Matching Spherical Panoramas and Planar Photographs

by

Gail Carmichael

A thesis submitted to
the Faculty of Graduate Studies and Research
in partial fulfilment of
the requirements for the degree of
Master of Computer Science

Ottawa-Carleton Institute for Computer Science
School of Computer Science
Carleton University
Ottawa, Ontario, Canada
August 2009

Abstract

Image matching and the epipolar geometry for a stereo pair has been a well-studied topic in the field of computer vision. There is a strong foundation for matching techniques between two planar images, and the case of two spherical panoramas has been more recently explored. This work establishes the geometry for a pair consisting of one planar image and one spherical panorama, while exploring matching techniques that will perform well for scenes with repetitive features. A pseudo-fundamental matrix is defined for use with one calibrated image and one uncalibrated. This allows a photograph to be used without calibration while a panorama can be more easily considered as a whole. A global context descriptor for Speeded Up Robust Features and Maximally Stable Extremal Regions improves matching results and automatically computed epipolar geometry for scenes with buildings having repetitive features.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The process of matching two images by finding points of interest (also called feature points) that correspond to one another has been a heavily researched topic in the field of computer vision. There are many applications of this research, from object recognition to the determination of the geometry between two cameras. Most of the focus has been on the case of matching two planar images, such as those captured as photographs. More recently, the case of finding correspondences between two spherical panoramas has been examined [18, 17, 11]. Combining these two types of images, this thesis looks at matching a panorama to a photograph.

Image correspondences can be used to retrieve an image from a database which is deemed to contain the same object as a query image. This is known as object recognition [25]. The first step in one particular approach to object recognition is to develop a database of feature points based on a set of training images. Then, each feature point in a query image is independently compared to the database, and a match (or several matches) found. The initial matches are filtered based on clustering techniques as well as geometric considerations.

Another common use of feature matching is known as wide baseline matching. Rather than simply determine what object is present in a single query image, this technique considers two images of the same scene taken with two different cameras. Feature point correspondences are used to compute an exact representation of the relative location and rotation between the cameras. This geometry can then be used to build a 3D model of the scene, or to insert virtual objects in a natural way to produce what is known as augmented reality.

Other uses of feature correspondences, as listed by Mikolajczyk and Schmid[30], include texture recognition, image retrieval, robot localization, video data mining, building panoramas, and recognition of object categories.

While these concepts are usually applied to sets of planar images, many can be useful for panoramas as well. As an example, consider the NAVIRE project [35] carried out by researchers at the University of Ottawa. The goal of this project was to "achieve effective and natural virtual navigation in image-based renditions of real environments." Panoramas with 360° fields of view were used to capture and display sites of interest. Users would be able to navigate from one panorama to another, and change the point of view for the current panorama.[1] Image matching between two panoramas has several uses in this scenario [18, 17, 11, 5]. First, the relative ordering of panoramas might be found by determining the geometry between them using feature matching results. Then, these panoramas might be aligned relative to each other to make navigating between them more natural. Finally, the geometry may be used to insert virtual objects into the panoramas.

A dense database of panoramic images for a site of interest could be potentially useful for some new and interesting applications of feature matching, discussed in a moment. Many of these ideas involve the ability of a person to take their own photograph and do something useful with it, since most people have access to some kind of camera (such as those integrated into their mobile devices). Using photographs and panoramas together requires new theory on how to match features between them, which is the focus of this thesis.

One application of matching photographs to panoramas is the idea of finding one's location. While GPS can often provide reasonable coordinates, it is not always available, especially indoors. Furthermore, not all phones can give GPS coordinates. In these cases, a general location could be obtained by finding the panorama that matches best with a photograph taken on location.

Another use of panoramas and photographs (and perhaps video) is for mobile augmented reality. Virtual objects and their positions relative to various panoramas could be added to the precomputed panorama database. When a new photograph is taken, it can be matched with the appropriate panorama to find the relative geometry between the two images. The geometry can then be used to transform the virtual object to the correct position in the photograph. The augmentations might be navigation aids or even elements of a game. They could be visualizations of how an area looked throughout history, or display future construction.

One might wonder why panoramas should be used for these applications (and others), rather than a set of regular photographs. Research on community photo collections [39], for instance, has successfully used publicly available photographs to build models of popular tourist sites. The advantage of panoramas comes from the fact that they capture a much wider field of view in a single

[1] The NAVIRE project is similar in concept to Google's Street View, found online at http://maps.google.ca/help/maps/streetview/

image, making them more compact than the dense set of photographs that would be required to do the same. Furthermore, panoramas can be more efficient to capture than the equivalent set of photographs when using multi-sensor, omnidirectional camera equipment.

1.1 Thesis Objective

The main focus of this thesis is on developing techniques that will facilitate matching between panoramas and photographs. Only one particular projection of a panorama is considered here - namely, a cube laid out in a cross pattern - but the concepts should extend to any chosen projection. The main goal is to establish the general geometric relationship found between a calibrated cube panorama and an uncalibrated photograph, and then find the best feature matching approach that addresses the challenges presented when comparing such different images. It is ultimately desirable to find enough good matches so that they may be used in calculating the geometry between the panorama and the photograph, which could then be used in, for example, the applications discussed above.

1.2 Thesis Outline

After this introductory chapter, Chapter 2 presents the background knowledge required for the rest of the thesis. The three main steps of features matching - detection, description, and correspondence - are discussed first; several popular feature detectors and descriptors are given. The pinhole camera projection model is introduced and the geometric relationship between two planar images is developed along with the fundamental and essential matrices that capture it numerically. The RANSAC method of outlier removal is given before discussing the capture, representation, and geometry of cube panoramas.

Chapter 3 builds on the cube panorama theory to establish geometry between a panorama and a photograph. First, a new entity called the pseudo-fundamental matrix is introduced as a way to represent the geometry of an image pair in which only one camera has been calibrated. It is shown that this matrix still exhibits the properties of the fundamental matrix, and experimental results confirm that it does indeed capture the geometry between a cube panorama and photograph. The chapter ends with details about the challenges that will be faced when trying to match panoramas with photographs, such as different fields of view and image qualities.

Chapter 4 is the first in a series that tests various feature detectors and descriptors for use

with eight panoramas and photograph pairs, and looks at Speeded Up Robust Features, or SURFs. Details of the SURF detector and descriptor are given before presenting experimental results on real images. The location of SURF points detected in the images are shown, and match results for basic thresholding and nearest-neighbour ratio are listed as percentages of correct matches. Chapter 5 has the same outline, but for features called Maximally Stable Extremal Regions, or MSERs.

The SURFs and MSERs of Chapters 4 and 5 are extended in Chapter 6 to be more distinctive, since the matching scenarios considered often involve buildings with repetitive features. A global descriptor that was developed for use with SIFT features is described first, followed by an adaptation of the concept to SURFs and MSERs. Experimental results follow, and a summary of results for all matching techniques looked at in the thesis concludes the chapter.

Chapter 7 combines the best matching results from the previous three chapters, and uses them to find the epipolar geometry for several panorama-photograph pairs. A pseudo-fundamental matrix is found for the best SURF results, MSER results, and the combination of the two. The epipolar geometry is shown on the images for the matrix found using combined results.

Finally, Chapter 8 summarizes the results found throughout the thesis, and suggests areas of future work.

1.3 Thesis Contributions

While exploring the problem of matching panoramas and photographs and the challenges involved, this thesis makes the following original contributions:

- A pseudo-fundamental matrix is proposed as a way to represent the geometry for an image pair in which only one camera is calibrated. Its application to the panorama and photograph image pair is demonstrated.

- New techniques for describing Speeded Up Robust Features (SURFs) and Maximally Stable Extremal Regions (MSERs) with global context are proposed to increase the distinctiveness of the features and improve matching results for repetitive images.

- Detailed experimental results are presented for matching images with repetitive features with several different feature detectors and descriptors, including the new global descriptors mentioned above. This includes several examples of the geometry found using the best matching results.

Chapter 2

Feature Matching, Epipolar Geometry, and Panoramas

The main goal of this thesis is to find a set of matching points between a spherical panorama and a planar photograph. To accomplish this, a good understanding of several key concepts is required. The background given here will first consider the more standard case of matching two planar scenes. State of the art techniques for detecting, describing, and matching points of interest will be outlined first, followed by the basic concepts of epipolar geometry. RANSAC will be demonstrated as a way of finding the epipolar geometry using a set of potentially matching points. After these fundamentals are established, the focus will move to panoramas, where the process to generate them will be shown and the geometry between a pair of panoramas will be discussed.

2.1 Feature Matching

The idea behind feature matching is to detect points or areas of interest in two images, find a distinctive way to describe these features, and then find correspondences between them. There are many applications of this process [30], including object and texture recognition, image retrieval, panorama construction, and image categorization. Each of the matching steps are described in the following subsections.

2.1.1 Feature Detection

The first step in matching features is, naturally, to reliably locate points or areas of interest. A comparative evaluation of detectors [37] explains that many detection methods aim to find points or corners in an image using image contours, image intensity, or parametric models. For example,

the popular Harris corner detector [15] uses image derivatives to locate intensity changes in two directions, indicating the presence of a corner. Because corners are detected throughout an image with good repeatability, it is one of the most popular detectors. Unfortunately, Harris corners are quite sensitive to changes in image scale, and become less repeatable under such conditions [37].

There has been much work on scale-invariant feature detection [7, 22, 28, 38], the highlight of which is Lowe's [25] Scale Invariant Feature Transform (SIFT) system. The SIFT detector works by taking advantage of scale space analysis [22], wherein a Gaussian convolution is applied to the image with varying degrees of blur. Successively blurred images are subtracted, and feature points are placed at maxima and minima of the difference images [23]. This is called a Difference of Gaussians (DoG) filter, and it approximates the Laplacian of Gaussians (LoG). Points with low contrast or points that are poorly localized along an edge are rejected. Coupled with the SIFT descriptor, feature points are translation, rotation, and scale invariant, and are often chosen as the best among other detectors and descriptors [30, 31]. However, like most detector/descriptor combinations, SIFT only works well up to about a 30° change in viewpoint between the two images being matched, with larger changes handled for images of planar surfaces [31].

Bay et al. developed Speeded Up Robust Features [3] to improve the runtime efficiency of SIFT while still retaining relatively good matching results. The SURF detector is, very briefly, based on the determinant of the Hessian matrix and second order Gaussian derivative approximations. According to an analysis comparing SIFT and SURF [2], SURF does indeed perform more efficiently than SIFT with a smaller but sufficient number of quality detected points.

Some detectors have been developed to improve the ability to reliably find features in images that view a scene from widely varying angles, and which thus experience some sort of geometric deformation. These features are classified as affine co-variant, since the deformation of the area immediately surrounding the features can be approximated by an affine transformation on a local scale [27]. One such detector is called Affine-SIFT [45], which is an extension to Lowe's SIFT that simulates various image tilts before detecting the feature points.

A high quality affine co-variant detector was developed to look for what are called Maximally Stable Extremal Regions, or MSERs [26]. In this case, the features are actually shapes rather than points or corners. This detector can be explained minimally by its similarity to the watershed algorithm [44] for image intensities. Suppose that the image represents a terrain viewed from above, where black areas are low ground and white areas are high ground. If the terrain is slowly flooded, certain areas will collect water in such a way that the pool does not change shape for some time.

These areas are considered to be the most stable and are chosen as features. When combined with certain descriptors, MSERs perform very well when detected on flat surfaces, and have average performance for use with images of 3D objects [31]. They can also work well for changes in illumination between images [14].

Both the SURF and MSER detectors will be discussed in more detail in Chapters 4 and 5 respectively.

2.1.2 Feature Description

After features have been located in an image, some unique way of describing them is required so that features in another image can be compared, and correspondences found. There are several ways to describe feature points, and there is a choice of which descriptor to combine with a particular detection method [30]. Note that the simplest descriptor of all would be a vector of image pixel values around the feature. The most common way to represent a feature is by a numerical vector; it is how this vector is formed that varies. This section looks at descriptors used with SIFT and MSER features, as they will be the focus of future chapters. Readers are referred to evaluation works [30, 31, 37] should they wish to learn more about other detectors and descriptors.

The goal of many feature descriptors is to compute descriptors that are invariant to a number of transformations between images. A particular feature that appears in one image might appear at a different size, rotation, or viewing angle in the other image. If a descriptor is computed from measurements made around a feature in exactly the same way in both images, the result will not be the same. Therefore, additional work is required to make sure the descriptors will match despite these transformations. Descriptors that succeed at this are called scale, rotation, and affine invariant.

The SIFT framework defines how to describe a feature point in addition to the detection method mentioned above. A scale for each feature is decided during SIFT's difference of Gaussians detection process. The scale determines the local working area around a feature point. The SIFT feature descriptor obtains rotation invariance by detecting one or more prominent orientations from the image gradient (obtained from the image derivative), and then rotating the working area to match. The rotated working area is divided into a 4x4 grid, totalling 16 regions. An 8-bin histogram is filled by the directions of the image gradients found in each region. The counts in these bins for all regions are used to form the SIFT descriptor, which will be a vector of size 128.

Ke and Sukthankar [19] use principal component analysis, or PCA, techniques to improve on the SIFT descriptor by making it shorter. After extracting a 41×41 patch centred on the feature point

and oriented according to the dominant directions found by the usual SIFT algorithm, an eigenspace is computed to express the gradient images of local patches. For each patch, the gradient image is found and then projected using the precomputed eigenspace, resulting in a compact feature vector. PCA-SIFT is shown to be more accurate than SIFT for several matching scenarios, as well as more efficient to compute.

The SURF descriptor relies on first order Haar wavelet responses in the x and y directions, differing from the use of gradients in SIFT. This, the authors claim, makes calculating the SURF descriptors more efficient. Like SIFT, SURF also assigns an orientation to each feature point. The typical SURF descriptor is a vector of length 64. Being smaller than the SIFT descriptor by half, fewer comparisons are needed for computing distances between feature descriptors while finding possible matches. Again, SURF will be described in further detail in Chapter 4.

Detected MSER features can be described in a variety of ways. In the original work describing MSERs [26], Matas et al. proposed an affine invariant procedure that uses several multiples of the original MSER as measurement regions, transforming the measurement region so its covariance matrix is diagonalized, and then computing rotational invariants based on complex moments. This method is based on the actual intensity values found in the image, but it is also possible to describe MSERs by their shape alone. Another approach [6] uses local affine frames defined from affine-invariant sets of three points chosen from the MSER contour and centroid. A more recent method given by Forssén and Lowe [14] works with affine-normalized patches that contain either the actual image values, or a binary image representing the MSER shapes. In this case, the SIFT descriptor is used to describe the patches, as it was evaluated as the best choice for describing MSERs [31]. These patches are explained further in Chapter 5.

2.1.3 Feature Correspondence

Once feature points have been located and described in two images, correspondences between the features may be found. There are several ways of finding similar features between images, but only methods that use the descriptors described above will be discussed here.

There are three common methods used to find matches between features described by numerical vectors. The first is to simply calculate the distance between the vectors using the appropriate metric (be it Euclidean, Mahalanobis, or χ^2), and keep all pairs of features whose distance falls below a certain threshold. This can clearly result in features having more than one possible match or false matches, so further processing may be required (for instance, RANSAC, described later,

might be used in conjunction with a geometric constraint). The second method involves finding the nearest neighbour for each feature descriptor in the first image in terms of the distance between it and the features in the second image. This will result in a one-to-one pairing of features, but may also result in mismatches in the presence of repetitive/indistinct features. The third method helps alleviate this problem by examining the ratio of the distances between the first and second nearest neighbours. The idea is that if the first and second neighbours have distances too similar to each other, then the match is not distinctive enough and should be discarded. It has been shown that this nearest-neighbour ratio method works quite well for a variety of descriptors and images [14, 25, 31].

A different approach is taken for MSER regions described with rotational moments based on complex moments [26]. For a feature in the first image, K features with the closest measurement values to the first feature's values are chosen from the second image. A vote is cast for each of these potential correspondences. After this process is finished, the feature pairs with the most votes are taken as potential matches. MSERs described with local affine frames are matched using a hash table, where the index comes from the triplet of points making up the descriptor [6].

When features are reasonably distinct, the above methods work well. For scenarios where this is not the case, some amount of information beyond the local area of a given feature may be needed to achieve good matching results. Mortensen et al. proposed [32] a global shape descriptor that would augment an existing SIFT descriptor. Another histogram is created, similar to an approach using shape contexts [4]: the maximum curvature is computed, and the number of edge points in a log-polar histogram are counted. The area of the image used to compute the global context is not relative to the actual scale of the feature point, making this technique sensitive to scale changes. Later work by Li and Ma [21] modifies the global context to be scale invariant, and adds colour information to help make the descriptors even more distinct.

Considering pairs of features can also help increase the distinctiveness of features for matching. Tell and Carlsson select pairs of Harris corners, and use the image intensities along a line between the two feature points as a basis for a descriptor [40]. The same voting strategy used for MSERs above is effective in finding individual features that match when voting in pairs. The pairs idea is also used by Forssén and Lowe [14] to match MSERs, though they don't require voting or the use of intensities since a concatenated SIFT descriptor used to describe the two features in a pair is supposed to be distinct enough.

2.2 Epipolar Geometry

Once a set of enough correct matches has been obtained from the methods described in 2.1, or any other method, the matches can be used to determine the geometry between the two cameras that captured the images. These two cameras can be referred to as a stereo pair. The geometry for a stereo pair is formally called epipolar geometry, and will be introduced in some detail in the following sections. First, the pinhole camera model is introduced. The geometry of two cameras builds on this, after which the fundamental and essential matrices are introduced. Finally, RANSAC is shown as a way to remove outliers from a set of potentially matching feature points. This entire section is based on the explanations of Hartley and Zisserman [16], unless otherwise specified.

2.2.1 Pinhole Camera Model

Before looking at the geometry between two images, it is important to first understand how an image is formed. The basic pinhole camera model, a specialization of the general projective camera model, describes how a 3D point in space is transformed into a 2D point on an image plane.

The basic concept is illustrated in Figure 1. The camera centre C is located at the origin of the world coordinate system. The image plane, which might be the film or digital sensor in a camera, is parallel to the XY plane. It crosses the Z-axis, which is also called the principal axis, at the principal point p. The principal point is located at a distance of f, the focal length, from the camera centre. The image plane has a 2D coordinate system of its own. Its origin is sometimes at the principal point, but this is not required. The mapping between a point in space to a point on the image plane is called central projection.

Represent a point $\mathbf{A}$ in 3D space and its projection $\mathbf{a}$ in homogeneous coordinates as $\mathbf{A} = (X, Y, Z, 1)^T$ and $\mathbf{a} = (x, y, 1)^T$. Let $\mathbf{p} = (p_x, p_y)^T$ be the coordinates of the principal point in the image plane's coordinate system. Then the central projection can be represented by (1).

$$\mathbf{a} = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{bmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{1}$$

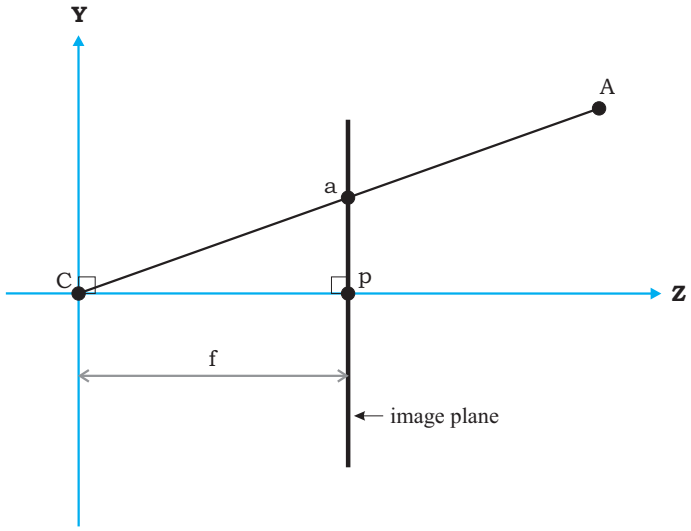


Figure 1: The pinhole camera model, where C is the camera centre, A is a 3D point in space, and a is the projection of A onto the image plane.

Let

$$K = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

so that (1) may be written in condensed form as

$$\mathbf{a} = K\,[I|\mathbf{0}]\,A \tag{3}$$

where $[I|\mathbf{0}]$ represents a 3×4 matrix whose leftmost 3×3 block contains the identity matrix, and whose rightmost column is the zero vector. K is known as the camera calibration matrix, since it encapsulates all the intrinsic camera properties needed to define the central projection of any point.

It is important to note that in (3), point $\mathbf{A}$ is written relative to the coordinate system whose origin is at the camera's centre; this is called the camera coordinate frame. Points in space are usually defined in terms of a more general world coordinate frame, which may not be the same as the camera coordinate frame. Thus it is sometimes necessary to find the coordinates of $\mathbf{A}$ with respect to the camera coordinate frame before the central projection can be applied. If $\tilde{C}$ is the camera centre C in world coordinates, and R represents the orientation of the camera coordinate frame, then (3) can be updated to become

$$\mathbf{a} = KR\left[I| - \tilde{C}\right]\mathbf{A} \tag{4}$$

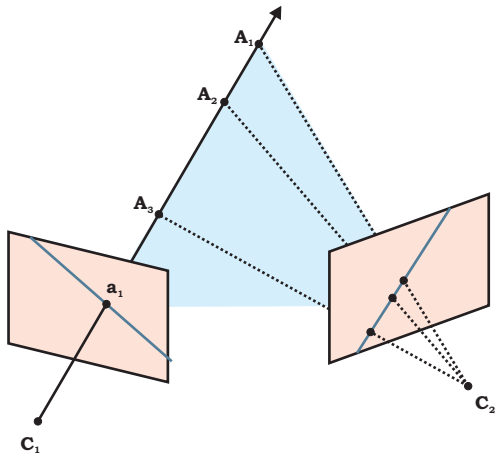


Figure 2: Point $\mathbf{a}_1$ might have been imaged from any point along a ray back-projected from C_1. The images of those points lie along a line on the second camera's image plane.

where $\mathbf{A}$ is given in the world coordinate frame. Taking the camera calibration matrix and its external orientation parameters R and $\tilde{C}$ together, the projection matrix P can be defined:

$$P = K\left[R|-R\tilde{C}\right] \tag{5}$$

The term $-R\tilde{C}$ is often written simply as t, giving $P = K\,[R|t]$.

2.2.2 Geometry Between Two Cameras

Regardless of what scene any two cameras are looking at, there is an intrinsic projective geometry between them. Similarly, there is a relationship between points projected on the image planes for each camera from the same point in space. This geometry is called epipolar geometry.

Refer to the scenario in Figure 2. Point $\mathbf{a}_1$ appears in an image captured by camera C_1. Using the pinhole camera model described in the previous section, the point in space that was projected to $\mathbf{a}_1$ must be located somewhere along the ray back-projected from the camera's centre through $\mathbf{a}_1$. Points $\mathbf{A}_1$, $\mathbf{A}_2$, and $\mathbf{A}_3$ are some of an infinite number of possibilities. Suppose that camera C_2 can also see the exact point in space $\mathbf{a}_1$ was projected from. Where on C_2's image plane can the projection of this point lie? It appears from the diagram that any point on that back-projected ray would lie on a line on the image plane. In fact, this is the case.

The reason for this is based on the idea that any point projected from a point $\mathbf{A}$ in space must lie on the plane formed by $\mathbf{A}$, C_1, and C_2. This is shown in the scenario depicted in figure 3. Here, $\mathbf{A}$ is projected onto the image planes for cameras C_1 and C_2, and a plane is defined by these three

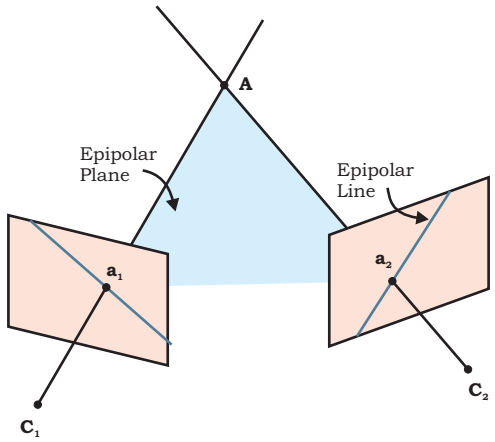


Figure 3: The epipolar plane is defined by a point **A** in space and two camera centres C_1 and C_2. The lines of intersection between the epipolar plane and image planes are called epipolar lines.

points. Depending on where exactly **A** is located on this plane, its projections $\mathbf{a}_1$ and $\mathbf{a}_2$ may lie anywhere on the intersection of the plane and the appropriate image plane. The plane defined by the camera centres and point in space is called the epipolar plane, and the lines of intersection between the epipolar and image planes are called epipolar lines.

When the point **A** in space is known, the exact projections $\mathbf{a}_1$ and $\mathbf{a}_2$ on the image plane are also known. When only one of $\mathbf{a}_1$ or $\mathbf{a}_2$ is known, the epipolar plane is defined by it and the two camera centres. Therefore, the other projection must lie somewhere along the epipolar line in the other image.

There is a special point on each camera's image plane called the epipole. This image point is actually the image of the other camera's centre, as if that centre were a point in space. It does not always appear within the image plane's viewable area, and can even lie at infinity if, say, the two image planes are coincident. The epipole will always be on the epipolar plane, no matter what point in space is considered, since the epipolar plane is partially defined by the two camera centres. Consequently, all epipolar lines will pass through the epipoles.

2.2.3 The Fundamental Matrix

The fundamental matrix encapsulates the epipolar geometry between two views. It can be constructed directly from the projection matrices of equation (5) for two cameras with different centres, and the projection matrices can be recovered from the fundamental matrix up to a projective ambiguity. It can be used to find the epipolar line in an image when searching for point correspondences,

and can act as a constraint for algorithms that seek to eliminate incorrect matches.

Let $\mathbf{x}$ and $\mathbf{x}'$ be corresponding points in two images written in the image plane's coordinate frame, and F be the fundamental matrix that represents the geometry between the images. Then

$$\mathbf{x}'F\mathbf{x} = 0 \tag{6}$$

Some properties of the fundamental matrix include:

1. If the fundamental matrix for a particular stereo pair of cameras P and P' is F, then the fundamental matrix for the reverse pair P' and P is F^T.

2. $F\mathbf{x}$ gives a 3-vector interpreted as the homogeneous representation of a line in 2D. This is the epipolar line in the second image corresponding to $\mathbf{x}$ from the first image. It is because $\mathbf{x}'$ must lie on the epipolar line given by $F\mathbf{x}$ that the dot product of the two gives zero. Similarly, $F^T\mathbf{x}'$ gives the epipolar line in the first image corresponding to $\mathbf{x}'$ from the second image.

3. Let the epipoles in the first and second images be $\mathbf{e}$ and $\mathbf{e}'$. Then the epipolar lines given by $F\mathbf{x}$ for $\mathbf{x}$ other than $\mathbf{e}$ pass through $\mathbf{e}'$. Similarly, all epipolar lines given by $F^T\mathbf{x}'$ for $\mathbf{x}'$ other than $\mathbf{e}'$ pass through $\mathbf{e}$.

Given at least 7 correct point correspondences, (6) can be used to find the unknown fundamental matrix. If a particular correspondence is known to be $\mathbf{x} = (x, y, 1)^T$ and $\mathbf{x}' = (x', y', 1)^T$, then equation (6) can be written as

$$x'xf_{11} + x'yf_{12} + x'f_{13} + y'xf_{21} + y'yf_{22} + y'f_{23} + xf_{31} + yf_{32} + f_{33} = 0 \tag{7}$$

If the entries of F are put into a vector $\mathbf{f}$ in row major order and multiple correspondences considered, then the following linear system is obtained:

$$\begin{bmatrix} x'_1x_1 & x'_1y_1 & x'_1 & y'_1x_1 & y'_1y_1 & y'_1 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x'_nx_n & x'_ny_n & x'_n & y'_nx_n & y'_ny_n & y'_n & x_n & y_n & 1 \end{bmatrix} \mathbf{f} = \mathbf{0} \tag{8}$$

When only seven correspondences are available, the system can be solved for one or three possible fundamental matrices. If there are more points available, a least squares solution using the singular value decomposition can be found to account for potential noise in the points. In this case, it is highly recommended that the points be normalized so that the centroid is at the origin and no

point is further from the origin than a root-mean-square distance of $\sqrt{2}$. These methods are called the 7-point algorithm and normalized 8-point algorithm, respectively. There are also non-linear approaches to finding F, and the interested reader is referred to Hartley and Zisserman's book [16] for more detailed information.

The fundamental matrix, once determined for a stereo pair, can be used to test the quality of a pair of features as a match. This is possible because for each of those features, an epipolar line in the other image may be found, and the distance from the other feature to that line is easy to compute. If both features are close to the corresponding epipolar lines in the same image, the feature pair is more likely to be a good match. To find the distance from a point to a line given in homogeneous coordinates, they should both be normalized first. The following normalizations for a point $\mathbf{x} = (x_1, x_2, x_3)$ and a line $l = (a, b, c)$ are used, respectively:

$$(x_1, x_2, x_3) \rightarrow \left(\frac{x_1}{x_3}, \frac{x_2}{x_3}, 1\right) \tag{9}$$

$$(a, b, c) \rightarrow \left(\frac{a}{\sqrt{a^2+b^2}}, \frac{b}{\sqrt{a^2+b^2}}, \frac{c}{\sqrt{a^2+b^2}}, \right) \tag{10}$$

Now the perpendicular distance d from the point $\mathbf{x}$ to the line l is given by the dot product

$$d = l \cdot \mathbf{x} \tag{11}$$

When the point is on the line, $d = 0$, as seen earlier.

2.2.4 The Essential Matrix

While points used with the fundamental matrix in equation (6) are given in the image's coordinate frame, it is sometimes convenient to use the camera's coordinate frame instead. Points in this frame are said to be normalized, and are obtained by applying the inverse of the camera's calibration matrix to the camera's projection matrix and the image point. If the image point is $\mathbf{x}$ and the camera projection matrix is $P = K[R|t]$, then the normalized point will be $\tilde{\mathbf{x}} = K^{-1}\mathbf{x}$ with respect to the normalized camera $K^{-1}P = [R|t]$.

Consider (6) again. If $\mathbf{x}$ is replaced with $K\tilde{\mathbf{x}}$ and $\mathbf{x}'$ is replaced with $K'\tilde{\mathbf{x}'}$ the equation becomes

$$(K'\tilde{\mathbf{x}}')^T F(K\tilde{\mathbf{x}}) = 0 \tag{12}$$

$$\tilde{\mathbf{x}}'^T(K'^T F K)\tilde{\mathbf{x}} = 0 \tag{13}$$

Let

$$E = K'^T F K \tag{14}$$

be called the essential matrix. Then the fundamental relationship between corresponding normalized coordinates in two views becomes

$$\tilde{\mathbf{x}}'^T E \tilde{\mathbf{x}} = 0 \tag{15}$$

2.2.5 RANSAC Outlier Removal

Despite there being much research on feature matching techniques, as seen in Section 2.1, incorrect matches usually remain after any standard matching process. Obviously incorrect matches are called outliers, and must be removed automatically from the initial match set. Fischler and Bolles [12] introduced RANSAC as a way to fit a model to experimental data while removing outliers from that data. The fundamental or essential matrix from the previous subsections provide a model that constrains the location of matching points to their corresponding epipolar lines. These models can be used with RANSAC to refine the match set to one that is consistent with a fundamental matrix.

The main idea behind RANSAC is to sample minimal random sets of data rather than try to incorporate the entire set at once. This way, grossly wrong outliers will not necessarily influence the fit of the model to the rest of the data. A predetermined number of iterations is performed, and a different randomly chosen subset of data is chosen at each iteration. The model is fit to this set, and the number of points in the overall data set that fit this model (called the inliers) is computed. The best model is tracked based on the largest number of inliers associated to it. The number of iterations required to find a good model generally depends on some knowledge of the actual data being considered, and can be computed based on the probabilities of choosing only true inliers for a particular randomly sampled set.

To apply RANSAC to a match set with the fundamental matrix as the model, seven or eight points are typically chosen as a random sample. The 7 point algorithm or normalized 8 point algorithm is applied to this minimal set. Inliers are determined using the epipolar lines computed from $F\mathbf{x}$ and $F^T\mathbf{x}$. A correspondence is considered an inlier if x' is close enough to $F\mathbf{x}$ and $\mathbf{x}$ is close enough to $F^T\mathbf{x}$. The threshold value used in the experiments to decide whether the distance to the epipolar line is small enough is 0.001, a default value of the implementation used. After RANSAC performs the required number of iterations, a final set of inliers is found based on the best performing model, and a final matrix is often computed using some non-linear method.

There have been several other methods of outlier removal proposed besides RANSAC, such as

MLESAC [41]. Because RANSAC is simple and because there are free implementations readily available [43], only it is used in the experiments here. Comparing results against other techniques might be the topic of future work.

2.3 Cubic Panoramas

A spherical panorama is an image with a 360^{o} field of view. There are many ways to represent a panorama, the most obvious being on a sphere. However, this is not very computationally attractive, and instead a cube shape is proposed for its geometric conveniences and efficiency for rendering on standard graphics hardware [5, 11]. A cube is also easily laid out onto a plane in a way that is easy to interpret visually. In this section, the process of capturing panoramic data and representing it as a cube is discussed. This is followed by an overview of the geometry between two cubic panoramas, which extends the geometry found between two planar images. The theory in this entire section was originally presented or developed by Kangni [18].

2.3.1 Capture and Representation of Cubic Panoramas

The images used in the experiments outlined in later chapters were captured with the Point Grey Ladybug camera [36], seen in Figure 4, or generated from images taken with a consumer digital single lens reflex camera.

The Ladybug has six 1024×768 camera sensors: five are placed around the circumference facing out, and one on the top is aimed upward. The Ladybug can be calibrated so that the exact position of all camera centres is known in addition to each camera's intrinsic properties, such as their focal length. This setup allows for fast and convenient capture of multiple panoramas, as each camera will record its image at the same time. Because of this, the Ladybug was used for the University of Ottawa NAVIRE project [35], which uses a dense collection of panoramas to allow a user to explore an area virtually (known as telepresence).

Once images have been captured by the Ladybug sensors, assuming the Ladybug has been accurately calibrated, the data can be fused together to form a nearly complete spherical panorama that can then be reprojected onto any surface, including the preferred cube. Cube generation, illustrated in Figure 5, was originally developed by Fiala et al. [5, 11], and extended by Kangni [17, 18].

Panoramas generated from Ladybug camera data, while convenient, do pose some challenges. The most notable disadvantage is the lower image quality that results (see, for instance, Section

Figure 4: The Ladybug2 spherical camera from Point Grey Research. Image from http://www.ptgrey.com/products/ladybug2/samples.asp (used with permission).

3.3.1). A sharper image can be obtained using a consumer camera (or cameras), but this requires stitching planar images together into a spherical panorama. Open Source Software such as Hugin [8] streamlines this task and automates where possible, but users often need to adjust control points so the program can align photos relative to each other. A panorama created this way can be seen in Figure 10 on page 25. A more expensive solution would be a carefully constructed and calibrated rig housing multiple consumer cameras that can be triggered together.

Regardless of how it is captured, a cube panorama is visualized on a flat surface by laying it out in a cross pattern, as in Figure 6. Faces of the cube are labelled using the same standard from previous work, with the top and bottom faces labelled *up* and *down*, and the faces across the middle from left to right labelled *left*, *front*, *right*, and *back*. Each face can thus be referenced by a single identifying letter from the set $\{U, L, F, R, B, D\}$.

2.3.2 Geometry of Cubic Panoramas

This section outlines the projective geometry of a single cube, extending the theory developed for the pinhole camera model in Section 2.2.1.

First, the coordinate frame of the cube must be defined as it was for a regular pinhole camera. The cube's centre is also its centre of projection, where the cube coordinate frame will have its origin. Following the OpenGL convention, the x-axis will point toward the *right* face, the y-axis to the *up* face, and the z axis toward the *back* face. Figure 7 shows three views of the cube with the coordinate axes superimposed.

One of the reasons for choosing the cube to represent panoramic data is that it is geometrically convenient to work with. This is because the six faces could each individually be said to represent

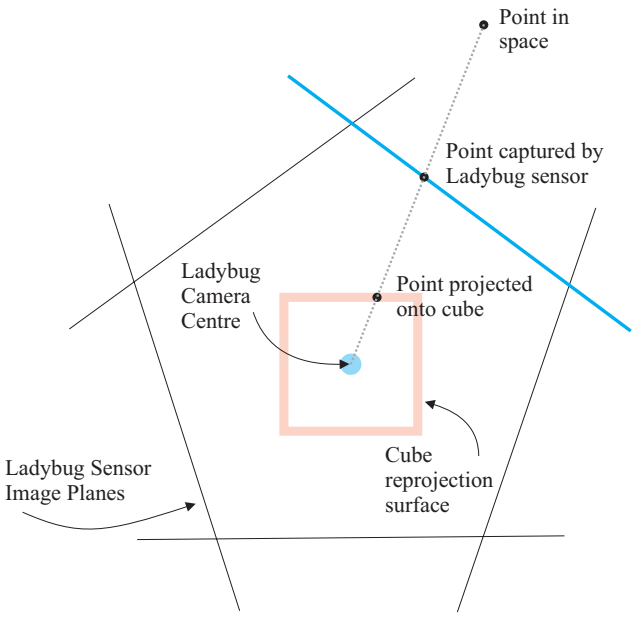


Figure 5: A point in space is captured by the Ladybug camera sensors first, then reprojected onto a cube.

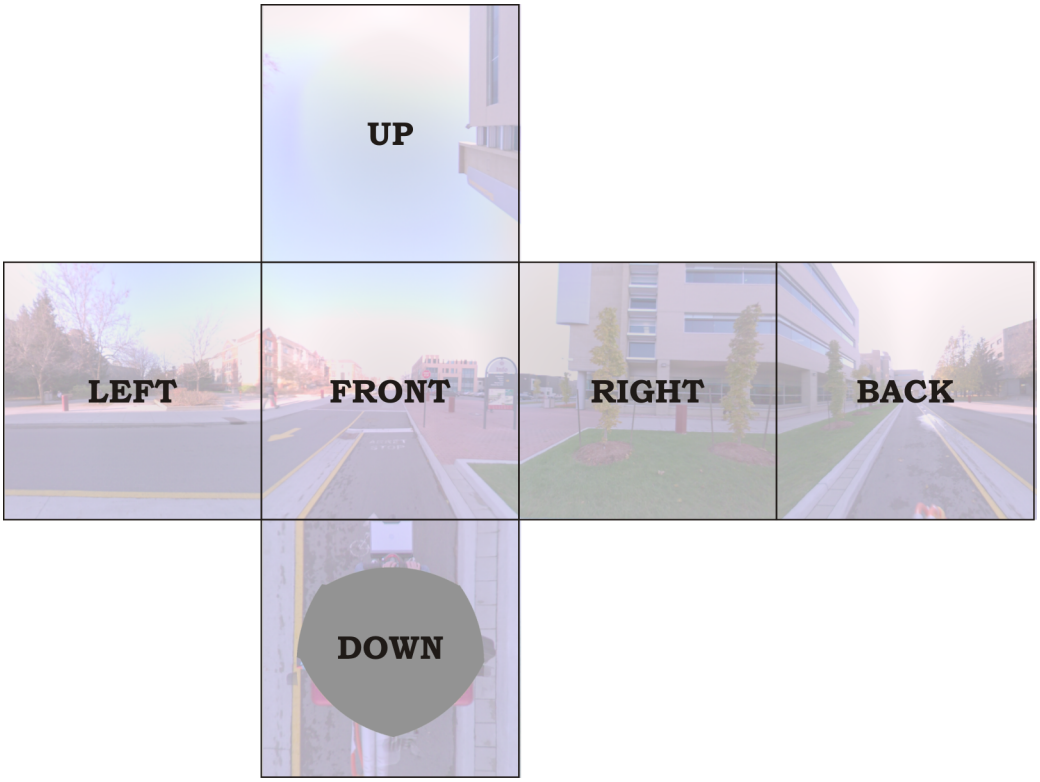


Figure 6: A cube panorama laid out in a cross pattern with its faces labelled.

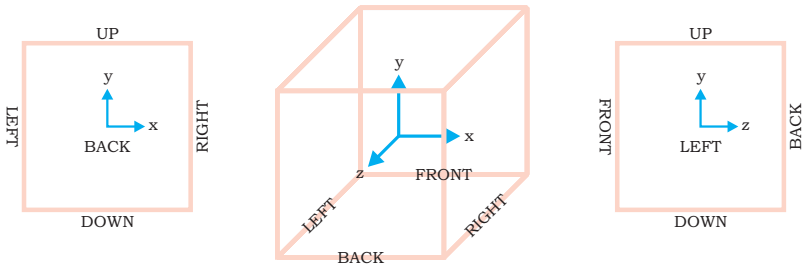


Figure 7: An illustration of the cube's reference frame.

the image plane of a pinhole camera; it just happens that each camera shares the same centre. When cubes are generated, users have the ability to determine their size. Suppose the length of an edge on a cube is L pixels. This means that the perpendicular distance from the cube centre to a face, which is also the focal length for that image plane, is $\frac{L}{2}$. It also means that the principal point of the pinhole camera for that face will be at $(\frac{L}{2}, \frac{L}{2})$, assuming image coordinates with the origin at the top left and the positive y-axis facing down. Together, these facts give the camera calibration matrix for any face on the cube:

$$K = \begin{bmatrix} \frac{L}{2} & 0 & \frac{L}{2} \\ 0 & \frac{L}{2} & \frac{L}{2} \\ 0 & 0 & 1 \end{bmatrix} \tag{16}$$

2.3.3 Geometry Between Two Cubes

The geometry between two cubes is similar to the scenario in Figure 3 on page 13. Figure 8 shows a point $\mathbf{A}$ in space projected onto a particular face of two cubes as $\mathbf{a}_1$ and $\mathbf{a}_2$. Simply removing the faces not involved with the projection would yield the epipolar geometry seen in Section 2.2.2. In other words, the faces from two cubic panoramas may be considered a set of six stereo pairs. This concept is simplified when the same or entire portion of a scene is visible in corresponding faces, but even when this is not the case, matches from other faces may be reprojected onto the appropriate face's supporting plane for the computation of the fundamental matrix. Each face of the cube is related to all other faces by a homography.

2.3.4 The Essential Matrix for Cubic Panoramas

Just as the epipolar plane introduced in Section 2.2.2 intersected the two image planes in question, the epipolar plane will intersect each of the supporting planes for the cube faces (possibly at infinity).

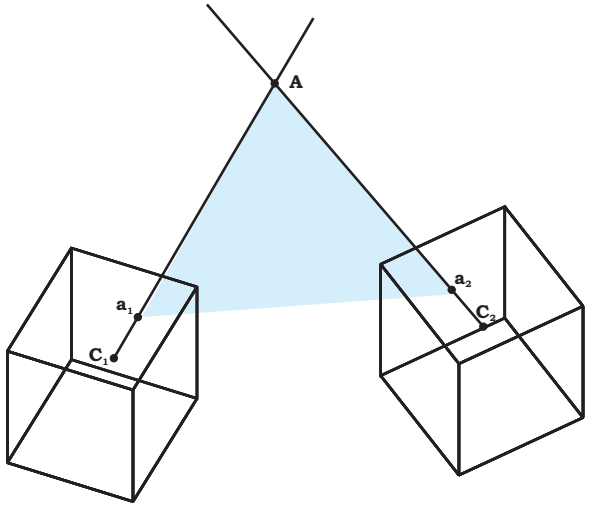


Figure 8: Epipolar geometry for two cubes. A point **A** in space is projected onto a particular face of each cube.

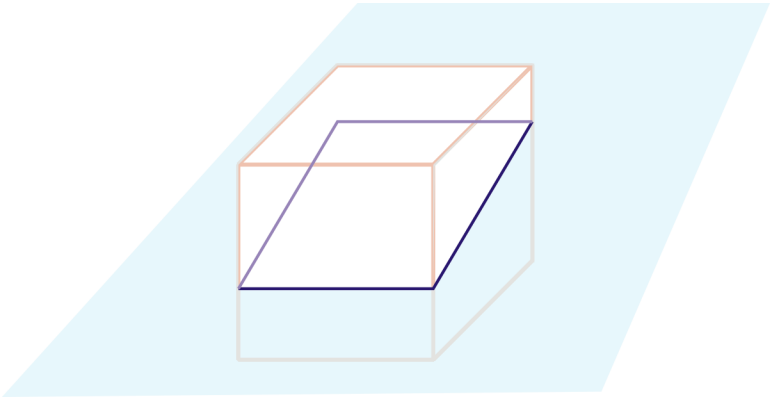


Figure 9: An epipolar plane intersecting a cube.

Since the epipolar plane must go through the cube's centre (which is also the camera centre for each face), it will actually intersect only four faces within the bounds of the cube edge lengths. Figure 9 shows an example to help visualize this. It would be nice to consider all these individual epipolar lines simultaneously. As will be seen below, using normalized coordinates for image points with an essential matrix achieves this and other advantages, allowing the entire cube to be considered as a whole rather than as individual stereo pairs.

As explained in Section 2.2.4, normalized coordinates are projected points given in camera frame coordinates. Because the coordinate frame for the cube is easily determined (see Section 2.3.2), it is not difficult to convert 2D image points to 3D camera coordinates. The exact conversion process

for points on a panorama in the cross pattern seen in Figure 6 is described in appendix B.

Recall that if $\mathbf{p}$ and $\mathbf{p}'$ are corresponding points in a stereo system, then the essential matrix relates them by

$$\mathbf{p}'^T E\mathbf{p} = 0 \tag{17}$$

There is a geometric interpretation of this constraint [10] that allows the use of the entire epipolar plane when considering the epipolar geometry between two cubes. In this interpretation, $E\mathbf{p}$ (respectively $E^T\mathbf{p}'$) gives the normal to a plane rather than a homogeneous representation of a line. The dot product of $\mathbf{p}$ (or $\mathbf{p}'$) with this normal gives zero because $\mathbf{p}$ and $\mathbf{p}'$ must be coplanar with the cube centres if they are true correspondences. Because the plane must pass through the second (or the first) cube's centre, it is uniquely defined by the normal $E\mathbf{p}$ (or $E^T\mathbf{p}'$) , relative to that cube's coordinate frame. Hence, $E\mathbf{p}$ (or $E^T\mathbf{p}'$) can be referred to as the epipolar plane with respect to the second (or first) cube's coordinate frame.

The essential matrix for a pair of cubes can be calculated from a set of correspondences using a variant of the method for computing the fundamental matrix shown in Section 2.2.3. The first step is the normalization of the image points, which is somewhat easier for the cube case, since all points are already centred around the origin of the cube's centre and reference frame. All that remains is the scaling of the points to ensure they are no more than $\sqrt{2}$ away from the centre. The easiest way to do this is simply divide by the maximum absolute value of $\frac{L}{2}$ where L is the length of the cube sides.

Now equation (17) must be rewritten in the form $Ah = 0$ where h represents the entries of the essential matrix in row-major order. Using n matches between $\mathbf{p}_i = (p_{ix}, p_{iy}, p_{iz})$ and $\mathbf{p}'_i = (p'_{ix}, p'_{iy}, p'_{iz})$, A will have the following form:

$$A = \begin{bmatrix} p_{1x}p'_{1x} & p_{1y}p'_{1x} & p_{1z}p'_{1x} & p_{1x}p'_{1y} & p_{1y}p'_{1y} & p_{1z}p'_{1y} & p_{1x}p'_{1z} & p_{1y}p'_{1z} & p_{1z}p'_{1z} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ p_{nx}p'_{nx} & p_{ny}p'_{nx} & p_{nz}p'_{nx} & p_{nx}p'_{ny} & p_{ny}p'_{ny} & p_{nz}p'_{ny} & p_{nx}p'_{nz} & p_{ny}p'_{nz} & p_{nz}p'_{nz} \end{bmatrix} \tag{18}$$

The solution to $Ah = 0$ can be found as a least squares solution using singular value decomposition, just as with the fundamental matrix.

Once the essential matrix has been computed, it is useful to note that finding the intersection of the epipolar planes with a particular cube is also simplified by the nature of the cubes. The fact that the epipolar plane passes through the cube coordinate frame's origin means the plane can be written in projective coordinates as $\pi = (a, b, c)$. Then, using the distance $\frac{L}{2}$ to each face, some

simple equations can be formulated to find the line of intersection for each face's supporting plane. Four of these lines will be within the bounds of the viewable image for their respective faces. The equations for finding the lines of intersection are listed in Appendix B.

The quality of matches between points in the cube's coordinate frame can be evaluated using a similar approach as that in Section 2.2.3. For cube points, distance to the epipolar plane is computed, rather than distance to an epipolar line. The distance d from a cube point $\mathbf{x} = (x_1, x_2, x_3)$ to a plane defined by a normal $n = (a, b, c)$ is, including normalization:

$$d = \frac{|n, \mathbf{x}|}{\|n\|} = \frac{|ax_1 + bx_2 + cx_3|}{\sqrt{a^2 + b^2 + c^2}} \tag{19}$$

2.4 Conclusion

In this background chapter, some basic knowledge about feature matching and geometry was introduced. The three steps for feature matching were outlined in some detail, and the geometry for a stereo pair of planar photographs explained. RANSAC was shown as a way to remove outliers from an initial match set by fitting a geometric model to the data. Then, the focus shifted to cubic panoramas, whose capture process was given and geometry demonstrated. In the next few chapters, these concepts will be put together to examine the case of matching a cubic panorama with a planar photograph. The geometry between these entities will prove to be quite similar to that already seen, but special care will be needed to find the most effective matching technique.

Chapter 3

A Cubic Panorama and Planar Photograph as a Stereo Pair

In chapter 2, two different geometries were examined: that of two standard cameras with flat image planes (Section 2.2), and that of two cubic panoramas (Section 2.3.3). This chapter combines the two, considering the geometry between one cubic panorama (which is generally assumed, without loss of generality, to be the first image) and one planar image. This combination might be used, for example, as a method to precapture panoramic data densely covering a geographic area, and later matching a regular photograph to the panorama for location, recognition or augmentation purposes. An example pair can be seen in Figure 10.

All the ideas detailed earlier can be extended to this new geometry. Refer to Figure 11. There are two camera centres: C_1 for the cube, and C_2 for the planar image. A point $\mathbf{A}$ in space is still projected to one face of the cube as $\mathbf{a}_1$, and the image plane as $\mathbf{a}_2$. An epipolar plane is defined by these points, and intersects the cube as epipolar lines on four of its faces, and as one epipolar line on the planar image.

In this chapter, a modification of the fundamental matrix is presented and then used to efficiently represent the geometry between a cube and a planar image. Some of the challenges faced when matching a panorama with a photograph are then given.

3.1 The Pseudo-Fundamental Matrix

Recall equation (6) from chapter 2 for two corresponding points $\mathbf{x}$ and $\mathbf{x}'$ written in image coordinates:

$$\mathbf{x}'F\mathbf{x} = 0 \tag{20}$$

(a) Panorama

(b) Photograph

Figure 10: Example of a cubic panorama-photograph pair where the scene depicted in the photograph appears on more than one face in the cube.

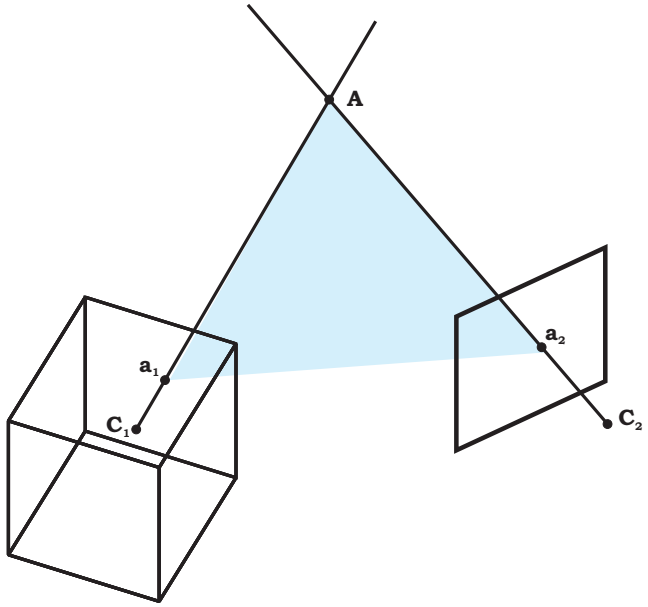


Figure 11: The basic geometry between a photograph and a cubic panorama.

This encapsulates the epipolar constraint for $\mathbf{x}$ and $\mathbf{x}'$ given as homogeneous image coordinates. Now recall equation (15), which also encapsulates the epipolar constraint, but for $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{x}}'$ written in normalized camera coordinates:

$$\tilde{\mathbf{x}}'^T E \tilde{\mathbf{x}} = 0 \tag{21}$$

The essential matrix E was defined in equation (14) as

$$E = K'^T F K \tag{22}$$

Sections 2.2.4 and 2.3.4 showed how knowing both camera calibration matrices for a stereo pair made the use of the essential matrix possible. What happens when calibration matrix K is known for the first camera, but calibration matrix K' is not known for the second? Normally, the known K would be ignored, and the fundamental matrix used instead of the essential matrix. However, in some circumstances, the use of normalized coordinates is actually quite desirable, as was the case with a stereo pair of cubic panoramas (see Section 2.3.4). A new entity, called a *pseudo-fundamental matrix*, is proposed to allow the use of normalized coordinates for one image (such as a panorama) and uncalibrated image coordinates for the other.

3.1.1 Definition of the Pseudo-Fundamental Matrix

Consider a stereo pair of cameras with projection matrices $P = K[R|t]$ and $P' = K'[R'|t']$. Let $\mathbf{x}$ and $\mathbf{x}'$ be corresponding points in image coordinates. These points in normalized camera coordinates are $\tilde{\mathbf{x}} = K^{-1}\mathbf{x}$ and $\tilde{\mathbf{x}'} = K'^{-1}\mathbf{x}'$. Substituting these values into equation (15) gives

$$(K'^{-1}\mathbf{x}')^T E(K^{-1}\mathbf{x}) = 0 \tag{23}$$

$$\mathbf{x}'^T K'^{-T} E K^{-1}\mathbf{x} = 0 \tag{24}$$

Notice that equation (24) can be combined with the definition of E in equation (22) to get the correspondence equation (20) in terms of F. Instead of combining both inverse camera calibration matrices with E, combine only K'^{-1}, leaving K^{-1} to normalize $\mathbf{x}$:

$$\mathbf{x}'^T(K'^{-T}E)(K^{-1}\mathbf{x}) = 0 \tag{25}$$

$$\mathbf{x}'^T(K'^{-T}E)\tilde{\mathbf{x}} = 0 \tag{26}$$

Let

$$G = K'^{-T}E \tag{27}$$

where G is called the pseudo-fundamental matrix. Then the following correspondence equation can be defined:

$$\mathbf{x}'^T G\tilde{\mathbf{x}} = 0 \tag{28}$$

The pseudo-fundamental matrix is simply a modified version of the fundamental matrix that facilitates the use of normalized camera coordinates for one image and non-normalized image coordinates for the other. In general, this would not be considered useful, since having normalized coordinates for only one image does not offer any advantage. In the case of any spherical panorama, on the other hand, normalized coordinates are very valuable, as they avoid the need to reproject all image points onto a plane in order to find a fundamental matrix between it and a planar image. Thus, the pseudo-fundamental matrix can be used to consider a cubic panorama as a whole when comparing it to a planar image, even when the planar image's camera is uncalibrated.

3.1.2 Properties of the Pseudo-Fundamental Matrix

To show that G is indeed a special case of the fundamental matrix and a correct representation of a partially calibrated stereo pair's geometry, the first three properties of a fundamental matrix listed in Section 2.2.3 will be verified for G.

First, it will be shown that if G is the pseudo-fundamental matrix for a pair of cameras (P, P'), then G^T is the pseudo-fundamental matrix for the opposite pair (P', P). The correspondence equation for (P', P) would be:

$$\tilde{\mathbf{x}}^T G^T \mathbf{x}' = 0$$

Substituting the definition of G from (27):

$$\tilde{\mathbf{x}}^T (K'^{-T} E)^T \mathbf{x}' = 0$$

and replacing $\tilde{\mathbf{x}}$ with the equivalent $K^{-1}\mathbf{x}$:

$$(K^{-1}\mathbf{x})^T (K'^{-T} E)^T \mathbf{x}' = 0 \tag{29}$$

$$\mathbf{x}^T K^{-T} E^T K'^{-1} \mathbf{x}' = 0 \tag{30}$$

$$\mathbf{x}^T (K'^{-T} E K^{-1})^T \mathbf{x}' = 0 \tag{31}$$

Rearranging (14) to isolate F gives

$$F = K'^{-T} E K^{-1} \tag{32}$$

Substituting (32) into (31) results in

$$\mathbf{x}^T F^T \mathbf{x}' = 0$$

which is the correspondence equation for (P', P) in terms of the fundamental matrix, known to be true.

The second property of the fundamental matrix is that the epipolar line $\mathbf{l}'$ corresponding to a point $\mathbf{x}$ from the first image is given by $\mathbf{l}' = F\mathbf{x}$, and the epipolar line $\mathbf{l}$ corresponding to a point $\mathbf{x}'$ from the second image is $\mathbf{l} = F^T\mathbf{x}'$. The following shows that $G\tilde{\mathbf{x}}$ is equivalent to $F\mathbf{x}$, proving that $G\tilde{\mathbf{x}}$ gives an epipolar line in the second image:

$$G\tilde{\mathbf{x}} = (K'^{-T} E)\tilde{\mathbf{x}} = (K'^{-T} E)(K^{-1}\mathbf{x}) = (K'^{-T} E K^{-1})\mathbf{x} = F\mathbf{x} \tag{33}$$

Similarly, the following shows that $G^T\mathbf{x}$ is equivalent to $E^T\tilde{\mathbf{x}}'$, which gives the normal to the epipolar plane in the first image (as in Section (2.3.4) and Faugeras et al. [10]):

$$G^T\mathbf{x}' = (K'^{-T} E)^T \mathbf{x}' = E^T K'^{-1} \mathbf{x}' = E^T \tilde{\mathbf{x}}' \tag{34}$$

Even though $\tilde{\mathbf{x}}'$ is not actually known in the considered scenario due to the second camera's intrinsic properties not being known, it does theoretically exist.

Finally, the third property states that the epipoles $\mathbf{e}$ and $\mathbf{e}'$ are contained in the epipolar lines $F\mathbf{x}$ and $F^T\mathbf{x}'$. To show this for G, it must be demonstrated that $\mathbf{e}'^T(G\tilde{\mathbf{x}}) = 0$ for all $\tilde{\mathbf{x}}$ other than

$\tilde{\mathbf{e}}$, and $(\mathbf{x}'^T G)\tilde{\mathbf{e}} = 0$ for all $\mathbf{x}'$ other than $\mathbf{e}'$. Note that the epipole in the first image is given in normalized camera coordinates, just as $\tilde{\mathbf{x}}$ is. For the first equation:

$$\mathbf{e}'^T(G\tilde{\mathbf{x}}) = \mathbf{e}'^T K'^{-T} E\tilde{\mathbf{x}} = \mathbf{e}'^T K'^{-T} E K^{-1}\mathbf{x} = \mathbf{e}'^T F\mathbf{x} = 0 \tag{35}$$

For the second equation:

$$(\mathbf{x}'^T G)\tilde{\mathbf{e}} = \mathbf{x}'^T K'^{-T} E\tilde{\mathbf{e}} = \mathbf{x}'^T K'^{-T} E K^{-1}\mathbf{e} = \mathbf{x}'^T F\mathbf{e} = 0 \tag{36}$$

Thus, the first three properties of the fundamental matrix hold for the pseudo-fundamental matrix.

3.1.3 Application to Cubic Panorama and Planar Image

The pseudo-fundamental matrix can be used to represent the geometry between a cube and a planar image, since the former is calibrated while the latter is not. The advantages of considering a cubic panorama as a whole were mentioned in Section 2.3.4, and still stand even when only one panorama is involved.

Suppose that neither the panorama nor the planar image were calibrated, and a set of point correspondences are given in image coordinates. If all of these matches happened to lie on a single face of the cube, then it would be easy to find a fundamental matrix between the two planes. Since this will not always be the case (such as in Figure 10), points on nearby faces would have to be reprojected onto the main face's supporting plane before finding the fundamental matrix. Not only that, but an extra step would have to be added to the matching process that would determine which face to use. This is why it is much easier to consider the cube as a whole.

To find the pseudo-fundamental matrix for a cubic panorama laid out in a cross pattern and a planar image, the cube's point must first be converted into 3D points in the cube's coordinate frame (see Appendix B). Then a matrix A must be constructed, similar to Sections 2.2.3 and 2.3.4. If there are n corresponding points $\mathbf{p}_i = (p_{ix}, p_{iy}, p_{iz})$ and $\mathbf{p}'_i = (p'_{ix}, p'_{iy}, p'_{iz})$, then A will be

$$A = \begin{bmatrix} p_{1x}p'_{1x} & p_{1y}p'_{1x} & p_{1z}p'_{1x} & p_{1x}p'_{1y} & p_{1y}p'_{1y} & p_{1z}p'_{1y} & p_{1x} & p_{1y} & p_{1z} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ p_{nx}p'_{nx} & p_{ny}p'_{nx} & p_{nz}p'_{nx} & p_{nx}p'_{ny} & p_{ny}p'_{ny} & p_{nz}p'_{ny} & p_{nx} & p_{ny} & p_{nz} \end{bmatrix} \tag{37}$$

$Ah = 0$ can be solved for h, where h is the pseudo-fundamental matrix written as a vector in row major order, the same way as the case for the fundamental matrix. A least squares solution is found via a singular value decomposition, and the two non-zero singular values forced to their average.

3.2 Experimental Results

The theory for the pseudo-fundamental matrix proposed in Section 3.1 is verified experimentally with a straightforward Matlab implementation. For a chosen pair of images (one panorama and one photograph), matching points are hand picked and their coordinates recorded. These points are spread out across the entire photograph and lie of various structures to avoid too many coplanar choices. Points in the panorama are converted to 3D cube coordinates. These points are used to construct A in equation (37) to find the pseudo-fundamental matrix with a singular value decomposition. The pseudo-fundamental matrix found is then used to display corresponding epipolar lines in the two images for each match. In the case of the cubic panorama, the pseudo-fundamental matrix gives the normal of the epipolar plane. The epipolar lines on the individual faces can be found using the method in Appendix B.

The first experiment consists of a scene with a large building in the background and some benches in the foreground. Eleven matching point sets were hand picked from the two images. The pseudo-fundamental matrix for this image pair was found to be

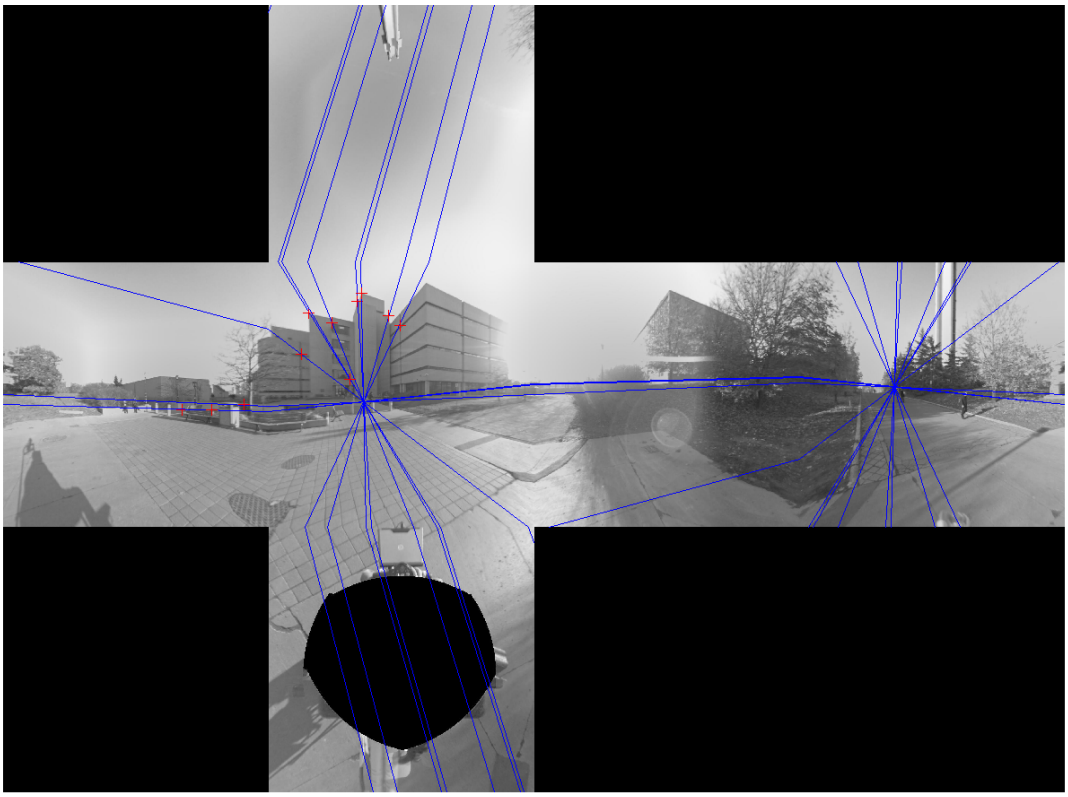
$$G = \begin{bmatrix} -0.0005 & 0.0074 & -0.0003 \\ 0.0073 & 0.0003 & -0.0021 \\ -2.8223 & -3.2588 & 1.0000 \end{bmatrix} \tag{38}$$

The resulting epipolar geometry is shown in Figure 12. Red crosses show the locations of the points chosen from the images. The blue lines are the epipolar lines found using the pseudo-fundamental matrix written above. The quality of the results can be checked visually by noticing that the epipolar lines pass through the points they are associated with. Quantitatively, Figure 13 shows how far away each match is from its corresponding epipolar line or plane in pixels. The results demonstrate that the pseudo-fundamental matrix does a good job of representing the image pair's geometry.

An additional but less detailed example is shown next in Figure 14. This image pair has the following pseudo-fundamental matrix:

$$G = \begin{bmatrix} 0.00034 & -0.00437 & -0.00002 \\ -0.00349 & 0.00028 & -0.00429 \\ 0.60940 & 2.40620 & 1.0000 \end{bmatrix} \tag{39}$$

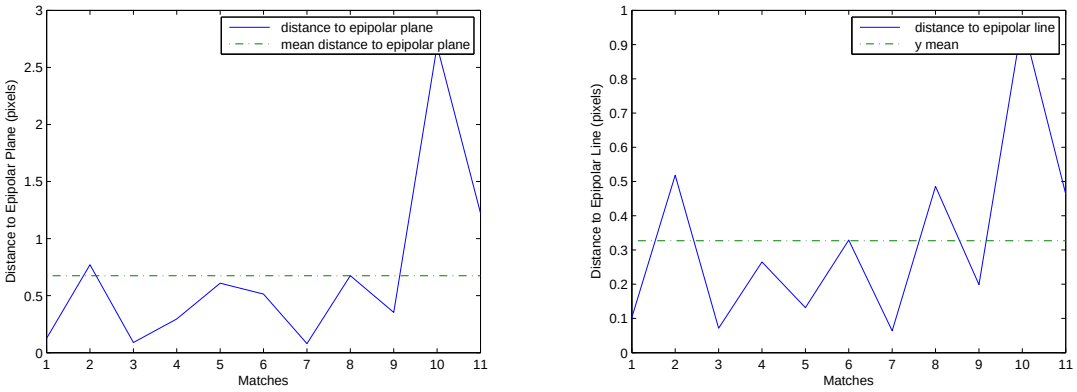
The distances from the matching points to their corresponding epipolar lines is shown in Figure



(a) Cubic panorama

(b) Photograph

Figure 12: First example of epipolar lines obtained with pseudo-fundamental matrix. Hand picked matches are shown as red crosses.



(a) Distance from matches to epipolar plane in panorama. (b) Distance from matches to epipolar line in photograph.

Figure 13: Distances of points to epipolar lines and planes for first example of finding the pseudo-fundamental matrix generated with hand-picked points.

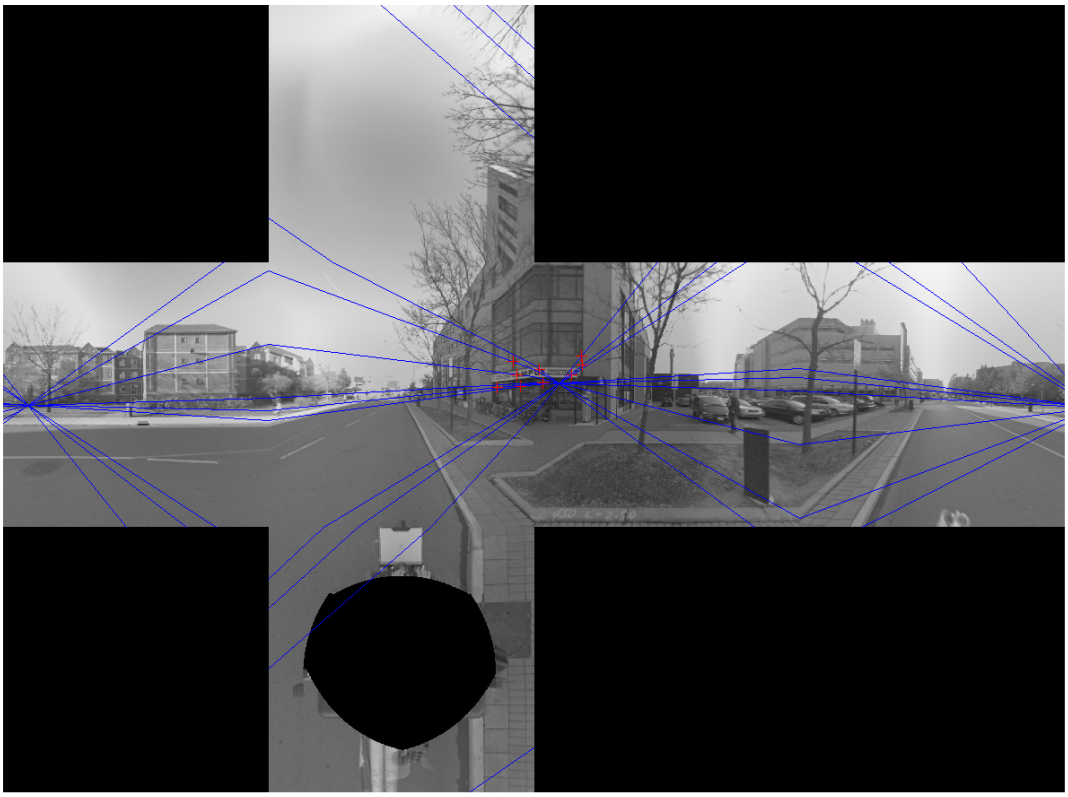
15. Again, the pseudo-fundamental matrix appears to be representing the geometry well based on a visual inspection of the distances from the hand-picked points and their corresponding epipolar lines. The points lie almost exactly on the lines, and all epipolar lines intersect at one epipole (or two, in the case of the cube), as expected.

3.3 Challenges in Matching Cubes and Photographs

Attempting to match images of a cubic panorama and planar photograph poses a new set of challenges not found when matching more similar images to each other. This section briefly outlines some of these challenges, which will become more evident in the discussion of the following chapters. Some of these issues are inevitable properties of spherical panoramas (such as field of view and focal lengths), while others are dependent on the technology used to capture the images (especially image quality). Others may depend on the type of projection used for the panorama (viewing angle) or the typical content of the matching application (repeating features).

3.3.1 Image Quality

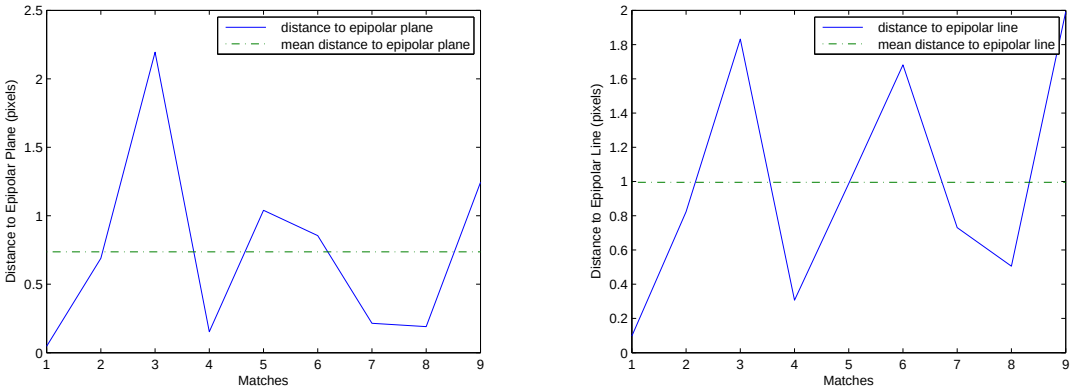
The first difference between some panoramas and photographs, most obvious when comparing many of them side-by-side, is the image quality. Panorama cubes that are built from Ladybug camera



(a) Panorama

(b) Photo

Figure 14: A second example of epipolar lines found with a pseudo-fundamental matrix generated from hand-picked points.



(a) Distance from matches to epipolar plane in panorama. (b) Distance from matches to epipolar line in photograph.

Figure 15: Distances of points to epipolar lines and planes for second example of finding the pseudo-fundamental matrix.

data, for example, are generated by fusing data from six relatively small 1024×768 CCD micro-sensors. Interpolation and data averaging can cause hazing and blurring effects in certain areas of the resulting images. At the same time, because so much of a scene is fit into a single panorama of manageable resolution, each individual building or other structure becomes small enough to have a relatively low resolution in its details.

Neither of these issues would be as much of a challenge if the task was to simply view the panoramas, or to match them to each other. When comparing them to photographs, which often focus on one structure with the same resolution as an entire panorama, these issues can make a noticeable difference. Figure 16 shows two pieces of a single panorama spanning several faces, and demonstrates how poor the image quality can be.

The situation is much improved when using panoramas created from consumer camera images, such as the example in Figure 10. More than five individual images were fused together for this panorama, and each one had a higher resolution than what a sensor on the Ladybug camera can capture. Quality can occasionally suffer with this method when images are not easily aligned to each other.

3.3.2 Field of View

By the very fact that panoramas cover a 360° field of view, a single cube image contains much more of a scene than a photograph can. Generally speaking, most photos show about as much as one and

Figure 16: Image quality of some cube panoramas.

a half to two cube faces at a maximum, depending on how wide the camera lens is. Most photos in the examples of this chapter show even less. For instance, on a camera with a sensor equivalent to 35mm film, 50mm focal lengths are common and give an approximately 47° field of view [9]. An ultra-wide angle lens with a 12mm focal length gives an approximately 121° field of view [9], but isn't nearly as common.

The challenge lies in knowing which subset of features in the panorama actually match the photo's features. Unless the relative orientation of the photo is known (say, from a compass reading), features from all faces of the cube are candidate matches to the features in the photo. While potential matches can likely be eliminated if most are clustered together in a single area, doing so after checking for matches amongst all the candidate pairs will be more computationally demanding than standard matching algorithms.

To further complicate the situation, the common subject between the photo and panorama might actually span more than one face of the cube. The features close to the seam between these faces may not have matchable features because of the sharp change in viewing angle. When feature descriptors are created on such points using information from both faces, the result will be different than the same feature described on a photograph without any viewing angle changes.

3.3.3 Repeating Features

For sets of panoramas that were captured outdoors, as is the case with all of the examples shown here, the main subject is often a large man-made structure, such as a building. This is especially true when images might be taken at varying times of the year, since as seasons change, snow covers the ground and leaves may be on or off the trees, making these structures unmatchable. The people and cars appearing in the images will always be different, unless they happen to be captured at the same moment. For all these reasons, buildings often provide the only constant features available for matching.

The downside of this is that many buildings contain repetitive features and large areas of isotropic texture. When one cannot tell apart the multiple indistinguishable areas in the two images, it is very challenging to sort out which feature matches up with which other feature. This particular challenge is not unique to matching with panoramas; rather, it might be commonly encountered when matching panoramas to photographs because of the need to rely on matching features from buildings to be successful.

3.3.4 Viewing Angle

Viewing the same parts of a building or object from various angles can make those areas look rather different in the final image. While humans can recognize what they are looking at up to fairly extreme angles, it is difficult for matching algorithms to do the same. As the tilt increases, the projected shapes in the image change, too.

Each face of a cubic panorama represents a change in viewing angle relative to all the others. If the viewing angle and subject matter of the photograph happens to be similar to a particular face of the cube, then there won't be an issue. It may even be possible to rotate the cube until this is the case. But if that subject spans multiple faces, then there is no choice but to compare it at varying viewing angles.

3.3.5 Focal Lengths

The Ladybug's micro lenses used to capture image data for panoramas each have a wide field of view with focal lengths of just 2.5 mm. Indeed, since there are five lateral lenses, each of them must cover at least 72° to capture the whole surrounding sphere.

While there are cameras that can be equipped with ultra-wide angle lenses, they are less common among basic consumer products such as compact or mobile digital cameras. This is another difference found between the images in panoramas and photographs, posing a new set of challenges when trying to match them.

Distortion effects are common at such a short focal length. The images are also likely to differ because of the effect of compression. The longer the focal length for a lens (or, equivalently, the more zoomed in the lens is), the closer together objects seem to be in the resulting image. This means that a tree or a sign will obscure the face of a building more for a regular camera with a longer focal length than it will for the panorama, even when the building is around the same size in both images. See Figure 17 for an example of this effect.

3.4 Conclusion

This chapter established the geometry between a cubic panorama and a planar photograph. It went on to show that the fundamental matrix could be modified to allow one image to be calibrated, resulting in a matrix proposed as the pseudo-fundamental matrix. This pseudo-fundamental matrix

Figure 17: An example demonstrating the effects of compression when using a longer focal length (right) than the panorama (left).

works well for representing the cube-photo pair geometry, as demonstrated with several sets of hand-picked points. Finally, some the challenges that will be faced in the next chapters while trying to match the panorama with the photograph automatically were mentioned. This thesis will address some of these challenges.

Chapter 4

Matching with Speeded Up Robust Features

This chapter is the first of several that evaluate matching techniques for use with panoramas and photographs. Basic image matching was introduced in Chapter 2. Given enough matches, the concepts established in Chapter 3 may be used in conjunction with RANSAC to eliminate incorrect matches and find the epipolar geometry between the panorama and photo.

The following sections describe how Speeded Up Robust Features, or SURFs, are detected and described. This is followed by an experimental evaluation of these features for a variety of typical panoramas, including several challenging cases.

4.1 Detecting and Describing Speeded Up Robust Features

Speeded Up Robust Features [3], or SURFs, are similar in concept to SIFT features [25]. The main idea behind their creation was to provide a detector and descriptor system that would perform at least as well as state of art systems, but that would work more efficiently. SURF is based on an approximation of the Hessian matrix where SIFT uses a basic Laplacian-based descriptor. The process for detecting and describing SURF is summarized in Algorithm 4.1.

4.1.1 Feature Detection

The SURF detector is based on the Hessian matrix for its efficiency and accuracy. The determinant of this matrix is used to select the location and scale of the interest points (blob-like structures are detected where the determinant is maximum). At a point $\mathbf{x} = (x, y)$, the Hessian matrix is defined

Algorithm 4.1 Detecting and describing SURF interest points.

1. Apply Hessian-approximating box filter to image at different scales.
2. Interest points are localized using a localized maximum search, where the local maximum is greater than all its neighbours.
3. For each interest point detected:
   (a) Find the orientation of the interest point using Haar wavelet responses.
   (b) Define a square measurement region 20 times the scale of the interest point, rotated to the orientation determined in the previous step.
   (c) Divide the measurement region into a 4×4 grid of subregions.
   (d) Compute Haar wavelet responses at 5×5 regularly spaced sample points.
   (e) For each subregion:
      i. Weigh responses with a Gaussian with $\sigma = 3.3s$ centred at the interest point.
      ii. Compute sum and absolute sum of responses in the vertical direction.
      iii. Compute sum and absolute sum of responses in the horizontal direction.
   (f) Collect 4 values from each of the 16 subregions to create an interest point descriptor, a vector of dimension 64.

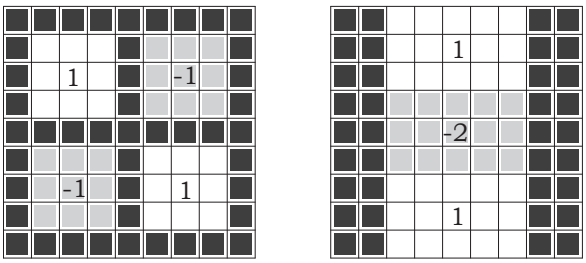


Figure 18: Box filter approximations for the second order Gaussian partial derivative of the in xy- (left) and y- (right) directions.

as

$$H(x, y, \sigma) = \begin{pmatrix} L_{xx}(x, y, \sigma) & L_{xy}(x, y, \sigma) \\ L_{xy}(x, y, \sigma) & L_{yy}(x, y, \sigma) \end{pmatrix} \tag{40}$$

where $L_{xx}(x, y, \sigma)$ is the second order derivative $\frac{\delta^2}{\delta x^2} g(\sigma)$ of the Gaussian for an image I at point $\mathbf{x}$ (similarly for L_{yy} and L_{xy}).

Instead of discretizing and cropping the Gaussian function as would normally be done, a more efficient, approximating box-filter is proposed. The filters for the xy- and y-directions are shown in Figure 18. These filters can be evaluated very efficiently; in fact, when used in conjunction with integral images, their computational cost can be independent of the filter size.

An integral image is like a tally of sums of pixel values in a rectangle defined by the image origin and a particular point $\mathbf{x} = (x, y)$. More formally,

$$I_\Sigma = \sum_{i=0}^{i \leq x} \sum_{j=0}^{j \leq y} I(i, j) \tag{41}$$

With this integral image precomputed, it only takes three additions to obtain the sum of all values within a rectangular region. For example, consider the box filter shown in Figure 19. To obtain the sum Σ of all the pixel values in the white box, where A, B, C, and D are the values of the integral image at the box corners, the following equation may be used:

$$\Sigma = A - B - C + D \tag{42}$$

After the sum of the box areas of the filters are found, the multiplying factors seen in Figure 18 can be applied.

Denote the filters seen in Figure 18 (plus the unillustrated filter in the x-direction) by D_{xx}, D_{yy}, and D_{xy}. These 9×9 boxes are used to obtain blob response maps by approximating the determinant

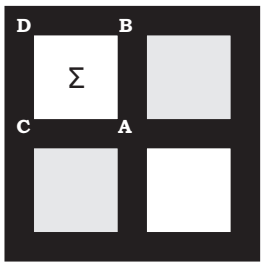


Figure 19: Three additions from the integral image can be used to get the total value Σ for the white box in the filter defined by A, B, C, D.

of the approximate Hessian matrix H_{approx} at a particular location $\mathbf{x}$ with

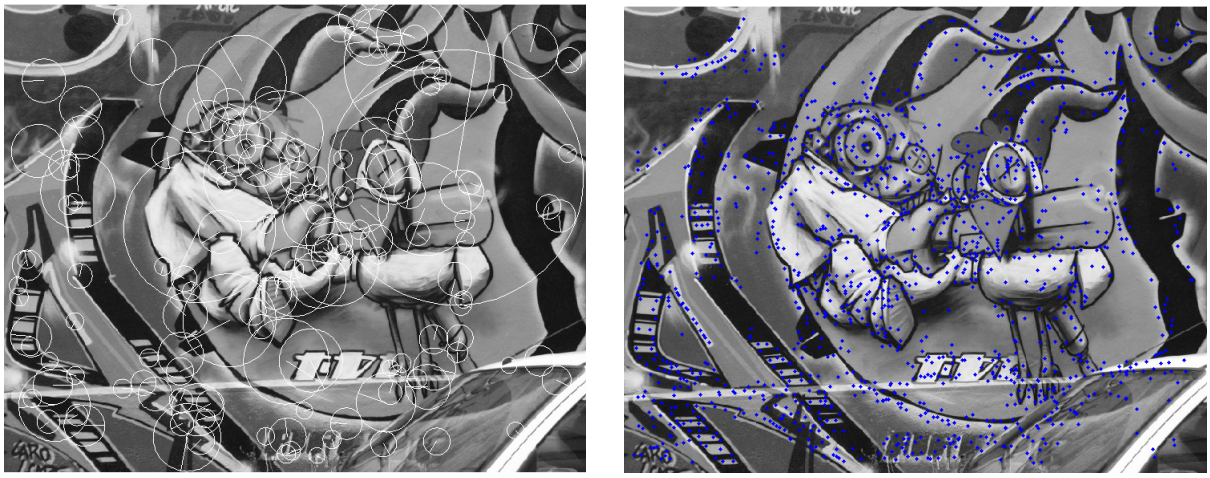
$$det(H_{approx}) = D_{xx}D_{yy} - (wD_{xy})^2 \tag{43}$$

where w is the relative weight of the filter responses used to balance the expression. The default value of w suggested by the authors, $w = 0.9$, is used for all experiments.

A scale space is used to detect feature points, but instead of reducing the image size iteratively (as is done by SIFT), the box filters can instead be scaled up. As mentioned above, applying these filters using the integral image is quite efficient, so working with progressively larger filters will be faster than resampling the image multiple times. The 9×9 filter shown in Figure 18 is used for the first scale. Because it approximates Gaussian derivatives with $\sigma = 1.2$, this scale is referred to as $s = 1.2$.

The scale space is divided into octaves, with each octave representing a series of response maps computed with filters of increasing size until they are fully doubled. Because of the discrete nature of the box filters, the size of each box in the filter must always increase by an even number of pixels to guarantee the presence of a middle pixel. Also, the first and last Hessian response maps in the stack cannot contain the maxima sought for image point localization. Thus filters of size 9×9, 15×15, 21×21, and 27×27 are applied in the first octave, even though this more than doubles the scale change. For each new octave, the filter size is increased by twice as much as it was for the previous octave. The filter sizes for the second octave are then 15, 27, 39, and 51, and the sizes for the third octave are 27, 51, 75, and 99. If the image size is larger than the corresponding filter sizes after the third octave, a fourth is added with sizes 51, 99, 147, and 195.

After the filters are applied, interest points are localized in the image and over the scales using a non-maximum suppression (a localized maximum search, where the local maximum is greater than all its neighbours) in a $3 \times 3 \times 3$ neighbourhood, and the maxima of the determinant of the Hessian



(a) 155 of 1549 detected SURFs, shown with the scales and rotations associated to the features.

(b) All 1549 detected SURFs plotted as points on the image.

Figure 20: Example of detected Speeded Up Robust Features on the graffiti photograph available online [1].

matrix are interpolated in image and scale space.

Figure 20 shows an example of detected SURF points on a photograph of graffiti, available online [1] for use in evaluating various feature detectors. Figure 20a shows a selection of SURF points with their associated scales, and Figure 20b pinpoints the location of all SURF points.

4.1.2 Feature Description

Feature description for SURF points is somewhat similar to the process used for SIFT. The first step in both is to determine an orientation for the interest point so the description of it will be robust to image rotations. For SURF, this is done using Haar wavelet responses.

The measurement region for determining orientation is circular, and is six times the size of the scale the point was detected at. The size of the wavelets is four times the scale. Figure 21 shows the two filters used. Based on their nature, integral images can again be used to compute the response. The result is weighted with a Gaussian with $\sigma = 2s$ (where s is the scale) centred at the SURF point.

The wavelet responses are represented as points in Euclidean space, with the responses in the x-direction along the x-axis and responses in the y-direction on the y-axis. A sliding window is used to find the largest sum of responses in a given window, represented as a vector. The orientation of this vector then becomes the orientation of the feature point. Rotation invariance can actually be

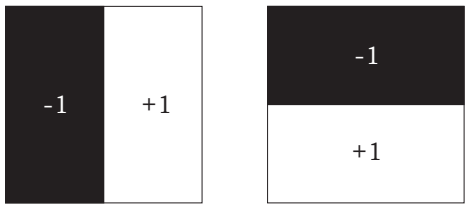


Figure 21: Haar wavelet filters in the x- and y-directions used to compute SURF point orientation and descriptor.

skipped altogether for applications that don't require robustness to rotations greater than $\pm 15^{\circ}$.

The measurement region for computing the actual descriptor is a square region twenty times the scale of the feature point, oriented as per the rotation determined in the previous step. This region is then split up into a 4×4 grid. For each sub-region, the Haar wavelet responses are computed at 5×5 regularly spaced sample points. The horizontal and vertical filters are oriented the same way as the measurement region. The responses are weighted with a Gaussian with $\sigma = 3.3s$ centred at the interest point. For each of the regions in the 4×4 grid, the wavelet responses are summed up in both the horizontal and vertical directions. The absolute values of the responses are also summed up, giving a total of 4 values for each of the regions. Using these values to make a descriptor for the feature point results in a vector of length 64.

The SURF descriptor is invariant to a bias in illumination, and is invariant to contrast when turned into a unit vector. SURF is also more resilient to noise than SIFT is, since SIFT uses gradient directions for its descriptor, while SURF integrates gradient information within the sub patches.

4.2 Performance in Panoramas and Photographs

A recent study [2] suggested that SURF was a better choice of feature detector and descriptor than SIFT because SURF is more efficient, yet gives matching results comparable to SIFT. The study measured the correct matches that both SIFT and SURF produced per second of actual runtime, and found that SURF produced at least twice as many matches per second than SIFT did. As mentioned earlier, SURF is also supposed to be more robust under noise, which may be beneficial when matching panoramas of lower image quality (see Section 3.3). The SURF detector also generally finds fewer interest points. Given that the entire panorama will be matched with a photograph, fewer points will make comparing the images easier, so long as there are enough good matches among the points in the first place. However, like SIFT, SURF is not affine covariant, and

will not work well for largely varying viewing angles.

In this section, SURF detection, description, and matching will be analyzed for a range of panorama and photograph image pairs.

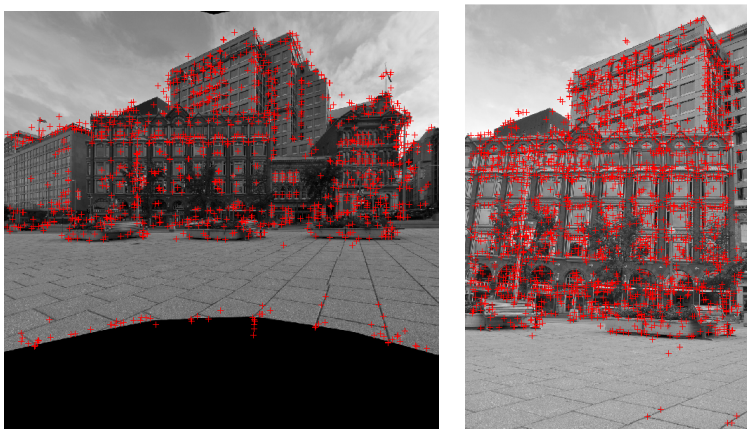
4.2.1 Detecting SURFs

The performance of SURF point detection will be explored through example. Several panorama-photograph image pairs are examined to see whether the feature points are repeatable (i.e., whether the detected SURF points appear in the same location for both views). Note that only the most relevant face of the panoramas is used in these experiments, and the panoramas were created with Ladybug camera data unless otherwise stated. The full panoramas are shown in Appendix C.

The first two examples come from a panorama created by the author using a standard digital single lens reflex camera and the freely available Hugin [8]. The photos used for matching and to make the panorama were taken at the National War Memorial on Elgin Street, downtown Ottawa. The full panorama was shown in Figure 10 on page 25. Both images in Figure 22 have very similar viewing angles, differing mostly by scale. As a result, there are many potential matches, even though the photograph contains more features. There is an even bigger scale difference between the images in Figure 23, but even still, there are many matching feature locations. This is more obvious in the zoomed version of the panorama seen in Figure 23c. The potential issue with the large scale changes is that there are many features in one image (the panorama in this case) that don't exist in the other image, but may be considered potential matches. (This issue will be exaggerated even more when considering the entire panorama, rather than a single face.)

The first example that includes a panorama made with the Ladybug camera is shown in Figure 24, where a biology building at the University of Ottawa has been captured. The area with the largest quantity of common points detected is around the doors. Though there is not a one-to-one correspondence, there are certainly many corresponding points. There are many points in the common areas of the images that don't have any match at all, but there are often points in matching locations nearby. For example, some of the small windows along the top of the building have multiple points in the photograph, but only one in the panorama. That one point in the panorama does appear in a consistent location in both images.

The next example is of a scene that includes a pharmacy sign, Figure 25. The features on this sign turn out to be the most reliable between the two images because of the large scale difference between the panorama and the photograph. It is interesting to note that the upper half of the



(a) Panorama (front face) (b) Photograph

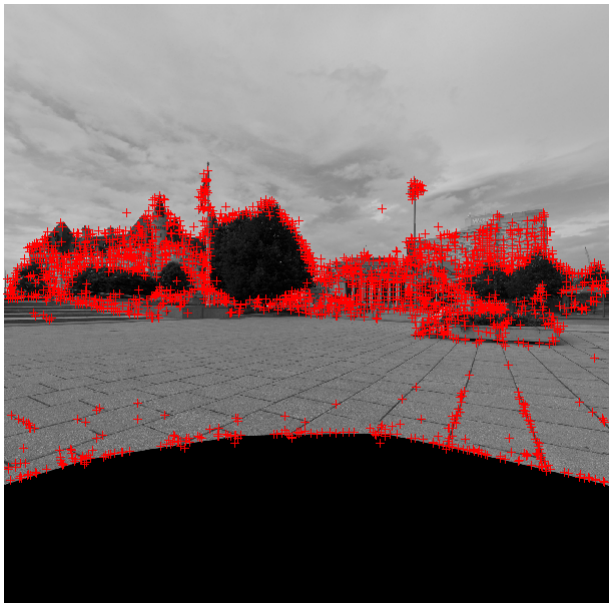
Figure 22: Elgin Street buildings image pair.

panorama has fewer features than any area in the photograph; this may be a result of the lower image quality and lower image contrast. Similarly, the sign itself has fewer features detected in the panorama.

The images in Figure 26 show the side of the building housing the pharmacy of Figure 25. The viewing angle here differs greatly between the panorama and the photograph, yet SURF points are detected more reliably than expected. The highlight on top of the building in the panorama impedes the ability to match features in that area, and is a typical effect that appears in panoramas created with Ladybug camera data. (In fact, the effect is often much worse, making some panoramas all but unusable.) Despite the reliable detection, matching is expected to be more difficult, given that SURF was not designed to be affine invariant.

The example in Figure 27 depicts a building called the Cube at the University of Ottawa. This building does not have many descriptive features, and so neither image in Figure 27 contains very many SURF points. (Note that the panorama contains part of an adjacent face so there is more overlap with the photograph.) Both the photograph and panorama have features around the windows, but only the photo has many around the other corners of the building, such as along the top where the vertical siding meets the roof line. Because of this, and the repetitive nature of the windows, matching will likely be challenging for this example.

The environmental genomics building at the University of Ottawa is shown in Figure 28. Though not completely visible in the one face shown of the panorama, there is a large hazing effect to the right of this building. The image quality of the building is not good, with a noticeable blur appearing

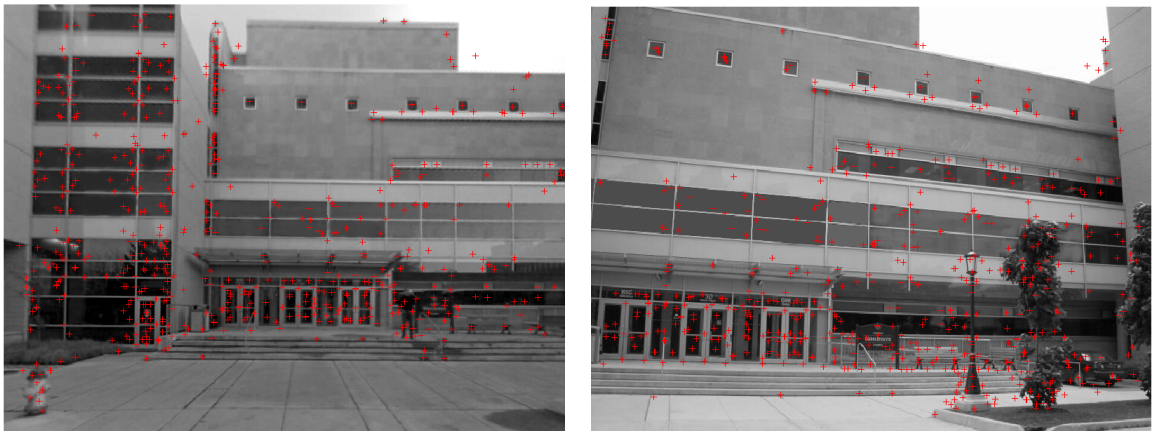


(a) Panorama (back face)

(b) Photograph

(c) Panorama zoomed in

Figure 23: Westin Hotel image pair.



(a) Panorama (right face) (b) Photograph

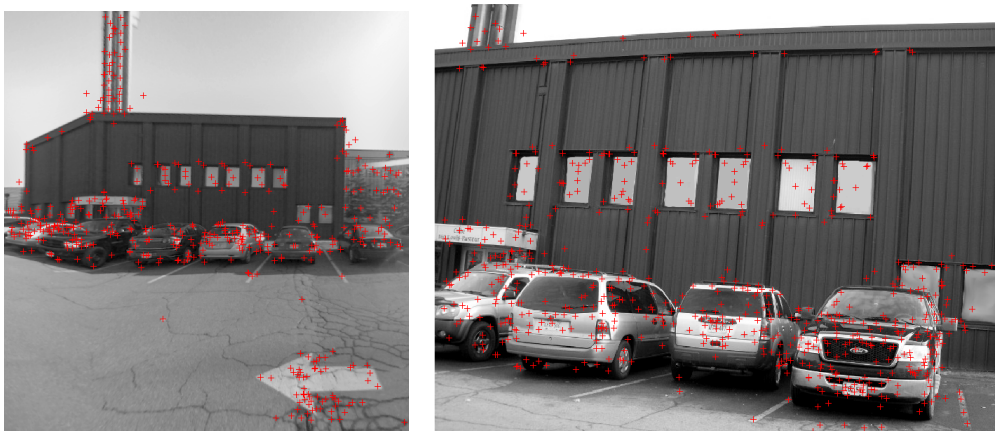
Figure 24: Biology building image pair.

(a) Panorama (right face) (b) Photograph

Figure 25: Pharmacy image pair.

(a) Panorama (left face) (b) Photograph

Figure 26: Side of pharmacy image pair.



(a) Panorama (right face) (b) Photograph

Figure 27: Cube building image pair.

(a) Panorama (left face) (b) Photograph

Figure 28: Environmental Genomics building image pair.

particularly around the windows. The long thin windows along the top of the building are barely recognizable compared to the clear shapes of the same windows seen in the photograph. There are a few points that appear in approximately the same location in these two images, but not many.

Matching results for these examples are presented in the next section.

4.2.2 Matching SURFs

For each example in Section 4.2.1, as well as the graffiti photograph pair in Figure 54, SURF descriptors were found and matched using the thresholding and nearest neighbour ratio thresholding techniques from Section 2.1.3. In both cases, the distance metric for evaluating the quality of individual matches is Euclidean distance. The graffiti pair is included in the results for comparison, since it is a relatively easy scene to match.

Threshold matching results are shown in Table 1and nearest-neighbour ratio results are shown in Table 2. The matches are evaluated manually in that each match is examined visually, and the number of correct matches tallied.

Most image pairs have either a low matching rate with at least twice as many incorrect matches as correct matches, or a higher matching rate with a small number of absolute matches. The exception is the graffiti image pair, a relatively easy image to match for its clear features and small change in viewing angle. From all examples, it is clear that relaxing the threshold results in having more good matches, but also allows for bad matches at a higher rate.

Image Pair	**Threshold 0.15 Number Correct**	**Threshold 0.15 Percent Correct**	**Threshold 0.2 Number Correct**	**Threshold 0.2 Percent Correct**	**Threshold 0.3 Number Correct**	**Threshold 0.3 Percent Correct**
Graffiti	137	95.8%	337	91.3%	537	53.5%
Elgin Street	39	70.9%	104	58.4%	232	28.4%
Westin	0 (out of 0)	-	4	80%	49	25.3%
Biology	7	46.7%	24	35.3%	62	16.3%
Pharmacy	15	93.8%	40	95.2%	64	28.4%
Side of Pharmacy	0 (out of 0)	-	0 (out of 10)	0%	6	3.1%
Cube	2	40%	5	45.5%	20	32.8%
Genomics	0 (out of 7)	0%	2	4.7%	7	7.5%

Table 1: SURF threshold matching results.

For some images, it appears that no reasonable threshold will result in a high number or percentage of correct matches. This is not surprising for the side of the pharmacy image or the genomics image: the former has a big change in viewing angle, which SURF was not designed for, and the latter has repetitive features, exhibits a moderate but noticeable angle change, and has a low quality panorama. However, it would be desirable to see more matches for the other image pairs.

The nearest neighbour ratio matching technique yielded better results for some image pairs (including the biology building, side of the pharmacy, and the Westin hotel). This implies that the descriptors for some of the matching points have a larger distance than the thresholds used above, and are thus only found with this strategy. In these cases, a simple change in the threshold used for basic matching is not always the answer since, as seen above, the percentage of good matches drops as the threshold raises. A better alternative would be to distinguish matches so the threshold could be raised without allowing too many false matches.

Image Pair	**NN Ratio 0.8 Number Correct**	**NN Ratio 0.8 Percent Correct**
Graffiti	637	95.2%
Elgin Street	155	68.9%
Westin	45	42.9%
Biology	29	38.2%
Pharmacy	61	56.5%
Side of Pharmacy	13	18.3%
Cube	21	30.0%
Genomics	2	4.2%

Table 2: SURF nearest neighbour ratio matching results.

4.3 Conclusion

This chapter evaluated the use of Speeded Up Robust Features with panoramas and photographs. Chosen for their speed and relative robustness to noise, SURFs performed well for some matching cases, but provided few reliable matches for the more difficult images. Based on the experimental results, increasing the threshold will not necessarily help, as this will simply allow more incorrect matches into the match set. The next chapter will evaluate another type of feature called Maximally Stable Extremal Regions, after which the issue of matching repetitive features will be addressed.

Chapter 5

Matching with Maximally Stable Extremal Regions

The previous chapter introduced Speeded Up Robust Features, or SURFs, and evaluated their detection and matching abilities for panorama-photograph pairs. This chapter explores a different feature detector that finds what are called Maximally Stable Extremal Regions, or MSERs. These features can be described in an affine covariant way, and may be useful for use with images of buildings, since the general shapes of windows and signs may be matchable even in cases of differing image quality.

A detailed explanation of how MSERs are detected opens this chapter. A method of describing MSERs is presented next, followed by an analysis of the detection and matching abilities of MSERs for panoramas and photographs.

5.1 Detecting and Describing Maximally Stable Extremal Regions

Before any matching can occur, a set of MSERs must be reliably detected in the two images to be matched. This section explains how MSERs are detected and described in the implementation used for the rest of the chapter. The process is summarized in Algorithm 5.2.

5.1.1 Feature Detection

Maximally stable extremal regions, or MSERs, were first introduced by Matas et al [26]. Informally, the definition of MSER regions can be thought of as follows. An image is imagined as a topographical map with a bird's eye view, where pixels with a greater intensity value represent higher ground. The image is then slowly flooded, with areas of lower ground filling with water first. Certain areas retain their shapes for long periods of time. For instance, a window in the image would have a strong

Algorithm 5.2 Detecting and describing MSER features.

1. Sort pixels in image in ascending order according to their intensity values as $x_1, x_2 \ldots, x_n$.
2. Add the first pixel x_1 to a forest of pixels.
3. For each pixel $x_i = x_2 \ldots, x_n$:
   (a) Find any pixels adjacent to x_i in the forest and add x_i as the parent of forest nodes representing all such pixels, or start a new tree.
4. Compute the stability of the regions represented by trees, and select those which are maximally stable.
5. Remove unnecessary regions: very small or very big regions, regions which have too high area variation, and duplicates.
6. For each remaining region:
   (a) Create a binary mask representing the region on the image.
   (b) Resample the relevant portion of the mask into a square patch, using its covariance to determine the affine normalization required to transform it into a circle. Call this the shape patch.
   (c) Repeat, creating the patch using pixels from the actual image instead of the binary mask. Call this the texture patch.
   (d) Compute a SIFT descriptor for each of the shape and texture patches.

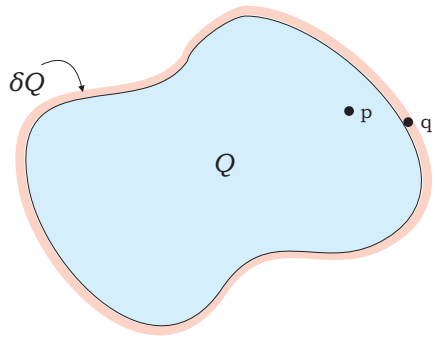


Figure 29: A region Q is a contiguous set of pixels in an image I with outer boundary δQ. Q is an extremal region if for all pixels $p \in Q$ and $q \in \delta Q$, $I(p) > I(q)$ for a maximum intensity region, or $I(p) < I(q)$ for a minimum intensity region.

contrast compared to the adjacent walls of the building, and as water filled the window area, it would retain its shape for some time. The process is then reversed so that high intensity pixels represent low ground. The most stable shapes become MSERs. This idea is also explained well by Chutorian and Trivedi [33].

More formally, an image I is defined as a mapping from a pixel coordinate to an intensity value $I : D \subset \mathbb{Z}^2 \rightarrow S$. For all experiments in this paper, the intensity values are $S = \{0, 1, \ldots, 255\}$, though MSERs can be detected on images with real values. A 4-neighbourhood adjacency relation is defined on the image. In other words, pixels p and q are adjacent (written as pAq) if and only if $\sum_{i=1}^{d} |p_i - q_i| \leq 1$ (that is, pixels adjacent to p are all those pixels q which have a distance from p of one or less).

A region Q in an image is defined as a contiguous subset of D. That is, for any two pixels p and q in region Q, there is a sequence $p, a_1, a_2, \ldots, a_n, q$ such that subsequent pixels are adjacent to each other, as in pAa_1, a_iAa_{i+1}, and a_nAq. The entity δQ is called the outer region boundary, and consists of pixels outside of Q that are adjacent to at least one pixel inside Q. Region Q is called an extremal region if for all pixels $p \in Q$ and $q \in \delta Q$, $I(p) > I(q)$ for a maximum intensity region, or $I(p) < I(q)$ for a minimum intensity region. These concepts are illustrated in Figure 29.

Consider a sequence of nested regions $Q_1, \ldots, Q_{i-1}, Q_i$, depicted with three regions in Figure 30. One of these regions Q_{i^*} is considered maximally stable if and only if $q(i) = |Q_{i+\Delta} \setminus Q_{i-\Delta}| / |Q_i|$ has a local minimum at i^*, where $|.|$ denotes cardinality and $\Delta \in S, \Delta > 0$ is chosen by the user of the detection algorithm.

Figure 31 shows an example of the graffiti image first seen in Figure 43, and twenty distinct

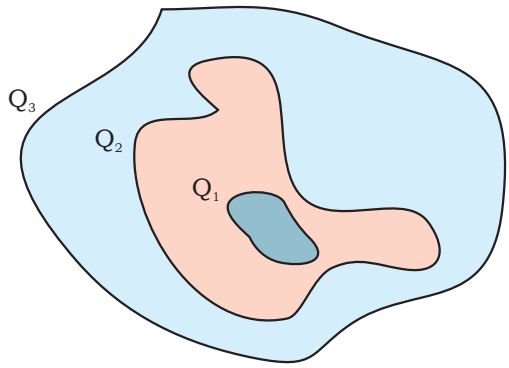


Figure 30: Several nested regions Q_1, Q_2, Q_3.

(a) Original image.

(b) First twenty MSERs detected.

Figure 31: Example of twenty MSERs from an image.

MSERs from this image superimposed on the right. In this example, all the MSERs shown were found as minimum intensity regions.

The implementation of MSER detection used in the experiments in this chapter is part of the freely available, open source VLFeat vision library [43], and is based on the concept of union-trees [33]. The reader is invited to read a corresponding technical report detailing the implementation [42]. The main ideas from it are briefly explained next.

In the VLFeat implementation, the criteria for which a particular extremal region is considered maximally stable is similar to that described above. A *level* $I(Q)$ is defined on an extremal region Q as the maximum intensity value within Q:

$$I(Q) = \max_{p \in Q} I(p) \tag{44}$$

As before, a $\Delta > 0$ is chosen by the user of the algorithm. Let $Q_{+\Delta}$ be the smallest extremal region that completely contains Q whose intensity values are at least Δ larger than those found in Q. Similarly, let $Q_{-\Delta}$ be the largest extremal region completely contained inside of Q whose intensity values are at least Δ smaller than those in Q. Now let the area variation ρ for Q be defined as

$$\rho(Q, \Delta) = \frac{|Q_{+\Delta}| - |Q_{-\Delta}|}{|Q|} \tag{45}$$

Region Q is considered maximally stable if $\rho(Q, \Delta)$ is smaller than $\rho(R, \Delta)$ for any extremal region R that is either immediately contained in Q or immediately contains Q in the sense that if R' is some third extremal region, $Q \subset R' \subset R$ or $R \subset R' \subset Q$ implies that $R = R'$.

To find extremal regions in an image, pixels are first sorted according to their intensity values so that pixels $x_1, x_2 \ldots, x_n$ have intensity values $I(x_1) \leq I(x_2) \leq \ldots \leq I(x_n)$. Then the pixels are added to a forest of pixel trees in increasing order. The first pixel x_1 is the pixel (or one of several pixels) in the image with the lowest intensity value. It automatically forms an extremal region of I_1. I_i is a subset of the image I which contains only the pixels processed so far (in the case of I_1, just the first). For each subsequent pixel x_{i+1}, all adjacent pixels y are found in the forest. For each tree containing a neighbour y, x_{i+1} is added as the parent of the root. This, in effect, joins the new pixel to the union of all the previous pixels in a particular region represented by a tree in the forest. Figure 32 shows a small example of this process.

In order to compute the stability of the extremal regions found above, these regions are placed into a tree where a region is the parent of another if it immediately contains it. To find the stability score of a particular region Q, the regions $Q_0 = Q$, $Q_1 = parent(Q_0)$, $Q_2 = parent(Q_1)$, and so on are examined. The region satisfying $R_{-\Delta} = Q_0$ with the maximum area is used to calculate the stability score. To satisfy $R_{-\Delta} = Q_0$, the following will hold:

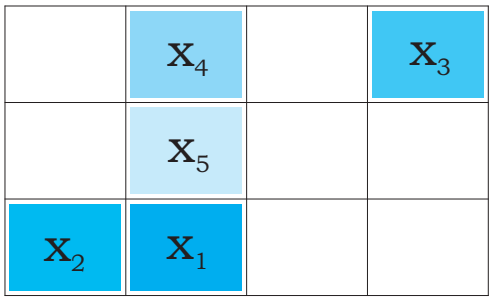
$$I(Q_0) \leq I(R_i) - \Delta < I(R_i) \tag{46}$$

Similarly, the region satisfying $R_{+\Delta}$ is found. In this case, there will be only one such region, and the following condition is sufficient to find it:

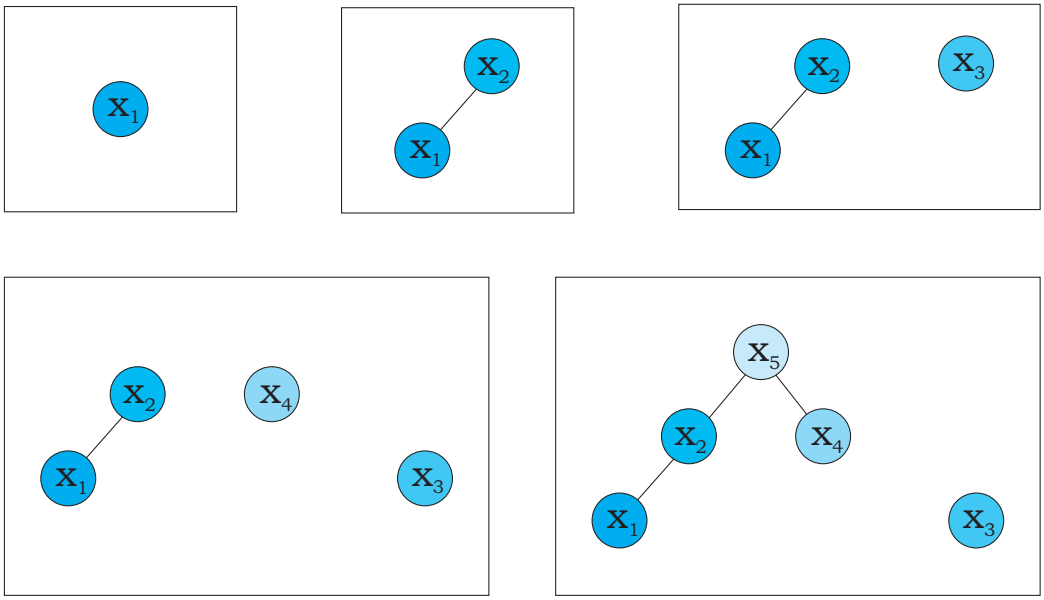
$$I(R_i) \leq I(R_0) + \Delta < I(R_{i+1}) \tag{47}$$

The final step of the detection process is to clean up regions that are deemed to not be useful. For instance, regions whose areas are too large or small may be removed, and regions that have too high an area variation might be eliminated even when those regions represent the minima. Finally, duplications can be deleted.

The example in Figure 31 was created using this VLFeat implementation of MSER detection.



(a) A small set of pixels. x_1 to x_5 have the five lowest image intensity values.



(b) A sequence of pixel forests, created by adding one of x_1 to x_5 at a time.

Figure 32: A simple example to demonstrate the forest of pixels generated from an image to find MSER regions.

5.1.2 Feature Description

Unlike the description of SIFT and SURF features (Section 2.1.2 and 4), where some measurement region around a single point was defined based on some predetermined scale for that feature, the shape of an MSER region itself gives a measurement area. Further to this, MSERs can be analyzed in an affine covariant manner by finding a bounding ellipse and transforming the area it contains so that the ellipse becomes a circle. After this transformation, any number of techniques can be applied to produce a descriptor for the MSER. The method of finding the bounding ellipse is shown first below, followed by a discussion on the affine normalization used in experiments.

The region's centre of gravity (or centroid) and covariance matrix are used to find the bounding ellipse of an MSER region [24, 34, 13]. The MSER region R is taken as a binary mask B of the original image I, where pixels are black if they are not part of the MSER, and white if they are.

$$B(x,y) = \begin{cases} 1 & (x,y) \in R \\ 0 & (x,y) \notin R \end{cases} \tag{48}$$

The area A of the region R is simply the number of pixels in the region.

$$A = \sum_{x,y \in I} B(x,y) \tag{49}$$

The centroid of the region $\mu = (\mu_x, \mu_y)$ can be determined using B and (49).

$$\mu_x = \frac{1}{A} \sum_{x,y \in I} xB(x,y) \tag{50}$$

$$\mu_y = \frac{1}{A} \sum_{x,y \in I} yB(x,y)$$

The covariance matrix Σ is found using (49) and (50) as

$$\Sigma = \begin{bmatrix} \sigma_x^2 & \sigma_{xy} \\ \sigma_{xy} & \sigma_y^2 \end{bmatrix} \tag{51}$$

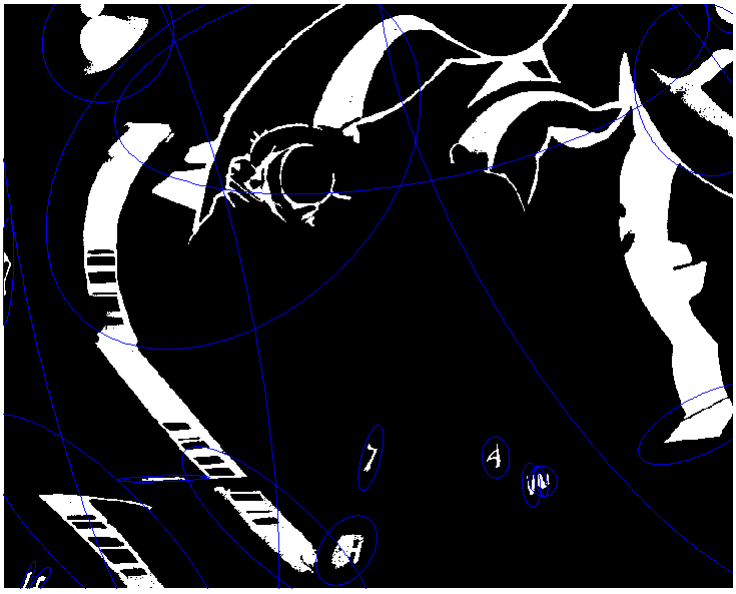


Figure 33: Bounding ellipses drawn for twenty MSER regions found in the image in Figure 31.

with

$$\begin{aligned}
\overline{x} &= x - \mu_x \\
\overline{y} &= y - \mu_y \\
\sigma_x^2 &= \frac{1}{A} \sum_{x,y \in I} \overline{x}^2 B(x,y) \\
\sigma_y^2 &= \frac{1}{A} \sum_{x,y \in I} \overline{y}^2 B(x,y) \\
\sigma_{xy} &= \frac{1}{A} \sum_{x,y \in I} \overline{xy} B(x,y)
\end{aligned}$$

The centre of the bounding ellipse is given by μ. The direction of the bounding ellipse axes can be obtained from the eigenvectors of Σ^{-1}. The ratio of the lengths of the ellipse axes is given by $\sqrt{\lambda_1}/\sqrt{\lambda_2}$, where λ_1 and λ_2 are eigenvalues of Σ^{-1}. Figure 33 shows the bounding ellipses for the twenty MSER regions in Figure 31.

While the bounding ellipse itself is useful for visualization, it is not strictly necessary for the normalization of MSER regions; only the covariance matrix Σ is needed. If a normalization of the entire image is desired, then an image point $\mathbf{x}$ can be transformed to its normalized position $\mathbf{x}'$ as follows:

$$\mathbf{x}' = \Sigma^{-\frac{1}{2}} \mathbf{x} \tag{52}$$

This normalization can work for any feature for which a covariance matrix can be computed.

Even in the case of single point features, like SIFT or SURF features, the area around the point can be used to find a covariance matrix [21]. However, in the case of the experiments shown later, a technique introduced by Forssén and Lowe [14] will prove to be very useful, and so the normalization used in that work is described next.

In this scenario, a set of small images called patches is produced. Every MSER yields at least one shape and one texture patch, and it is these patches that are later given a descriptor to match with. Shape patches are resampled versions of a binary mask representing the MSER, while texture patches are resampled versions of the actual image pixels. Shape patches will only be matched to other shape patches, and texture only to texture. The patches will be a fixed size, and this gives the representation scale invariance, where scale invariance refers to the fact that the same feature may appear with different sizes in two different image projections, but will have the same (or very close to the same) descriptor, without accounting for large differences in blur.

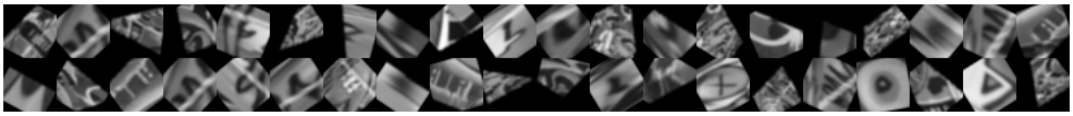
Before the normalization, the input image is blurred with a Gaussian kernel of scale σ_i (to be defined momentarily). The eigenvalue decomposition of the MSER mask's covariance matrix $\Sigma = RDR^T$ ($\det R > 0$) along with the mask's centroid μ are used to define the transformation between an image point in the mask $\mathbf{x}$ and its position in the patch $\mathbf{x}'$:

$$\mathbf{x} = sA\mathbf{x}' + \mu \tag{53}$$

where $A = 2RD^{1/2}$. The parameter s is a scaling factor that brings in a certain amount of extra area surrounding an MSER to help, for instance, make a patch more distinguishable. Bilinear interpolation is used to get the equivalent image points from the patch points $\mathbf{x}' \in [-1, 1]^2$. The resulting patch is then blurred again with another Gaussian kernel σ_p. The size of the patch is $N_s \times N_s$. The value of σ_i is determined as $\sigma_i = bs/N_s$, where b is the length of the minor axis of the bounding ellipse, and s and N_s are as given above. The authors' suggested default patch size is $N_s = 41$, and this is the size used in all experiments and demonstrations found in this chapter.

Another interesting suggestion in this framework is to not only make use of the image texture itself in the patches, but also the shape of the MSER regions. The same process is followed for both types of patches, except that instead of sampling points for the patch from the original image, the binary mask is used in its place. The shape patches can be advantageous especially for situations of varying light, or anything else that might alter the image texture of otherwise matching regions. Some parameters are tuned to work well with either shape or texture patches. For instance, the Forssén and Lowe found that $s = 1.2$ and $\sigma_p = 1.2$ work well for shape, while $s = 2.5$ and $\sigma_p = 1.0$ work well for texture, based on detailed experimentation. Figure 34 shows forty shape and texture

(a) Shape patches.



(b) Texture patches.

Figure 34: Forty shape and texture patches from the image in Figure 31.

patches for the MSERs found in the image of Figure 31. Only one type of patch may be chosen for some applications, or both may be used together, as they are in the experiments shown here.

To describe the patches once they are created, Forssén and Lowe [14] propose using a slightly customized SIFT descriptor [25]. A reference orientation is found at the maximum of a histogram of gradients in the patch, and the patch is rotated accordingly. Multiple orientations might be used if the histogram has local maxima that are at least 80% as large as the overall maximum. The rotated patch or patches are divided into $4 \times 4 = 16$ squares, for which individual histograms of gradients are computed. These histograms have eight bins each, so when put all together, they form a descriptor of size $4 \times 4 \times 8 = 128$. The descriptor is normalized to unit length. The SIFT descriptor has been chosen as the best descriptor for use with MSERs [31].

5.2 Performance in Panoramas and Photographs

5.2.1 Detecting MSERs

The nature of typical outdoor panoramic images used in this study suggests that MSERs could be a good detector for matching panoramas and photographs. For instance, the most useful outdoor panoramas are likely to contain buildings. Furthermore, it may often be the case that these buildings provide the only reliable commonality between images that may have been taken in different seasons and lighting conditions, with varying crowds in the scene, and with small-scale construction changes. This means that not only are they available for matching, but possibly required.

Buildings generally contain planar surfaces likely to be photographed from varying viewing angles. Buildings also contain such recognizable features as windows, which are generally homogeneous regions with distinctive boundaries (aside from reflections on the glass). While MSERs have proved

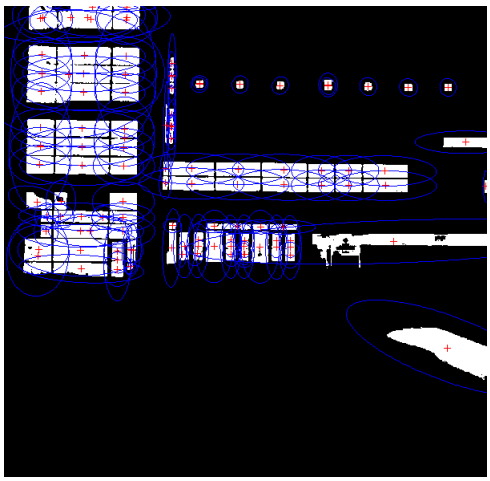
to be reliable for all these scenarios, they have been shown, unfortunately, to be susceptible to changes in blur between images [27]. This could certainly pose a problem for comparisons between panoramas and photographs since the former are often more blurry than the latter due to the fusion of multiple images into one panorama. Detecting MSERs in a scale space and eliminating duplicates does seem to improve invariance to changes in scale and blur [14], but won't be explored here. Problems arising from the repetitiveness of features commonly found on buildings will be addressed in the next chapter.

As seen in the previous section, MSER detection results in a set of regions for an image, and for each region, small square patches representing the shape and image texture in normalized form. While the MSER region and the shape and texture patches can be used to find initial matches of MSERs between two images, a specific point is needed to determine the epipolar geometry. For this, the centre of gravity may be used, since it will always be the same for a particular shape of region. Therefore, when evaluating the performance of the MSER regions for use with panoramas and photographs, two criteria must be used: first, are the same image regions detected as MSERs in both images, and second, is the centre of gravity in the same location?

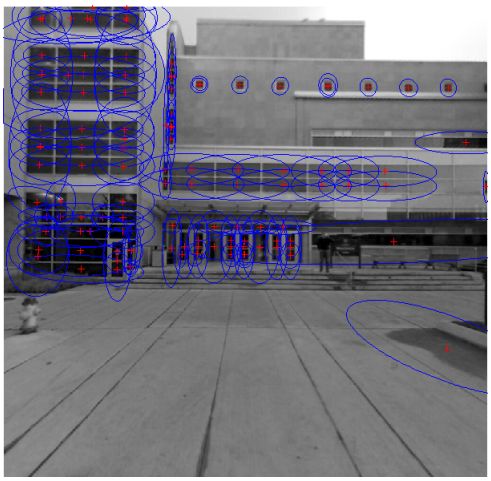
This section demonstrates MSER detection for panoramas and images. Because the actual MSER matching process will be discussed later, where more detailed analysis will be possible, it is sufficient to simply show several examples of detected MSERs which can be visually judged for quality. These examples show only MSERs detected as darker regions, though the inverse images would then be used to find even more regions. Some matches may occur between regions detected in both scenarios, especially if the lighting changes.

It is important to note that a certain amount of preprocessing may be required before detecting MSERs in some images. For example, panoramas generated from Ladybug camera data may not have enough contrast, so a simple adjustment of tone levels can improve MSER detection greatly. Furthermore, the current algorithms are fairly sensitive to reflections in the windows. Certain windows act almost like mirrors, and the reflections seen in them can influence the shape of MSERs detected. In some of the experiments here, images have been manually retouched to smooth out or remove reflections. Handling this more automatically might be the topic of future work.

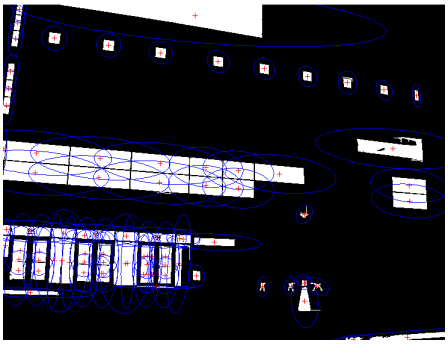
The first detection example is shown in Figure 35. Only one relevant face of the panorama is used (full panoramas are shown in Appendix C), and only the MSERs found on the original image (where darker regions are detected), as opposed to MSERs found on the inverted image (where lighter regions in the original image are detected), are displayed in all cases but one, where the opposite



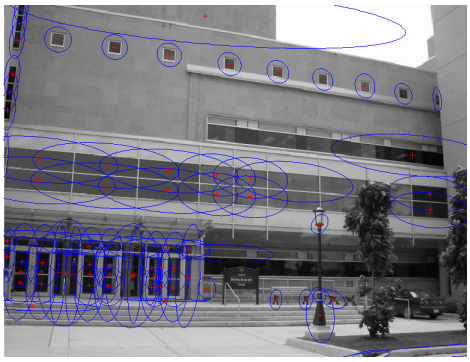
(a) MSERs on merged binary masks, panorama (right face).



(b) MSERs on original image, panorama (right face).

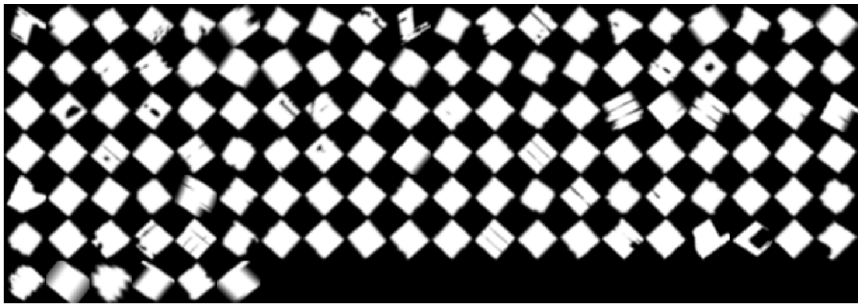


(c) MSERs on merged binary masks, photograph.

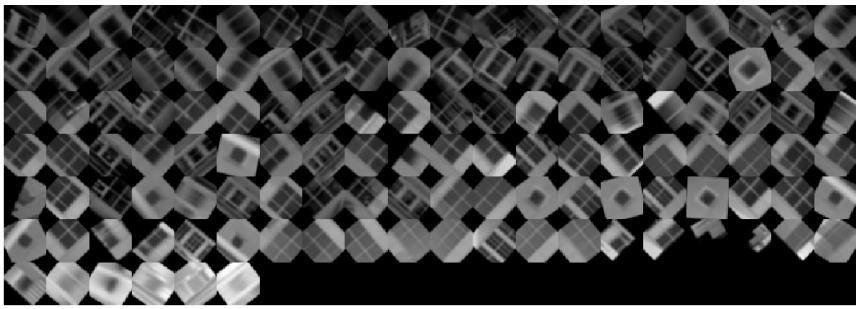


(d) MSERs on original image, photograph.

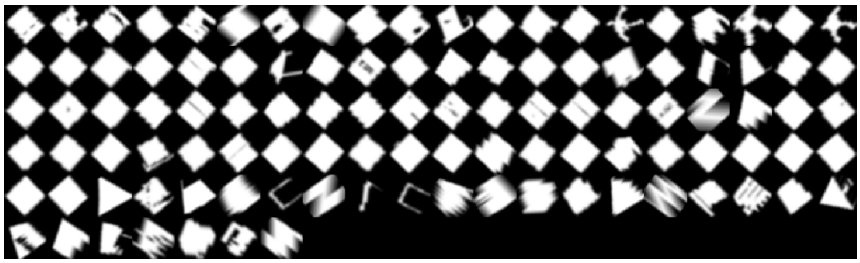
Figure 35: Detailed example of MSER detection between a panorama and photograph.



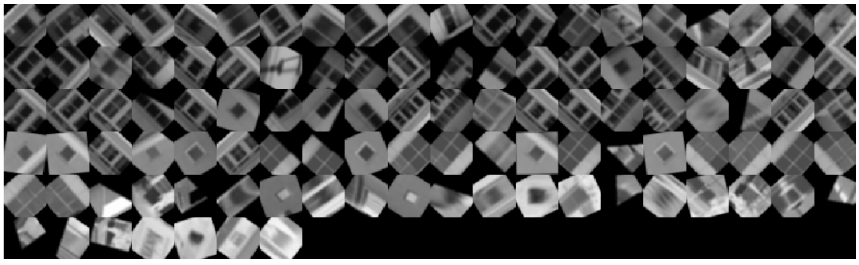
(a) Shape patches for MSERs from Figure 35a.



(b) Texture patches for MSERs from Figure 35b.

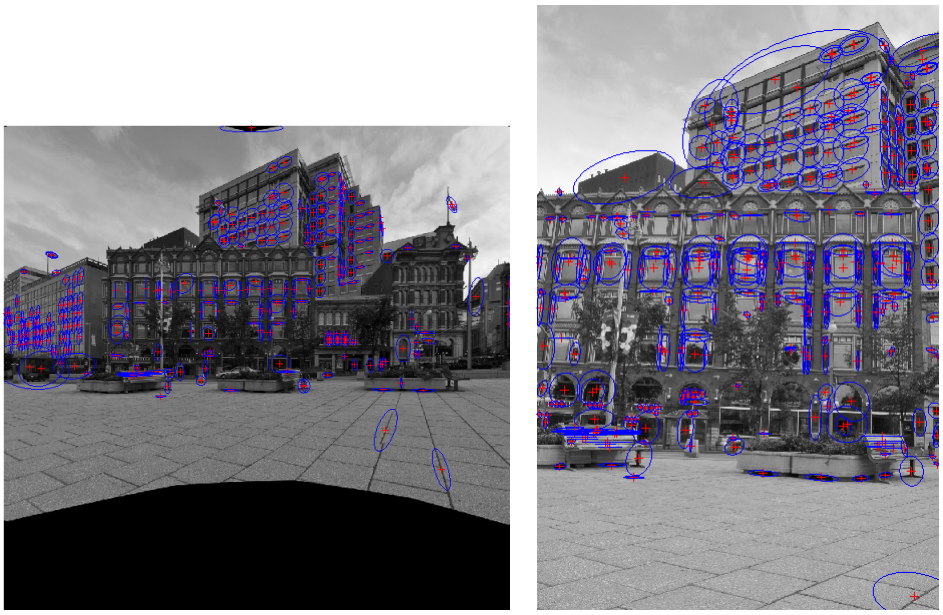


(c) Shape patches for MSERs from Figure 35c.



(d) Texture patches for MSERs from Figure 35d.

Figure 36: Shape and texture patches for the images of Figure 35.



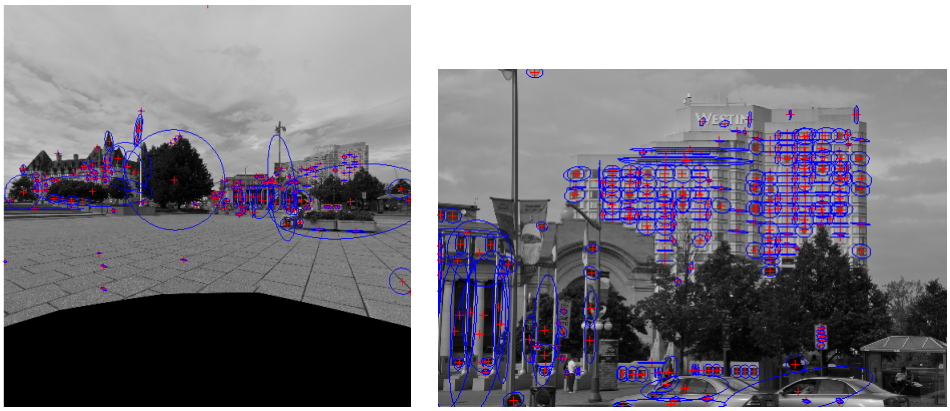
(a) Panorama (front face) (b) Photo

Figure 37: MSER detection for the Elgin Street pair.

is shown. Notice that the centroids are always located in the middle of the windows, and that the bounding ellipses contain the same area in both images. The shape and texture patches for these MSERs are shown in Figure 36 (without extra patches created for rotation invariance). Though there is repetition, the patches that should match do look the same up to rotation. Although it is immediately obvious that there are fewer MSER features than there would be SURF points, the MSER centroids are usually located in areas of the image that SURF points are not found, making them a complementary feature.

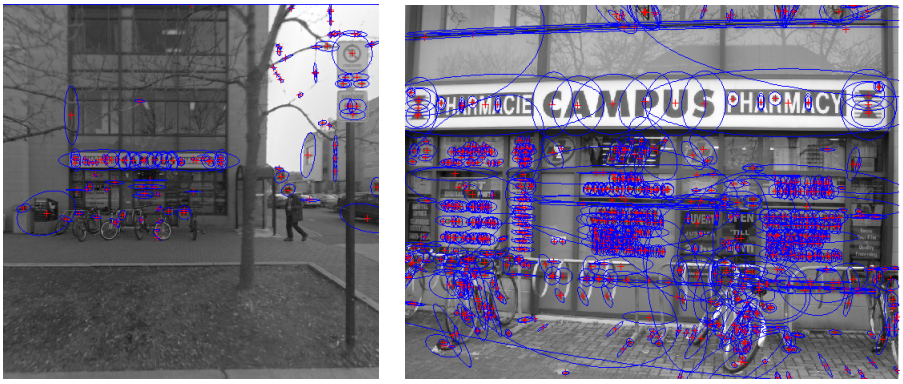
Figure 37 shows buildings in the downtown Ottawa area, taken from the walkway of the National War Memorial on Elgin Street. Because the panorama was taken with a standard camera and lens, it does not suffer as much from the hazing and blurring effects seen in the others. The images used to build the panorama were not touched up at all. The building itself in this example has more interesting features, including a larger quantity of windows and slightly more variety in their shape. More MSERs were detected, and the different shapes of the window regions imply that one might expect to have an easier time matching them.

While there aren't as many MSERs detected in the Westin hotel image pair is shown in Figure 38, there are certainly enough to make several good matches with. The Westin sign, like the pharmacy



(a) Panorama (back face) (b) Photo

Figure 38: MSER detection for the Westin Hotel pair.



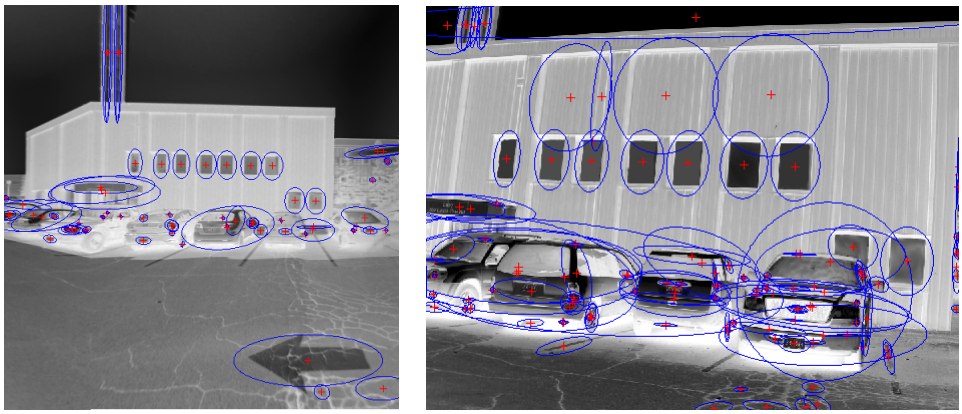
(a) Panorama (right face) (b) Photo

Figure 39: MSER detection for the pharmacy pair.

sign, has strong shapes that should match well. The regions are not shown here, since the letters in the sign are lighter than the surrounding wall.

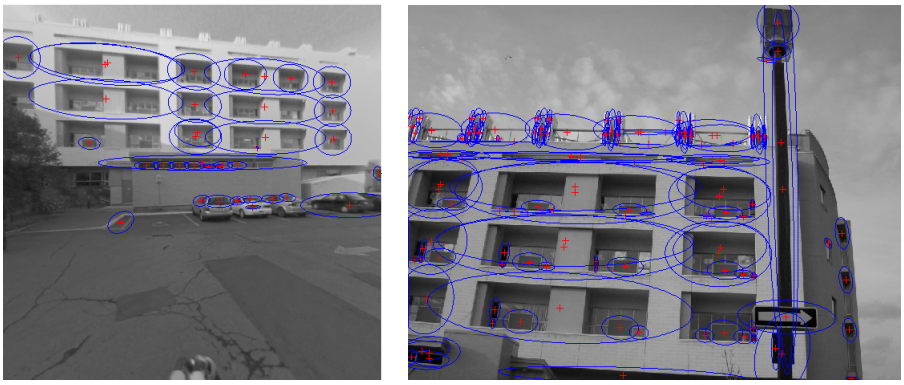
The pharmacy pair is shown in Figure 39. Many of the features detected are situated around the main pharmacy sign, similar to the distribution of SURF points seen in Figure 25. Signs tend to have strong shapes that are usually detected as MSERs.

The Cube pair in Figure 40 is shown in its reversed state, since most MSERs were detected that way. This building is fairly non-descript and contains few features. The windows, the awning above the door, and the pipes on the roof are the only matchable features, and the windows are the most reliable of these. The pipes would be reasonable matches, but different amounts appear in the two images. The MSER shape is altered because of this, and the centroids don't match. If the vehicles



(a) Panorama (right face) (b) Photo

Figure 40: MSER detection for the Cube pair.



(a) Panorama (left face) (b) Photo

Figure 41: MSER detection for the Genomics pair.

outside were the same (for instance, if the images were captured together), they would provide more good features, but panoramas are typically taken at very different times from the photographs.

Finally, the Genomics building in Figure 41 is an example of a panorama being much blurrier than the photograph. Even without implementing a scale-space MSER detection scheme, there are several MSERs that appear to match. Since this can only improve with scale-space detection, it suggests that MSERs might be a good choice for image pairs with differing image qualities and blurs.

In conclusion, while fewer features are detected as compared to SURF points, MSER locations can still be repeatable between a panorama generated with Ladybug data and a standard photograph. MSERs perform better with higher quality panoramas created from photographs, as did SURF

points. How well they perform depends on how close the main subject of the images are in terms of blur (until multi-scale detection is used), the effect of reflections and of occlusions from vegetation or people, and how many useful features there are away from the edges of the image or cube face.

5.2.2 Matching MSERs

The previous sections outlined how to detect MSERs, transform them into small square patches containing image texture of a binary mask representing the shape of the MSER, and how to describe them with modified SIFT descriptors. These descriptors can now be matched using standard matching techniques, such as thresholding the descriptors or finding their nearest neighbours (see Section 2.1.3).

In this section, both basic thresholding and the slightly more complicated nearest neighbour ratio thresholding are demonstrated for several different image pairs. The results are summarized in Table 3. As in the previous chapter, matches are evaluated manually in that each match is examined visually, and the number of correct matches tallied.

The graffiti photograph pair gives good matching results, yielding both a high number of correct matches and a good percentage of correct matches. This happens because the photographs do not change greatly in terms of viewing angle, focal length, and other factors. In addition, there are many clear and distinct shapes that should match to each other well.

The panoramas generated from Ladybug data performed rather poorly. In the case of the biology building and the pharmacy, it appears that matches are possible based on the nearest neighbour results. The restrictive threshold of 0.05, however, does not allow for many patches to match. Higher thresholds, not shown, generally allow for more correct matches, but also far more incorrect matches. Results from Tables 1 and 2 were much more encouraging, and this may suggest that a more distinctive MSER descriptor could make matching these images more successful.

On the other hand, there are three panorama image pairs that were completely unsuccessful: the side of the pharmacy, the Cube building, and the Genomics building. Some of these had some good matches with SURF points, but none had especially high numbers or percentages of matches. It doesn't seem likely that a more descriptive MSER would improve these results, since each pair exhibits a particular challenge other than repetitive features. For instance, the side of the pharmacy and the Genomics building have a different viewing angle between the images, and the panoramas have some quality issues. The Cube building has few distinct features. These examples are mainly included as an illustration of the challenges faced when matching panoramas made with Ladybug

Image Pair	**Threshold 0.05 Number Correct**	**Threshold 0.05 Percent Correct**	**NN Ratio 0.8 Number Correct**	**NN Ratio 0.8 Percent Correct**
Graffiti	132	80%	246	60%
Elgin Street	- (>7000 potential matches)	-	181	18.2%
Westin	- (>1682 potential matches)	-	2	1.3%
Biology	9	3.1%	21	16.7%
Pharmacy	4	23.5%	44	7.2%
Side of Pharmacy	1	2.7%	6	5.3%
Cube	0 (out of 1)	0%	5	6.1%
Genomics	0 (out of 10)	0%	0 (out of 83)	0%

Table 3: MSER matching results for basic thresholding and nearest neighbour (NN) ratio thresholding.

data, as they are not a rare exception.

The Elgin Street and Westin pair both use a panorama generated with a standard consumer camera. Even with a threshold as restrictive at 0.05, there were too many potential matches to verify. The nearest-neighbour ratio thresholding produced better results for the Elgin Street pair than the Westin, but the Westin appears to have more repetitive features. A more distinctive MSER descriptor would likely benefit images like these.

5.3 Conclusion

This chapter provided detail about detecting and matching Maximally Stable Extremal Regions, or MSERs. MSER detection worked reliably for most image pairs, even without scale-space detection. The way they are described is affine-covariant, allowing for more drastic changes in the viewing angles between images. All this makes MSER a good choice of feature for matching panoramas to photographs, but the repetitive nature of the scenes depicted causes problems. Enough good matches are found for some pairs, but far too many incorrect matches are also found. The next chapter addresses this issue and suggests ways to make MSER features more distinct.

Chapter 6

Matching Repetitive Features

Chapter 4 established that matching with Speeded Up Robust Features works well with panoramas created using standard photographs, and reasonably well for some panoramas generated with Ladybug camera data. Chapter 5 showed that Maximally Stable Extremal Regions could be detected reasonably well in both types of panoramas, but the repetitive features found in typical scenes with buildings and other man-made structures made accurate matching difficult. This issue is addressed in this chapter, where a global context descriptor is employed.

The use of global descriptors has been previously proposed [32, 21] for use with SIFT [25] descriptors. The main motivation was to give local descriptors some kind of context about their place in an image to help make repetitive features more distinguishable.

In the first section of this chapter, the method for building global descriptors for use with SIFT features is described. This global descriptor is adapted for use with SURF and MSER features. Experimental results are promising. A final descriptor combining MSER features and a global context descriptor built from SURF points is also explored. Finally, the chapter concludes with a comparison of all matching methods from this chapter and the previous two.

6.1 Global Descriptors with SIFT Features

Previous work has established two similar frameworks for augmenting a SIFT descriptor with a global context vector. The first work by Mortensen et al [32] uses the whole image to gather context from all surrounding pixels, while the second by Li and Ma[21] uses a local measurement area for each feature and gathers context only from other feature points within this area. Since the second method builds on and improves the first, and inspires the use of global context with SURFs and MSERs, it is described next.

An important contribution by Li and Ma [21], in addition to improvements to the global context vector, is the adjustment of the measurement region for SIFT descriptors. Normally, SIFT descriptors are computed in a circular neighbourhood around a detected point, where the size of this neighbourhood is determined relative to the scale at which the feature point was detected. When a neighbourhood around a feature point undergoes an affine transformation between two views, however, this enclosing circle may have different contents in the two images, introducing the possibility of incorrect matching of SIFT descriptors. Thus, an elliptical neighbouring region based on the second moment matrix of the intensity gradient is proposed instead, as this has been used to create affine covariant features in previous work [29]. This elliptical affine region is estimated using an iterative procedure, and then normalized before the dominant orientation is determined and before the gradient histogram is computed for the SIFT descriptor.

The same local measurement region that is used to compute the SIFT descriptor is used for the global context descriptor. Doing so balances the need for distinctiveness while maintaining robustness in terms of image transformations, such as scale. This improves on earlier work [32], which used the entire image as the measurement region.

Once the measurement region for feature $\tilde{\mathbf{x}} = (\tilde{x}, \tilde{y})^T$ is normalized to become circular, the principal curvature is computed for each other feature point $\mathbf{x} = (x, y)^T$ whose normalized location is contained within the measurement region. A 2×2 Hessian matrix $H(x, y)$ is computed for $\mathbf{x}$ as

$$H(x,y) = \begin{pmatrix} L_{xx} & L_{xy} \\ L_{xy} & L_{yy} \end{pmatrix} = I(x,y) * \begin{pmatrix} g_{xx} & g_{xy} \\ g_{xy} & g_{yy} \end{pmatrix} \tag{54}$$

with second partials of the image in x and y as L_{xx} and L_{yy}, and L_{xy} the second cross-partial. The Gaussian second derivatives g_{xx}, g_{yy}, and g_{xy} are used to compute the image's second derivative by convolution. The principal curvature $c(x, y)$ is taken as the maximum absolute eigenvalue of the Hessian matrix.

A log-polar histogram is now built for the feature $\tilde{\mathbf{x}}$, seen in Figure 42. The histogram has 5 radial bins, and 12 angular bins for each radial bin. The curvature values of those other SIFT features $\mathbf{x}$ whose normalized locations are within the bounds of the measure region of feature $\tilde{\mathbf{x}}$ are collected and weighted by an inverse Gaussian $w(x, y)$

$$w(x,y) = 1 - e^{-((x-\tilde{x})^2 + (y-\tilde{y})^2)/(2\sigma^2)} \tag{55}$$

where σ is the local scale $\tilde{\mathbf{x}}$ was detected on. This weighted curvature value is added to the appropriate bin in the histogram. The weighting places more importance on the actual SIFT descriptor

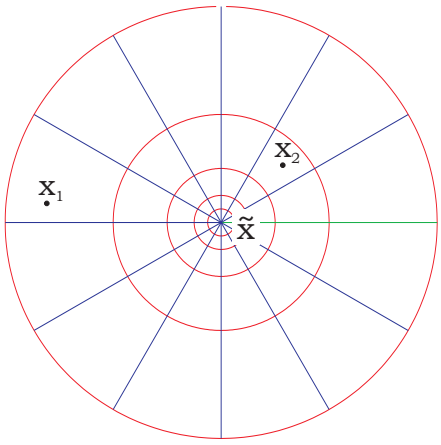


Figure 42: Log-polar graph created for feature point $\tilde{\mathbf{x}}$. Weighted curvature values for surrounding features $\mathbf{x}_1$ and $\mathbf{x}_2$, whose normalized locations fall within the bounds of the measurement region of $\tilde{\mathbf{x}}$, will be added to the appropriate bins.

closer to the centre of the measurement region, with a smooth transition from SIFT to the global descriptor. Once the curvatures of all nearby features have been added to the histogram, the 60 regions become the global descriptor, and can be matched using any of the usual techniques (basic thresholding and nearest-neighbour ratio thresholding) using the χ^2 metric.

The affine co-variant SIFT framework with global descriptors is shown by its authors to be invariant to scale changes, translation, rotation, affine transformations, and image contrast.

6.2 Global Descriptors with SURF Features

The SURF matching results in Chapter 4 were quite good, but could be improved with the added distinctiveness of a global descriptor. This section provides a simple algorithm for finding a global descriptor in a way similar to that used for SIFT points above. Algorithms 6.3 and 6.4 summarize matching with global context. Experimental results follow.

6.2.1 An Algorithm for Computing Global Descriptors with SURFs

Essentially the same algorithm that was used for computing the global context descriptor for SIFT points may be used with SURF. The only difference in the implementation suggested here is that the SURF points will not have affine covariant measurement regions. While this can occasionally be a disadvantage for certain image pairs with large differences in viewing angle, it is a simple addition

Algorithm 6.3 Threshold matching with global context.

1. Detect and describe interest points in both images, as in Algorithm 4.1 for SURFs or Algorithm 5.2 for MSERs.
2. Compute a global context descriptor for all feature points in both images using the desired method.
3. For each feature point p in the first image:
   (a) For each feature point q in the second image:
      i. Find the Euclidean norm between the feature descriptors for p and q.
      ii. Find the χ^2 norm between the global context descriptors for p and q.
      iii. If the feature descriptor norm is below one particular threshold $t_{feature}$ and the global descriptor norm is below another threshold t_{global}, then mark the pair p, q as a match.

Algorithm 6.4 Nearest neighbour ratio matching with global context.

1. Detect and describe the features in both images, as in Algorithm 4.1 for SURFs or Algorithm 5.2 for MSERs.
2. Compute a global context descriptor for all feature points in both images using the desired method.
3. For each feature point p in the first image:
   (a) Find the two nearest neighbours from the second image in terms of the norm between the feature descriptors.
   (b) If the ratio between the norm of the first nearest neighbour q and the second nearest neighbour r is less than a threshold t_{NN}:
      i. Compute the χ^2 norm between the global descriptors for p and q. If this norm is less than a second threshold t_{global}, then p and q are marked as a match.

to the existing SURF detection and description algorithm and requires fewer steps. As will be seen in the next section, experimental results are satisfactory.

The computation of the SURF global context descriptor is outlined in Algorithm 6.5. The main idea is to build the global context descriptor using other SURF points that lie within a radius of K times the scale at which the SURF point in question was detected. The curvature is computed once for the entire image and queried as needed by individual SURF points that are included in the global descriptor computations.

In more detail, the first step is to compute the curvature of the entire image using the Hessian matrix (54). Then, for each SURF point $\tilde{\mathbf{x}} = (\tilde{x}, \tilde{y})$ a measurement region is defined as a circle with radius K times the scale at which the point was detected. All SURF points $\mathbf{x}$ that lie within the circular measurement region are found. Each of these nearby SURF points $\mathbf{x} = (x, y)$ in the measurement region are placed in the appropriate angular bin φ and radial bin ρ. There are 12 angular and 5 radial bins, as in the previous method. The bin indices are computed as

$$\varphi = \left\lfloor \frac{6}{\pi} \left(\arctan\left(\frac{x - \tilde{x}}{y - \tilde{y}} \right) - \alpha \right) \right\rfloor \tag{56}$$

and

$$\rho = \max\left(1, \log_2\left(\frac{r}{r_{max}}\right) + 6\right) \tag{57}$$

where α is the angle at which the SURF point $\tilde{\mathbf{x}}$ was detected and r_{max} is the radius of the measurement region.

The curvature value at $\mathbf{x}$ is weighted with the inverse Gaussian of equation (55) and added to the appropriate bins computed as above. When all pixels in range have been added, the bins are flattened into a single vector with dimension 60 and the vector is normalized to have unit length one. This is the global context vector, which can be matched with any preferred method, such as thresholding or nearest-neighbour ratio thresholding. In the experiments, the global context vector is compared using the χ^2 metric, just as the SIFT descriptors are.

6.2.2 Experimental Results

In this section, the performance of the SURF global descriptor is demonstrated for a variety of image pairs. The experiments are first run on a pair of photographs that are expected to be easily matchable, followed by trials with the panorama-photograph pairs seen in the previous chapters. Matching is done using the methods in Algorithms 6.3 and 6.4. As usual, the correct number of matches is manually determined in all cases by visual inspection.

Algorithm 6.5 Compute the global context vector for a single SURF feature located at $\tilde{\mathbf{x}}$.

1. Compute the curvature of the entire image by finding the maximum absolute eigenvalue of the Hessian matrix for each pixel.
2. Compute the circular measurement region with radius R as K times the scale of the SURF feature.
3. For each other SURF feature whose location $\mathbf{x}$ is within R pixels of the SURF feature point location $\tilde{\mathbf{x}}$:
   (a) Compute radial and angular bin for SURF location $\mathbf{x}$.
   (b) Weigh curvature value found in the curvature image at $\mathbf{x}$ with an inverse Gaussian and chosen σ.
   (c) Add weighted curvature value to computed bin.
4. Create vector by flattening radial and angular bins.

There are different ranges of thresholds used in the experiments for various images in addition to trials with the nearest neighbour ratio matching method. Higher quality images, such as the Graffiti photograph pair, can have tighter thresholds than other image pairs, since the features tend to be strong, clear, and similar to those in the other image. Four thresholds for basic thresholding were chosen based on how well they exhibited the matching behaviour of the image pair for one of two values of K, and one threshold used with nearest-neighbour ratio threshold matching is shown.

For each set of results for a particular choice of K, a star is placed in the last column for the row that might be considered to contain the best results. The choice is made based on finding the best balance between a high percentage and high number of correct matches; in many cases, more than one choice would be reasonable. Higher percentages tend to be favoured. The chosen best results are summarized at the end of the section.

The first results are shown in Table 4, where the simple Graffiti photograph pair seen in Figure 54 on page 125 was used to demonstrate the abilities of the SURF global context descriptor for an easy image to match. It is immediately obvious that the larger measurement area for the global context, obtained from a multiple $K = 10$ of the SURF's detected scale, performs better. Both the number and percentage of matches is higher. Using the larger multiple factor K could cause problems in some images, particularly those with occlusions, but works well here. For both values

SURF Thresh	Global Thresh	Number Correct	Percent Correct	Best
0.15	1.0	32	88.9%	
0.15	1.3	35	83.3%	
0.15	1.6	39	84.8%	
0.3	1.0	163	39.7%	
NN Ratio 0.8	1.0	94	83.2%	*

(a) Matching results for $K = 5$.

SURF Thresh	Global Thresh	Number Correct	Percent Correct	Best
0.15	1.0	70	100%	
0.15	1.3	102	98.1%	
0.15	1.6	123	96.1%	
0.3	1.0	358	91.2%	
NN Ratio 0.8	1.0	350	98.6%	*

(b) Matching results for $K = 10$.

Table 4: Matching results for SURF features with global context for Graffiti photograph pair.

of K, the nearest neighbour ratio matching method gave better results, as it did when using SURF without the global context descriptor. The global context simply helped remove incorrect matches.

For the Elgin Street example in Table 5, the better results are again obtained with the larger value of K. There are fewer matches overall than for the Graffiti example, but the two images do not contain the same proportion of the scene in this case. While the nearest neighbour ratio matching results are again chosen as best, and have a very reasonable percentage of correct matches (over 80% for both values of K), a higher percentage may be favoured over the higher number of matches for some applications, and it is possible to obtain it for this image pair.

The difference in scale of the Westin Hotel building makes the numbers of correct matches in Table 6 somewhat lower than the previous two examples. However, both the number and percentage of these matches are still more than enough to use in a RANSAC process of outlier elimination. This suggests that matching with the global context descriptors is scale invariant as the SIFT global descriptor is.

The Biology Image pair is the first example that uses a partial panorama made from Ladybug data, causing a difference in quality between the two images. No results in Table 7 have particularly high numbers of matches, but some percentages may be enough to find the epipolar geometry. Combining these results with another match set would probably help.

The Pharmacy pair is another example of Ladybug data being matched with photographs. Table

SURF Thresh	Global Thresh	Number Correct	Percent Correct	Best
0.15	1.0	26	96.3%	
0.15	1.3	37	92.5%	
0.15	1.6	39	88.6%	
0.3	1.0	105	42.0%	
NN Ratio 0.8	1.0	64	85.3%	*

(a) Matching results for $K = 5$.

SURF Thresh	Global Thresh	Number Correct	Percent Correct	Best
0.15	1.0	32	88.9%	
0.15	1.3	35	83.3%	
0.15	1.6	39	84.8%	
0.3	1.0	163	39.7%	
NN Ratio 0.8	1.0	94	83.2%	*

(b) Matching results for $K = 10$.

Table 5: Matching results for SURF features with global context for Elgin Street image pair.

SURF Thresh	Global Thresh	Number Correct	Percent Correct	Best
0.15	1.6	0 (out of 0)	-	
0.3	1.0	11	47.8%	
0.3	1.3	35	45.5%	
0.3	1.6	47	34.1%	
NN Ratio 0.8	1.3	26	59.1%	*

(a) Matching results for $K = 5$.

SURF Thresh	Global Thresh	Number Correct	Percent Correct	Best
0.15	1.6	0 (out of 0)	-	
0.3	1.0	30	63.8%	
0.3	1.3	39	39.0%	
0.3	1.6	45	32.1%	
NN Ratio 0.8	1.3	24	80.0%	*

(b) Matching results for $K = 10$.

Table 6: Matching results for SURF features with global context for the Westin Hotel image pair.

SURF Thresh	Global Thresh	Number Correct	Percent Correct	Best
0.15	1.3	5	62.5%	
0.15	1.6	8	66.7%	
0.2	1.0	12	66.7%	
0.3	1.0	25	32.9%	
NN Ratio 0.8	1.0	16	61.5%	*

(a) Matching results for $K = 5$.

SURF Thresh	Global Thresh	Number Correct	Percent Correct	Best
0.15	1.3	7	70.0%	
0.15	1.6	7	53.9%	
0.2	1.0	13	65.0%	
0.3	1.0	25	25.3%	
NN Ratio 0.8	1.0	15	60.0%	*

(b) Matching results for $K = 10$.

Table 7: Matching results for SURF features with global context for the Biology image pair.

SURF Thresh	Global Thresh	Number Correct	Percent Correct	Best
0.15	1.3	5	100%	
0.15	1.6	8	88.9%	
0.25	1.3	14	41.2%	
0.3	1.0	2	9.5%	
NN Ratio 0.8	1.0	16	53.3%	*

(a) Matching results for $K = 5$.

SURF Thresh	Global Thresh	Number Correct	Percent Correct	Best
0.15	1.3	12	100%	
0.15	1.6	15	93.8%	*
0.25	1.3	25	50.0%	
0.3	1.0	11	33.3%	
NN Ratio 0.8	1.0	9	90.0%	

(b) Matching results for $K = 10$.

Table 8: Matching results for SURF features with global context for the Pharmacy image pair.

SURF Thresh	Global Thresh	Number Correct	Percent Correct	Best
0.25	1.3	3	12.5%	
0.3	1.0	3	10.7%	
0.3	1.3	5	7.9%	
0.3	1.6	5	4.4%	
NN Ratio 0.8	1.0	5	25.0%	*

(a) Matching results for $K = 5$.

SURF Thresh	Global Thresh	Number Correct	Percent Correct	Best
0.25	1.3	3	7.9%	
0.3	1.0	4	10.8%	
0.3	1.3	6	8.0%	
0.3	1.6	6	5.7%	
NN Ratio 0.8	1.0	7	24.1%	*

(b) Matching results for $K = 10$.

Table 9: Matching results for SURF features with global context for the Side of Pharmacy image pair.

8 shows that the results with this pair aren't much better than those for the Biology pair. The letters on the sign provide clear features that should be more easily matchable, but entire words are repeated, making the global context less useful in distinguishing the surroundings of individual features. The matches on the letters are still some of the most reliable in the image, since not only is there a difference between the image quality and blur, but there is also a large difference in scale. Despite all this, 16 matches may again be enough to find the epipolar geometry, especially when combined with another set of matches.

The Side of Pharmacy is a much more difficult image to match than any of the others so far. The scale difference is not as large, but the viewing angle is quite different, and the partial panorama is quite blurry. Many of the matches from the results in Table 9 were actually quite reasonable given the circumstances, but not very many happen to be correct.

The Cube pair is quite similar to the Biology pair in terms of the image quality and scale difference. The Cube building, however, has fewer discernible features than the Biology building, and the cars in front are completely different in the panorama and photograph. The results in Table 10 may be sufficiently good for certain applications, such as object/location recognition.

The Genomics pair, like the Side of Pharmacy pair, suffers from low image quality. The partial panorama exhibits a hazing effect and is blurry. Very few matches were found to be correct in Table

SURF Thresh	Global Thresh	Number Correct	Percent Correct	Best
0.25	1.3	5	71.4%	
0.3	1.0	8	80.0%	
0.3	1.3	12	54.6%	
0.3	1.6	17	40.5%	*
NN Ratio 0.8	1.6	13	54.2%	

(a) Matching results for $K = 5$.

SURF Thresh	Global Thresh	Number Correct	Percent Correct	Best
0.25	1.3	6	40.0%	
0.3	1.0	9	45.0%	
0.3	1.3	14	50.0%	
0.3	1.6	18	45.0%	*
NN Ratio 0.8	1.6	17	36.2%	

(b) Matching results for $K = 10$.

Table 10: Matching results for SURF features with global context for the Cube image pair.

SURF Thresh	Global Thresh	Number Correct	Percent Correct	Best
0.25	1.3	0 (out of 23)	0%	
0.3	1.0	1	6.7%	
0.3	1.3	2	4.6%	
0.3	1.6	4	5.6%	*
NN Ratio 0.8	1.6	0 (out of 16)	0%	

(a) Matching results for $K = 5$.

SURF Thresh	Global Thresh	Number Correct	Percent Correct	Best
0.25	1.3	0 (out of 8)	0%	
0.3	1.0	3	25.0%	
0.3	1.3	4	13.8%	
0.3	1.6	4	69.0%	*
NN Ratio 0.8	1.6	1	4.2%	

(b) Matching results for $K = 10$.

Table 11: Matching results for SURF features with global context for the Genomics image pair.

Image Pair	K=5, Number Correct	K=5, Percent Correct	K=10, Number Correct	K=10, Percent Correct
Graffiti	94	83.2%	350	98.6%
Elgin Street	64	85.3%	94	83.2%
Westin	26	59.1%	24	80.0%
Biology	16	61.5%	15	60.0%
Pharmacy	16	53.3%	15	93.8%
Side of Pharmacy	5	25.0%	7	24.1%
Cube	18	45.0%	17	40.5%
Genomics	4	5.6%	4	69.0%

Table 12: Summary of the best matching results for SURF with global context for $K = 5, 10$ (marked with * in previous tables).

11 because the areas of the panorama that should match with the photograph are exactly those areas with the most hazing and blur. These results are not sufficient to find the epipolar geometry between the Genomics images.

As mentioned in the introduction of this section, one result for each value of K for each image pair was selected as the best result. These choices are marked with an asterisk, and summarized in Table 12. These results show that matching with SURF points augmented with a global context descriptor computed from the curvature at nearby SURF points usually produces a sufficient number and percentage of correct matches. In some cases, the number of correct matches is lower than 20, and it would be desirable to combine the matches found here in these cases with matches produced from some other technique. The next section will look at creating a global context descriptor for MSER points, and if good results are obtained, the matches from this method may be combined with the results of this section.

6.3 Global Descriptors with MSER Features

A global context framework was also developed for use with Maximally Stable Extremal Regions to improve matching abilities in image pairs with repetitive features. The general idea is the same as finding the global context descriptor for SIFT and SURF feature points in that nearby curvature values are collected in a log-polar histogram for each MSER feature. The two main questions are what the measurement region should be, and how to best collect these curvature values. These are addressed next, after which the algorithm is described in detail. Algorithms 6.3 and 6.4 summarize matching with global context.

6.3.1 Defining the Measurement Region and Collecting Curvature Values

The measurement region for the global context descriptor for SIFT features is related to the scale at which the SIFT point was detected and an elliptical affine region around the feature point. The natural application of this to MSERs, then, would be to use the bounding ellipse of the MSER shape itself as the measurement region.

This idea does not work here because of the way MSERs are detected. Recall from Chapter 5 that MSERs are areas with relatively stable intensity values. This means that there often won't be any significant curvature values within the MSER region, and that the bounding ellipse may not contain many curvature values beyond the actual boundaries of the shape. In other words, whatever curvature is available within the bounding ellipse often does not provide much context in terms of surrounding features.

A potentially good measurement region, then, may not be the bounding ellipse of the feature itself, but perhaps some multiple K of it. This way, a certain amount of surrounding context will be measured beyond what is available in the SIFT patch descriptors. Choosing K involves balancing between having a more distinct descriptor, and remaining robust to image transformations between two views. Two values of K are evaluated in the experimental results section later in this chapter.

Within the measurement region of the SIFT or SURF approach for global context, curvature values were collected at each other nearby feature point. The initial instinct is to do the same for MSERs: collect curvature values at the centroids of MSERs that fall within a particular measurement region. As discussed above, however, there is often not much or any curvature within an MSER's bounds, where the centroid usually falls. Even where there are non-zero curvature values, it is clear that this one value does not do a good job of representing the feature as a whole. The location of other features might not be represented in the global context when the centroid does not have a

Algorithm 6.6 Compute the global context vector for an MSER feature.

1. Normalize the MSER region (shape or texture, same as the original MSER patch) into a square patch of size $2R$ by $2R$. The area included in the patch is K times the bounding ellipse of the MSER region. The patch will be sampled from the original image for both shape and texture MSER features.

2. Compute curvature of new patch by finding the maximum absolute eigenvalue of the Hessian matrix at each pixel.

3. For each pixel within R pixels of the centre of the patch:

   (a) Compute radial and angular bin for pixel.

   (b) Weigh curvature value found at that pixel with an inverse Gaussian and chosen σ.

   (c) Add weighted curvature value to computed bin.

4. Create vector by flattening radial and angular bins.

positive curvature value, and additional information distinguishing the features is lost.

To incorporate more information about the features found in the measurement region, one might consider using all of the curvature values within the bounds of each nearby MSER (or a slight expansion of the region to ensure boundary pixels are considered). Each of these values would be individually placed in the appropriate log-polar bins. While this would indeed consider the shapes of MSERs in their entirety, there is a disadvantage. If an MSER with significant size was detected in one image but not the other, its many curvature values would increase the difference between the global context vectors significantly.

Instead, the approach from earlier work [32] is borrowed. There, all curvature values within the measurement region (which happened to be the entire image in that case) are incorporated into the global context vector. Instead of using the entire image, only pixels within K times the MSER bounding ellipse would be considered. This approach ensures that the number of nearby MSERs need not agree in the two images.

6.3.2 An Algorithm for Computing Global Descriptors with MSERs

Using the ideas in Section 6.3.1, this section outlines an algorithm for computing a global context vector for a particular MSER feature $\tilde{\mathbf{x}}$. The process is summarized in Algorithm 6.6.

The first step is to normalize the region, transforming the bounding ellipse into a circle. This is accomplished using the same method as shown in Section 5.1.2. The scaling factor s in (53) is now K. Variable N_s can be given any value, but should be larger than the patches created in 5.1.2 since they must capture more surrounding information. The experiments use $N_s = 100$.

The global context descriptor is computed for both shape and texture types of MSER patches. However, in both cases, the pixels from the original image are resampled into the new patch. Thus, the global context descriptor should be the same for a shape and texture patch that are based on the same MSER feature.

Next, the curvature image is found for the patch. The Hessian matrix in equation (54) is computed, and the curvature values set as its maximum absolute eigenvalues. By using the patch version of the MSER to obtain the curvature image, a constant σ may be used. In the experiments, $\sigma = 0.5$.

The log-polar graph introduced in Section 6.1 is used again here. To achieve rotation invariance, the same feature angle α used when computing the SIFT descriptor for MSER patches (Section 5.1.2) will be used to position the first angular bin. In other words, the log-polar graph will be rotated by the same angle.

Now each pixel $\mathbf{x} = (x, y)$ whose distance to the centroid r is within the maximum radius $r_{max} = N_s/2$ is placed in the appropriate angular bin φ and radial bin ρ. There are 12 and 5 bins respectively, as in the previous methods. The centroid of the feature $\tilde{\mathbf{x}}$ is at the centre of the patch as $\tilde{\mathbf{x}}' = (\tilde{x}', \tilde{y}')$. The bin indices are again computed as

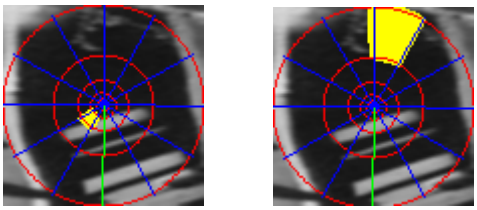
$$\varphi = \left\lfloor \frac{6}{\pi} \left(\arctan\left(\frac{x - \tilde{x}'}{y - \tilde{y}'} \right) - \alpha \right) \right\rfloor \tag{58}$$

and

$$\rho = \max\left(1, \log_2\left(\frac{r}{r_{max}}\right) + 6\right) \tag{59}$$

Figure 43 shows a patch with the log-polar graph superimposed. The green line indicates where the first angular bin begins, based on the feature's angle calculated using image gradient when computing a SIFT descriptor. In this case, the y-axis points down, so positive angles are in the clockwise direction. Yellow pixels are used to roughly highlight which pixels belong to a particular bin.

The curvature value at $\mathbf{x}$ is weighted with the inverse Gaussian of (55) and added to the appropriate bin. As before, the bins are flattened and normalized to have unit length one. The global context vector is again compared using the χ^2 metric.



(a) Angular bin 2, radial bin 3.

(b) Angular bin 7, radial bin 5.

Figure 43: The log-polar graph of a patch shown with a particular bin roughly highlighted.

6.3.3 Experimental Results

The experimental results in this section are presented in the same way as those for SURF (see Section 6.2.2 for more information). While larger values of $K = 5, 10$ were chosen as multipliers of the measurement region for the global context descriptor computed using SURF points, smaller values were explored for the all-pixel curvature-based descriptor used for MSER, since more information is considered within a smaller radius with this approach.

The first demonstration is again for the Graffiti pair of photographs. These images of graffiti on a wall were taken under the same conditions and represent just a small change in viewing angle. There are plenty of distinct shapes detected as MSER regions. Based on all this, these images should match easily. Match results for several thresholds for both the SIFT patch descriptor and the global context descriptor are shown in Table 13. More correct matches were found for the lower values of K because the MSER regions in this image are large and well defined; larger multiples of larger regions would result in less robustness near the image edges. This is the opposite of what happened with SURF's global context, suggesting that using fewer samples of the curvature values in the measurement region would make the method more robust to larger measurement regions.

The panorama in the Elgin Street pair was constructed from images taken with a standard digital single lens reflex camera, and is thus similar in quality to the actual photograph. The buildings are photographed at very similar viewing angles. The biggest difficulty in this image is the repetition. Matching results for $K = 3, 4$ are shown in Table 14. The addition of the global context descriptor is clearly helping distinguish between matches, allowing for a higher percentage of correct matches than was possible with MSERs alone. Although the nearest neighbour approach in Table 3 has a larger number of correct matches and than results seen here, the percentage of correct matches with the global context is better here, and the number of correct matches is more than enough to reliably determine the epipolar geometry.

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.05	0.05	85	100%	
0.10	0.05	135	100%	
0.15	0.05	159	98.8%	
0.15	0.075	181	97.3%	
NN Ratio 0.8	0.075	198	99.5%	*

(a) Match results for $K = 3$.

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.05	0.05	74	100%	
0.10	0.05	120	100%	
0.15	0.05	146	100%	
0.15	0.075	159	99.4%	
NN Ratio 0.8	0.075	176	100%	*

(b) Match results for $K = 4$.

Table 13: Threshold matching results for MSER with global context for Graffiti photograph pair.

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.10	0.05	61	37.4%	
0.10	0.10	84	14.2%	
0.15	0.05	91	34.0%	
0.15	0.075	150	25.4%	
NN Ratio 0.8	0.05	55	82.1%	*

(a) Match results for $K = 3$.

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.10	0.05	66	37.9%	
0.10	0.10	91	20.6%	
0.15	0.05	108	35.9%	
0.15	0.075	152	28.1%	
NN Ratio 0.8	0.05	51	79.7%	*

(b) Match results for $K = 4$.

Table 14: Threshold matching results for MSER with global context for Elgin Street image pair.

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.10	0.05	0 (out of 0)	-	
0.10	0.10	1	10.0%	
0.15	0.05	1	33.3%	
0.15	0.075	3	27.3%	
NN Ratio 0.8	0.075	2	50.0%	*

(a) Match results for $K = 3$.

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.10	0.05	6	42.8%	*
0.10	0.10	9	14.8%	
0.15	0.05	12	13.8%	
0.15	0.075	19	10.9%	
NN Ratio 0.8	0.05	0 (out of 2)	0%	

(b) Match results for $K = 4$.

Table 15: Threshold matching results for MSER with global context for Westin Hotel image pair.

The Westin Hotel image pair comes from the same panorama as the Elgin Street pair. Matching results are shown in Table 15. The improvement over the results in Table 3 is greater than what the numbers show. The actual building in this image pair has hundreds of windows that all look the same. With regular MSERs, matches were made between windows in the photograph and completely irrelevant elements in the other image, including cobblestones, trees, and features on the other building to the left. With the global context descriptor added, windows were only matched to other windows. A more sophisticated method than RANSAC for removing outliers might be useful to allow for a more relaxed threshold and thus more correct matches.

The next demonstration involves the Biology image pair. This image pair also contains a large amount of repetition, both in terms of the normalized shapes of the MSERs (due to many similar windows), and possibly in terms of the global context as well (for example, the areas surrounding the three sets of front doors are quite similar). Thus, any improvement in the results obtained from matching MSERs without global context would be welcome. Table 16 summarizes various matching results. These results are indeed an improvement over the basic thresholding without global context. The best results give just enough good matches to find a consistent epipolar geometry between the images, but also a reasonably low number of poor matches. Combining these results with the SURF with global context results could result in an even more accurate geometry.

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.10	0.05	8	88.9%	
0.10	0.10	14	31.8%	
0.15	0.075	14	57.6%	
0.15	0.15	37	11.4%	
NN Ratio 0.8	0.075	14	87.5%	*

(a) Match results for $K = 3$.

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.10	0.05	6	54.6%	
0.10	0.10	18	35.3%	
0.15	0.075	23	45.1%	
0.15	0.15	37	13.3%	
NN Ratio 0.8	0.075	16	64.0%	*

(b) Match results for $K = 4$.

Table 16: Threshold matching results for MSER with global context for Biology image pair using global context.

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.10	0.10	4	13.3%	
0.15	0.05	2	9.5%	
0.15	0.075	6	8.2%	
0.15	0.15	9	4.9%	
NN Ratio 0.8	0.1	7	50.0%	*

(a) Match results for $K = 3$.

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.10	0.10	4	17.4%	
0.15	0.05	2	15.4%	
0.15	0.075	5	11.4%	
0.15	0.15	9	6.6%	
NN Ratio 0.8	0.1	9	100%	*

(b) Match results for $K = 4$.

Table 17: Threshold matching results for MSER for Pharmacy image pair using global context.

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.10	0.10	2	4.8%	
0.15	0.05	0 (out of 9)	0%	
0.15	0.075	3	4.8%	
0.15	0.15	16	3.2%	
NN Ratio 0.8	0.1	4	40.0%	*

(a) Match results for $K = 3$.

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.10	0.10	0 (out of 0)	-	
0.15	0.05	0 (out of 1)	0%	
0.15	0.075	3	23.1%	
0.15	0.15	12	4.8%	*
NN Ratio 0.8	0.1	3	25.0%	

(b) Match results for $K = 4$.

Table 18: Threshold matching results for MSER with global context for the Side of Pharmacy image pair.

Results for the Pharmacy image pair, seen in Table 17, show that the addition of the global context does not always improve results. While matching with MSERs alone using basic thresholding didn't yield many matches, this method resulted in even fewer. One problem that occurred in particular was that letters in the main sign were matched with the same letter in the wrong place; even the global context can't necessarily help with this since entire words of the sign are repeated. There is a large amount of clutter that is not the same in both images, and the most reliable features in this pair - the main pharmacy sign and some of the smaller signs in the windows - are quite different in scale and blur. This might imply that more work is needed to ensure that the global context descriptor is less sensitive to these types of image transformations.

The Side of Pharmacy pair, seen in Table 18, has similar performance as matching with SURF global descriptors. Matching with MSERs and global context gives more correct matches, but a lower percentage. As mentioned in the SURF experiments, this image pair is difficult to match because of the large differences in viewing angle and image quality. While MSERs are often useful for matching large differences in viewing angle, the detection process used here does not take place in scale space. Doing so might help for images with some blur, though it may still be difficult to improve results for this particular pair.

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.10	0.10	0 (out of 7)	0%	
0.15	0.05	0 (out of 0)	-	
0.15	0.075	0 (out of 6)	0%	
0.15	0.15	6	13.3%	
NN Ratio 0.8	0.15	5	71.4%	*

(a) Match results for $K = 3$.

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.10	0.10	0 (out of 4)	0%	
0.15	0.05	0 (out of 0)	-	
0.15	0.075	0 (out of 1)	0%	
0.15	0.15	3	12.0%	
NN Ratio 0.8	0.15	2	66.7%	*

(b) Match results for $K = 4$.

Table 19: Threshold matching results for MSER with global context for the Cube image pair.

The Cube building in the pair shown in Table 19 does not have many distinguishable features. This is especially true when matching with MSERs; whereas SURF can be detected at the corners of the building and other such areas of interest, MSERs are detected in more solid areas such as windows (see Figure 40). Many of the incorrect matches occur amongst the features on the cars parked in front of the building, and there is little context surrounding the windows to help with matching. All this leads to a low number (and often percentage) of correct matches even in the best case.

Finally, in the case of the Genomics building, Table 20, the same problem observed with SURF matching occurs here: the areas of the panorama that should match with the photograph are so low in quality that it is not possible. Areas that look the same but that technically aren't correct matches do match well in this image. The good news with this example and the other difficult pairs (Cube and Side of Pharmacy) is that, with a low enough threshold, it is very unlikely to get false positives, which might be useful in some matching applications.

The best results of this section are summarized in Table 21. While MSERs with global context performed reasonably well for many image pairs, the results are almost always better matching for matching SURFs with global context. Still, for those images that MSERs give a high enough number and percent of correct matches, there is potential to combine results with those for SURF for the

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.10	0.10	0 (out of 1)	0%	
0.15	0.05	0 (out of 0)	-	
0.15	0.075	0 (out of 5)	0%	
0.15	0.15	1	3.1%	*
NN Ratio 0.8	0.15	0 (out of 13)	0%	

(a) Match results for $K = 3$.

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.10	0.10	0 (out of 4)	0%	
0.15	0.05	0 (out of 0)	-	
0.15	0.075	0 (out of 6)	0%	
0.15	0.15	0 (out of 13)	0%	*
NN Ratio 0.8	0.15	0 (out of 9)	0%	

(b) Match results for $K = 4$.

Table 20: Threshold matching results for MSER with global context for the Genomics image pair.

purpose of finding the epipolar geometry. This will be discussed in the next chapter.

6.4 Using SURF Points to Build Global Context for MSERs

In the previous section, a global context descriptor was built using curvature values surrounding Maximally Stable Extremal Regions. A multiple of the size of the MSER bounding ellipses was used as the measurement region, and a log-polar histogram collected all of the curvature values within the measurement region with an inverse Gaussian weighting. This differed from the approach used for SIFT and SURF points, where only curvature values at nearby feature points were considered. This section explores the effect of taking a similar approach with MSERs.

Although it was already established that nearby MSER regions should not be used when building global context, it might be possible to use nearby SURF points instead when both are detected in an image. The experiments listed below test this using Algorithm 6.7, which a slight variation of Algorithm 6.6.

Image Pair	**K=3, Number Correct**	**K=3, Percent Correct**	**K=4, Number Correct**	**K=4, Percent Correct**
Graffiti	198	99.5%	176	100%
Elgin Street	55	82.1%	51	79.7%
Westin	2	50.0%	6	42.8%
Biology	14	87.5%	16	64.0%
Pharmacy	7	50.0%	9	100%
Side of Pharmacy	4	40.0%	12	4.8%
Cube	5	71.4%	2	66.7%
Genomics	1	3.1%	0 (out of 13)	0%

Table 21: Summary of the best matching results for MSER with global context for $K = 3, 4$ (marked with * in previous tables).

Algorithm 6.7 Compute the global context vector for an MSER feature.

1. Normalize the MSER region (shape or texture, same as the original MSER patch) into a square patch of size $2R$ by $2R$. The area included in the patch is K times the bounding ellipse of the MSER region. The patch will be sampled from the original image for both shape and texture MSER features.

2. Compute curvature of new patch by finding the maximum absolute eigenvalue of the Hessian matrix at each pixel.

3. For each SURF point p found within R pixels of the centre of the patch:

   (a) Compute radial and angular bin for p.

   (b) Weigh curvature value found at p with an inverse Gaussian and chosen σ.

   (c) Add weighted curvature value to computed bin.

4. Create vector by flattening radial and angular bins.

6.4.1 Experimental Results

The experiments in this section are run the same way as the previous two (see Section 12 for more information). The multiple factor K is shown as 5 and 10 as in the SURF experiments. This differs from the MSER experiments because so many fewer pixels are considered in this scheme. With smaller multiples, it is often the case that no SURF points appear within the measurement region at all. When this happens, potential point matches must be ignored, since there is no additional information to distinguish similar features. Put another way, if two otherwise unrelated features have an all-zero global context descriptor but pass the SIFT thresholding, they will be incorrectly matched. This may sacrifice some true matches, but eliminates many bad ones.

The results for each of the image pairs are shown in Tables 22 through 29. Based both on the number and percentage of matches, this method did sometimes performs better than the MSER global context that uses all surrounding pixels, but is generally worse than the SURF global context results. A closer look at the individual numbers reveals why.

Like the previous two techniques for describing global context, it is possible to get perfect matching with the Graffiti image in Table 22. The highest number of correct matches comes from nearest neighbour matching, but was not chosen as the best result because of the slightly lower percentage. This matching method appears to be a reasonable choice for images with clear features like this pair,

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.05	1.0	77	100%	
0.05	1.3	91	100%	
0.05	1.6	96	98.0%	
0.10	1.0	124	93.2%	*
NN ratio 0.8	1.0	156	81.7%	

(a) Match results for $K = 5$.

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.05	1.0	92	100%	
0.05	1.3	97	98.9%	
0.05	1.6	97	97.0%	
0.10	1.0	149	88.2%	*
NN ratio 0.8	1.0	194	83.3%	

(b) Match results for $K = 10$.

Table 22: Threshold matching results for MSER with SURF global context for the Graffiti image pair.

even if it is not the best.

There are far fewer matches for the Elgin Street pair using this method (Table 23) than the previous two. When creating global context for SURF, other nearby SURF points are used, just as in this method, but there tend to be more SURF points clustered together than there are near MSERs. Compare the detection results in Chapters 4 and 5 to see this. This is why using all nearby pixels instead of specific features works better when building global context for MSERs. This is more noticeable in this example than the previous because the MSER features are generally smaller, so even a large multiple of the bounding ellipse may not include many (or any) SURF points. The Westin Hotel pair, Table 24, proves this further, as its features are even smaller.

The Biology images, Table 25, have larger features, and thus better results than the Westin pair. In this case, matching with a global context based on SURF is approximately equal to the previous two approaches. Using MSERs with the Pharmacy pair in Table 26, which has a dense distribution of SURF features around the sign, fared better when the global context was built with SURFs rather than all nearby pixels. Regardless, the all-SURF method performs better than either type of global context for MSERs.

Results for the Side of Pharmacy, Cube, and Genomics image pairs (Tables 27, 28 and 29) are

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.05	1.0	8	38.0%	
0.05	1.3	12	31.6%	
0.10	0.75	17	48.6%	
0.15	0.75	27	40.9%	
NN ratio 0.8	0.75	33	61.1%	*

(a) Match results for $K = 5$.

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.05	1.0	9	27.3%	
0.05	1.3	12	26.7%	
0.10	0.75	38	35.2%	
0.15	0.75	59	29.8%	
NN ratio 0.8	0.75	59	55.1%	*

(b) Match results for $K = 10$.

Table 23: Threshold matching results for MSER with SURF global context for the Elgin Street image pair.

fairly poor as before. Both methods using MSERs give similar results with the SURF results slightly better.

A summary of the best results from each image pair and value of K is shown in Table 30. Comparing this with the previous two summaries, the conclusion is that building global context for MSERs using SURF is a reasonable choice for higher quality images with large, clear shapes to be detected as MSERs. However, for lower quality images or those with smaller features, this method does not work especially well.

6.5 Comparison of All Matching Techniques

In this section, matching techniques presented in this chapter and the previous two are compared and discussed. Figures 44-49 summarize a selection of matching methods by plotting each result with the percentage of correct matches on the horizontal axis and the number of correct matches on the vertical. A legend for the data points can be found in Table 31. In the case of global context matching, the best results from Tables 12, 21, and 30 are used for results **F** through **K**. Six images are summarized; the Side of Pharmacy and Genomics buildings are not, since their results are an

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.05	1.6	2	4.6%	
0.10	1.0	1	5.6%	
0.10	1.3	3	5.3%	
0.10	1.6	3	1.6%	
NN ratio 0.8	1.0	3	25.0%	*

(a) Match results for $K = 5$.

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.05	1.6	0 (out of 51)	0%	
0.10	1.0	2	12.5%	
0.10	1.3	6	8.1%	
0.10	1.6	6	3.0%	
NN ratio 0.8	1.0	3	16.7%	*

(b) Match results for $K = 10$.

Table 24: Threshold matching results for MSER with SURF global context for the Westin Hotel image pair.

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.05	1.0	7	9.2%	
0.05	1.3	14	7.0%	
0.10	0.75	9	22.0%	
0.10	1.0	15	11.0%	
NN Ratio 0.8	1.0	13	46.4%	*

(a) Match results for $K = 5$.

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.05	1.0	15	8.0%	
0.05	1.3	15	4.8%	
0.10	0.75	18	12.2%	
0.10	1.0	28	8.0%	
NN Ratio 0.8	1.0	24	45.3%	*

(b) Match results for $K = 10$.

Table 25: Threshold matching results for MSER with SURF global context for the Biology image pair.

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.10	1.0	1	20%	
0.10	1.3	4	30.8%	
0.15	1.3	7	20.6%	
0.15	1.6	12	20.7%	*
NN Ratio 0.8	1.6	13	13.8%	

(a) Match results for $K = 5$.

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.10	1.0	3	27.3%	
0.10	1.3	3	27.3%	
0.15	1.3	6	21.4%	
0.15	1.6	11	22.5%	*
NN Ratio 0.8	1.6	13	15.5%	

(b) Match results for $K = 10$.

Table 26: Threshold matching results for MSER with SURF global context for the Pharmacy image pair.

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.10	1.0	2	9.1%	
0.10	1.3	3	3.6%	
0.15	0.75	5	38.5%	*
0.15	1.0	7	11.7%	
NN Ratio 0.8	0.75	2	33.3%	

(a) Match results for $K = 5$.

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.10	1.0	3	3.3%	
0.10	1.3	4	2.0%	
0.15	0.75	7	8.2%	
0.15	1.0	10	4.1%	
NN Ratio 0.8	0.75	5	15.2%	*

(b) Match results for $K = 10$.

Table 27: Threshold matching results for MSER with SURF global context for the Side of Pharmacy image pair.

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.10	1.0	1	14.3%	
0.10	1.3	2	15.4%	
0.15	1.0	6	18.8%	*
0.15	1.3	7	14.6%	
NN Ratio 0.8	1.3	4	22.2%	

(a) Match results for $K = 5$.

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.10	1.0	1	5.9%	
0.10	1.3	2	10.0%	
0.15	1.0	7	15.6%	
0.15	1.3	8	12.9%	*
NN Ratio 0.8	1.3	5	17.2%	

(b) Match results for $K = 10$.

Table 28: Threshold matching results for MSER with SURF global context for the Cube image pair.

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.10	1.0	0 (out of 2)	0%	
0.10	1.3	1	14.3%	
0.15	1.3	1	5.3%	
0.15	1.6	3	7.9%	*
NN Ratio 0.8	1.6	0 (out of 39)	0%	

(a) Match results for $K = 5$.

SIFT Threshold	Global Threshold	Correct Matches	Percent Correct	Best
0.10	1.0	0 (out of 1)	0%	
0.10	1.3	0 (out of 6)	0%	
0.15	1.3	0 (out of 19)	0%	
0.15	1.6	1	1.8%	*
NN Ratio 0.8	1.6	0 (out of 37)	0%	

(b) Match results for $K = 10$.

Table 29: Threshold matching results for MSER with SURF global context for the Genomics image pair.

Image Pair	**K=5, Number Correct**	**K=5, Percent Correct**	**K=10, Number Correct**	**K=10, Percent Correct**
Graffiti	124	93.2%	149	88.2%
Elgin Street	33	61.1%	59	55.1%
Westin	3	25.0%	3	16.7%
Biology	13	46.4%	24	45.3%
Pharmacy	12	20.7%	11	22.5%
Side of Pharmacy	5	38.5%	5	15.2%
Cube	6	18.8%	8	12.9%
Genomics	1	1.8%	3	7.9%

Table 30: Summary of the best matching results for MSER with SURF global context for $K = 5, 10$ (marked with * in previous tables).

A	SURF, basic threshold 0.15
B	SURF, basic threshold 0.3
C	SURF, NN ratio threshold 0.8
D	MSER, basic threshold 0.05
E	MSER, NN ratio threshold 0.8
F	Best SURF with global $K = 5$
G	Best SURF with global $K = 10$
H	Best MSER with global, $K = 3$
I	Best MSER with global, $K = 4$
J	Best MSER with SURF global, $K = 5$
K	Best MSER with SURF global, $K = 10$

Table 31: Legend for matching results in Figures 44-49.

example of much more difficult matching scenarios, and aren't of great interest for detailed analysis.

The Graffiti image pair is summarized in Figure 44. It is immediately obvious that matching using nearest-neighbour ratio thresholding with SURF, but without global context, performs the best. The results using global context for either SURF or MSER are still very good; they simply have smaller numbers of correct matches to go along with their high percentages. This is likely due to the fact that the scene in this pair has many clear and distinct features that are able to match well with no supplementary context. When adding the global context, false matches may be eliminated, but so too are some good matches, particularly for features near the boundaries of the image, where the global context descriptor may vary too much. In this case, performance for matching MSER features is certainly improved with global context, since many MSER features are less distinct than SURFs, particularly for shape patches.

Figure 45 shows results for the Elgin Street Matching pair. The first two techniques for computing global context for SURF and MSER result in high percentages of correct matches, and although the

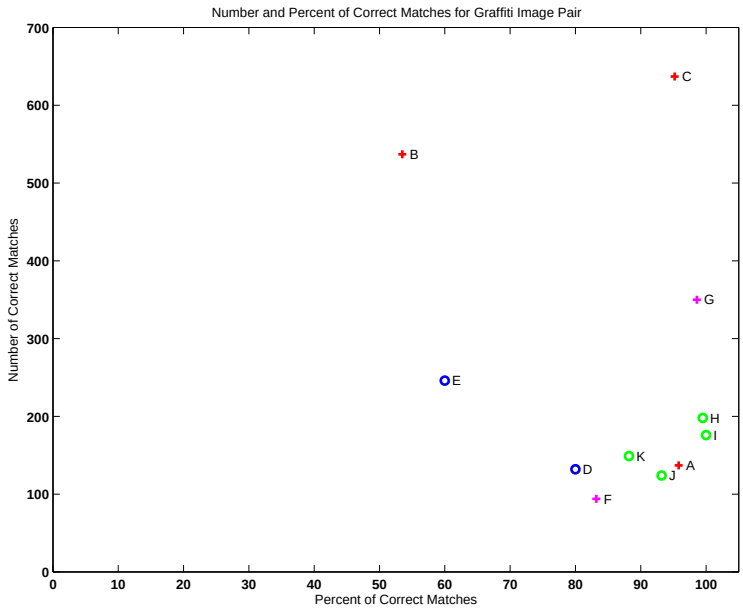


Figure 44: Matching results for Graffiti image pair.

SURF nearest-neighbour result has a higher number, its percentage is lower. The percentages of correct matches for MSER global context using SURF are noticeably lower than the other techniques, though the larger measurement area has a similar number as the others. This pair has good image quality, but more repetitive features than the Graffiti pair. Results for MSER thresholding are not shown, since there were too many possibilities to count by hand.

The Westin Hotel results are in Figure 46. The scale on the vertical axis and the prominence of results along the bottom left immediately indicate the difficulty of this image pair. Some of the SURF results appear to be sufficient to potentially find the epipolar geometry for this pair, but the MSER results contribute little. Results for MSER thresholding are again not shown for the same reason as above. There are a few possibilities for this, as discussed earlier in this chapter. For instance, the hotel building appears at a very different scale in the two images, and has many small, repeating features (the windows). The measurement region for SURF is based on the scale on which it was detected rather than the size of the feature as is the case with MSER, so smaller MSERs can include less context than larger MSERs.

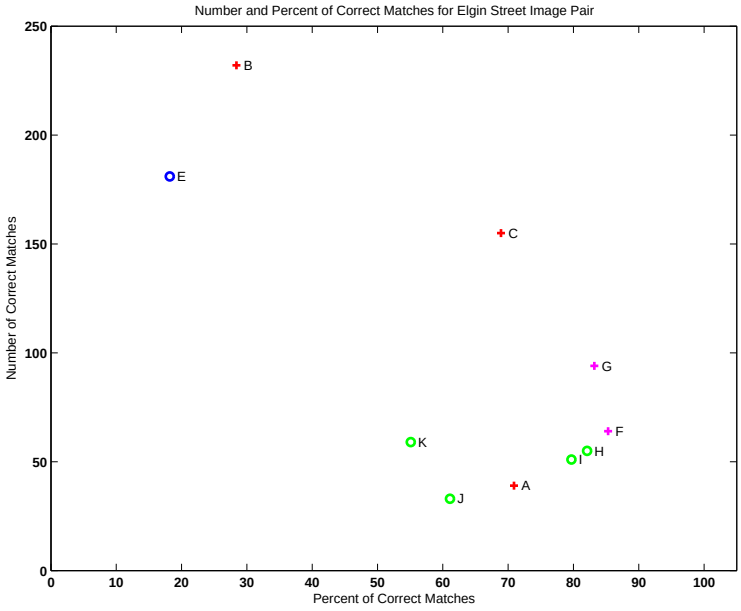


Figure 45: Matching results for Elgin Street image pair.

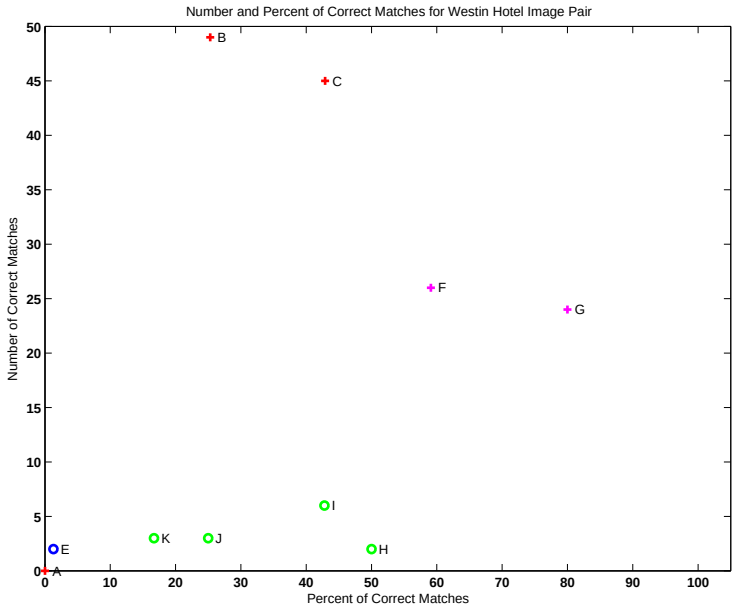


Figure 46: Matching results for Westin Hotel image pair.

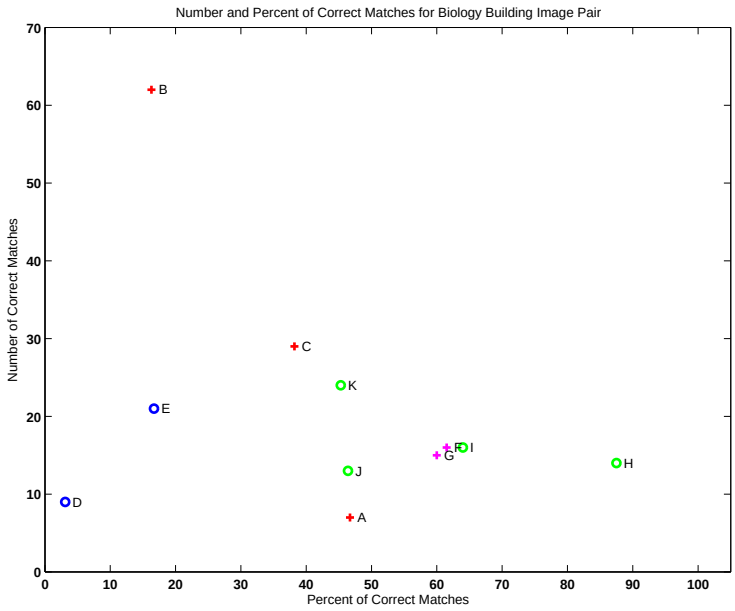


Figure 47: Matching results for Biology building image pair.

The biology building, summarized in Figure 47, differs from the previous two image pairs by having lower image quality in the panorama. This seems to result, in general, in fewer correct matches in general. In terms of percentages, it is interesting that, for the first time, MSER with global context far outperforms the others. This is a good example of an image pair whose epipolar geometry might be most easily found by combining the lower numbers of correct matches from the best performing SURF and MSER results in terms of high percentages.

The Pharmacy image pair (Figure 48) is another difficult example. The image quality of the panorama is lower than in the Biology pair, and the scale difference between the images is greater. SURF without global context fared better than in the Biology pair, though the percentage is still lower than some of the other results. There are results with high percentages for both MSER and SURF with global context, but again these would likely have to be combined to find epipolar geometry due to the lower numbers.

Finally, the Cube image pair (Figure 49) represents a case where there are not many available features to match. The building itself is fairly non-descript, and the cars parked in front are different in

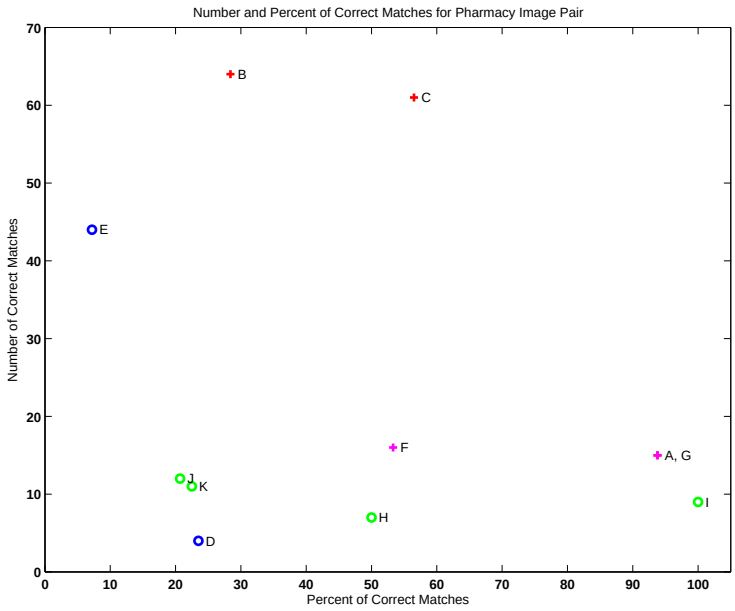


Figure 48: Matching results for Pharmacy image pair.

the two images (otherwise, they would probably match well). The SURF with global context results may be the most useful here, given the higher number of matches. The MSER with global context results would likely be a useful addition to the SURF results thanks to their higher percentages.

Looking at all the graphs, a few conclusions can be made. When matching images with clear, distinct features at similar scales and image qualities, matching SURFs without global context using nearest-neighbour ratio thresholding is a very good choice. Alternatively, either SURF or MSER with global context can be used to achieve a high percentage of matches but smaller numbers. On the other hand, image pairs with differences in scale and image quality seem to benefit more from using global context. In most cases, SURF methods outperform MSER techniques. Using SURFs to build global context for MSERs does not perform as well as the other global context methods, but can be better than basic thresholding, usually in terms of percentage of correct matches.

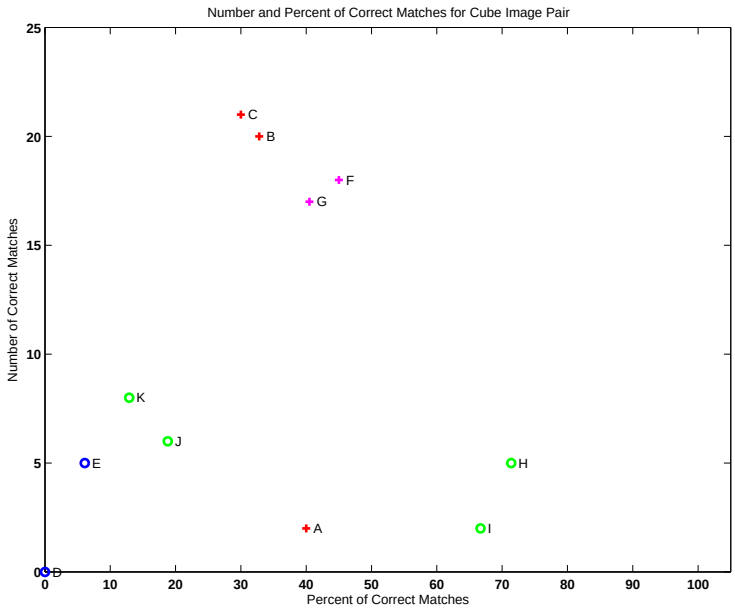


Figure 49: Matching results for Cube image pair.

6.6 Conclusion

This chapter explored three new uses of the global context descriptor originally designed for SIFT features: SURF with global context using other SURF points, MSER with global context using all pixels in measurement region, and MSER with global context using SURFs. The new feature descriptors were evaluated for a basic photograph pair and several photograph and panorama pairs. SURF with global context generally outperformed the other two, and MSER with all pixels performed better than MSER with SURF. Features with global context generally had higher percentages of correct matches than features without, but lower numbers.

The next chapter will explore using the best results from each image to find the epipolar geometry between a photograph and panorama. Higher percentages of matches will be favoured, making features with global context the natural choice.

Chapter 7

Combining Match Results to Find Epipolar Geometry

Various features were explored in the previous chapters in the context of matching photographs to panoramas, including matching with Speeded Up Robust Features, and with Maximally Stable Extremal Regions, both with and without additional global context. For some image pairs - particularly those with many repetitive features and that include a panorama made with Ladybug camera data - each matching strategy resulted in barely acceptable rates of good matches. The highest *percentage* of correct matches often came with a low *number* of matches. This chapter examines the results when the best matches from each technique are combined, and tests whether the epipolar geometry can be found when matching the entire panorama to the photograph.

7.1 Introduction to Experiments

The main goal of the experiments in this chapter is to determine whether enough progress has been made in matching techniques to make it possible to find the geometry between an entire panorama and a photograph. If enough correct matches are found, the RANSAC method can be used to find a pseudo-fundamental matrix that represents the epipolar geometry.

Each image pair considered so far in this thesis has presented a matching challenge. Panoramas created with Ladybug data did not always have the same image quality as the photographs being matched to them. Some panoramas, such as the Genomics building, were poor enough that a good match set does not seem feasible at this point. It is not expected that finding an epipolar geometry with the current matching techniques will be possible in these cases. Therefore, the focus of this chapter will be on four panorama-photograph pairs that have a sufficient number of SURF and/or

MSER matches.

The experiments in the next section will seek a consistent epipolar geometry using a basic RANSAC implementation [20] that fits a pseudo-fundamental matrix to a set of matches. Matching techniques will be chosen based on a high percentage of correct matches from previous experiments, and not necessarily a particularly high number of correct matches (though the more correct matches that are available, the better the final results should be). One matching technique using SURF with global context and another using MSER with global context will be evaluated first separately. Then, the results from both techniques will be combined to see if a higher quality result can be produced.

The pseudo-essential matrix found during the RANSAC process will be used to find the distances of hand-picked points to their corresponding epipolar lines. The average distance will be reported. Note that the matching algorithms from Chapter 6 will be used with the photograph as the first image since matching occurs in one direction only, and the features of the photograph are a subset of those in the panorama.

7.1.1 Experimental Results

The four image pairs used in this chapter are those that have a sufficient number of matches with which to find the epipolar geometry: Elgin Street, Biology, Pharmacy, and Cube. The Westin hotel probably has enough SURF matches to find the geometry, but won't be examined because the comparison between using only SURF or MSER and the two combined is also important for this chapter; this pair has poor results with MSER. The Cube pair also has very few correct MSER matches, but unlike the Westin pair, the percentage is high, and the number of SURF matches is also low; this makes the Cube pair worth examining.

The results for these images are summarized in Table 32. The first number listed in each column is the average distance, in pixels, from hand picked points to their associated epipolar line in the photograph. The second number is the average distance to the epipolar plane in the panorama. Around 16 matching points were chosen throughout the image. The epipolar geometry for each image pair is visualized in Figure 50 through Figure 53.

The results show that the quality of the pseudo-fundamental matrix is improved by combining SURF and MSER matches in most cases. This is not only because of the higher support extra matches provide, but also because SURF and MSER features are generally located in different areas of the images. That is, SURF points tend to be detected around blobs, while an MSER centroid

Image Pair	**Average Distance: Best SURF**	**Average Distance: Best MSER**	**Average Distance: Both**
Elgin Street	7.4243 4.0222	9.5874 2.2707	8.1541 1.7600
Biology	3.3729 4.3441	2.6981 3.6264	1.0711 1.7592
Pharmacy	20.9664 6.4114	48.9112 66.2059	9.4063 8.2276
Cube	18.7797 13.9011	(not enough features)	20.9793 16.5355

Table 32: Quality of pseudo-fundamental matrix from matching with MSER, SURF, and both. In each cell, the first number indicates the average distance from the hand-picked points in the panorama to their corresponding epipolar lines computed with the pseudo-fundamental matrix found automatically before. The second number is the same average for points in the photograph.

would be located at the centre. The more spread out the matches are, the better the pseudo-fundamental matrix will represent the geometry of the image pair.

In three out of the four pairs (that is, all but the Cube pair), SURF features alone are superior to MSER features alone. The only exception is the biology building, which agrees with the matching summary graph in Figure 47 where MSER with global context performed the best. The Cube pair ended up not even having enough correctly matching MSER features to find the epipolar geometry. This makes sense given that few matches were found even when using one face of the cube. Nearest neighbour matching was used again for the entire panorama since no other method worked nearly as well. Unfortunately, this means that some features from the photograph were potentially matches to features in the wrong face on the panorama. If thresholding were used, this might be eliminated, but many incorrect matches would have been selected in addition to the correct matches. Exploring how to handle this issue might be the topic of future work.

The same three pairs also saw the overall performance of the computed pseudo-fundamental matrix improve. Even the Pharmacy pair has better results with both types of features, even though

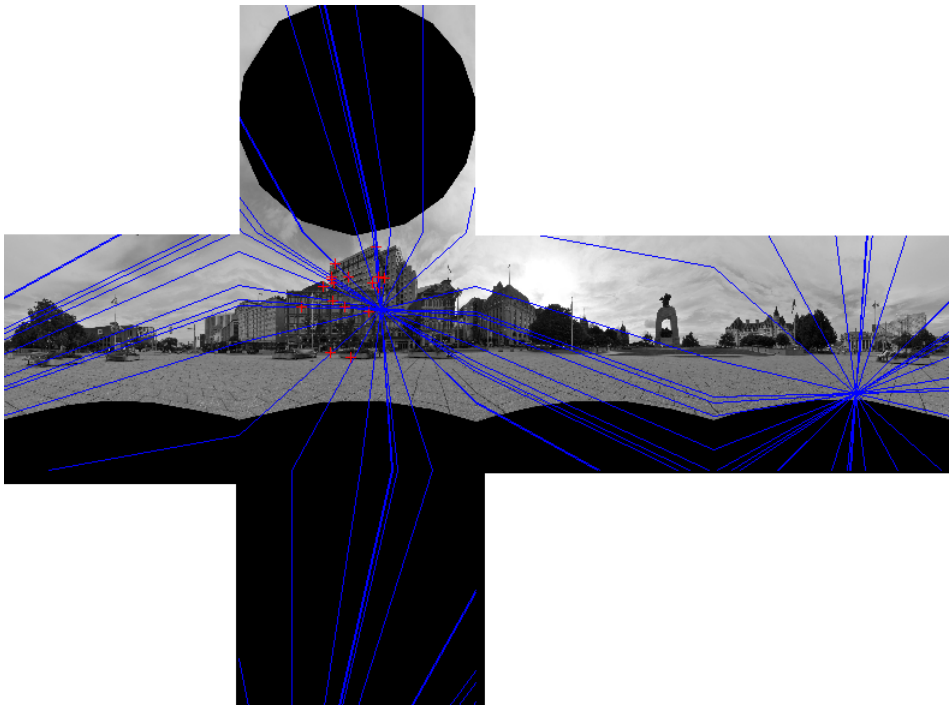
MSERs alone were significantly worse than SURFs. This supports the hypothesis that having the two types of features with different locations can really help. When the MSERs were added to the SURF matches for the cube, the results unfortunately worsened. However, the difference in the average distance is within a few pixels, suggesting that it may be safe to assume that all MSER and SURF matching sets can be combined without fear of occasionally changing the results by very large amounts.

Looking at the epipolar geometry in Figures 50 through 53 gives some more insight into the quality of the results. In these diagrams, it is possible to see how far each match is from its corresponding epipolar line. In this way, it can be determined whether a high average distance is due to many points being far away, or only one or two. In most cases, what appears to be happening is the matches found automatically are not spread across the entire area of the photograph. Those areas without matches are where the hand picked matches perform the worst. For example, the Elgin Street pair has very good results on the buildings, but hand picked points are farther away from the epipolar lines on the planter boxes and light post. The Cube provides an even more obvious example: the points chosen on the parking lines painted on the pavement are very far away. No features were matched here because the cars would have made the global context too different. If more matches could be made throughout the entire common area between the images, a better pseudo-fundamental matrix would result.

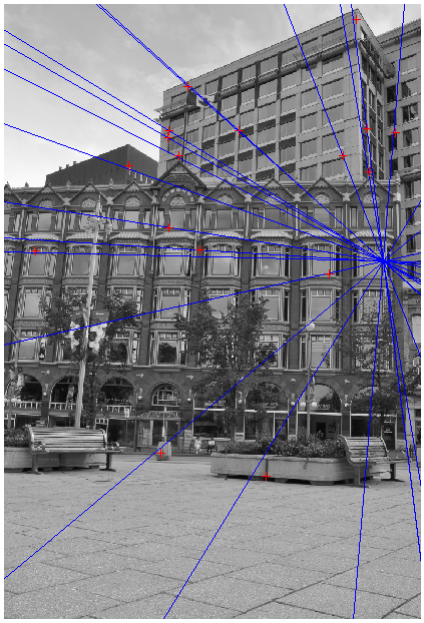
The diagram for the Cube image pair, shown in Figure 53, as well as the numerical results in Table 32, suggest that this pair is actually a failure case when a general accurate epipolar geometry is desired. The epipolar lines are too far from some points, as explained above. Using the geometry to do general augmentations may cause visually incorrect results. However, the results might be used in a more forgiving application, such as object or location recognition.

7.2 Conclusion

This chapter showed that it is possible to find the epipolar geometry between a photograph and panorama automatically using the matching techniques described in previous chapters. It also demonstrated that combining matches made with SURF and MSER features generally improves the pseudo-fundamental matrix representing this geometry. However, there are still improvements to be made in future work, such as finding ways to better match features throughout the photograph.

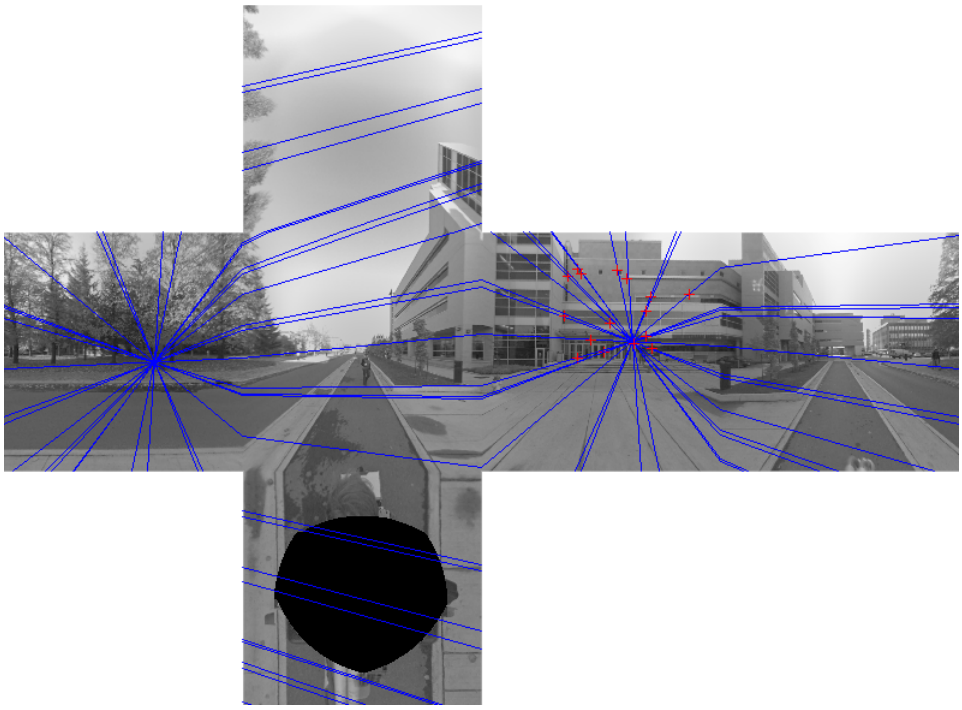


(a) Panorama

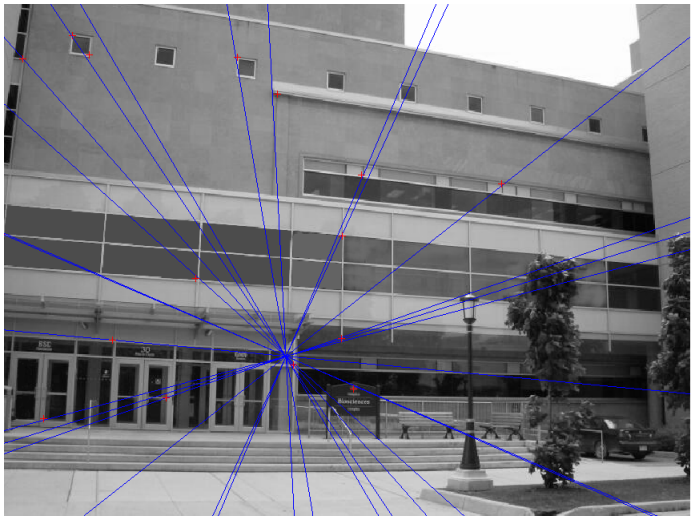


(b) Photograph

Figure 50: Epipolar geometry for Elgin Street image pair based on combined MSER and SURF match results, shown with hand picked matching points.

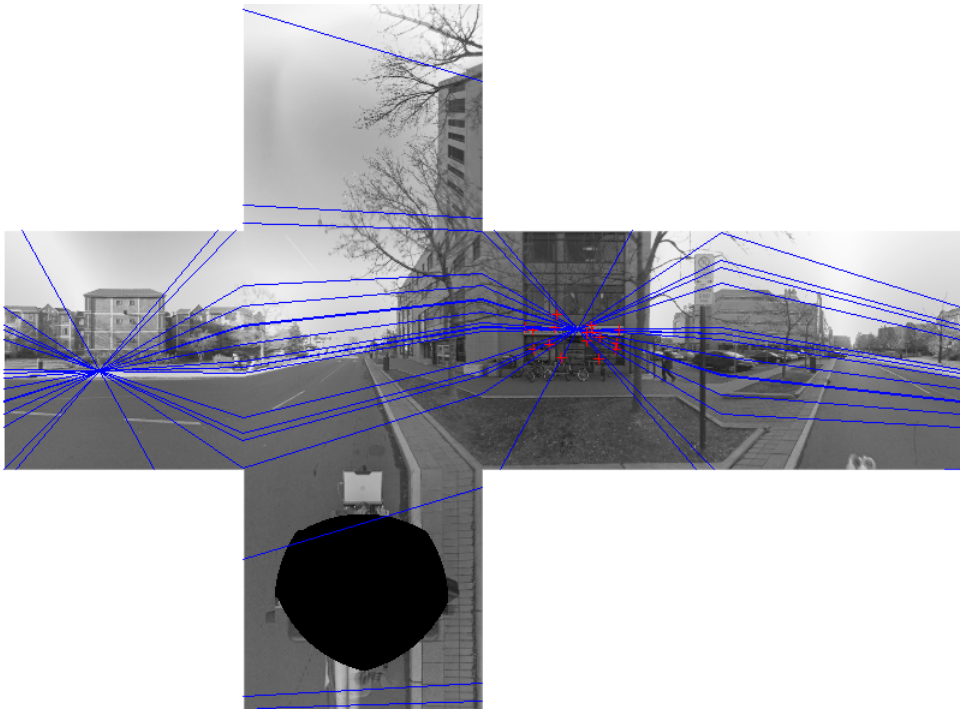


(a) Panorama

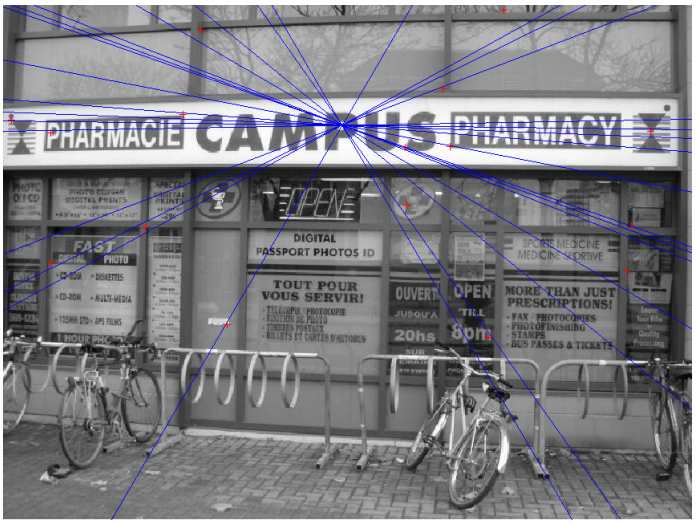


(b) Photograph

Figure 51: Epipolar geometry for Biology image pair based on combined MSER and SURF match results, shown with hand picked matching points.

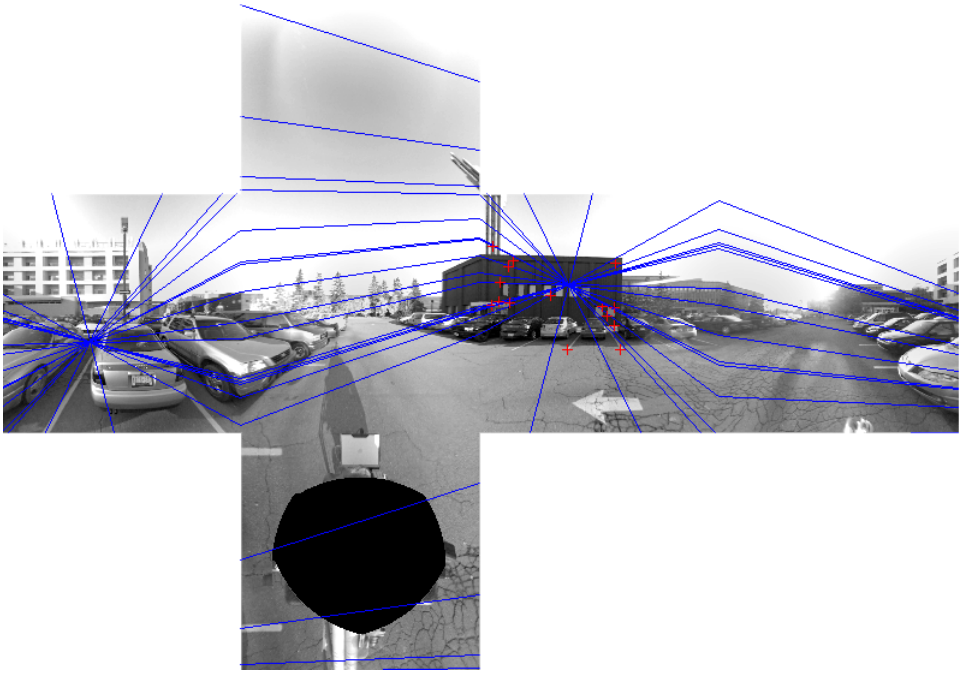


(a) Panorama



(b) Photograph

Figure 52: Epipolar geometry for Pharmacy image pair based on combined MSER and SURF match results, shown with hand picked matching points.



(a) Panorama

(b) Photograph

Figure 53: Epipolar geometry for Cube image pair based on combined MSER and SURF match results, shown with hand picked matching points.

Chapter 8

Conclusion and Future Work

This thesis began looking at the problem of matching a spherical panorama to a planar photograph. After giving the basics of matching and epipolar geometry for two photographs and two panoramas in Chapter 2, the focus shifted to the case of one photograph and one panorama in Chapter 3. Chapters 4 and 5 explored detecting and matching SURF and MSER features using standard techniques. Chapter 6 expanded this by adding a global context descriptor to increase the distinctiveness of the features. Finally, Chapter 7 showed four examples of the epipolar geometry obtained using SURF and MSER features alone and combined.

This chapter summarizes the thesis conclusions and outlines areas of future work.

8.1 Geometry Between Panoramas and Photographs

The geometry between two photographs has been studied for some time. The fundamental matrix and epipolar geometry are well understood in the case of a pair of two planar images. More recently, this theory was extended for use with a pair of spherical panoramas, where the essential matrix was deemed the most intuitive way to represent the epipolar geometry. Cube panoramas were studied specifically, but the concepts are easily extended to any other type. This thesis described the geometry between a spherical panorama and planar photograph.

Many of the same concepts apply to the panorama-photograph case, except for the use of the essential matrix. The essential matrix was deemed a more convenient way to represent the geometry between two spherical panoramas, a normalized pixel coordinates of points on those panoramas are easily obtained. Since the properties of many consumer cameras are not known so that photographs are thus not always calibrated, and since it would still be convenient to use the normalized panorama coordinates, a slightly modified version of the fundamental matrix was proposed in Chapter 3. The

pseudo-fundamental matrix is partially calibrated so that normalized coordinates for one image may be used while standard image coordinates may be used for the other. The properties of this matrix were shown to agree with the properties of the fundamental matrix, and examples show experimentally that the idea is sound.

Several challenges appear when matching these two very different types of images - panoramas and photographs - including a large difference in image quality (which is more pronounced when comparing photographs with panoramas generated with data captured by the Ladybug camera equipment), a larger field of view in the panorama, differences in viewing angle and focal lengths, and repetitive features. The thesis focused on solving the problem of repetitive features, though some of the other challenges became apparent in several examples.

8.2 Matching With SURF and MSER

Speeded Up Robust Features, or SURFs, have become a popular alternative to Scale Invariant Feature Transform, or SIFT, features, as they are claimed to be faster and at least as robust. Matching with these features using basic thresholding or nearest-neighbour ratio thresholding often produced many correct matches, but also many incorrect matches. Image pairs that included panoramas of lower image quality had far fewer matches. Results for matching with Maximally Stable Extremal Regions (MSERs) with the same techniques were even worse. Many of the features detected as MSERs were simply not distinctive enough.

Both SURF and MSER features performed better when a global context descriptor is used to augment the existing feature descriptors, increasing their distinctiveness. In many cases, fewer correct matches were found, but the percentage of correct matches often rose significantly. The SURF global descriptors proposed are computed by collecting image curvature values at nearby SURF points in a log-polar histogram. The SURF descriptors with global context performed better than any global descriptor for MSER, even though SURF features have no affine normalization as MSERs do. For MSERs, two new types of global descriptors were compared. The first used curvature values at all nearby pixels, and the other collected only curvature values at nearby SURF points. The latter yielded reasonable results, but the former outperformed it in general.

Chapter 7 experimented with the best matching results found in previous chapters. A pseudo-fundamental matrix was found using the matches determined automatically, and the quality of this matrix was evaluated using a set of hand-picked matches. The matrix was found individually for the best SURF and MSER results (in terms of the best percentage of correct matches with a reasonable

number of good matches), then for the combination of the two. In three out of four cases, the combination improved the results, and in the one case it did not, the results did not change greatly.

8.3 Future Work

Though the geometry of a cube panorama has been established and new matching techniques proposed, there is more work to be done to improve the ability to match panoramas to photographs. Additionally, the many applications discussed in the introduction of the thesis would be worth exploring.

First, several assumptions were made when matching with MSERs, causing limitations to the method. For instance, buildings with mirror-like windows caused reflections that make detecting reliable MSERs very difficult, so some images were preprocessed to eliminate these reflections. In addition, the contrast was manually adjusted in both panoramas and photographs to obtain better MSERs during the detection process. This is not a problem for a database of panoramas that is made before any matching occurs, but manual intervention is not possible for photographs which are taken on the fly for many applications.

Some research [14] has suggested that detecting MSERs in a scale-space can improve matching results. This was not explored here, but there were several image pairs considered earlier that might have benefited from the idea, particularly those with lower quality panoramas made from Ladybug data. A comparison of results with and without this change in detection would determine whether it would help match these types of images.

More work on the measurement region for the MSER global context descriptor might yield better results. For instance, it was observed that image pairs with smaller MSER features, such as the Westin Hotel pair, had poor performance with these features. Perhaps a minimum radius might be used for all measurement regions to ensure that enough context is considered.

While the idea of computing the global context descriptor for MSERs using nearby SURF points was deemed to underperform compared to the descriptors that used all nearby pixels, the reasons why were not looked into. It may be worth exploring this in more depth to both understand why this is the case, and to see whether there is a way to improve the result. Since SURF features with the global descriptor worked so well, it is reasonable to assume that MSERs might benefit from the same approach. It may even be possible to find a way for the MSER and SURF combination to outperform SURF in some cases, since MSERs have the added advantage of being affine covariant. The differences in the measurement regions for SURFs and MSERs may provide a clue.

Finally, the main matching challenge addressed in this thesis was the prevalence of repetitive features. However, addressing the other challenges would not only improve the results shown here, but expand the abilities of any real implementation of the applications suggested in the thesis introduction. For instance, the bigger the difference in image quality, the more difficult it is to find reliable matches. The shape patches used to describe MSERs might be beneficial for these scenarios, but a more sophisticated matching and outlier removal scheme must be found, such as a relative ordering heuristic to remove impossible matches. Also, many of the best results when matching one panorama face come from nearest-neighbour ratio thresholding, but when increasing the field of view to include the entire panorama, many incorrect matches are suddenly introduced. Because basic thresholding often introduces far too many incorrect matches, it may be worth exploring the best way to work around the nearest-neighbour problem.

With improvements to some general but applicable matching techniques, as well as solutions to the challenges specific to matching this type of image pair, matching panoramas with photographs can become a very useful tool in many applications, and is worth studying further.

Appendix A

Rotation Matrices for Cube Face Coordinate Frames

The projection matrices for individual faces of a cubic panorama were first discussed in section 2.3.2. It was mentioned that each face has a rotation matrix to transform its coordinate frame to the world coordinate frame, placed at the front face's frame. Let $R_{axis}(\theta)$ be a rotation of amount θ around *axis* x, y, or z. Then the rotation matrices R_i, derived by Kangni [18], are as follows:

$$R_U = R_x(\tfrac{\pi}{2}) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \quad R_L = R_y(\tfrac{\pi}{2}) = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix} \quad R_F = R_x(0) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R_R = R_y(-\tfrac{\pi}{2}) = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \quad R_B = R_y(\pi) = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \quad R_D = R_x(-\tfrac{\pi}{2}) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix}$$

Appendix B

Converting Cube Image Points to 3D and Finding Plane Intersections

Kangni laid out a series of transformations that would take a point in face coordinates to a 3D point in the cube's coordinate frame [18]. Each cube face has a coordinate system whose origin is at the top left of the face, with the y-axis point down. When a cube is laid out in a cross pattern, the face coordinates can be obtained by subtracting the appropriate offsets from the overall image point, using the known length L of each face.

The transformations below make use of the fact that, for each face, one coordinate will be constant at $x = \pm\frac{L}{2}$ for the *right* and *left* faces, $y = \pm\frac{L}{2}$ for the *top* and *down* faces, and $z = \pm\frac{L}{2}$ for the *front* and *back* faces. Multiply a face point with the corresponding matrix below to get the 3D cube coordinates.

$$T_U = \begin{bmatrix} 1 & 0 & -\frac{L}{2} \\ 0 & 0 & \frac{L}{2} \\ 0 & -1 & \frac{L}{2} \end{bmatrix} \quad T_L = \begin{bmatrix} 0 & 0 & -\frac{L}{2} \\ 0 & -1 & \frac{L}{2} \\ -1 & 0 & \frac{L}{2} \end{bmatrix} \quad T_F = \begin{bmatrix} 1 & 0 & -\frac{L}{2} \\ 0 & -1 & \frac{L}{2} \\ 0 & 0 & -\frac{L}{2} \end{bmatrix}$$

$$T_R = \begin{bmatrix} 0 & 0 & \frac{L}{2} \\ 0 & -1 & \frac{L}{2} \\ 1 & 0 & -\frac{L}{2} \end{bmatrix} \quad T_U = \begin{bmatrix} -1 & 0 & \frac{L}{2} \\ 0 & -1 & \frac{L}{2} \\ 0 & 0 & \frac{L}{2} \end{bmatrix} \quad T_D = \begin{bmatrix} 1 & 0 & -\frac{L}{2} \\ 0 & 0 & -\frac{L}{2} \\ 0 & 1 & -\frac{L}{2} \end{bmatrix}$$

The known coordinates of cube faces are also useful in finding the intersection of an epipolar plane given by $E\mathbf{p}$ for a point $\mathbf{p}$ in 3D cube coordinates. The plane is given by $\pi = (a, b, c)$ in

face i	a_i	b_i	c_i
U	a	$-c$	$g_U(b)$
L	$-c$	$-b$	$g_L(-a)$
F	a	$-b$	$g_F(-c)$
R	c	$-b$	$g_R(a)$
B	$-a$	$-b$	$g_B(c)$
D	a	c	$g_D(-b)$

Table 33: Coordinates of lines of intersection between the epipolar plane and cube.

projective coordinates since it passes through the centre of the cube. The line of intersection on the supporting plane of each face will be $l_i = (a_i, b_i, c_i)$, where the values of a_i, b_i, and c_i are given in Table 33. Function g_i is given by

$$g_i(m) = \frac{L}{2}(m - (a_i + b_i))$$

Appendix C

Image Pairs Used in Experiments

Figures 54-60 provide references to the panoramas and photographs used in the experiments of Chapters 4-6.

(a) Photograph 1 (b) Photograph 2

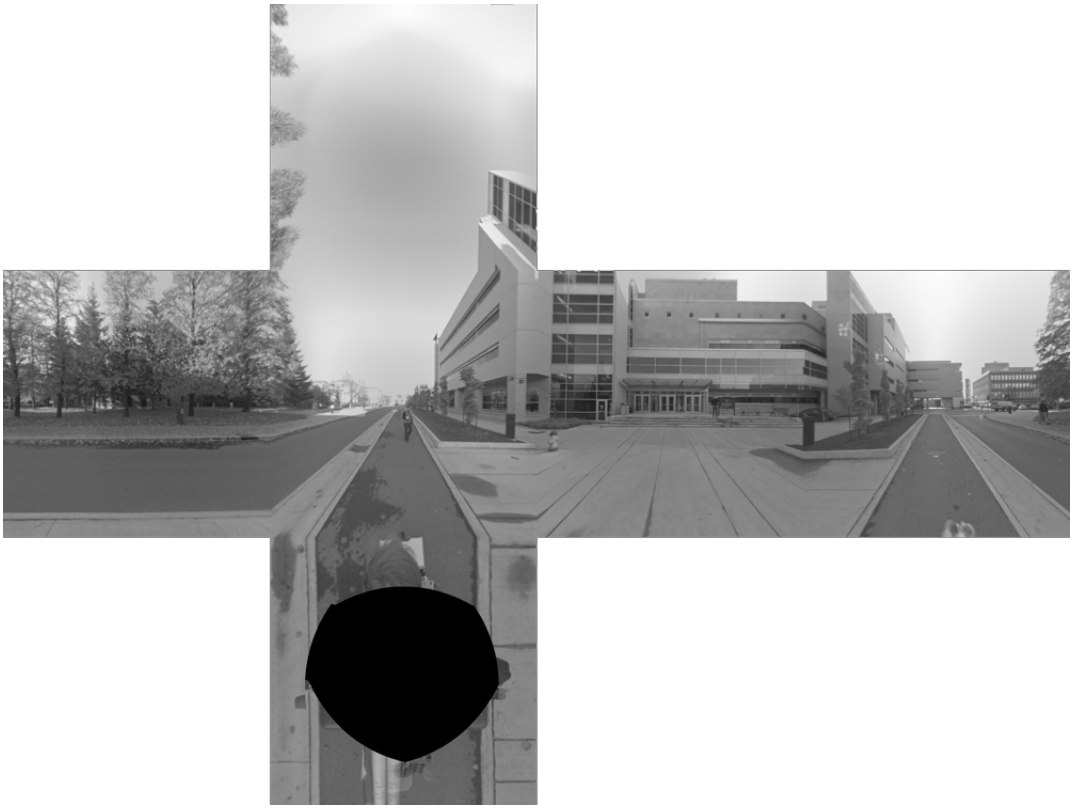
Figure 54: Graffiti photograph pair

(a) Panorama (top and bottom faces truncated)

(b) Elgin Street photograph

(c) Westin Hotel photograph

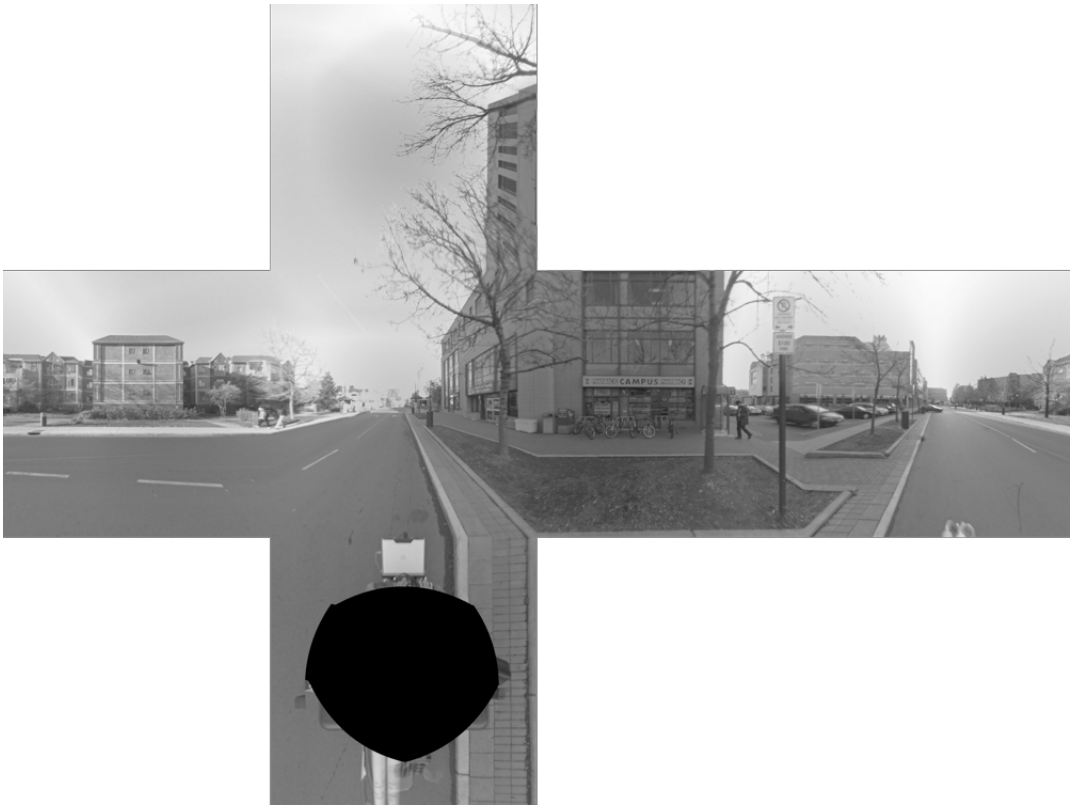
Figure 55: Elgin Street and Westin Hotel image pair



(a) Panorama

(b) Photograph

Figure 56: Biology image pair



(a) Panorama

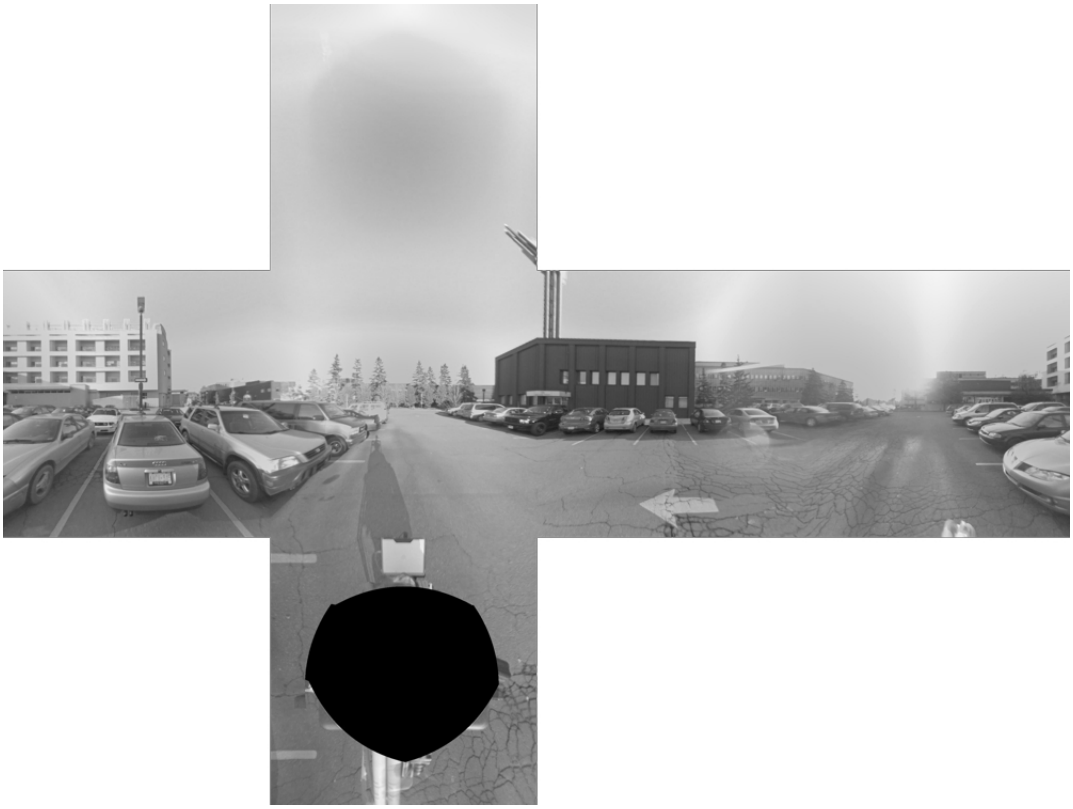
(b) Photograph

Figure 57: Pharmacy image pair

(a) Panorama

(b) Photograph

Figure 58: Side of pharmacy image pair



(a) Panorama

(b) Photograph

Figure 59: Cube building image pair

(a) Panorama

(b) Photograph

Figure 60: Genomics building image pair

Bibliography

[1] Affine covariant features. http://www.robots.ox.ac.uk/ vgg/research/affine/.

[2] Johannes Bauer, Niko Sünderhauf, and Peter Protzel. Comparing several implementations of two recently published feature detectors. In *Proc. of the International Conference on Intelligent and Autonomous Systems*, 2007.

[3] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. SURF: Speeded up robust features. In *ECCV*, pages 404–417, 2006.

[4] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(4):509–522, 2002.

[5] Derek Bradley, Alan Brunton, Mark Fiala, and Gerhard Roth. Image-based navigation in real environments using panoramas. In *IEEE International Workshop on Haptic Audio Visual Environments and their Applications*, 2005.

[6] Ondrej Chum and Jiri Matas. Geometric hashing with local affine frames. In *CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 879–884, Washington, DC, USA, 2006. IEEE Computer Society.

[7] Alice C Crowley, James L. Parker. A representation for shape based on peaks and ridges in the difference of low-pass transform. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-6, Issue:2:156–170, 1984.

[8] Pablo d'Angelo, Kai-Uwe Behrmann, Alexandre Jenny, Sebastian Nowozin, and Ed Halley. Hugin. http://hugin.sourceforge.net/.

[9] Bojidar Dimitrov. Lens focal length and field of view. http://kmp.bdimitrov.de/technology/fov.html.

[10] Olivier Faugeras, Quang-Tuan Luong, and T. Papadopoulou. *The Geometry of Multiple Images: The Laws That Govern The Formation of Images of A Scene and Some of Their Applications.* MIT Press, Cambridge, MA, USA, 2001.

[11] M Fiala and G Roth. Automatic alignment and graph map building of panoramas. In *IEEE International Workshop on Haptic Audio Visual Environments and their Applications*, pages 103 – 108, October 2005.

[12] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Readings in computer vision: issues, problems, principles, and paradigms*, pages 726–740, 1987.

[13] G.L. Foresti, C. Micheloni, and C. Piciarelli. Detecting moving people in video streams. *Pattern Recognition Letters*, 26(14):2232 – 2243, 2005.

[14] Per-Erik Forssén and David G. Lowe. Shape descriptors for maximally stable extremal regions. In *International Conference on Computer Vision (ICCV)*, October 2007.

[15] C. Harris and M. Stephens. A combined corner and edge detection. In *Proceedings of The Fourth Alvey Vision Conference*, pages 147–151, 1988.

[16] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[17] F. Kangni and R. Laganiere. Orientation and pose recovery from spherical panoramas. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8, Oct. 2007.

[18] Florian Kangni. Rectification and pose recovery for spherical images. Master's thesis, University of Ottawa, 2007.

[19] Yan Ke and Rahul Sukthankar. PCA-SIFT: A more distinctive representation for local image descriptors. pages 506–513, 2004.

[20] P. D. Kovesi. MATLAB and Octave functions for computer vision and image processing. School of Computer Science & Software Engineering, The University of Western Australia. http://www.csse.uwa.edu.au/∼pk/research/matlabfns/.

[21] Canlin Li and Lizhuang Ma. A new framework for feature descriptor based on SIFT. *Pattern Recogn. Lett.*, 30(5):544–557, 2009.

[22] Tony Lindeberg. Scale-space theory: A basic tool for analysing structures at different scales. *Journal of Applied Statistics*, 21:224–270, 1994.

[23] Tony Lindeberg. Feature detection with automatic scale selection. *International Journal of Computer Vision*, 30:79–116, 1998.

[24] Tony Lindeberg and Jonas Gårding. Shape-adapted smoothing in estimation of 3-D shape cues from affine deformations of local 2-D brightness structure. *Image and Vision Computing*, 15(6):415 – 434, 1997.

[25] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004.

[26] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *In British Machine Vision Conference*, pages 384–393, 2002.

[27] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. *International Journal of Computer Vision*, 65:2005, 2005.

[28] Krystian Mikolajczyk and Cordelia Schmid. Indexing based on scale invariant interest points. In *In Proc. ICCV*, pages 525–531, 2001.

[29] Krystian Mikolajczyk and Cordelia Schmid. Scale & affine invariant interest point detectors. *Int. J. Comput. Vision*, 60(1):63–86, 2004.

[30] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(10):1615–1630, 2005.

[31] P. Moreels and P. Perona. Evaluation of features detectors and descriptors based on 3D objects. 73(3):263–284, July 2007.

[32] E.N. Mortensen, Hongli Deng, and L. Shapiro. A SIFT descriptor with global context. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 184–190 vol. 1, June 2005.

[33] E. Murphy Chutorian and M.M. Trivedi. N-tree disjoint-set forests for maximally stable extremal regions. 2006.

[34] Stepan Obdrzalek and Jiri Matas. *Toward Category-Level Object Recognition*, chapter Object Recognition Using Local Affine Frames on Maximally Stable Extremal Regions, pages 83–104. 2006.

[35] University of Ottawa. NAVIRE. http://www.site.uottawa.ca/research/viva/projects/ibr/.

[36] Point Grey Research. Ladybug2 camera. http://www.ptgrey.com/.

[37] Cordelia Schmid, Roger Mohr, and Christian Bauckhage. Evaluation of interest point detectors, 2000.

[38] Ali Shokoufandeh, Ivan Marsic, and Sven J. Dickinson. View-based object recognition using saliency maps, 1998.

[39] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Photo tourism: Exploring photo collections in 3d. In *SIGGRAPH Conference Proceedings*, pages 835–846, New York, NY, USA, 2006. ACM Press.

[40] Dennis Tell and Stefan Carlsson. Wide baseline point matching using affine invariants computed from intensity profiles. In *ECCV '00: Proceedings of the 6th European Conference on Computer Vision-Part I*, pages 814–828, London, UK, 2000. Springer-Verlag.

[41] P. H. S. Torr and A. Zisserman. Mlesac: A new robust estimator with application to estimating image geometry. *Computer Vision and Image Understanding*, 78:2000, 2000.

[42] A. Vedaldi. An open implementation of the SIFT detector and descriptor. Technical Report 070012, UCLA CSD, 2007.

[43] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/, 2008.

[44] L. Vincent and P. Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 13(6):583–598, Jun 1991.

[45] G. Yu and J.M. Morel. A fully affine invariant image comparison method. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2009.