intensity-based methods seem to be very popular for doing image registration, especially for medical applications. I believe that one contribution here is to show the advantages of feature-based image registration.

When running experiments, we struggled a lot with parameter selection for the different software toolkits, trying to make them produce good results. By comparison, the proposed methods are very easy to setup and run because they avoid a lot of parameters. Having spent much time trying to figure out the best settings for the toolkits, it is clear to me that this is an important factor. Especially if the methods are run automatically, or by people not super-interested in image analysis, like for instance radiologists.

Because of its robust properties, the method will in many cases serve well for computing initial solutions, that other methods then could refine.

One line of interesting future work could be to extend these methods for doing feature-based nonrigid registration.

Chapter 9

Shift-map-based nonrigid registration

This chapter concerns itself with the problem of nonrigid dense registration of 2D images. The problem of estimating a deformation field is formulated as a multi-label discrete optimization problem, using the Shift-map framework. Shift-map image processing was introduced by Pritch et al. [86], who applied their framework to image inpainting, content-aware resizing, texture synthesis and image rearrangement. The work presented in this chapter extends the shift-map framework to image registration. The results obtained with shift-map registration seem promising.

This chapter is based on [106].

9.1 Problem formulation

We have a base image $\mathbf{B}(i,j)$ and an input image $\mathbf{I}(i,j)$. These two images need not have the same size. The goal is to register the pixels of the input image onto the base image using a shift-map

$$\mathbf{T}(i,j) = \Big(t_i(i,j) \ , \ t_j(i,j)\Big). \tag{9.1}$$

The pixel $\mathbf{I}(i,j)$ is registered onto

$$\mathbf{B}\Big(i + t_i(i,j) \ , \ j + t_j(i,j)\Big). \tag{9.2}$$

Figure 9.1 shows the input and base images and the resulting image obtained by moving all pixels in the input image as specified by the computed shift-map.

Each possible shift-map is assigned an energy, based on *a priori* assumptions on what a good shift-map typically looks like and how well the two images match each other. The goal is then to find the optimal shift-map, that is, the shift-map with the lowest energy:

$$E(\mathbf{T}) = \sum_{(i,j)} \left(\alpha\, E_d^{ij}(\mathbf{T}(i,j)) + \sum_{(i',j')\in\mathcal{N}(i,j)} E_s^{ij}(\mathbf{T}(i,j), \mathbf{T}(i',j')) \right), \quad (9.3)$$

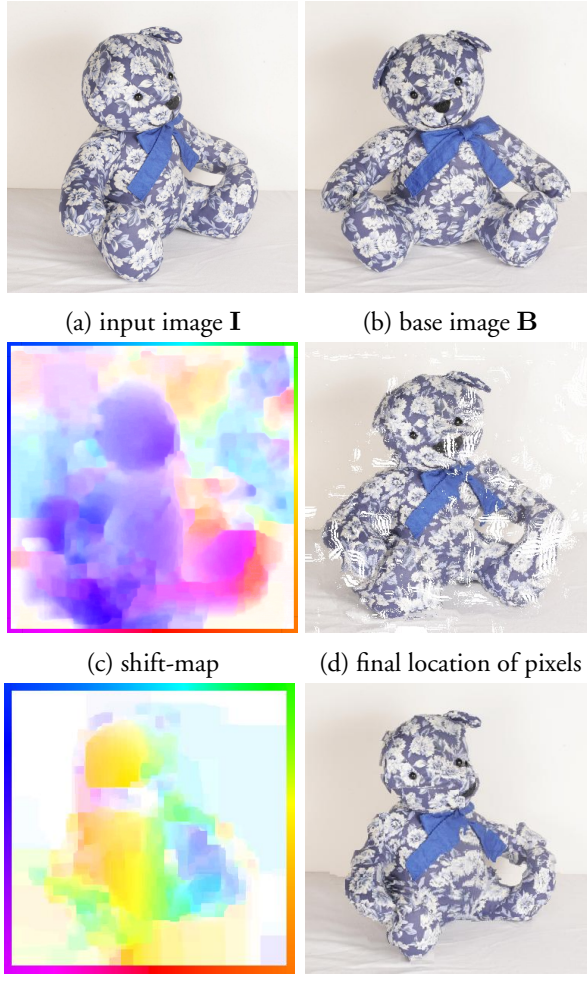
where outer sum is over all pixels in the image and the inner sum over all (i', j') in a neighborhood $\mathcal{N}(i, j)$ of (i, j). Figure 9.2 shows one such neighbor. We will use 4-connectivity of adjacent pixels exclusively. E_d^{ij} and E_s^{ij} are the data terms and smoothness terms, respectively. They will be described in separate sections below.

9.2 Registration energy terms

The methods in [86] deal with constructing a new image from an old one and the registration problem is about finding a map between two existing images. Hence the energy previously used for finding shift-maps is not suitable for registration and new energy terms must be constructed.

Comparing pixels. A related problem to image registration is dense depth estimation from two images of the same object with known camera positions. This problem has been studied extensively, see for example [56]. Recently a new descriptor, DAISY, was proposed by Tola et al. [108], tailored to dense stereo estimation where the position of the two cameras differ by a large amount. This descriptor is shown to outperform other approaches (e.g. SIFT, SURF and pixel differences) in extensive experiments. Therefore, it seems relevant to try and apply this descriptor to the related problem of estimating a dense image registration.

Not unlike SIFT [72], a DAISY descriptor constructs a histogram of the image gradient orientations. Eight different orientations at three different scales are used. By sampling these fields at different points around the feature location, a descriptor of dimensionality 200 is obtained. Since the same fields are used for all image locations, a dense field of descriptors can be computed in a couple of seconds. The main goal of the DAISY



(a) input image $\mathbf{I}$ (b) base image $\mathbf{B}$

(c) shift-map (d) final location of pixels

(e) SIFT Flow result [70] (f) SIFT Flow final locations

Figure 9.1: Registration of two images using a shift-map. Each pixel in the input image is placed on the base image as described by the shift-map. For illustration purposes, the vector-valued shift-map (image c) is visualized using colors, where different directions are coded as the border of image (c).

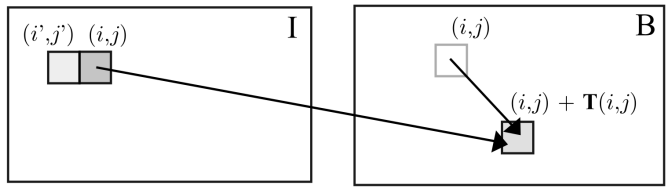


Figure 9.2: Shift-map between two images

descriptor was efficient dense computation. In order to choose relevant parameters, the work by Winder et al., [110], is helpful.

Data terms. The data terms E_d^{ij} were previously used in [86] to enforce hard constraints on the shift-map. When inpainting an image, the data term makes sure no pixels in the "hole" are used in the output image by assigning such shifts a cost of ∞.

In this paper, where image registration is considered, we need to develop more complex data terms to incorporate the fact that we want to find a mapping between two images such that similar pixels are mapped to similar pixels The data terms dictates that similar parts of the images should end up on top of each other. To measure similarity, dense DAISY is used.

It might only be possible to register parts of the input image, so shifting pixels outside the base image is permitted, at a constant cost P per pixel. The data terms are then given by

$$E_d^{ij}(\mathbf{T}) = \begin{cases} \left|\left|\hat{\mathbf{I}}(i,j) - \hat{\mathbf{B}}\Big((i,j) + \mathbf{T}(i,j)\Big)\right|\right| \\ P \text{ when } (i,j) + \mathbf{T}(i,j) \text{ is outside } \mathbf{B}, \end{cases} \tag{9.4}$$

where $\hat{\mathbf{I}}(i,j)$ is the DAISY descriptor describing the image $\mathbf{I}$ at pixel location (i,j). If the shift takes pixel (i,j) outside the bounds of the base image, a constant cost is issued. Otherwise, dissimilarity of the pixels determines the cost of the assignment. Figure 9.6f shows a heat map of the distance from the circled feature in the first row to all locations in the image in row 2.

Smoothness terms. The smoothness terms are used to enforce global consistency to the shift-map, while allowing discontinuities at a limited number of places. In [86], the smoothness terms compared the color and gradient pixel-wise. Where a discontinuity in the shift-map occurs, the penalty is computed as the difference in color and gradients.

The smoothness function takes the form of the Euclidean distance between the endpoints of the two shifts:

$$E_s^{ij}(\mathbf{T}(i,j),\mathbf{T}(i',j')) = \\ ||(i',j') + \mathbf{T}(i',j') - (i,j) - \mathbf{T}(i,j)||. \tag{9.5}$$

Here, (i,j) and (i',j') are neighboring pixels, see (9.3). We also tried using the shift difference,

$$E_s^{ij}(\mathbf{T}(i,j),\mathbf{T}(i',j')) = ||\mathbf{T}(i',j') - \mathbf{T}(i,j)||, \tag{9.6}$$

but this turned out to penalize smoothly varying shift-maps too much, as well as penalizing rotations. The proposed smoothness term in (9.5) obviously has a bias towards a contracting one image, but for reasonable smoothness weights, we never noticed this effect in practice.

Color information. The DAISY descriptor does not use color information, yet intuitively it makes little sense to match pixels of very different colors. Because of this, we have also made experiments where the color information of the images is incorporated in the above data terms. The color model used assigned a cost of P to pixels with large difference in hue, given that the intensity and saturation allowed a reliable value of the hue. This model improved the result of the registration in Fig. 9.6. We did not use color information in the experiment shown in Fig. 9.1.

9.3 Energy minimization

To minimize the energy in (9.3), *α-expansions* as described by Boykov and Veksler [12] was used, with the graph algorithms described in [11, 55].

Each possible shift value $\mathbf{T}(i,j) \in \{-m \dots m\} \times \{-n \dots n\}$ is mapped to a 1D label space. Naturally, this makes the number of labels, even for very moderately sized images, very large. To make the problem

tractable, the image is down-sampled in a Gaussian pyramid. This allows for a dramatic decrease in the number of labels needed for most cases.

For the images in Fig. 9.6, an initial image size of 128×23 was used. The size was doubled 3 times until the final resolution of 1024×179 was reached. Each doubling of the image size is followed by a linear interpolation of the shift-map. This shift-map was used as a starting guess for the optimization at the larger level. At each level after the first, only 9 possible shifts then need to be considered: $\{-1, 0, 1\}$ in each direction.

9.4 Experiments

9.4.1 Inpainting with DAISY

To verify our implementation, we inpainted an example image used in [86], see Fig. 9.5. We tried to follow their implementation as closely as possible. The result was different, but qualitatively similar. We did not allow the pixels outside the area to be removed to move at all, which is in contrast to [86], where all pixels except the border of the image were allowed to be shifted.

9.4.2 Registration

Figures 9.1 and 9.3 show shift-map registration results. The bear image in Fig. 9.1 shows the same object from two different views and is from [60]. The building images in 9.3 register correctly, except for the light pole, which is very thin and does not have a large enough data term.

We have also conducted an experiment where we used shift-maps to recover a known image deformation. The results are displayed in Fig. 9.4.

During large-scale reconstruction of a city using images taken with a cylindrical camera, we have encountered many difficult image pairs where SIFT is unable to provide useful correspondences. The top two rows in Fig. 9.6 show one of the hardest. Computed SIFT features for the two images (794 and 1019 feature points, respectively) only yielded 3 correct matches. The main reason for this was the image geometry and large, repetitive patterns. Using shift-map we obtained a dense, mostly correct map between the images. This was then used as an aid to compute SIFT correspondences. We then obtained 28 matches, of which 12 were correct. The runtime for this image was about 2 minutes.

The method has also been compared to the recent SIFT flow algorithm [70]. Both algorithms worked well for simple distortions of small magnitudes, which can be seen in Figure 9.4. However, for the other, more challenging experiments, we were not able to get any satisfactory results using SIFT flow. An example is shown in Figure 9.1.

The shift-map registration method was also compared to the optical flow algorithm described in [113]. This algorithm did not produce useful results for the street images, see bottom of Figure 9.6. Optical flow techniques are arguably not suitable for this kind of registration tasks.

9.5 Conclusion and further work

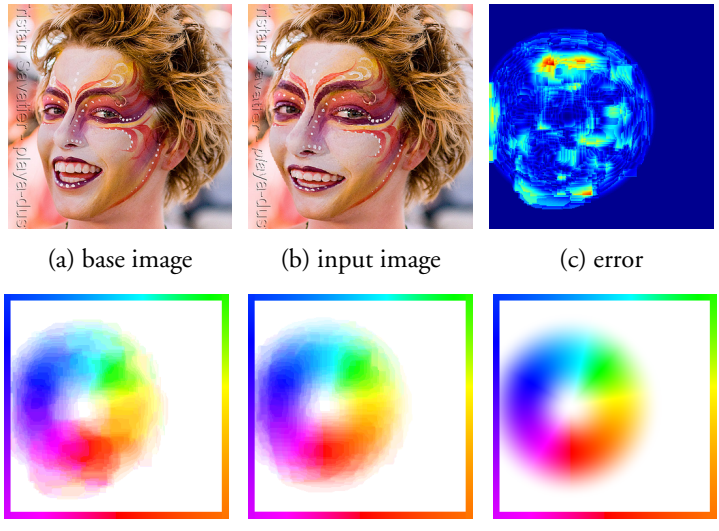
Computing the smoothness term with color and gradient differences as in [86] did not give satisfactory results when extended to image registration, but we found a great improvement with the dense DAISY descriptor. For relatively easy cases (Figs. 9.1 and 9.3), we obtained very good results. For very hard cases (Fig. 9.6) we obtained results which proved very useful for obtaining correspondences between the images. We compared shift-map registration to the optical flow algorithm described in [113] (Fig. 9.6e), which was significantly less accurate. One interesting future line of work would be to investigate whether shift-map inpainting can be improved by the DAISY descriptor as well. We have also not investigated large rotations in this paper, which would require additional considerations. One drawback of the shift-map technique is that it handles large rotations poorly. However, in many applications, i.e., registering CT- and MRI-images, rotations are small. Thus, another interesting direction would be to extend the method to registration of 3D medical images.

(a) input image **I**

(b) base image **B**

(c) final location of pixels

Figure 9.3: Registration of two images of a building.



(a) base image (b) input image (c) error

(d) Shift-map result (e) SIFT Flow result (f) Ground truth

Figure 9.4: Recovering a known image distortion. The maximum and mean error was 7.3 and 0.7 pixels, respectively. As can be seen in the images, the largest errors are in an area without texture, which is to be expected. SIFT Flow performs well in this case. Photo by Tristan Savatier obtained through Flickr.

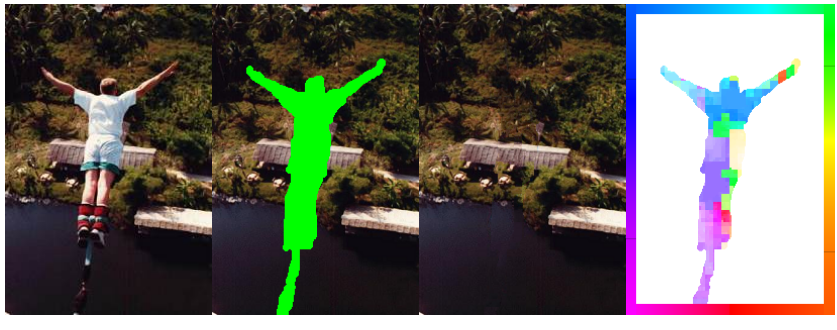
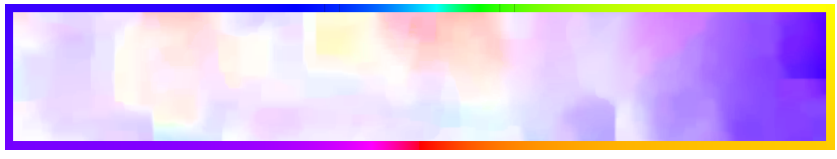


Figure 9.5: Our reimplementation of the algorithm in [86]. The complete running time for this example was 3.1415 seconds.

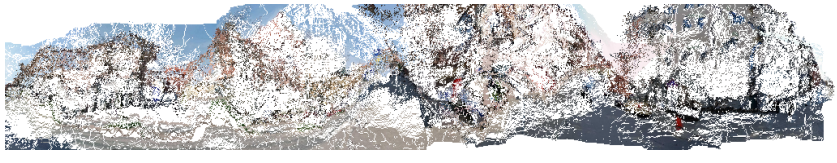
(a) input image **I**

(b) base image **B**

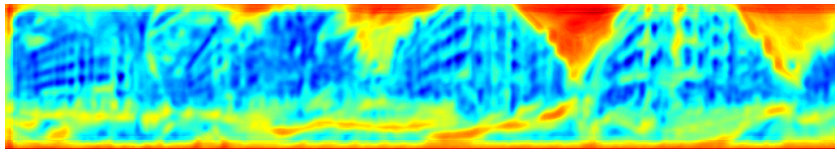
(c) final locations of the pixels in **I**



(d) resulting shift-map



(e) result using the method in [113]



(f) DAISY distance between the circled feature in **I** to all pixel locations in **B**

Figure 9.6: Registration of 1024×179 Hitta images. We note that we achieved a dense, highly nonlinear registration. This shift-map allowed us to obtain useful point-correspondences between the images, which was not possible using SIFT alone.

Bibliography

[1] Lis331dlh mems motion sensor. http://www.st.com, 2013. Cited on page 70.

[2] A.C. Aitken. On least squares and linear combination of observations. *Proceedings of the Royal Society of Edinburgh*, 1934. Cited on page 43.

[3] S. Allaire, J. Kim, S. Breen, D. Jaffray, and V. Pekar. Full orientation invariance and improved feature selectivity of 3D SIFT with application to medical image analysis. In *Conf. Computer Vision and Pattern Recognition*, Anchorage, USA, 2008. Cited on pages 100 and 101.

[4] E. Ask, O. Enqvist, and F. Kahl. Optimal geometric fitting under the truncated L_2-norm. In *Conf. Computer Vision and Pattern Recognition*, 2013. Cited on pages 53, 55, 72, 74, 82, 89, 92, 96 and 102.

[5] E. Ask, O. Enqvist, L. Svärm, F. Kahl, and G. Lippolis. Tractable and reliable registration of 2d point sets. In *European Conference on Computer Vision*. 2014. Cited on pages 2 and 83.

[6] M.A. Audette, F.P. Ferrie, and T.M. Peters. An algorithmic overview of surface registration techniques for medical imaging. *Medical Image Analysis*, 2000. Cited on page 81.

[7] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. *Computer Vision and Image Understanding*, 2008. Cited on pages 4 and 100.

[8] P.J. Besl and N.D. McKay. A method for registration two 3-d shapes. *Trans. Pattern Analysis and Machine Intelligence*, 1992. Cited on page 81.

[9] A. Blake and A. Zisserman. *Visual Reconstruction*. MIT Press, 1987. Cited on pages 11 and 12.

[10] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. Cited on page 30.

[11] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *Trans. Pattern Analysis and Machine Intelligence*, 2004. Cited on page 115.

[12] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *Trans. Pattern Analysis and Machine Intelligence*, 2001. Cited on page 115.

[13] T.M. Breuel. Implementation techniques for geometric branch-and-bound matching methods. *Computer Vision and Image Understanding*, 2003. Cited on pages 53 and 82.

[14] B. Buchberger. A theoretical basis for the reduction of polynomials to canonical forms. *ACM SIGSAM Bulletin*, 1976. Cited on page 25.

[15] P. Burt and T. Adelson. The laplacian pyramid as a compact image code. *Trans. on Communications*, 1983. Cited on page 4.

[16] M. Byröd, K. Josephson, and K. Åström. A column-pivoting based strategy for monomial ordering in numerical gröbner basis calculations. In *European Conference on Computer Vision*, 2008. Cited on pages 25 and 26.

[17] M. Byröd, K. Josephson, and K. Åström. Fast and stable polynomial equation solving and its application to computer vision. *International Journal of Computer Vision*, 2009. Cited on pages 25 and 27.

[18] W. Cheung and G. Hamarneh. n-SIFT: n-dimensional scale invariant feature transform. *Trans. Image Processing*, 2009. Cited on page 81.

[19] S. Choudhary and P.J. Narayanan. Visibility probability structure from sfm datasets and applications. In *European Conference on Computer Vision*, 2012. Cited on pages 52 and 66.

[20] O. Chum and J. Matas. Optimal randomized ransac. *Trans. Pattern Analysis and Machine Intelligence*, 2008. Cited on page 52.

[21] A. Collignon, F. Maes, D. Delaere, D. Vandermeulen, P. Suetens, and G. Marchal. Automated multi-modality image registration based on information theory. In *Information Processing in Medical Imaging*, 1995. Cited on pages 100 and 106.

[22] T. F. Cootes, G. J. Edwards, and Taylor C. J. Active appearance models. *Trans. Pattern Analysis and Machine Intelligence*, 23(6):681–685, 2001. Cited on page 79.

[23] D. Cox, J. Little, and D. O'shea. *Using algebraic geometry*. Springer Verlag, 2005. Cited on page 22.

[24] W. R. Crum, T. Hartkens, and D. L. G. Hill. Non-rigid image registration: theory and practice. *The British Journal of Radiology*, 2004. Cited on page 82.

[25] A. Dalalyan and R. Keriven. L1-penalized robust estimation for a class of inverse problems arising in multiview geometry. 2009. Cited on page 39.

[26] R.D. Datteri and B.M. Dawant. Estimation and reduction of target registration error. In *Conf. Medical Image Computing and Computer-Assisted Intervention*, 2012. Cited on page 79.

[27] S. Doyle, A. Madabhushi, M. Feldman, and J. Tomaszeweski. A boosting cascade for automated detection of prostate cancer from digitized histology. In *Conf. Medical Image Computing and Computer-Assisted Intervention*, 2006. Cited on page 85.

[28] J. Dreo, J.C. Nunes, and P. Siarry. Robust rigid registration of retinal angiograms through optimization. *Comput Med Imaging Graph*, 2006. Cited on page 81.

[29] O. Enqvist, E. Ask, F. Kahl, and K. Åström. Robust fitting for multiple view geometry. In *European Conference on Computer Vision*, 2012. Cited on page 53.

[30] O. Enqvist, E. Ask, F. Kahl, and K. Åström. Tractable algorithms for robust model estimation. *International Journal of Computer Vision*, 2014. Cited on pages 14, 17, 19, 53, 55 and 63.

[31] O. Enqvist, K. Josephson, and F. Kahl. Optimal correspondences from pairwise constraints. In *International Conference Computer Vision*, 2009. Cited on page 82.

[32] J.-C. Faugère. A new efficient algorithm for computing gröbner bases without reduction to zero (f5). In *International Symposium on Symbolic and Algebraic Computation*, 2002. Cited on page 25.

[33] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting. *Commun. Assoc. Comp. Mach.*, 1981. Cited on pages 12, 82, 96, 100 and 102.

[34] A.W. Fitzgibbon. Robust registration of 2d and 3d point sets. *Image and Vision Computing*, 2003. Cited on page 81.

[35] J.M. Fitzpatrick, J.B. West, and C.R. Maurer. Predicting error in rigid-body point-based registration. *Trans. Medical Imaging*, 1998. Cited on page 79.

[36] F. Fraundorfer, P. Tanskanen, and M. Pollefeys. A minimal case solution to the calibrated relative pose problem for the case of two known orientation angles. *European Conference on Computer Vision*, 2010. Cited on page 53.

[37] W. Fulton. *Algebraic topology: a first course*. Springer, 1995. Cited on page 18.

[38] V. M. Govindu. Robustness in motion averaging. In *Asian Conference Computer Vision*, 2006. Cited on pages 36 and 38.

[39] X. Han. Feature-constrained nonlinear registration of lung CT images. *Medical Image Analysis for the Clinic: A Grand Challenge*, 2010. Cited on pages 81 and 100.

[40] R. Haralick, H. Joo, C. Lee, X. Zhuang, V. Vaidya, and M. Kim. Pose estimation from corresponding point data. *Systems, Man and Cybernetics*, 1989. Cited on page 51.

[41] T. Hartkens, K. Rohr, and H. Stiehl. Evaluation of 3D operators for the detection of anatomical point landmarks in MR and CT images. *Computer Vision and Image Understanding*, 2002. Cited on page 81.

[42] R. Hartley, K. Aftab, and J. Trumpf. L1 rotation averaging using the weiszfeld algorithm. In *Conf. Computer Vision and Pattern Recognition*, 2011. Cited on page 38.

[43] R. Hartley and F. Schaffalitzky. L_∞ minimization in geometric reconstruction problems. In *Conf. Computer Vision and Pattern Recognition*, 2004. Cited on pages 9, 36 and 37.

[44] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004. Cited on pages 7 and 8.

[45] J. Hays and A. Efros. Im2gps: estimating geographic information from a single image. In *Conf. Computer Vision and Pattern Recognition*, 2008. Cited on page 52.

[46] B. K. P. Horn. Closed-form solution of absolute orientation using unit quaternion. *Journal of the Optical Society of America A*, 1987. Cited on pages 81 and 96.

[47] Z. Hu, U. Keiichi, H. Lu, and F. Lamosa. Fusion of vision, 3d gyro and gps for camera dynamic registration. In *Int. Conf. Pattern Recognition*, 2004. Cited on page 36.

[48] A. Irschara, C. Zach, J-M Frahm, and H. Bischof. From structure-from-motion point clouds to fast location recognition. In *Conf. Computer Vision and Pattern Recognition*, 2009. Cited on page 52.

[49] A. Jemal, F. Bray, M.M. Center, J. Ferlay, E. Ward, and D. Forman. Global cancer statistics. *CA Cancer J Clin*, 2011. Cited on page 84.

[50] M. Jenkinson, P.R. Bannister, J.M. Brady, and S.M. Smith. Improved optimization for the robust and accurate linear registration and motion correction of brain images. *Neuroimage*, 2002. Cited on pages 100 and 106.

[51] F. Kahl. Multiple view geometry and the L_∞-norm. In *International Conference Computer Vision*, 2005. Cited on pages 9 and 36.

[52] F. Kahl and R. Hartley. Multiple view geometry under the L_∞-norm. *Trans. Pattern Analysis and Machine Intelligence*, 2008. Cited on page 53.

[53] Q. Ke and T. Kanade. Quasiconvex optimization for robust geometric reconstruction. In *International Conference Computer Vision*, 2005. Cited on page 36.

[54] Q. Ke and T. Kanade. Quasiconvex optimization for robust geometric reconstruction. *Trans. Pattern Analysis and Machine Intelligence*, 2007. Cited on pages 9 and 53.

[55] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *Trans. Pattern Analysis and Machine Intelligence*, 2004. Cited on page 115.

[56] V. Kolmogorov and R. Zabih. Graph cut algorithms for binocular stereo with occlusions. In *Handbook of Mathematical Models in Computer Vision*. Springer, 2006. Cited on page 112.

[57] K. Konolige, M. Agrawal, and J. Sola. Large scale visual odometry for rough terrain. In *International Symposium on Robotics Research*, 1996. Cited on page 36.

[58] Z. Kukelova, M. Bujnak, and T. Pajdla. Automatic generator of minimal problem solvers. In *European Conference on Computer Vision*, 2008. Cited on page 27.

[59] Z. Kukelova, M. Bujnak, and T. Pajdla. Closed-form solutions to minimal absolute pose problems with known vertical direction. In *Asian Conference Computer Vision*, 2010. Cited on pages 53 and 68.

[60] A. Kushal and J. Ponce. Modeling 3D objects from stereo views and recognizing them in photographs. In *European Conference on Computer Vision*, 2006. Cited on page 116.

[61] J.T. Kwak, S.M. Hewitt, S. Sinha, and R. Bhargava. Multimodal microscopy for automated histologic analysis of prostate cancer. *BMC Cancer*, 2011. Cited on pages 79 and 85.

[62] D. Kwon, I.D. Yun, K.H. Lee, and S.U. Lee. Efficient feature-based nonrigid registration of multiphase liver CT volumes. In *British Machine Vision Conf.*, 2008. Cited on page 81.

[63] D. Lazard. Resolution des systemes d'equations algebriques. *Theor. Comput. Sci.*, 1981. Cited on page 23.

[64] M. Lhuillier. Fusion of gps and structure-from-motion using constrained bundle adjustment. In *Conf. Computer Vision and Pattern Recognition*, 2011. Cited on pages 36 and 46.

[65] H. Li. A practical algorithm for L_∞ triangulation with outliers. In *Conf. Computer Vision and Pattern Recognition*, 2007. Cited on page 53.

[66] H. Li. Consensus set maximization with guaranteed global optimality for robust geometry estimation. In *International Conference Computer Vision*, 2009. Cited on pages 53 and 82.

[67] X. Li, C. Wu, C. Zach, S. Lazebnik, and J-M Frahm. Modeling and recognition of landmark image collections using iconic scene graphs. In *European Conference on Computer Vision*, 2008. Cited on page 52.

[68] Y. Li, N. Snavely, and D. Huttenlocher. Location recognition using prioritized feature matching. In *European Conference on Computer Vision*, 2010. Cited on pages 52, 64, 65 and 66.

[69] Y. Li, N. Snavely, D. Huttenlocher, and P. Fua. Worldwide pose estimation using 3d point clouds. In *European Conference on Computer Vision*, 2012. Cited on pages 52, 65 and 66.

[70] C. Liu, J. Yuen, A. Torralba, J. Sivic, and W. T. Freeman. SIFT flow: Dense correspondence across different scenes. In *European Conference on Computer Vision*, 2008. Cited on pages 113 and 117.

[71] P. Lothe, S. Bourgeois, F. Dekeyser, E. Royer, and M. Dhome. Towards geographical referencing of monocular slam reconstruction using 3d city models:applications to real-time accurate vision based localization. In *Conf. Computer Vision and Pattern Recognition*, 2009. Cited on page 36.

[72] D. Lowe. Object recognition from local scale-invariant features. In *International Conference Computer Vision*, 1999. Cited on pages 4 and 112.

[73] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 2004. Cited on pages 5 and 100.

[74] D. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial & Applied Mathematics*, 1963. Cited on page 9.

[75] D. Martinec and T. Pajdla. Robust rotation and translation estimation in multiview reconstruction. In *Conf. Computer Vision and Pattern Recognition*, 2007. Cited on pages 9, 36 and 38.

[76] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image Vision Computing*, 2004. Cited on page 4.

[77] J. Michot, A. Bartoli, and F. Gaspard. Bi-objective bundle adjustment with application to multi-sensor slam. In *3DPVT*, 2010. Cited on page 36.

[78] A. Myronenko and X. Song. Point-set registration: Coherent point drift. *Trans. Pattern Analysis and Machine Intelligence*, 2010. Cited on page 81.

[79] O. Naroditsky, Z. Zhu, A. Das, S. Samarasekera, T. Oskiper, and R. Kumar. Videotrek: A vision system for a tag-along robot. In

Conf. Computer Vision and Pattern Recognition, 2009. Cited on page 53.

[80] K. Nguyen, B. Sabata, and A.K. Jain. Prostate cancer grading: Gland segmentation and structural features. *Pattern Recognition Letters*, 2012. Cited on page 85.

[81] D. Nistér, O. Naroditsky, and J. Bergen. Visual odometry. In *Conf. Computer Vision and Pattern Recognition*, 2004. Cited on page 36.

[82] C. Olsson and O. Enqvist. Stable structure from motion for unordered image collections. In *Scandinavian Conf. on Image Analysis*, 2011. Cited on page 66.

[83] C. Olsson, A. Eriksson, and R. Hartley. Outlier removal using duality. In *Conf. Computer Vision and Pattern Recognition*, 2010. Cited on page 53.

[84] T. Oskiper, Z. Zhu, S. Samarasekera, and R. Kumar. Visual odometry system using multiple stereo cameras and inertial measurement unit. In *Conf. Computer Vision and Pattern Recognition*, 2007. Cited on page 53.

[85] S. Ourselin, A. Roche, G. Subsol, X. Pennec, and N. Ayache. Reconstructing a 3D structure from serial histological sections. *Image and Vision Computing*, 2001. Cited on pages 81, 82, 92 and 100.

[86] Y. Pritch, E. Kav-Venaki, and S. Peleg. Shift-map image editing. In *International Conference Computer Vision*, 2009. Cited on pages 111, 112, 114, 115, 116, 117 and 119.

[87] T. Pylvanainen, L. Fan, and V. Lepetit. Revisiting the pnp problem with a gps. In *International Symposium on Visual Computing*, 2009. Cited on page 36.

[88] A. Rangarajan, H. Chui, E. Mjolsness, S. Pappu, L. Davachi, P. Goldman-Rakic, and J. Duncan. A robust point-matching algorithm for autoradiograph alignment. *Medical Image Analysis*, 1997. Cited on page 81.

[89] M. Reuter, H. Rosas, and B. Fischl. Highly accurate inverse consistent registration: a robust approach. *Neuroimage*, 2010. Cited on pages 82, 100 and 106.

[90] D. Rueckert, L. Sonoda, C. Hayes, D. Hill, M. Leach, and D. Hawkes. Nonrigid registration using free-form deformations: application to breast MR images. *Trans. Medical Imaging*, 1999. Cited on page 100.

[91] T. Sattler, B. Leibe, and L. Kobbelt. Fast image-based localization using direct 2d-to-3d matching. In *International Conference Computer Vision*, 2011. Cited on pages 52 and 66.

[92] T. Sattler, B. Leibe, and L. Kobbelt. Improving image-based localization by active correspondence search. In *European Conference on Computer Vision*, 2012. Cited on pages 52 and 66.

[93] G. Schindler, M. Brown, and R. Szeliski. City-scale location recognition. In *Conf. Computer Vision and Pattern Recognition*, 2007. Cited on page 52.

[94] Y. Seo, H. Lee, and S. W. Lee. Outlier removal by convex optimization for l-infinity approaches. In *Pacific-Rim Symposium on Image and Video Technology*, 2009. Cited on page 39.

[95] K. Sim and R. Hartley. Removing outliers using the L_∞-norm. In *Conf. Computer Vision and Pattern Recognition*, 2006. Cited on pages 39 and 53.

[96] N. Snavely, S.M Seitz, and R. Szeliski. Photo tourism: Exploring photo collections in 3d. *SIGGRAPH*, 2006. Cited on page 9.

[97] A. Sotiras, C. Davatzikos, and N. Paragios. Deformable medical image registration: A survey. *Medical Imaging, Transactions on*, 2013. Cited on page 82.

[98] B. Steder, G. Grisetti, S. Grzonka, C. Stachniss, A. Rottmann, and W. Burgard. Learning maps in 3d using attitude and noisy vision sensors. In *Intelligent Robots and Systems*, 2007. Cited on page 53.

[99] H. Stewénius, C. Engels, and D. Nistér. Recent developments on direct relative orientation. *Journal of Photogrammetry and Remote Sensing*, 2006. Cited on page 13.

[100] C. Strecha, T. Pylvanainen, and P. Fua. Dynamic and scalable large scale image reconstruction. In *Proc. Conf. Computer Vision and Pattern Recognition, Colorado springs, USA*, 2010. Cited on page 36.

[101] D. Strelow and S. Singh. Motion estimation from image and inertial measurements. *Int. Journal Robotics Research*, 2004. Cited on page 36.

[102] J. Sturm. Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones. *Optimization methods and software*, 1999. Cited on page 31.

[103] C. Sunglok, K. Taemin, and Y. Wonpil. Performance evaluation of ransac family. In *British Machine Vision Conf.*, 2009. Cited on page 13.

[104] L. Svärm, O. Enqvist, M. Oskarsson, and F. Kahl. Accurate localization and pose estimation for large 3d models. In *Conf. Computer Vision and Pattern Recognition*, 2014. Cited on pages 2, 51 and 71.

[105] L. Svärm and M. Oskarsson. Structure from motion estimation with positional cues. In *Scandinavian Conf. on Image Analysis*. 2013. Cited on pages 2 and 35.

[106] L. Svärm and P. Strandmark. Shift-map image registration. In *Int. Conf. Pattern Recognition*, 2010. Cited on pages 2 and 111.

[107] M. Toews and W.M. Wells III. Efficient and robust model-to-image alignment using 3D scale-invariant features. *Medical Image Analysis*, 2013. Cited on pages 82, 100 and 106.

[108] E. Tola, V. Lepetit, and P. Fua. Daisy: An efficient dense descriptor applied to wide baseline stereo. *Trans. Pattern Analysis and Machine Intelligence*, 2009. Cited on page 112.

[109] W. Triggs, P.F. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment—a modern synthesis. In *Vision algorithms: theory and practice*. 2000. Cited on page 9.

[110] S. Winder, G. Hua, and M. Brown. Picking the best DAISY. In *Conf. Computer Vision and Pattern Recognition*, 2009. Cited on page 114.

[111] J. Yu, A. Eriksson, T.-J. Chin, and D. Suter. An adversarial optimization approach to efficient outlier removal. In *International Conference Computer Vision*, 2011. Cited on page 53.

[112] C. Zach, M. Klopschitz, and M. Pollefeys. Disambiguating visual relations using loop constraints. In *Conf. Computer Vision and Pattern Recognition*, 2010. Cited on pages 36 and 38.

[113] C. Zach, T. Pock, and H. Bischof. A duality based approach for realtime TV-L1 optical flow. In *Pattern Recognition (Proc. DAGM)*, 2007. Cited on pages 117 and 120.

[114] C. Zach and M. Pollefeys. Practical methods for convex multi-view reconstruction. In *European Conference on Computer Vision*, 2010. Cited on pages 9 and 38.