Spherical Mosaics with Quaternions and Dense Correlation

Ph.D Seminar Report

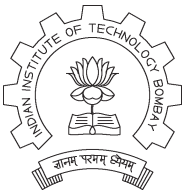
by

Rhushabh Goradia

under the guidance of

Prof. Sharat Chandran
Computer Science and Engineering
IIT-Bombay

April 26, 2005



Department of Computer Science and Engineering
Indian Institute of Technology, Bombay
Mumbai

Acknowledgment

I would like to thank my guide, **Prof. Sharat Chandran**, for his valuable guidance, consistent motivation and directions he has fed into my work.

— **Rhushabh Jitendra Goradia**

List of Figures

Abstract

The need for mosaicing arises when we want to stitch two or more images together so as to view them as a single continuous image. There are various ways to construct mosaics of images, one of them being Spherical Mosaics. Several algorithms have been designed to compute spherical mosaics. This report describes an algorithm for constructing spherical mosaics from a collection of images taken from a common optical center. It also compares two different optimization techniques (local and global) and shows why global optimization technique is much more superior to the local one. Partially overlapping images, an adjacency map relating the images, initial estimates of the rotations relating each image to a specified base image, and approximate internal calibration information for the camera form the inputs for the algorithm. The algorithm's output is a rotation relating each image to the base image and revised estimates of the camera's internal parameters.

This algorithm, based on global optimization technique, offers several advantages. First, image capture instrumentation provides both an adjacency map for the mosaic, and an initial rotation estimate for each image. Second, it optimizes an objective function based on a global correlation of overlapping image regions. Third, representation of rotations as quaternions signicantly increases the accuracy of the optimization.

Contents

Chapter 1

Introduction

The need for mosaicing arises when we want to stitch two or more images together so as to view them as a single continuous image. There are various ways to construct mosaics of images, one of them being Spherical Mosaics. As the name suggests, it allows any number of images to be merged into a single seamless view, simulating the image that would be acquired by a camera with a spherical field of view. Images shown here describes a somewhat hemispherical point of view but it can be easily extended to full spherical arrangements of images.

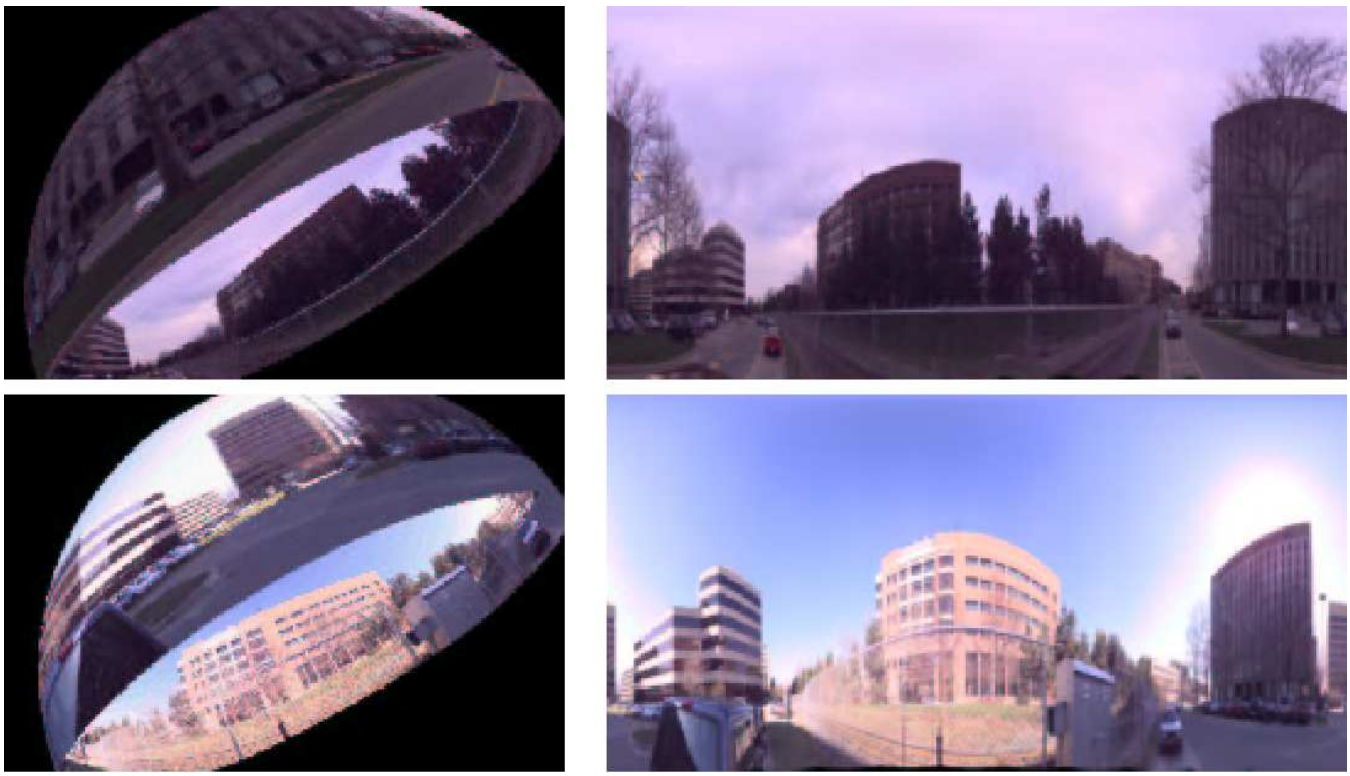


Figure 1.1: Two typical mosaics shown as sphere and cylinder

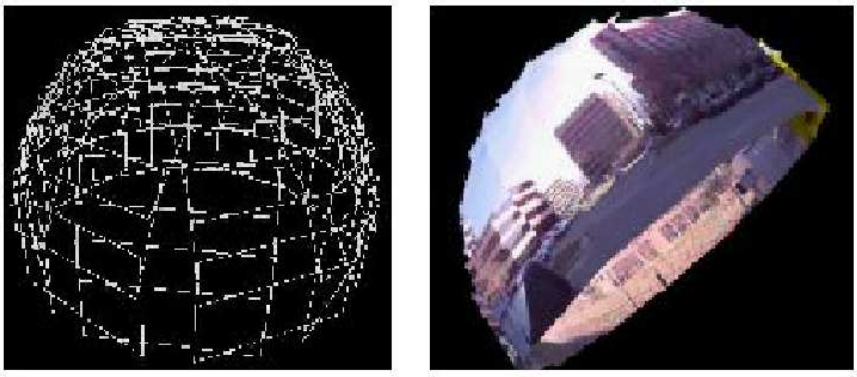


Figure 1.2: The roughly hemispherical tiling for a node of the dataset

The camera instrument used annotates each acquired image with an estimate of absolute 6-DOF pose (exterior orientation) — 3 DOF of position, and 3 DOF of orientation for the acquiring camera. Thus the acquisition system provides both an adjacency map for images in the mosaic, and an initial estimate of the rotations relating each image to its neighbors. This forms one of the advantages of using this method. But these estimates are not that accurate and calls for some *pose-refinement* algorithm. How to optimize and refine these estimates is what this report describes.

1.1 Organization of Report

In chapter two we will see what is some of the work done related to this topic of discussion. Chapter three will briefly tell us about some basic concepts of image formation via perspective projection and give an introduction to quaternions as a form of representation along with some of its nice properties. We will also see how quaternions and its properties help us in representing rotations in a very efficient way as discussed in [5] and [2].Chapter four reviews 2-D projective transformations and methods to compute them. Chapter five presents a closed-form method to decouple projective transformations into two parts, one describing the intrinsic parameters of the camera, and another describing the pure rotations to which the camera has been subjected. While theoretically elegant, this technique is sensitive to errors in image formation and generally yields poor results for real imagery due to the usage of local optimization technique for refining the parameters. Solution to the above problem is addressed in Chapter 6 which describes a global optimization technique that computes revised rotations and camera internal parameters directly from correlations among images. Constraining the optimization to manipulate pure rotations produces significantly more accurate mosaics. Concluding remarks are made in chapter seven.

Chapter 2

Related Work

Much work has been done related to this topic of discussion. Of fundamental interest in mosaic computations is the warp relating a pair of overlapping images. The simplest method to compute this warp uses four point correspondences between the two images. However, identifying suitable features and correspondences is a difficult problem, and known methods yield good results only for images with significant overlap and minimal projective distortion.

An alternative method would use correlation of color or luminance information present in images to compute the warp by nonlinear optimization (e.g., [4]). Although such techniques avoid the need for feature detection and correspondence, they do not guarantee that a series of pairwise warps will produce a globally consistent set of relative orientations as they use the local optimization technique. The example for the same is described in chapter five.

The choice of rotation also plays an important role. Various methods are available to represent rotations. Some of them are

- The **Axis-angle** representation for estimating small rotations [4]
- The **Quaternion** representation because of its convenience and compactness [5]

This algorithm makes use of the quaternion representation.

Full view spherical panoramas have been computed by [3]. However, their global alignment algorithm requires a combination of both correlation-based and feature-based optimization. In contrast, the algorithm described here optimizes correlation directly to perform global alignment, avoiding both the need to identify and correspond suitable features.

Chapter 3

A Brief Review of the Basic Concepts

3.1 Image formation by perspective projection

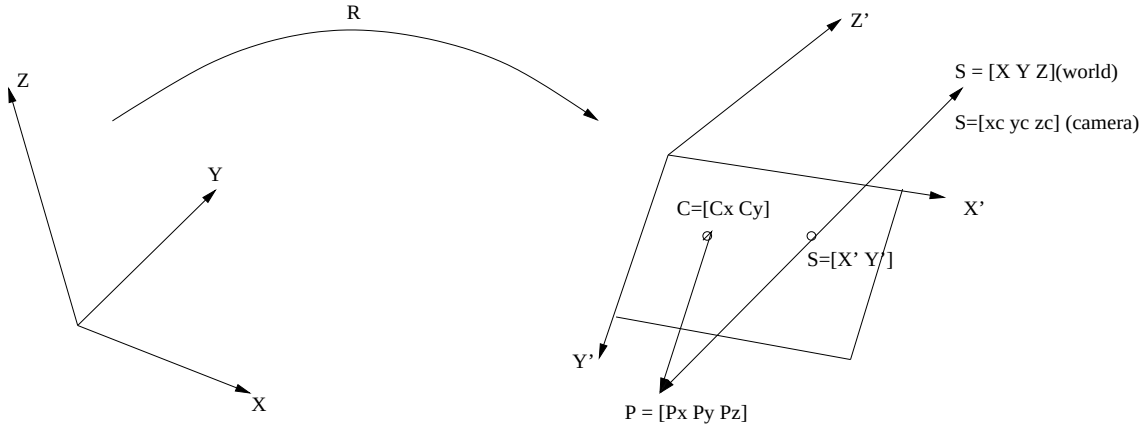


Figure 3.1: Overview of Perspective Projection

Above figure shows the process of image formation by perspective projection, illustrated for a point $s = [x\ y\ z]$ as viewed by a camera at world-space position $p = [p_x\ p_y\ p_z]$. The rotation from the global coordinate system **XYZ** to the camera coordinate system $X'Y'Z'$ is specified by a 3×3 rotation matrix R.

Now,

$$s_c = R(s - p)$$

where $s_c = [x\ y\ z]$ are the coordinates of s in the camera's coordinate system. Now we scale the x and y co-ordinates by depth as

$$x_I = \frac{x_c}{z_c} \; y_I = \frac{y_c}{z_c}$$

After scaling it, we now apply the intrinsic camera parameters to complete the transformation. The final values are:

$$x' = fx_I + c_x \quad y' = fy_I + c_y$$

Thus, the entire transformation can be represented in a matrix form as :

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} \cong KM \begin{bmatrix} R & -R_p \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{3.1}$$

where **K**the 3×3 (upper-triangular) internal camera parameter matrix:

$$\begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

and M is the 3×4 canonical perspective projection matrix:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

3.2 Introduction to Quaternions

3.2.1 Why Quaternions ?

A common problem in computer vision is solving for rigid body motions or poses consisting of a rotation and translation in 3D space. For example, given a set of points x_i and correspondences p_i, it is often of interest to compute the 3×3 rotation matrix R and 3-vector translation t such that

$$Rx_i + t = p_i$$

Although this system of equations is essentially linear, a number of problems arise when formulating solutions that account for the non-linear constraints on the components of R. The constraints arise from using nine values of rotation matrix R to represent three independent variables of 3D rotation. The rotation matrix is constrained to be orthogonal which is satisfied when $R^T R = I$ (i.e., the rows and columns are orthonormal). Also, the rotation must not be a reflection; this is satisfied when the determinant is 1 ($\mid R \mid = 1$).

A number of techniques have been developed to deal with this added complexity. One of the most convenient is the quaternions representation.

Here are some of the advantages and mathematical niceties of the quaternion representation of rotation.

- Quaternions can be composed/multiplied in a straightforward manner to accumulate the effects of composed rotations.
- The inverse of a quaternion (specifying the inverse rotation) is obtained by simply negating 3 components of the quaternion vector.
- The rotation between two rotations can be computed by multiplying one quaternion with the inverse of the other.
- One can easily transform a quaternion into an axis-and-angle representation. Using this and the previous item, one can compute a rotational distance metric between two rotations — the angle of rotation between them.
- Quaternions can be easily transformed to a 3×3 rotation matrix for efficient computation when rotating vectors.
- With the quaternion representation, the rotation can be solved for in closed form when correspondences between three-dimensional point sets are available.
- Maintaining the constraints (orthogonal with unit determinant) of rotation is made simple with quaternions by standard vector normalization.
- The unit quaternion representing the best rotation is the eigenvector associated with the most positive eigenvalue of a symmetric 4×4 matrix. The elements of this matrix are combinations of sums of products of corresponding coordinates of the points.
- Suppose that we are given the coordinates of a number of points as measured in two different Cartesian coordinate systems. The photogrammetric problem of recovering the transformation between the two systems from these measurements is referred to as that of *absolute orientation*. Let us call the two coordinate systems "left" and "right." A desirable property of a solution method is that, when applied to the problem of finding the best transformation from the right to the left system, it gives the exact inverse of the best transformation from the left to the right system. Symmetry is guaranteed when one uses unit quaternions to represent rotation.
- It is much simpler to enforce the constraint that a quaternion have unit magnitude than it is to ensure that a matrix is orthonormal.

3.2.2 What are Quaternions ?

The quaternion $\mathbf{q}$ is a four vector $[q_x,\ q_y,\ q_z,\ q_0]^T$ which is often considered as a three-vector $\boldsymbol{u} = [q_x,\ q_y,\ q_z]^T$ and a scalar $s = q_0$. Also it has the property that ${q_0}^2 + {q_x}^2 + {q_y}^2 + {q_z}^2 = 1$. Quaternion $\mathbf{q}$ is generally referred to as $[u, s]^T$ for notational simplicity.

The dot product and vector norm for quaternions is defined as usual

$$q_1 \cdot q_2 = u_1 \cdot u_2 + s_1 s_2$$
$$\mid q \mid = (q \cdot q)^{-\frac{1}{2}}$$

Multiplication is defined over quaternions as

$$q_1 q_2 = [[s_1 u_2 \ + \ s_2 u_1 \ + \ u_1 \times u_2],\ s_1 s_2 \ - \ u_1 \cdot u_2]$$

The complex conjugate of a quaternion is defined by negating the vector component and is denoted $\overline{q} = [-u, s]^T$. The complex conjugate of a unit quaternion, $|q| = 1$, is the inverse of the quaternion with respect to multiplication, i.e., $q\overline{q} = q_I$, where $q_I = [0, 0, 0, 1]^T$. Also $qq_I = q_I q = q$ which is why q_I is referred as the identity quaternion.

3.2.3 Rotations as Quaternions

A unit quaternion $\mathbf{q}$ can be used to perform a rigid rotation of a vector $\mathbf{x} = [x, y, z]^T$ by two quaternion multiplications

$$x' = q \begin{bmatrix} x \\ y \\ z \\ 0 \end{bmatrix} \overline{q},$$

where the scalar component of $\mathbf{x}$ is simply set to zero. Observe that quaternion multiplication is not commutative; this is consistent with the fact that general three-dimensional rotations do not commute; however, quaternion multiplication is associative and distributive.

Working from this definition of quaternion rotation, one can derive a formula for the corresponding orthogonal (Euclidean) 3×3 rotation matrix from a unit quaternion

$$R_u(q) = \begin{bmatrix} ({q_0}^2 + {q_x}^2 - {q_y}^2 - {q_z}^2) & 2(q_x q_y - q_0 q_z) & 2(q_x q_z + q_0 q_y) \\ 2(q_y q_x + q_0 q_z) & ({q_0}^2 - {q_x}^2 + {q_y}^2 - {q_z}^2) & 2(q_y q_z - q_0 q_x) \\ 2(q_z q_x - q_0 q_y) & 2(q_z q_y + q_0 q_x) & ({q_0}^2 - {q_x}^2 - {q_y}^2 + {q_z}^2) \end{bmatrix}$$

The subscript u in R_u is used to denote that this is the rotation matrix when given a unit quaternion. Given an arbitrary quaternion, R_u would no longer be unitary but rather a scaled rotation matrix.

Chapter 4

2-D Projective Transformations

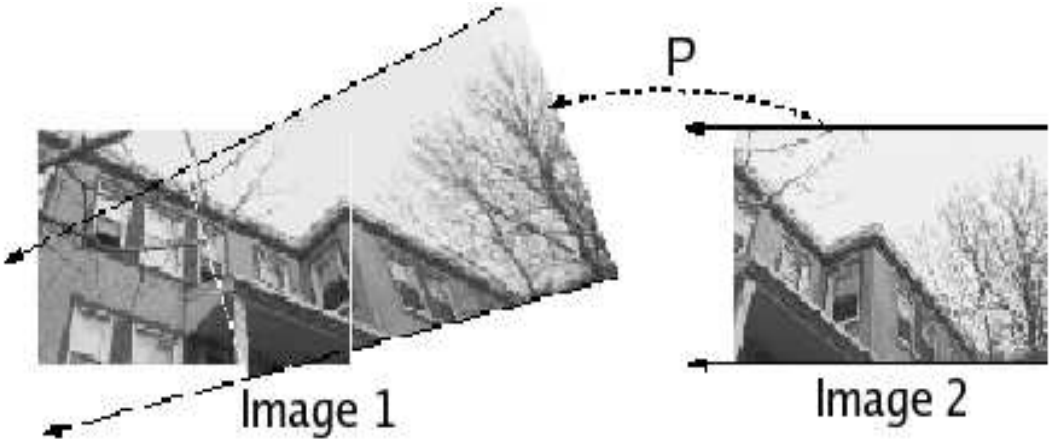


Figure 4.1: Transforming pixels from image 1 to space of image 2

Figure illustrates the relationship between two images taken from a fixed optical center, but with differing orientations. In such cases, pixels in one image can be mapped to the other image by a 2-D projective transformation [1]. As stated in [1], *Given a pair of images taken by cameras with the same internal parameters from the same location, then there is a projective transformation P taking one image from the other. Furthermore, P is of the form $P = KRK^{-1}$ where R is a rotation matrix and K is the calibration matrix. Unlike simpler 2-D transformations (translation, rotation, affine), the projective transformation does not preserve parallel lines. This is evident in the above figure, where the lines bounding image 1 intersect after transformation.*

As depth effects do not occur across two images taken from the same optical center [1, 4], the general perspective projection (Equation 3.1) simplifies to:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \cong KR \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{4.1}$$

Inverting Equation 4.1 yields:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} \cong R^{-1}K^{-1} \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \tag{4.2}$$

Above equation converts image co-ordinates to 3-D. Thus pixel coordinates in image 2 can be obtained by projecting back into image 2's space using Equation 4.1:

$$\begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} \cong KR_2R_1^{-1}K^{-1} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} \tag{4.3}$$

Thus the 2-D projective transformation that maps pixel (x_1, y_1) of image 1 to pixel (x_2, y_2) of image 2 is :

$$P = KR_2R_1^{-1}K^{-1} \tag{4.4}$$

As a consequence, only eight parameters are needed to describe the matrix $\mathbf{P}$. Thus 2-D projective transformations are also known as *8-parameter warps*.

Chapter 5

Computing Warps and Orientation Estimates with Local Optimization

5.1 Computing Warps

This section presents a novel idea to compute the warps [4]. The idea is to compute a warp that (locally) minimizes image-space error by using nonlinear optimization after having the initial estimates. The error function for this optimization simply measures the difference in brightness between two images 1 and 2 (in the overlap region), after pixels in image 1 are mapped to image 2's space. The difference in brightness is measured by a sum-of-squared difference (SSD) error metric using the luminances **L1** and **L2** of images 1 and 2, respectively:

$$E_{12} = \sum_{x_1,y_1}(L_1(x_1, y_1) - L_2(P(x_1, y_1)))$$

The SSD form is well suited for numerical optimization, as only first order derivatives are required to compute updated values for the iteration.
We define a single error term as

$$e^2_{x_1,y_1} = (L_1(x_1, y_1) - L_2(x_2, y_2))^2$$

The optimization consists of analytically determining derivatives of the above term with respect to **P**. It use Levenberg-Marquardt (LM) non-linear optimization technique for the same. The derivative $\frac{\partial e_{x_1,y_1}}{\partial P}$ is expressed as an 8-component vector consisting of derivatives w.r.t each entry of P.

The steps are:

- Initialise the warp with value that is closed to optimum. Several techniques have been proposed for this step. In this algorithm, the initial rotation estimates are provided by the acquisition instrumentation itself and we also know the approximate camera calibration. It is thus straightforward to compute a good initial estimate using Equation 4.4 with the internal parameter matrix determined by camera calibration and rotations supplied by physical instrumentation.
- The overall gradient term **G** is computed by accumulating over all error terms:

$$G = -\sum_{x_1,y_1} e_{x_1,y_1} \frac{\partial e_{x_1,y_1}}{\partial P}$$

- Similarly, the Hessian term corresponding to two adjacent images 1 and 2 is:

$$H = -\sum_{x_1,y_1} \frac{\partial e_{x_1,y_1}}{\partial P} \left(\frac{\partial e_{x_1,y_1}}{\partial P}\right)^T$$

- Thus, the optimization proceeds by incrementing the value of P by

$$\boxed{\Delta P = -(H + \lambda I)^{-1} G}$$

where λ is a stabilization parameter set initially to a high value, and reduced to 0 as the optimization converges.

The following figures shows images in which the first image represents image after the initial estimates of the warp.Note that incorrect transformations arising from inaccurate estimates of camera pose result in mismatches between pixels, causing the blurring and ghosting as seen. The second image is the image after optimization.

Figure 5.1: Part(a) shows one image of a hemispherical tiling blended with its adjacent images. Part(b) illustrates blurring due to incorrect pose estimates. Part(c) shows the same view after optimization.

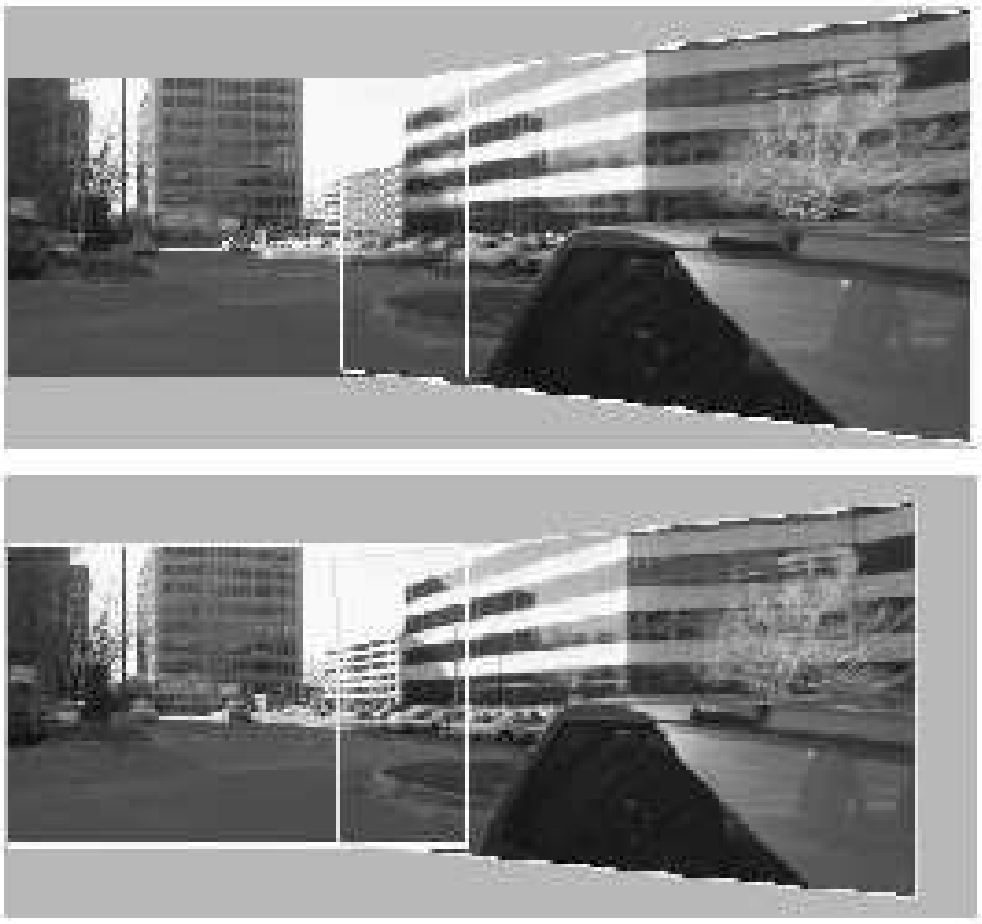


Figure 5.2: The projective warp between two images before and after optimization

5.2 Orientation Estimates from Warps

After having calculated and optimized the warps we now can recover rotation from warps using Equation 4.4 if the camera calibration is known accurately. If not, the following closed-form solution can be used to derive camera calibration from the warp itself.

5.2.1 Closed-Form Solution for Internal Parameters

We have the following equation with us:

$$P = KR_2R_1^{-1}K^{-1}$$

Let $R = R_2R_1^{-1}$. Therefore the equation becomes

$$R = K^{-1}PK$$

if we solve for R. Since $R = R^{-T}$ (the orthonormality condition for rotations), we have,

$$K^{-1}PK = K^T P^{-T} K^{-T}$$

Let $C = KK^T$. Therefore we get,

$$\boxed{PC = CP^{-T}}$$

were

$$C = \begin{bmatrix} f^2 + c_x^2 & c_x c_y & c_x \\ c_x c_y & f^2 + c_y^2 & c_y \\ c_x & c_y & 1 \end{bmatrix}$$

But how do we obtain matrix C ?

Now, the solution for **C** can be expressed in terms of eigen-vectors of the projective matrix **P**. The eigen-values of **P** are the same as that of **R**, since they are related by a *similarity transform*. We know the eigen-values of the rotation **R** are 1, $e^{i\theta}$, *and* $e^{-i\theta}$, where θ is the angel of rotation. Let e_0, e_1 *and* e_2 be the eigen-vectors of **P** corresponding to these three eigen-values, respectively. Therefore the most general symmetric solution to **C** is

$$C = c_0 e_0 e_0^T + c_1(e_1 e_2^T + e_2 e_1^T)$$

The coefficients c_0 *and* c_1 are solved by enforcing the constraints that the $C_{33} = 1$ and $C_{12} = C_{13}C_{23}$.

After having known **C** we can now easily estimate internal camera parameter matrix **K** and then the refined and optimized rotation matrix **R**.

5.3 Demerits of above discussed Technique

From emperical evidences it has been seen that this technique of estimating rotations seems to be inaccurate. This can be observed in form of large gaps between image borders although within the region of overlap, the two warps appear identical.

Figure 5.3: Image after local optimization

At first glance, it might appear that the problem is due to incorrectly recovered internal parameters. However, even though a different set of internal parameters might yield other rotations, the angle of rotation is completely determined by the eigen-values of the projective warp $\mathbf{P}$. As this is unaffected by the method by which internal parameters are computed, this **problem is inherent in the warp solution itself.** This calls for imposing a rotational constraint during the optimization to obtain quantitatively correct results and remove the gaps found in the image resulting from the above discussed technique.

Also, more fundamentally, relying on local pairwise warps to compute global quantities can lead to inconsistencies in the computed internal parameters and rotations, and more gaps in the mosaiced images. Thus, we need to go for a global optimization procedure.

Chapter 6

Computing Warps and Orientation Estimates with Global Optimization

The optimization described in this chapter produces the best possible rotations for each image, given initial estimates. The advantage of this approach is that global consistency is guaranteed by computing a unique rotation for each image. That is, the pairwise rotations inferred from this representation have the property that the aggregate rotation along any cycle in the image adjacency map is the identity. In this manner, our representation avoids the possibility of gaps arising from inconsistent pairwise estimates.

The approach followed is to optimize a global correlation function defined for adjacent images with respect to all orientations (represented as quaternions). As a by-product, the algorithm computes a spherical mosaic, a composite of all images corresponding to a single node.

6.1 Optimization

The algorithm minimizes a **Global Error Function**:

$$\boxed{E = \sum_{i,j} E_{ij} + E_{ji}}$$

where E_{ij} is the *SSD* error between luminance values of adjacent images i and j.

$$E_{ij} = \sum_{x_i, y_i} (L_i(x_i, y_i) - L_j(P_{ij}(x_i, y_i)))^2$$

and P_{ij} maps coordinates of image i to those of image j. This correlation function is computed only for pairs of adjacent images in the spherical tiling, and only for pixels of image i that map to a valid pixel of image j. As in the pairwise warp estimation, this function is minimized by computing *derivatives* w.r.t each orientation and using **LM Nonlinear Optimization** starting from the initial orientations.

The various steps of computation are:

- The single error term for images i and j is given by:

$$e^2_{x,y} = (L_i(x,y) - L_j(x", y"))^2 \quad \text{with} \quad x" = \frac{x'}{z'} \;\; y" = \frac{y'}{z'}$$

and

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = v' = P \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = KR'R^{-1}K^{-1} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- The derivatives for the above error term are calculated as follows:

$$\frac{\partial v'}{\partial q} = KR'(\frac{\partial R^{-1}}{\partial q})v$$

where

$$v = K^{-1} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- Then, the *derivative* of the term $e_{x,y}$ w.r.t the quaternion $\mathbf{q}$ is given by:

$$\frac{\partial x"}{\partial q} = \frac{\frac{\partial x'}{\partial q} - x" \frac{\partial z'}{\partial q}}{z'} \quad \frac{\partial y"}{\partial q} = \frac{\frac{\partial y'}{\partial q} - y" \frac{\partial z'}{\partial q}}{z'}$$

- and finally we have ...

$$\frac{\partial e_{x,y}}{\partial q} = \frac{\partial L_j}{\partial x"} \frac{\partial x"}{\partial q} + \frac{\partial L_j}{\partial y"} \frac{\partial y"}{\partial q}$$

- These *derivatives* $\frac{\partial L_j}{\partial x"}$ and $\frac{\partial L_j}{\partial y"}$ are approximated using the following convolution matrices applied at $(x", y")$

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

The **Gradient** term corresponding to the quaternion q_i is computed over all terms that depend on q_i:

$$G_i = \sum_{x_i,y_i} e_{x_i,y_i} \frac{\partial e_{x_i,y_i}}{\partial q_i}$$

It is computed in the coordinates of image i, w.r.t the quaternion q_i.Similarly, the **Hessian** term corresponding to two adjacent images i and j is:

$$H_{ij} = \sum_{x_i,y_i} \frac{\partial e_{x_i,y_i}}{\partial q_i} \left(\frac{\partial e_{x_i,y_i}}{\partial q_i}\right)^T$$

Now, in an unconstrained optimization, the increments would be computed as $-H^{-1}G$. Applying these increments directly to the q_i, however, would produce **Non-Unit Quaternions** which do not correspond to pure rotations !!

To constrain the updated quaternions to be unit vectors, we enforce the following additional constraints on the increments

$$\delta q_i \; : \; \forall i : q_i \cdot \delta q_i = 0$$

Using **Lagrange multipliers** λ_i to enforce these constraints, the equation for computing the increments becomes:

$$\begin{bmatrix} H & Q \\ Q^T & 0 \end{bmatrix} \begin{bmatrix} \Delta Q \\ \Lambda \end{bmatrix} = - \begin{bmatrix} G \\ 0 \end{bmatrix}$$

where

$$Q = \begin{bmatrix} q_1 & 0 & \dots & 0 \\ 0 & q_2 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & q_n \end{bmatrix}, \Delta Q = \begin{bmatrix} \delta q_1 \\ \delta q_2 \\ \vdots \\ \delta q_n \end{bmatrix}, \Lambda = \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_{4n} \end{bmatrix}$$

The optimization solves the above equation for ΔQ and Λ. Convergence is detected when the value of the objective function changes by less than some threshold.

6.2 Optimization of Internal Camera Parameters

In addition to estimating orientations, the algorithm also performs an optimization on the internal camera parameters. The overall optimization alternates between a step that updates all rotations, and a step that updates internal parameters of the camera. The new parameters are computed using derivatives of v' w.r.t the camera focal length $\mathbf{f}$ and image principal point (c_x, c_y) as shown:

$$\frac{\partial v'}{\partial f} = \left(\begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} R'R^{-1}K^{-1} + KR'R^{-1} \begin{bmatrix} -\frac{1}{f^2} & 1 & \frac{c_x}{f^2} \\ 0 & -\frac{1}{f^2} & \frac{c_y}{f^2} \\ 0 & 0 & 0 \end{bmatrix} \right) \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\frac{\partial v'}{\partial c_x} = \left(\begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} R'R^{-1}K^{-1} + KR'R^{-1} \begin{bmatrix} 0 & 0 & -\frac{1}{f} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \right) \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\frac{\partial v'}{\partial c_y} = \left(\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} R'R^{-1}K^{-1} + KR'R^{-1} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{f} \\ 0 & 0 & 0 \end{bmatrix} \right) \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

And here is the optimized image...

Figure 6.1: Projective warp between two images after direct optimization

6.3 Relative Importance of Camera Parameters

Are all the internal parameters equally important for this optimization? A simplified analysis shown below tells us that determining the **focal length** accurately is more important than determining the coordinates of the image principal point.

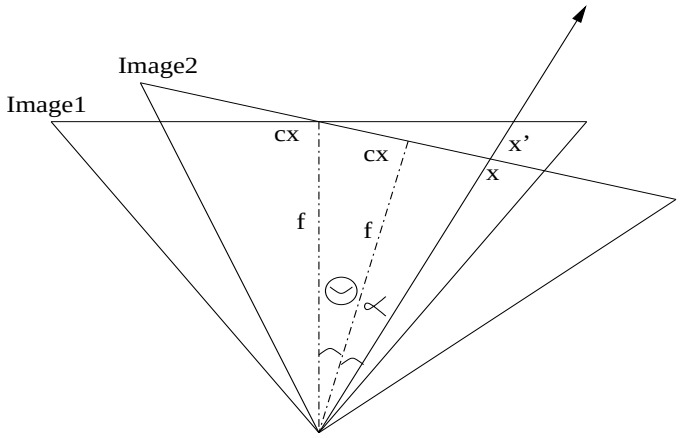


Figure 6.2: Rotation and camera parameters in 2-D

Figure shows two 1-D images rotated by an angle θ.Here the transformation between pixel x (with offset angle α from the center) in *image 2* to pixel x' in *image 1* is given by:

$$x' = c_x + f\tan(\theta + \alpha) = c_x + f\frac{\tan\theta + \tan\alpha}{1 - \tan\theta\tan\alpha}$$

Now $\tan\theta\tan\alpha \ll 1$ for small angles.Thus

$$x' \approx c_x + f\tan\theta + \tan\alpha = c_x + f\tan\theta + x - c_x = f\tan\theta + x$$
$$x' = f\tan\theta + x$$

Thus we see, to 1^{st} order, the mapping is **insensitive to the principal point (c_x, c_y) and more sensitive to the focal length f**. Thus the image center can be used as an initial value for optimization !!!

So we have seen how a global optimization function and representing rotations as quaternions helps in reducing the gaps between images and gives a very much acceptable composite image as its output.

Chapter 7

Concluding Remarks

7.1 A Brief Review of the Paper

Two methods for optimization of the spherical mosaic formed from the initial estimates were described. The first method use the local optimization technique while the second used the global optimization technique.

Here are a few advantages the above reviewed algorithm provides:

- Global Optimisation technique provides better and accurate results than the local optimisation technique used previously.
- It provides robust automatic estimation of internal camera parameters as a by-product.
- The results are quite independent of the errors in estimating the initial principal-point estimate.
- It produces an image with an effectively super-hemispherical field of view, eliminating the ambiguity between camera translation and camera rotation found in narrow field-of-view images.
- The spherical mosaiced image formed can be of any desired effective resolution, subject to the choice of optics and number of raw images that are composited.
- Spherical mosaicing allows the resulting mosaic to be treated as a rigid, composite image.

A few demerits of the above discussed technique are:

- Straightforward implementation of our algorithm will fail where there are large textureless regions.
- High-pass filtering alone introduces many discontinuities into previously smooth image regions, corrupting the derivative computations and preventing convergence. Thus their implementation must filter the images to remove textureless regions, while simultaneously preserving smoothness. Thus band-pass filtering is preferred.

- The traversing and mapping of pixels in each image and the accumulation of the global derivatives increases the computational costs.
- The algorithm fails to converge for large errors in estimating focal lenght. Thus, fairly good camera calibration is required to provide initial focal length estimates.
- The algorithm uses a gradient based approach for optimization and may get stuck with the problem of local minima. Other optimization techniques like *Simulated Annhealing* might just provide better results.

7.2 Conclusion

This report described two methods to recover relative rotations and internal camera parameters for the set of images acquired from a common optical center.

The first method is a closed-form solution using eigen-vectors of 8-parameter warps. This method is theoretically elegant, but yields quantitatively inaccurate results as it just performs *local optimization*. The second method solves this problem by computing rotations and internal parameters directly from image-space correlation using a *global optimization* technique.

We saw that this method gives better and accurate results than the former one. Overall, spherical mosaicing allows the resulting mosaic to be treated as a rigid, composite image and provides a huge field of view. If proper optimization of the estimates is performed, better results are delivered.

Bibliography

[1] Richard Hartley. Self-calibration of stationary cameras. In *IJCV, 22(1)*, pages 5–23, 1997.

[2] Berthold K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. In *Journal of the Optical Society of America A, 4(4),*, April 1987.

[3] H.-Y. Shum and R. Szeliski. Construction and re nement of panoramic mosaics with global and local alignment. In *In Proceedings of Sixth ICCV,*, pages pages 953–958, January 1998.

[4] Richard Szeliski. Video mosaics for virtual environments. In *IEEE Computer Graphics and Applications, 16(2)*, pages 22–30, March 1996.

[5] M. Wheeler and K. Ikeuchi. Iterative estimation of rotation and translation using the quaternion. In *Technical Report CMU-CS-95-215, Carnegie Mellon University,*, 1995.

- Name:
- Roll No:
- Guide:
- Examiner:
- Chair-person:
- Date
- Time:
- Venue:

- Name:
- Roll No:
- Guide:
- Examiner:
- Chair-person:
- Date
- Time:
- Venue: