Markerless Augmented Reality for Panoramic Sequences

by

Christopher R. Warrington

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
In partial fulfillment of the requirements
For the M.A.Sc. degree in
Electrical Engineering

School of Information Technology and Engineering
Faculty of Engineering
University of Ottawa

Abstract

This thesis describes the development and implementation of a system for inserting virtual objects into a panoramic image sequence. Virtual objects are inserted into the panoramic scene without the use of physical fiducial markers in the scene; placement is done through the detection of natural feature points within the panoramic imagery. An author inserts a planar image, text, or a three-dimensional model in one panorama, and the system automatically places these augmentations in a perspective-correct manner in other panoramic images near the insertion location.

In the case of planar augmentations, the system also corrects occlusion of the plane by foreground objects. By applying transparency to the planar augmentation, it's possible to give the occlusion the appearance of lying behind foreground objects. This enhances the realism of the augmentation, by giving a correct representation of relative depth.

Acknowledgements

Thank you to my supervisors, Dr. Eric Dubois and Dr. Gerhard Roth for their advice, suggestions and support throughout my thesis program. Special thanks to my two brothers and my sister and especially my parents; their support was invaluable.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

This chapter provides motivation for the work of this thesis, a description of the advantages and disadvantages of panoramic imaging with comparison to VR, a summary of prior work on similar systems, and finally the contributions of this thesis.

1.1 Motivation: Virtual Walkarounds

The creation of virtual environments has several practical applications. By generating a three-dimensional model of an environment, it is possible to provide a user with the ability to navigate through a relatively realistic environment. This environment can dynamically change with user interaction, for instance objects can be inserted or removed from the scene dynamically. These objects can also be animated, changing in position or appearance over time.

Traditionally, these environments have been fully modelled by 3D graphic artists, but this is a time-consuming process. Panoramic imaging allows us to rapidly generate an environment for navigation, but it does not allow the easy insertion of additional perspective-correct virtual content like is possible with VR. This thesis addresses the topic of the insertion of virtual objects into panoramic scenes.

With virtual reality (VR) users can exploit these modeled three-dimensional environments for a wide variety of purposes. VR is widely used in entertainment, where players in a video game can wander in an environment generated by graphic artists. Education provides another application; students can examine historical sites modeled from historical documents, or walk through virtual museums containing virtual objects or other multimedia. Other applications include real-estate, where buyers may be interested in the structure or interior layout of a building prior to construction, or in training where a student might benefit from going through an experience in some pre-defined environment.

A major problem with the generation of these VR environments is the time and cost involved with their creation. It can take teams of graphic artists days or weeks to design an environment to an acceptable level of realism. Even after the environment is created, it may still have significant differences in appearance from the real environment. The use of photography allows us to address these problems. Where it can take days to generate a realistic model of a scene, a photograph can be taken within seconds with high degrees of realism. Unfortunately, traditional planar photographs do not present the same degree of interactivity possible with a 3-D VR model; in the VR model, it is possible to move our viewing position, and look in arbitrary directions.

1.2 Comparison of VR to Panoramic Environments

The problems mentioned above can be partially addressed through the use of *panoramic imaging*. Unfortunately, panoramic imaging alone bears certain disadvantages when compared with traditional VR modeling. Some of these disadvantages include:

1. Limited movement of the viewer.
2. Limited to viewing static scenes.
3. Inability to modify scenes.

An advantage of virtual reality (VR) environments versus panoramic environments is freedom of movement for the viewer. In a VR environment, there is a great deal of geometrical information on the scene and the objects contained within. This allows the rendering of a view from a wide range of positions and orientations within the scene. A panoramic sequence limits the viewer to those positions at which panoramas were taken; this limitation is partly addressed by capturing a sufficiently dense sequence to provide a satisfactory number of viewpoints. It is possible to construct panoramic videos at very high spatial density, at the cost of considerable storage capacity. This is the approach used in [1], which uses a panoramic video to capture a scene. Ideally, we would like to capture a relatively small number of images, and use methods for interpolating views from between these positions. The interpolation of inter-frame panoramas is not a topic covered in this thesis.

The second and third items are problems addressed in this thesis. The two items are related to a single basic problem; using the base panoramic imagery, there is the limitation of displaying what was present in the scene at the moment the image was

captured by the camera. It is clear that it is highly desirable and it should be possible to modify the panoramic imagery to insert virtual objects. This is a much more difficult task than the case of VR environments, where a full geometric model of the environment exists, thereby making it relatively easy to insert a virtual object. In the case of the panoramic imagery, we do not have any (direct) geometric information regarding the scene.

When considering only a single image, it is straightforward to manually insert an artificial object into the image. However, this thesis discusses the insertion of an object into a *sequence* of panoramas. Recall that each frame in this panoramic sequence represents a viewpoint of the user: an object inserted into one frame of the sequence will (in most cases) have a different appearance in another frame, due to the change in projection between the two viewpoints. It would be possible for an author to manually insert these projections in each frame of the panoramic sequence, but this is a tedious process.

1.3 NAVIRE Project

The work for this thesis was conducted as part of the NAVIRE [2] project at the University of Ottawa. The NAVIRE project, whose names stands for *Virtual Navigation in Image-Based Representations of Real World Environments*, was created with the goal of generating a system to allow a user to navigate an image-based real-world environment. The environment is composed of a database of panoramic images, which are accessed and viewed by custom navigation and rendering software. Project participants conducted research related to many aspects of the generation, processing and display of the panoramic environments. Perceptual research was also conducted to compare the relative quality of fully virtual environments versus the panoramic image-based environments explored in

the project.

1.4 Other Virtual Tour Systems

The concept of connecting panoramas into a sequence and allowing virtual movement is not unique to this project. Prior work has been done on generating "virtual tours" by inter-connecting panoramas, which can be viewed by a user using navigation software.

1.4.1 Quicktime VR

QuickTime VR [3] is a popular software package for generating panoramic images in several formats, including cubic panoramas similar to those used for this thesis. The CubicConnector [4] add-on for QuickTime VR allows an author to inter-connect panoramas, which is used to create the virtual tour. The CubicConnector software allows users to select "hotspots" within the panorama; when the user clicks on a particular location on the panorama, an image or movie will be displayed. The system does not allow the insertion of virtual objects, interaction is performed by selecting regions of the original panoramic imagery.

1.4.2 Microsoft Research System

The goals and methods of the NAVIRE project are very similar to a system by Microsoft Research, described in [5]. In fact, the Point Grey Ladybug panoramic camera used by

the NAVIRE project was created for use by the Microsoft project. In a similar fashion to the NAVIRE project, the Microsoft system provides a method for a user to navigate through a pre-captured panoramic sequence.

In the Microsoft system, rendered panoramas are projected onto a pentagonal prism rather than a cube, but many of the techniques used are fundamentally similar to the ones from the NAVIRE project.

One key difference between the Microsoft and NAVIRE systems regards the topic of this thesis, the augmented reality component. The Microsoft system also addresses augmented reality, but does so using fiducial markers inserted into the scene. In the Microsoft system, a planar blue grid is inserted into the scene during the capture process. During the user's viewing of the scene, this planar blue grid is replaced by a planar image or video. Planar occlusions are handled by analyzing occlusions of the blue planar pattern.

For the NAVIRE system, as described in this thesis, augmented reality is done without the use of any fiducial markers inserted into the scene. We detect texture inherent to the scenes using SIFT or PCA-SIFT features, and do not require the insertion of any objects into the physical scene. In the case of planar augmentations, we also handle occlusions by detecting changes in the augmentation region between frames. Finally, in the NAVIRE system we also allow the insertion of three-dimensional augmentations.

Both the NAVIRE and Microsoft systems introduce additional augmentations placed into the scene without the use of augmented reality techniques. Both systems insert virtual direction indicators into the scene, which act as a cue to which directions of movement are permitted. Both systems also provide an overhead map, which aids users in determining their present location and simplifies the users' navigation task. The

Microsoft system also provides for localized audio, by attenuating the volume of audio clips according to the panoramas' positions from a designated source point.

1.4.3 Systems with Range Sensors

An important class of virtual tour generating system are those that do not use purely image-based methods for generating their environments. One example of such a system is given in the work of Asai et al [6]. In this system, a virtual environment is created by a combination of laser rangefinding sensors as well as a Ladybug panoramic camera. In this case, the environment is recovered as a 3-dimensional model, and the image data from the panoramic camera is used to apply a texture to the modelled environment.

The generation of three-dimensional models of scenes for the creation of virtual tours is another common approach in literature. However, this thesis discusses the generation of augmented reality for a fully-optical system, where no direct range data is available from an external sensor or prior modelling of the scene.

1.4.4 Shortcomings of Current Systems

This thesis addresses a key shortcoming from the previous system; the insertion of virtual objects into the scene without requiring the use of fiducial markers during the panorama capture process. We wish to be able to augment panoramic scenes with virtual objects, without requiring any modification to the environment during capture time.

1.5 Contributions of the Thesis

The goal of this thesis is to create a system which enables the insertion of virtual objects into a single panoramic image. The system will automatically propagate these objects into nearby panoramas in a perspective-correct fashion.

The contributions of the thesis are to enable the insertion of augmentations into a panoramic scene. Most pre-existing work on augmented reality relates to the insertion of virtual objects into traditional scenes, rather than the panoramas used here. We also present a method of handling occlusions of these inserted augmentations which greatly improves their appearance. The work in this thesis is unique in its addressing of performing marker-free augmented reality in panoramic scenes, while simultaneously handling occlusion.

The work for this thesis was presented in poster and demo form at International Symposium on Mixed and Augmented Reality (ISMAR 2006) [7]. A demo of the work was also presented at the Intelligent Interactive Learning Object Repositories conference (I^2LOR 2006).

1.6 Thesis Organization

This thesis begins with an overview of panoramic imaging and augmented reality. Next we present background information regarding planar homographies, a concept used extensively used in the system developed for this thesis. This is followed by a presentation of the full virtual navigation system that was developed, highlighting the component addressed by this thesis. The augmentation method is explained in more depth, and

results are presented. Finally, potential future work and conclusions are given.

Chapter 2

Planar Homographies

The principal mathematical tool used in this thesis is the planar homography. This concept is not a novel development of this thesis, and can be found described in many sources such as [8]. However, since it is used extensively in the thesis, it is presented in detail in this chapter.

The purpose of the planar homography is to specify a point-by-point mapping between pixel locations in one image to corresponding pixels in a second image. This mapping is done using an assumption of *planarity* between the two images: points on the two images are mapped under the assumption that the object being imaged is planar.

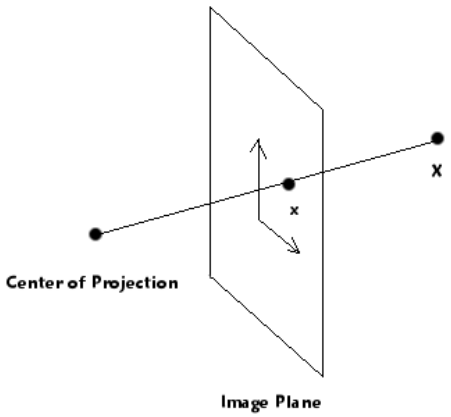


Figure 2.1: Central projection model

2.1 Central Projection Model

The equations shown in this section are based on the concept of *central projection*. A point on the image plane $\mathbf{x} = \begin{bmatrix} x_s & x_t & 1 \end{bmatrix}^T$, is formed by the projection of a world point $\mathbf{X} = \begin{bmatrix} X_x & X_y & X_z & 1 \end{bmatrix}^T$ onto the image plane by a ray from the world point to the center of projection of the camera. This is shown in Figure 2.1.

Any coordinate $\mathbf{X}$ laying in front of the camera will project to a unique point $\mathbf{x}$ on the image plane. However, the inverse projection is non-unique. A image point $\mathbf{x}$ may correspond to any point along the ray extending out from the center of projection through the image point. This presents an ambiguity, which is represented by the factor λ in equation 2.1. The image coordinates $\mathbf{x}$ and the world coordinates $\mathbf{X}$ are in homogenous form.

$$\begin{pmatrix} x_s \\ x_t \\ 1 \end{pmatrix} = \lambda \mathbf{P} \begin{pmatrix} X_x \\ X_y \\ X_z \\ 1 \end{pmatrix} \tag{2.1}$$

$$\mathbf{P} = \mathbf{K}\,[\mathbf{R}|\mathbf{t}] \tag{2.2}$$

Here, P is the projection matrix corresponding to the action of the camera, and is expanded in equation 2.2. $\mathbf{R}$ is a rotation matrix giving the rotational orientation of the camera, and $\mathbf{t}$ is a translation vector giving the camera's position. $\mathbf{K}$ is formed by the intrinsic parameters of the camera. The intrinsic parameters of the camera describe internal parameters of the camera that affect the projection onto the image plane, including the scaling factors along the x and y axes, any skewing of the alignment of the x and y axes of the camera, and the center of projection of the camera. For further details see [8].

2.2 Problem Statement

Given two images of a single scene from two viewpoints, we assume the existence of pairs of pixels, one from each image, that correspond to the same physical location in the scene. These matching pixels, referred to as pixel correspondences, are denoted as x_{1i} and x_{2i}. These pixel correspondences are represented in homogenous coordinates of the form $\begin{bmatrix} u_{mi} & v_{mi} & 1 \end{bmatrix}^T$. If we assume that pixel correspondences are for image points lying on a planar object, we obtain a simple linear relationship between x_{1i} and x_{2i}. The following section will show that their relationship is given as $x_{2i} = \mathrm{H}\ x_{1i}$, with H being a

3×3 matrix. Hartley and Zisserman [8] discuss the derivation of this linear relationship between the image pixels. To see this relationship, consider equation 2.3; we begin by assuming no planarity to the object. These equations illustrate how we project the three-dimensional spatial coordinates of the target, X, into the two-dimensional coordinates of the image plane. Matrix M represents rigid transformation, ie. rotation and translation, of the target in the world from a reference position. P represents the effect of the pinhole projection model.

$$\mathbf{x} = \mathbf{PMX}$$

$$\begin{pmatrix} x'_x \\ x'_y \\ x'_w \end{pmatrix} = \begin{pmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{pmatrix} \begin{pmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X_x \\ X_y \\ X_z \\ 1 \end{pmatrix} \quad (2.3)$$

The problem is greatly simplified when we introduce the additional constraint of a planar object. We have the freedom of selecting an arbitrary base coordinate system for the object. With a planar object, it is natural to select that all points on the object lie along the zero coordinate of one of the axes, ie. parallel to the plane formed by the other two axes. If we take the reference position such that the target is parallel to the XY-plane, we can set the X_z term to zero. This simplifies these equations to 2.4. From this, we see that we obtain a 3×3 matrix; the homography H.

$$
\begin{aligned}
\begin{pmatrix} x'_x \\ x'_y \\ x'_w \end{pmatrix} &= \begin{pmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{pmatrix} \begin{pmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X_x \\ X_y \\ 0 \\ 1 \end{pmatrix} \\
&= \begin{pmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{pmatrix} \begin{pmatrix} m_{11} & m_{12} & 0 & m_{14} \\ m_{21} & m_{22} & 0 & m_{24} \\ m_{31} & m_{32} & 0 & m_{34} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X_x \\ X_y \\ 0 \\ 1 \end{pmatrix} \\
&= \begin{pmatrix} p'_{11} & p'_{12} & 0 & p'_{14} \\ p'_{21} & p'_{22} & 0 & p'_{24} \\ p'_{31} & p'_{32} & 0 & p'_{34} \end{pmatrix} \begin{pmatrix} X_x \\ X_y \\ 0 \\ 1 \end{pmatrix} \\
&= \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} \begin{pmatrix} X_x \\ X_y \\ 1 \end{pmatrix} \\
&= \mathbf{HX} \\
\begin{pmatrix} x_x \\ x_y \\ 1 \end{pmatrix} &= \lambda \mathbf{HX}
\end{aligned} \tag{2.4}
$$

Each pixel correspondence provides us with two equations, as shown in equation 2.5. Four pixel correspondences provide us with eight equations, which we can use to determine the value of $\mathbf{H}$ up to the multiplicative constant λ. By placing the eight equations in a single homogenous matrix, we can employ singular value decomposition or other techniques to solve for the parameters of h. This follows the DLT algorithm described in [8].

$$\begin{aligned} x_x &= \lambda(h_{11}X_x + h_{12}X_y + h_{13}) \\ x_y &= \lambda(h_{21}X_x + h_{22}X_y + h_{23}) \\ \text{where}\lambda &= \frac{1}{h_{31}X_x + h_{32}X_y + h_{33}} \end{aligned} \tag{2.5}$$

2.3 Chained Homographies

The homography discussed above provides a relation between coordinates in the rectangular reference pattern, $\mathbf{X}$, and image pixel points $\mathbf{x}$. However, we want to be able to track points within the planar region between perspectively-projected images.

Consider two images of a planar region, as shown in Figure 2.2, where for each image there will have a homography relating the reference position of the plane to the pixels within the image. Thus, the relationship from one image to the other is as given in equation 2.6.

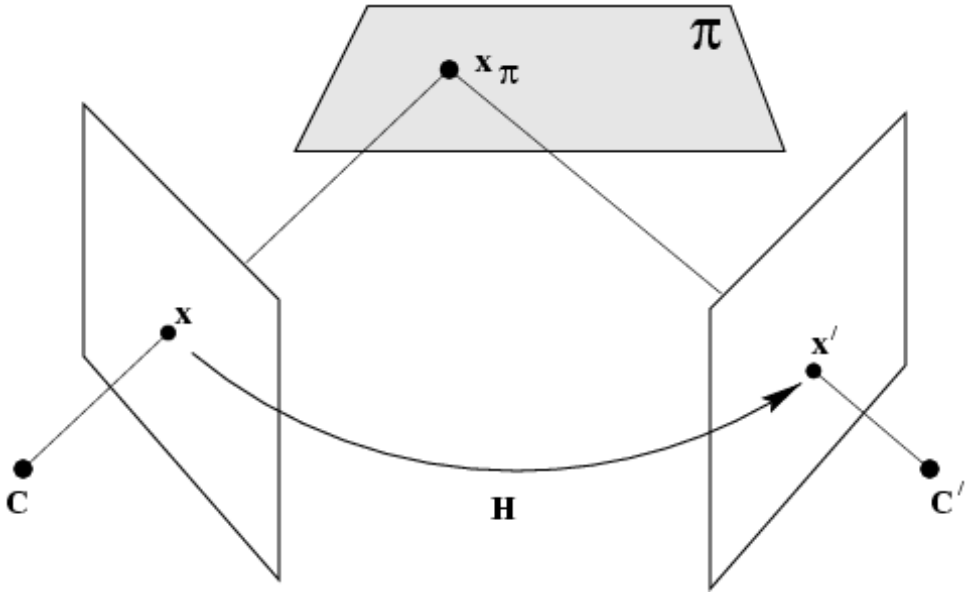


Figure 2.2: Homography relating two views of one plane

$$
\begin{aligned}
\mathbf{x_1} &= \lambda_1 \mathbf{H_1} \mathbf{x_r} \\
\mathbf{x_2} &= \lambda_2 \mathbf{H_2} \mathbf{x_r} \\
\mathbf{x_3} &= \lambda_3 \mathbf{H_3} \mathbf{x_r} \\
\\
\mathbf{x_r} &= \begin{pmatrix} x_r \\ y_r \\ 1 \end{pmatrix} \\
\\
\mathbf{x_2} &= \lambda_a \mathbf{H_2} \mathbf{H_1^{-1}} \mathbf{x_1} \\
\mathbf{x_3} &= \lambda_b \mathbf{H_3} \mathbf{H_2^{-1}} \mathbf{x_2}
\end{aligned}
\tag{2.6}
$$

Equation 2.6 illustrates how we can chain together homographies to obtain the pixel positions for points in the plane. A potential problem with the use of chained homographies is the accumulation of errors. In the example from above, we note that any errors

in the homography $\mathbf{H_1}$ will influence the values for the pixels $\mathbf{x_2}$, and in turn $\mathbf{x_3}$ or any further pixel positions calculated along the chain. With a chain of homographies, we need to check if the current composite homography (generated by the chain of homographies) appears sensible. If the homography does not meet this test, then one possible resolution is to re-compute the homography using a different chain. For instance, we can recalculate the homography with reference to a different position: in the example from equation 2.6, we can calculate $\mathbf{x_3}$ with reference to $\mathbf{x_1}$ rather than $\mathbf{x_2}$. For chains longer than the length two example from 2.6, we can match against any previous member of the chain rather than the immediate neighbour to obtain a correct match.

2.4 The RANSAC Algorithm

The calculation of the homography is prone to the effect of *outliers* from the presence of mismatches. In certain cases, one feature may be matched to a similar feature that does not represent the same physical location. The mismatched feature may be located in a substantially different location than the "correct" feature, which can introduce significant error into the calculation. This phenomenon presents a situation where certain matches introduce error of a non-Gaussian nature [8]. We can improve the homography by identifying and ignoring the outlier matches.

A commonly-used approach for dealing with these outlier matches is RANSAC (random sampling consensus) [9]. This approach attempts to iteratively separate the input data into two sets: inliers and outliers. Considering the application of RANSAC to homography calculation, which requires at least four sets of point correspondences. In each iteration of the RANSAC, we randomly select four of the correspondences and calculate a homography H_1. The remaining unused correspondences, which will be referred to as

the *test correspondences*, are used to test the validity of this homography.

Each of the test correspondences provides a match between a pixel position x_{1i} in one image to a pixel position x_{2i} in a second image while the RANSAC-generated homography provides a mapping from x_{1i} to a point x'_{2i}. We iteratively step through all the test correspondences, and calculate their respective x'_{2i} values using the x_{1i} value and the H_1 homography. We are attempting to determine a *support* value for the homography; an indication of the correctness of the homography. For each correspondence, if x'_{2i} lies sufficiently close to the correspondence's x_{2i} value, we mark this correspondence as supporting our homography. If $x_2 i'$ is outside a distance threshold, the correspondence does *not* support the homography. The set of supporting correspondences, which is said to comprise the *inlier* set, and the non-supporting correspondences comprises the *outlier* set.

After stepping through all the correspondences in the test set, we count the number of supporting correspondences. If the percentage of supporting correspondences exceeds some threshold of acceptability, we consider the inlier and outlier sets to have been properly assigned. We discard all correspondences from the outlier set, and recalculate a homography using only those correspondences from the inlier set.

After having computed H_1, we can optionally perform another reclassification step, determining inliers and outliers and calculating a new homography, H_2. This process can be repeated until the similarity between H_n and H_{n+1} falls below some convergence threshold. One similarity metric suggested in [8] is the number of inliers in the set.

2.5 Summary

This chapter has described planar homographies, and the RANSAC algorithm. Planar homographies generated from point correspondences by RANSAC is the primary tool used for placing augmentations in this thesis.

Chapter 3

Background on Panoramic Imaging

The following chapter presents background on the advantages of panoramic imaging, augmented reality, the registration of virtual objects in photographic scenes, and markerless versus marker-based augmented reality.

3.1 Panoramic Imaging

With the standard pinhole camera model for a traditional photographic image, the image represents light passing through a rectangular image *plane* placed between the center of projection of the camera and the subject. The view is limited to the angles subtended by the rectangular plane. With a panoramic image, we use a projection allowing a wider-angle view of the scene. Some commonly-used projections are shown in Figure 3.1. In each case, the center of the projection lies in the center of the geometric object.

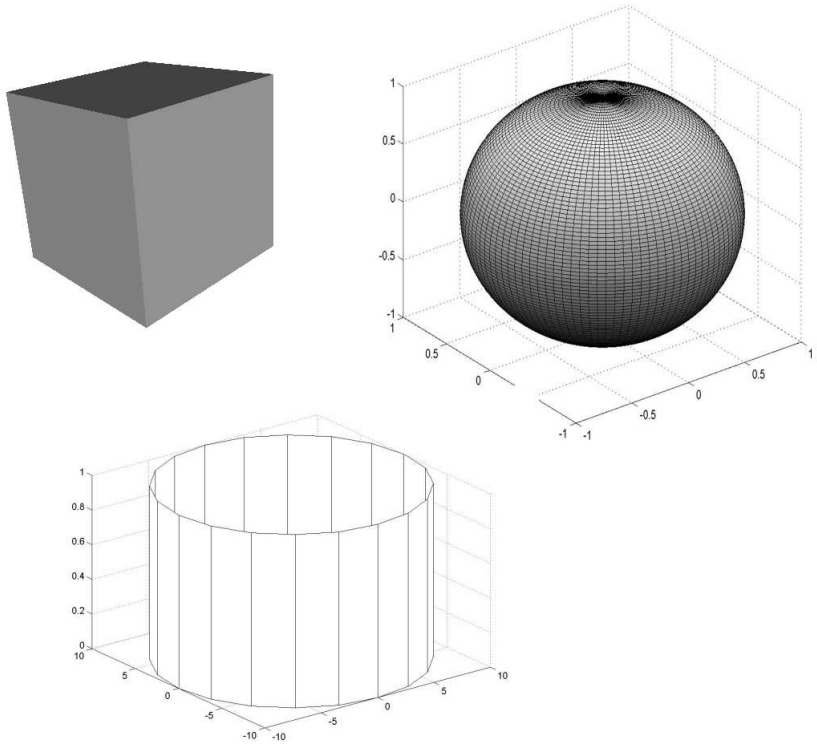


Figure 3.1: Cubic, spherical and cylindrical projections

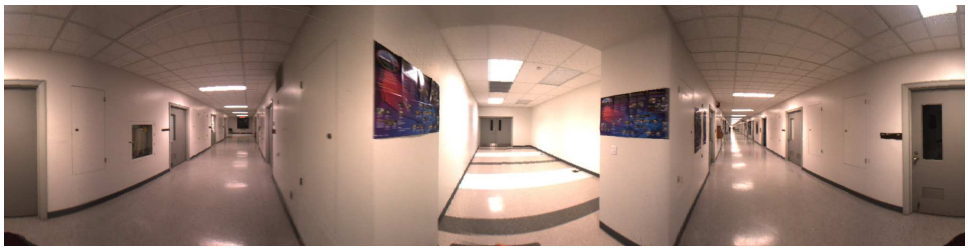


Figure 3.2: Warping of linear edges in cylindrical projection

In all cases, the panoramic images are obtained by considering the projections of the real-world environment onto the surface of a geometric object. If the panoramic image is directly observed, we may observe distortion of the objects from the scene. In the case of cylindrical panoramas, straight lines from the real-world scene become curved in the cylindrical panorama as shown in 3.2. In the case of polyhedral projections, direct viewing of the panoramic imagery will show seams at the boundaries of the polyhedron. For example, Figure 3.3 shows a cubic panorama in which the scene is projected onto six square sides around the center of projection, with each side being the face of a cube. Viewing the concatenated faces of the cube, the edges of each face are apparent. However, within each individual face there is no distortion of the image.

The above-described distortions and seams come from a mismatch between the panoramic projection's geometry and the planar surface that is used for viewing. The solution is to use *viewing software* that will project a partial portion of the panorama onto the planar rectangular viewing surface of the screen. The projection of geometric objects onto planar viewports can be done using any model rendering software; the OpenGL [10] rendering library is supported on a wide range of computing platforms, and is used as the rendering engine for this thesis.

The cubic panoramic projection is the projection used in this thesis. The cubic projection offers the advantage of simplicity for the rendering engine; the six-sized cube

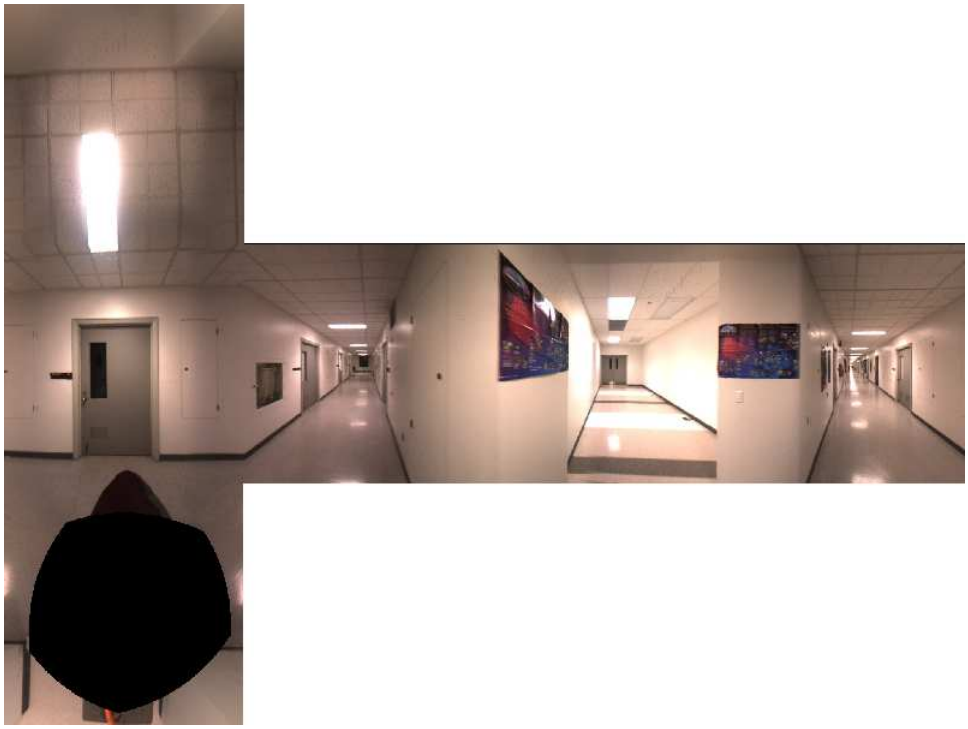


Figure 3.3: Seams in cubic projection

is rendered with the panorama's six faces as textures.

A viewer, looking from the center of the cube outwards is able to view the scene in any arbitrary viewing direction, though still from a fixed viewing position. By using a sequence of panoramas, it is possible to recreate views of the scene from several viewing positions and with freedom of viewing direction.

3.2 Overview of Augmented Reality

The research for this thesis is strongly related to research in the area of *augmented reality*. This term refers to the insertion of virtual objects into still images or video, by identifying camera pose or object positions through use of image processing techniques.

In [11], Azuma defines a set of criteria that characterize a typical augmented reality system. These are listed below.

1. Combines the real and virtual.
2. Registered properly in 3-D space.
3. Interactive in real-time.

The first two items clearly apply to the system designed for this thesis. The goal of the system is to insert rendered virtual objects into a panoramic scene, in a perspectively-correct fashion. Since the system involves the creation of a combination of real imagery and virtual objects, the first criterion is satisfied. The second criterion also follows, since the goal of the system is to place the objects such that they appear in-place three

dimensionally within the scene. For instance, a planar poster inserted into a scene might appear to be affixed to a real wall within the scene. As another example, it might be desirable to have a three-dimensional tree appear as a naturally-occurring object within the scene.

The third requirement, requiring real-time interaction needs more explanation. The goal of this system is to insert virtual objects into a pre-captured panoramic sequence, as opposed to a live-captured video. The rendered environments are interactive in real-time, but the registration of the virtual and real environments is performed in a pre-viewing authoring stage. Because of the existence of this *authoring* stage, it is acceptable that the system run at rates slower than real-time. However, it is *not* acceptable for the system to require an arbitrary amount of time. If the propagation of augmentations is performed more slowly than an author manually inserting the augmentation in each frame, the author will become unduly frustrated with the process.

In the case of planar augmentations, the work for this thesis is similar to work by Ferrari et al [12] in markerless augmented reality, whereby virtual signs are inserted into a photographic scene. The primary difference between this thesis' work and that of the virtual advertisement system is the use of panoramic imagery. As discussed in Section 6, the system also address the issue of occlusion of planar augmentations as well as the insertion of three-dimensional models into the scene.

3.3 Registration of Virtual Objects

The goal of augmented reality systems is to insert virtual objects into real-world photographic imagery. For this section, I limit the discussion to traditional planar image-plane

cameras, rather than panoramic imaging.

To place the virtual object in a perspectively-correct fashion, one of the following two conditions must exist:

1. To identify the pose of the virtual object relative to the camera position.
2. To identify a projection matrix which allows the object to be rendered in a perspectively-correct fashion.

Conceptually, the first approach is the most intuitive. If we are able to recover the pose of the object relative to the camera's position, we are clearly able to render the object correctly in the scene. However, in certain cases, such as planar objects, we may not require a full three-dimensional pose recovery in order to render the object correctly. It is possible to generate a projective matrix that allows the virtual object to be inserted with an appropriate perspective-correct appearance, without requiring full pose knowledge of the inserted object.

3.4 Marker-based Versus Markerless Systems

There are two principal types of augmented reality system. Those based on the insertion of physical markers into the scene to register augmentations, and those that use natural feature-points as a basis. Both methods are described below.

3.4.1 Marker-based Augmented Reality

To insert a virtual object into a photographic scene, a common approach is to place a physical object into the scene at or near the desired position of the virtual object. By detecting and extracting the pose of this known object, it is possible to obtain a camera position relative to the object location. Using the transformation between the camera and the object, a virtual object can be rendered to appear over-top of the real-world marker.

This approach, of inserting a physical object into the photographed scene in order to overlay a virtual object nearby is typically referred to as a *marker-based system*. This is the approach used by the ARToolkit [13] [14] and ARTag [15] systems. In these two systems, planar black and white images are used as fiducial markers. The pose of the marker is extracted from live video, and a virtual model associated with the marker is superimposed onto the video in a perspective-correct fashion. An example of this type of system is given in Figure 3.4.

A major advantage of a marker-based system is that the markers can be designed in such a fashion to ensure they remain relatively well-detected under significant pose and lighting changes. The obvious disadvantage of marker-based systems is that physical modification of the scene is required during the capture process.

3.4.2 Markerless Augmented Reality

The alternative to marker-based systems are the *markerless* augmented reality systems. Rather than extracting camera position by analyzing the pose of a known object in the

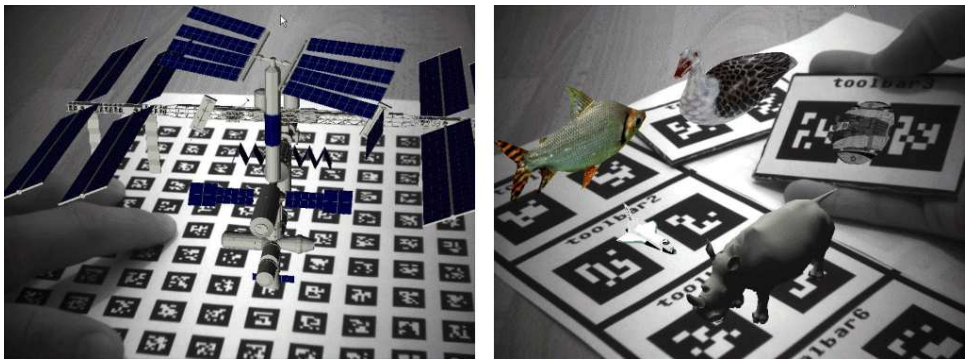


Figure 3.4: ARTag markers with registered objects

scene, camera pose may be extracted from naturally occurring features within the scene. Ideally, a virtual object can be inserted in a position relative to objects within the scene without the need for inserting a physical fiducial marker as a reference point.

Feature detectors, such as the Harris corner detector [16] or Lowe's SIFT detector [17] identify unique points within an image that can be robustly computed in the correct position in other images. The Harris detector attempts to identify areas of high gradient changes in both horizontal and vertical directions; these sections will often represent a corner in an image (for instance, the corner of a window). As such, the Harris detector and other feature detectors are frequently referred to as *corner detectors*. SIFT detects features with a high magnitude response to convolution by a filter, where the filter is a difference of Gaussian filters of identical size but varying scale. The SIFT features have advantages over Harris in terms of invariance to scale, and improved invariance to rotations, both parallel and out of the camera plane [18].

Feature-point detectors generate a list of features within an image, and associate with each feature point a *descriptor* providing characteristic values for the feature. By comparing these descriptor lists between two images, we attempt to classify whether two feature points (one from each image) actually correspond to the same point in space.

This is done by computing some type of distance metric between the feature points in the two images. In the case of the SIFT features, a commonly used metric is the Euclidean distance between the descriptor vectors. From the list of features from each image, we identify the pair with the least Euclidean distance. If the distance associated with this pair is below some threshold, we declare that the two points represent the same spatial position, thereby forming a *correspondence* between the two points.

Markerless augmented reality systems are a concept that have been applied to other systems in the past; principally to planar image or video sequences. Some systems of this class include work by Skrypnyk and Lowe [19], which tracks SIFT features to establish a model of the real-world environment, and simultaneously calculate camera parameters. The model and parameters are used to insert virtual objects into the photographic scene. Li, Laganiere and Roth [20] present a system using trifocal tensors to place augmented-reality augmentations by tracking natural feature points, with an initial setup process using three images of the placement region. Ferrari [12] which tracks planar patches undergoing affine transformation in video sequences.

3.4.3 Application to Thesis

For this thesis, we use a markerless augmented reality approach. This allows us to add augmentations to panoramic scenes, without requiring any physical modification of the scene.

An author enters an augmentation into a single scene. We obtain high-level features of the scene as described in Section 3.4.2. By tracking movement of these features between panoramas in the sequence, we are able to calculate the correct placement of augmentations in other panoramas, without the author needing to place the augmenta-

tion manually in each scene.

Chapter 4

System Overview

A basic overview of the panoramic sequence generation process is given in Figure 4.1. This thesis is principally concerned with the augmentation authoring component of the process, but a brief overview of the entire system will be provided.

4.1 Camera Capture

For image capture, we employ the Point Grey Ladybug panoramic camera. The Ladybug is composed of six wide-angle CCD cameras, with five placed pointing in a radial, horizontal circle and the sixth pointing straight upwards. The Ladybug provides images in a raw, Bayer-mosaic bitmap format, which is transmitted via a FireWire connection to a capture laptop. An image of the camera is shown in Figure 4.2. An image produced by the camera (after Bayer demosaicing) is shown in Figure 4.3.

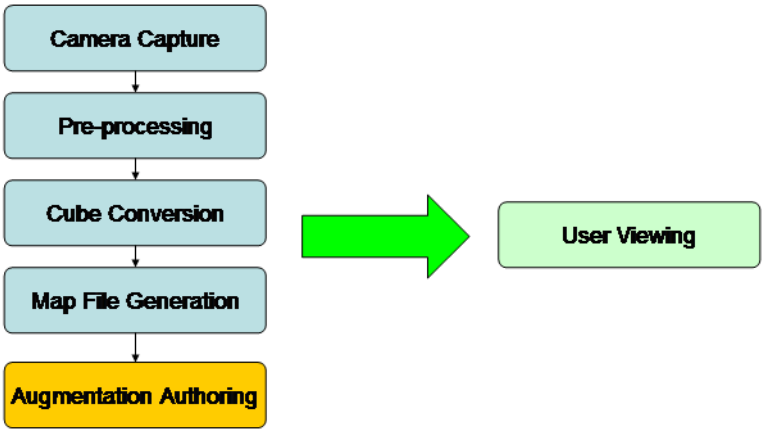


Figure 4.1: Overview of panoramic sequence generation

Figure 4.2: Point Grey Ladybug

Figure 4.3: Raw Ladybug image

To facilitate identifying camera position in outdoor scenes, the capture software also records the camera position using an attached GPS receiver. The position information provided by the GPS is of some use for outdoor scenes, where inter-frame movement of the camera is frequently significant, but is less useful in indoor scenes where inter-frame movement falls below the GPS resolution. In this thesis, no GPS position information was employed for placing augmentations; a purely image-based approach was employed.

Some important features of the raw image files can be noted. First, it is important to note that the six images from the camera do *not* correspond to the six faces of the cube, the panoramic representation used in this thesis. The conversion between the raw image format and the cubic panorama representation is performed in a subsequent stage. The raw images contain significant radial distortion, and there is considerable overlap between adjacent camera images [21].

4.2 Pre-processing

Prior to converting the raw images to cubes, we can perform some pre-processing of the image. The raw Bayer mosaic images are converted into a three-colour RGB image, denoising filters may be applied, and any other necessary pre-processing can be done prior to the conversion into the cubic panorama format.

4.3 Cube Generation

We convert the raw images from the Ladybug into a cubic panorama projection. Point Grey provides software for rendering their camera images onto a polygonal approximation of a sphere. By rendering this sphere onto square planar projections in the six face directions (forward, left, back, right, up and down) we can obtain the cubic projection and generate the six textures representing the faces of the cube.

Viewed from the center, the seams at the intra-side boundaries are not visible. Since some portions of the cubic panorama are generated using pixel values from multiple camera images, there may be some "ghosting" of the images in these areas; a solution for this is addressed by other work in the project, and will not be discussed in this thesis. Further details can be found in [22].

A major advantage of the cubic projection is the simplicity of the geometry for rendering. The model to be rendered for display is simply six squares, each with a separate texture; the rendering process is very fast. Note that there is no need for the cube faces to be aligned with the physical positions of the camera: we are able to apply a rotation to the cube faces captured from the spherical image prior to the projection of the cube faces. This allows us to align the cubes in a sequence; for instance we can place the center of the front face along the axis of movement of the camera (as the direction of movement need not correspond to the center of one of the Ladybug's images). If the cubes are aligned in this, or a similar fashion, we can improve the results from the augmented reality component; this will be described in a later section.

An example of a generated cube is shown in Figure 4.4. The cube is stored as six concatenated colour RGB bitmap files.

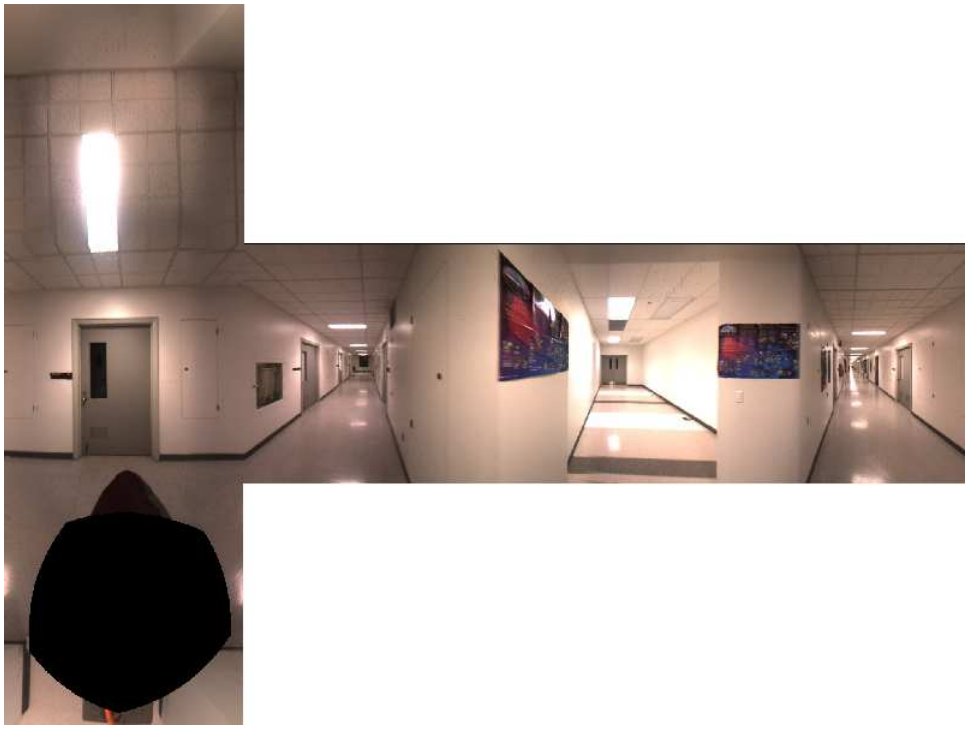


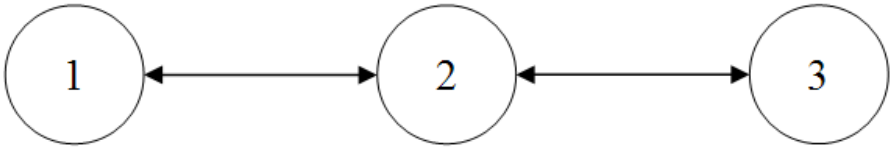
Figure 4.4: Cube File

After the cubes are generated, we are able to perform feature point extraction on the images. Doing feature point extraction at this point, rather than during the actual augmentation authoring, allows us to improve the computation times in the authoring stage. This results in reduced waiting for the author, and a corresponding reduction in frustration in the authoring process.

4.4 Map File Generation

We also generate a *map file* to associate with the cubic panoramas. This map file describe a coarse geometric interconnection between the panoramas in the sequence. These interconnections can be established either manually, or by an automated system. The map file is composed of a list of panorama nodes. Each node represents a panorama in the sequence, and is assigned two elements of data; a two-dimensional Cartesian position, and a list of neighbours. Each neighour is assigned a one-dimensional angle giving a direction from the node towards the neighbour. An example of a small three-member map is given in Figure 4.5. Each circle represents a node in the graph, and sample data is given for the center node.

For this thesis, it's assumed that the position and direction information encoded into the map file is too coarse to be used as an aid for the augmented reality process. The thesis system employs only the list of neighbours, which is used as a basis for propagating augmentations.




```
<cube id="2" angle="0.0" filename="frame2.cube">
        <neighbour id="1" angle="180" />
        <neighbour id="3" angle="0" />
        <mapCoords map="0" x="420.0" y="300.0" />
        <gpsData lat="N45'25.1" long="W75'42.3" />
</cube>
```

Figure 4.5: Sample Map File

4.4.1 Augmentation Authoring

The topic of augmentation authoring is the main focus of this thesis, and will be described in detail in following chapters.

Chapter 5

Placement of Virtual Augmentations

This chapter describes the authoring process for the augmentation. The goal of this process is to add the appropriate augmentations to the panoramic image sequence. The authoring process for the augmentations proceeds in the following steps:

1. User navigates to a panorama in which to insert the augmentation.
2. User selects the augmentation to place.
3. User selects a textured, planar region to anchor the augmentation in the panoramic scene.
4. User places the augmentation relative to the planar region.
5. System calculates image feature points within the planar region.
6. System matches feature points in the selected region versus adjacent panorama feature points to generate homographies.

7. (For planar augmentations) Detect occlusion of the planar region by foreground objects.

8. Save a descriptor for the augmentation.

9. Repeat the matching recursively for any attached panoramas.

After we complete this process, we will have taken the augmentation manually placed by the author, and propagated it automatically (in perspective-correct fashion), to panoramas near to the insertion point. Each of these steps are described in more detail in this section. Before proceeding, we note conventions regarding coordinate values used in this thesis.

5.1 Coordinate Frames

The image pixels obtained from the authoring software's input system are in the ranges $s_{pix} = [0, im_{width} - 1]$, $t_{pix} = [0, im_{height} - 1]$. im_{width} is the maximum pixel width of the image, e.g. 640 in a 640x480 window. im_{height} is defined in an analogous manner. Image coordinates in this section are typically specified in the standardized ranges $s = [-1, 1]$ and $t = [-1, 1]$, with the original s_{pix} and t_{pix} being mapped uniformly to the corresponding s and t ranges: $s_{pix} = 0$ corresponds to $s = 0$, and $s_{pix} = im_{width} - 1$ corresponds to $s = 1$. t is calculated in an identical manner. This results in a potentially non-uniform scale in s and t where im_{height} is not equal to im_{width}, but the use of this rescaling simplified implementation.

World-frame coordinates are specified according to the model for the panoramic cube. In this model, the panorama is viewed from the origin (0,0,0) and the sides of

the panoramic cube are arbitrarily chosen to be 200 units wide. The ground plane of the panorama is taken as the xz-plane, with the up vector is along the y axis. The author's view is parameterized by four values: θ, ϕ and the fields of view $\mathbf{fov_x}$ and $\mathbf{fov_y}$. $\mathbf{fov_y}$ represents the angle subtended in the vertical direction by the window, $\mathbf{fov_x}$ is then found by the aspect ratio of the window. In normal viewing, the author will typically only modify θ and ϕ by motion of the mouse, but changes to the field of view are also permitted; in certain cases, enlarging the field of view may simplify certain selection operations.

5.2 Navigation

To navigate to a panorama to augment, the author uses software similar to the one employed by the end user. The author views a planar-rendered version of the panoramic environment, and can move between panoramas using a point-and-click interface.

To improve the success of the augmentation process, it is best to place the augmentation in a highly visible planar region. More specifically, the planar region should have the largest possible field of view, to increase the number of detected feature points. Even with the occlusion detection feature enabled, the planar region must not be occluded by a foreground object in the initial placement panorama.

A screenshot from the authoring system is shown in Figure 5.1.

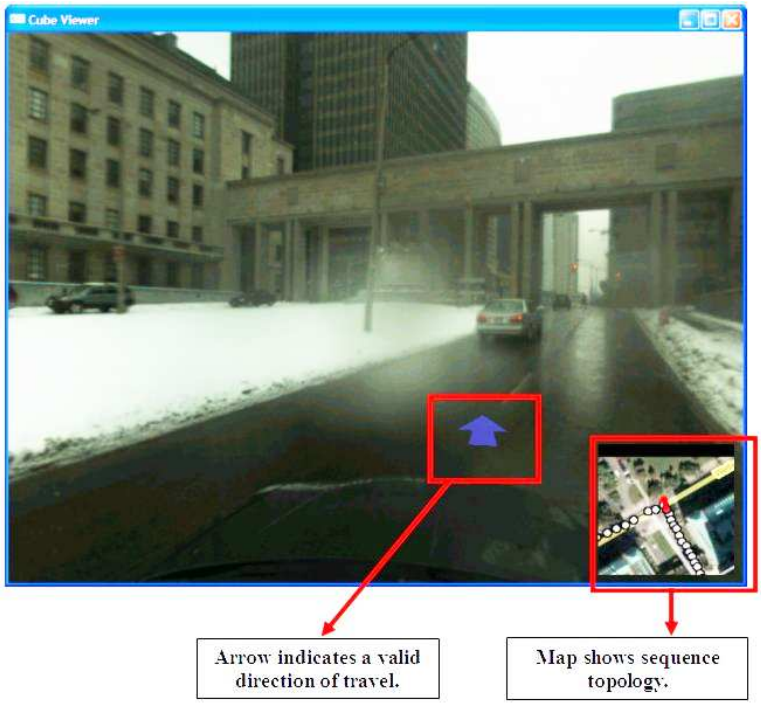


Figure 5.1: User navigation

5.3 Augmentation Selection

The author must select the type of augmentation to be inserted into the scene. The system allows two augmentation types: planar augmentations and model augmentations. The author chooses the augmentation by specifying a filename, and the nature of the augmentation is determined by the filename and extension.

Planar augmentations take the form of text or images that are inserted into the scene and are made to appear as though they are parallel to the planar region. The image, or bitmap image of text, is specified using a filename of extension BMP (a Windows bitmap file). In the current system, text is treated in an identical fashion to planar images. The user can specify an optional *refresh* value to update the planar texture at a regular rate; this is useful if the planar augmentation is an image taken from a remote

source, for instance imagery from a camera updated at a relatively slow rate. This allows the insertion of video textures.

The second type of augmentation is 3-D models. The system currently supports models of the 3D Studio (3DS) model type, which are loaded and rendered using a freely available 3D model library.

5.3.1 Planar Region Selection

To place the augmentation, the author must identify a *planar* and *textured* surface on which to anchor the augmentation. Identification of the planar region is done by selecting four corners of a rectangular (scene-wise) region: in the current implementation, only rectangular matching regions are supported. For instance, a user can select the corners of a poster or wall, which is rectangular in the physical scene, but not rectangular in the image. The planar region selection can be expanded in a straightforward manner to a polygonal planar region of arbitrary size.

In the case where the augmentations are planar, the rectangular area identified will typically be larger than the dimensions of the augmentation. This allows for detection of a greater number of feature points within the selected region, which will assist with the feature matching process. An increase in the number of feature matches increases the probability of success along with the stability of the homographies that are calculated. Some degree of non-planarity in the selected region is permissible; if most of the detected features lie within a plane the matching is likely to succeed. Feature points laying off the plane are likely to be discarded by the RANSAC process. This simplifies the identification of a planar area in certain cases, for instance if there is a large planar wall with a foreground object present, it is possible to select the entire wall as the anchoring

region.

5.3.2 User Places the Augmentation

After the planar region has been selected, the author places the augmentation relative to the planar region. The method for doing so differs slightly between the planar and three-dimensional model cases. In both cases, the user selects a rectangle (in reference to the scene object, not the image) to size and place the augmentation on the planar region. The region is rectangular in the real-world reference frame, but in the general case the region selected on-screen will be a non-rectangular quadrilateral.

5.3.3 Placement of Planar Augmentations

For planar augmentations, the augmentation will be scaled to fit the author's rectangle. This is done by calculating a homography between a front-facing reference model of the augmentation and the pose given by the augmentation region. In the front-facing model, the corners are assigned the coordinates [-0.5,-0.5], [0.5,-0.5], [0.5,0.5] and [-0.5,0.5]. These coordinates are in the standard coordinate frame specified at the start of this section. The positions of the author's selected rectangle are obtained by converting from image pixel coordinates to the standard [-1,1] range bounds. A homography, H_{ref}^{aug} is calculated to give the conversion from reference image points to corresponding points on the current view.

Note that the homography H_{ref}^{aug} is only meaningful for matching between the reference rectangle and the view of the author *at the time of augmentation placement*. The s and

t coordinates of the augmentation provide a homography that is intrinsically tied to the θ and ϕ values of their view, as well as the dimensions of the window and current field of view. To give a representation of the augmentation that is independent of these factors, we back-project the planar region spanned by the augmentation to the surface of the panoramic cube. Figure 5.2 illustrates the back-projection process.

To project the augmentation, we start by defining a unit vector $\mathbf{v_1}$, which is a ray from the center of projection to a point on the augmentation. The reference frame associated with $\mathbf{v_1}$ is aligned with the image plane during the augmentation process; the x-axis points to the right of the image, the y-axis points to the top of the image, and the z-axis points towards the viewer. The calculation of the unit vector $\mathbf{v_1}$ from the image coordinates $\mathbf{x_1}$ is given in equation 5.1. The a factor represents the image aspect ratio, and *fovy* represents the vertical field of view in radians.

$$\begin{aligned}
\begin{pmatrix} v'_{1x} \\ v'_{1y} \\ v'_{1z} \end{pmatrix} &= \begin{pmatrix} x_{1x} a \tan \frac{fovy}{2} \\ x_{1y} \tan \frac{fovy}{2} \\ -1 \end{pmatrix} \\
\begin{pmatrix} v_{1x} \\ v_{1y} \\ v_{1z} \end{pmatrix} &= \begin{pmatrix} v'_{1x} \\ v'_{1y} \\ v'_{1z} \end{pmatrix} / ||\mathbf{v'_1}||
\end{aligned} \tag{5.1}$$

Knowing the values for θ and ϕ, we can construct an associated rotation matrix $\mathbf{R_1}$ as shown in equation 5.2. The sub-matrices used to calculate $\mathbf{R_1}$ are the standard rotation matrices along the y and x axes, but with the direction of rotation inverted. The reason for this inversion is clearer if we consider the process of changing the cube view, i.e. altering θ and ϕ, as a rotation of the cube (around the origin) from a reference position to a viewing position. We are finding coordinates on the *rotated* position corresponding

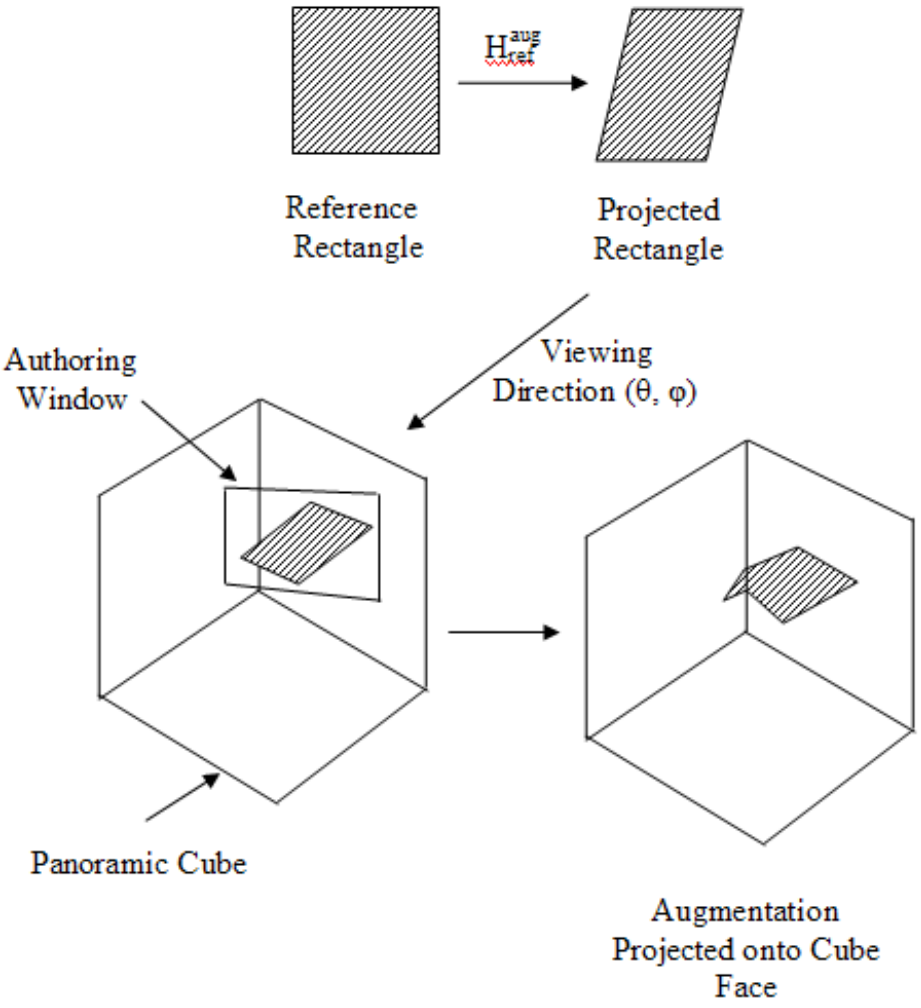


Figure 5.2: Back-projection of planar augmentation

to viewing vectors in the *reference* position. The vector $\mathbf{v_{1r}}$ represents the augmentation unit vector in the rotated reference frame (equation 5.3).

$$\begin{aligned}\mathbf{R_1} &= \begin{pmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{pmatrix} \\ &= \begin{pmatrix} \cos\theta & \sin\theta\sin\phi & -\sin\theta\cos\phi \\ 0 & \cos\phi & \sin\theta \\ \sin\theta & -\cos\theta\sin\phi & \cos\theta\cos\phi \end{pmatrix} \end{aligned} \tag{5.2}$$

$$\mathbf{v_{1r}} = \mathbf{R_1 v_1} \tag{5.3}$$

After calculating $\mathbf{v_{1r}}$, we wish to find the cube face intersected by the vector $\mathbf{v_{1r}}$. This is done by calculating the distance from a point (the origin) to the planes, along the unit vector. The plane with the shortest positive distance is the plane of intersection. Calculation of the intersection is shown in equation 5.4. Here, $\mathbf{N_i}$ represents a normal to the plane, and $\mathbf{P_i}$ represents a point on the plane. As an example, $\mathbf{N_0} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$ and $\mathbf{P_0} = \begin{bmatrix} 0 & 0 & -100 \end{bmatrix}^T$ for the front-side plane.

$$d_i = \frac{\mathbf{N_i} \cdot \mathbf{P_i}}{\mathbf{N_i} \cdot \mathbf{v_{1r}}} \tag{5.4}$$

Having identified the lowest positive value for d_i, the coordinate on the cube is simply $d_i \mathbf{v_{1r}}$.

We back-project the corners of the augmentation, as well as a regularly spaced grid of points within the augmentation. This is required to avoid unnatural stretching or skewing of the texture by the rendering software.

Calculating the back-projection of the augmentations allows for the possibility of generating augmentation "layers", where the augmentations would be saved as a separate cube in a nearly identical format to the original cube: augmentations would be represented as six bitmap images, with a pixel-by-pixel transparency value to allow the original cube to be visible beneath in regions with no augmentation. In the current implementation, this approach is not used. While it might provide some benefit in data consistency between the actual panoramic cube images and the artificial augmentations, the size requirement for storing the augmentations makes this approach infeasible.

5.3.4 Placement of Model Augmentations

In the case of placing a three-dimensional model, a slightly different approach is used. A homography will accurately represent the projection of points lying on a planar augmentation, but three-dimensional augmentations require correct placement of points lying off the plane. Again, the user selects four corners of a rectangle to make an initial placement of the model. The rectangle selected by the user will be used to define the orientation and scale of a coordinate system for displaying the model.

From the corners of the model, we use a direct geometric approach for determining the pose of the model. The direct approach assumes that the region selected is square, so it is desirable for the author to select a region that is very close to square.

The direct geometric approach follows the method from [9]. Consider the geometry

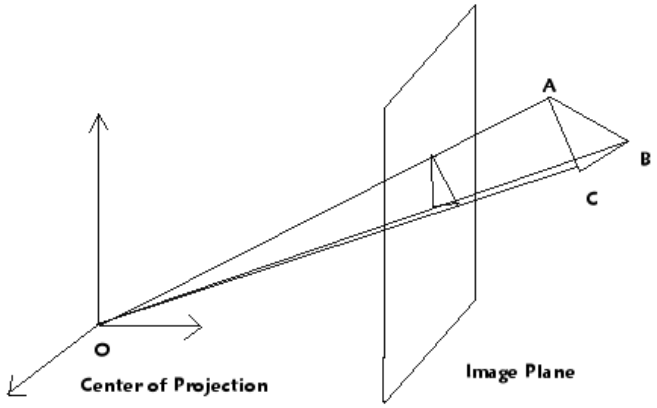


Figure 5.3: Direct geometric solution

illustrated in Figure 5.3. We assume we have prior knowledge of the angles and lengths of triangle ABC. The side lengths of the triangles are related to the ray lengths from the center of projection by using the cosine rule, as shown in equation 5.5.

$$\begin{aligned} AB^2 &= OA^2 + OB^2 - 2(OA)(OB)\cos\theta_{AOB} \\ BC^2 &= OB^2 + OC^2 - 2(OB)(OC)\cos\theta_{BOC} \\ AC^2 &= OA^2 + OC^2 - 2(OA)(OC)\cos\theta_{AOC} \end{aligned} \tag{5.5}$$

We can obtain the θ_{iOj} angles from equation 5.5 by determining the pixel coordinates of the triangle corners on the image plane. The distance from the center of projection to the image plane is fixed to the camera focal length, thus the angles can be determined by using the cosine rule. With knowledge of the side lengths AB, BC and AC, as well as the θ_{iOj} angles, we arrive at three equations in three unknowns, with the unknowns being the projection lengths OA, OB and OC. [9] describes a method to convert this problem into a 4^{th} order polynomial, whose real roots are used to solve for the projection lengths.

5.3.5 Calculation of Feature Points

After the augmentation is placed, we are able to begin the process of propagating the augmentation to the rest of the panoramic sequence. The first step in this process is to identify feature points that can be used for matching. These feature points are taken from the planar region selected by the author.

We use the rendered image from the authoring system's interface as the input for the feature point detector. We repeat rendering of the cube in the same orientation as selected by the author, but with the interface hidden. The frame buffer is captured, and passed to the feature point detectors.

SIFT and PCA-SIFT feature points are calculated using sample code from David Lowe [17] and Yan Ke [18], respectively. Accuracy results were found to be similar using the two methods, but the PCA-SIFT features take significantly less time to calculate.

5.3.6 Using pre-rendered features within selection region

An alternative, faster method is to use the author's selected region to select from feature points pre-calculated for the cube faces. However, there will be differences between the feature points calculated using these two methods. In the case of pre-computed feature points, the features are taken from a "front-on" view of the cube faces. In the case of calculating the feature points from the author's view, the cube face rotated according the view parameters θ and ϕ. We use feature points calculated using the author's view, as the region selected by the user will typically have the planar region in a front-facing orientation perpendicular to the author's viewing direction.

Figure 5.4: Restriction of feature region

To illustrate this, consider Figures 5.4 and 5.5. In the first case, we will have calculated feature points from all the shown cube faces, and will restrict our matching to use the feature points from the white region on the left. In the second case, we calculate feature points using the image shown, taken from the rendering on the user's window. The features calculated from these two methods may differ.

This alternative method did not work well when the matching region was significantly out-of-plane with respect to the cube faces. The final approach used the re-rendering of the cube as described in section 5.3.5.

5.3.7 Inter-frame Matching

Having obtained the feature points from the user's planar region, we attempt to match these features against an adjacent region. The feature points for the adjacent panoramas

Figure 5.5: Calculation of features from rendered region

are calculated from the cube faces prior to the start of the authoring process. A "match" is identified in a different fashion depending on whether SIFT or PCA-SIFT matches are used, but are functionally similar.

5.3.8 Feature Match Calculation

In the case of SIFT feature points, a match is identified when the most similar pair of feature points, one from each image, is more distinct than the next-most similar pair. Similarity is defined as the Cartesian distance between the feature descriptors. This is expressed algebraically in equation 5.6. In this case, $\mathbf{FV_{1a}}$ represents the descriptor vector from the feature we are testing for matches, $\mathbf{FV_{1b}}$ represents the descriptor vector of the most closely-matched feature from the other image. $\mathbf{FV_{2b}}$ represents the descriptor vector of the second-best matching feature.

$$match\ for\ \mathbf{FV_{1a}} \quad = \quad \begin{cases} \frac{||\mathbf{FV_{1a}}-\mathbf{FV_{2b}}||}{||\mathbf{FV_{1a}}-\mathbf{FV_{1b}}||} < \mathbf{thresh} & match \\ otherwise & no\ match \end{cases} \tag{5.6}$$

In the case of PCA-SIFT features, matching is done by associating each feature point descriptor from one image to the closest (using a Euclidian distance measure) feature point descriptor in the other set. If the match distance between the two features falls below a predetermined threshold for acceptability, we consider the match as successful. This is shown in equation 5.7.

$$match\ for\ \mathbf{FV_{1a}} \quad = \quad \begin{cases} ||\mathbf{FV_{1a}} - \mathbf{FV_{2a}}|| < \mathbf{thresh} & match \\ otherwise & no\ match \end{cases} \tag{5.7}$$

The errors that occur during the matching process can be separated into two categories: false positive errors, and false negative errors. A false positive occurs when a feature pair is marked as a *match*, when in reality the corresponding pixels do not represent the same physical location. A false negative would occur when two feature points correspond to the same physical location, but fail the matching test. A method to reduce the occurrence of these errors is described in the following section.

5.3.9 Limiting Match Region

We can control two aspects to reduce the effect of errors. The most direct way to affect the result is to alter the matching threshold. The other method is to restrict the features we search through on the matching panoramas using a proximity constraint.

By decreasing the threshold, we require a higher level of certainty before declaring a particular pair of features as a "match." This will decrease the number of false positives, with a corresponding increase in false negatives. However, there persists a problem with *outliers*, where a feature may match very well with an incorrect feature from the second set. An adjustment to the threshold will not, by itself, correct difficulties arising from outliers. In the case of the SIFT-style matching, false negatives can occur if a feature elsewhere in the panoramic scene is very similar to the corresponding feature from the scene. A typical case where this might occur is a region of repeated texture: for instance, windows of a building will have many similar features. A particular corner of a window will be very similar to several other corners, and will not be sufficiently distinct for matching purposes.

A common approach to correcting this problem is the use of *random sampling*, which is described in more detail in the homography calculation section. We use random sampling for the calculation of the final augmentation homography, but we are also able to use a geometric constraint to improve match success. We expect the matching feature points to be located in relative close proximity in the panoramic image, as they were selected within the author's viewing window with a limited field of view. Therefore, we use a two-stage process to generate the matches.

In the first stage of matching, we generate a list of matches using a relatively low-valued threshold. We use the matching strategy above to generate a first-round set of matches. Using these matches, we obtain an estimate as to the angle of the match location using the following method.

1. Assign a horizontal angle θ to each match, corresponding to the angle between the "straight forward" position in the panorama and the position of the match.

2. Calculate an average θ, θ_{av} from all the match angles.
3. Calculate a variance using θ_{av}.
4. Drop any matches with a θ lying outside three standard deviations of the mean angle.
5. Calculate a new average θ, θ'_{av} using only those matches within the three standard deviation limit.

Calculation of the average angle is done using a vector-addition approach. For each match's θ, we calculate the x and y-components of an associated unit vector. The components from all the matches are summed, corresponding to a vector addition of all the unit vectors. The average angle will be the angle corresponding to the summed vector. This is shown in equation 5.8.

$$\begin{aligned} x_i &= \cos\theta_i \\ y_i &= \sin\theta_i \\ \\ x_{av} &= \sum_{i=1}^{N} x_i \\ y_{av} &= \sum_{i=1}^{N} y_i \\ \theta_{av} &= \tan^{-1}\frac{y_{av}}{x_{av}} \end{aligned} \tag{5.8}$$

To reduce outlier effects, we then calculate a variance on the match angle, using θ_{av}. This is shown in equation 5.9. $\mathbf{u_x}$ represents a unit vector in R^2, at a rotation angle x.

$$\begin{aligned} var(\theta) &= \frac{1}{N}\sum_{i=1}^{N}\Delta\theta(i) \\ \text{where}\Delta\theta(i) &= \cos^{-1}(u_{\theta_{av}} \cdot u_{\theta_i}) \end{aligned} \tag{5.9}$$

We then recalculate the average angle, θ'_{av}, rejecting those with a θ outside a $3\sqrt{var(\theta)}$ range. Now that we have calculated a rough estimate of the position of the planar region, we can then re-generate a match list, rejecting any matches that lie far from the expected match direction. In most cases, this pre-filtering step will reduce the number of mismatches significantly. An example of this process is shown in Figure 5.6. In this figure, we can see the outliers as matches far from the matching region. It is likely that these outliers would be ignored using a random-sampling approach to generating the homography. However, by dealing with them in this way we can potentially increase the number of valid matches. Some features which were previously causing mismatches will now be excluded from the matching area.

After we have restricted the matching region, we perform a second pass of matching. We have increased confidence on the accuracy of our matches in this stage, and the match threshold is increased. An example of the second stage of matching is shown in Figure 5.7.

We use the matches from this stage to compute a homography. This homography is calculated using a random-sampling approach, as described in the Section 2 of this thesis. In the case of planar homographies, we can use the homography calculated at this point to perform occlusion detection. This is described in more detail in the following chapter on Occlusion Detection. The two-stage approach described above is a unique contribution of this thesis, and we found it allows for efficient, accurate calculation of augmentation placement.

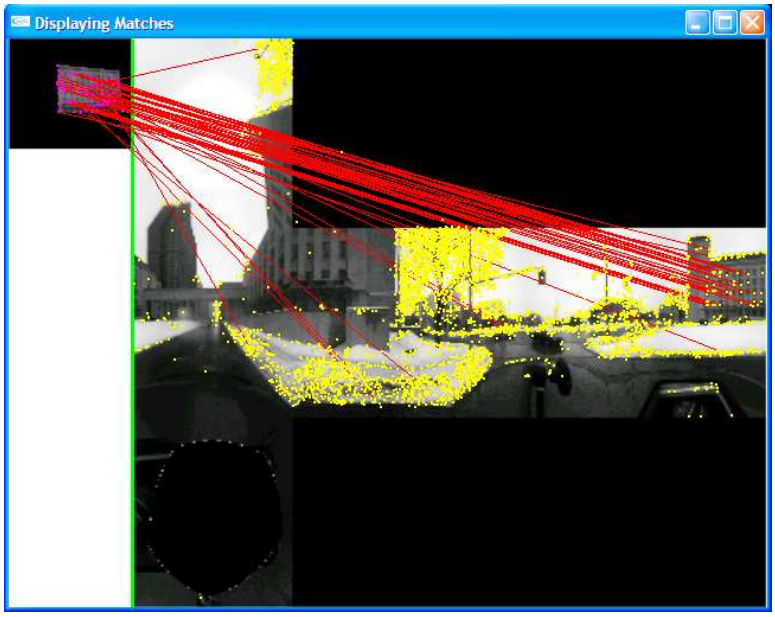


Figure 5.6: First pass: match versus entire cube

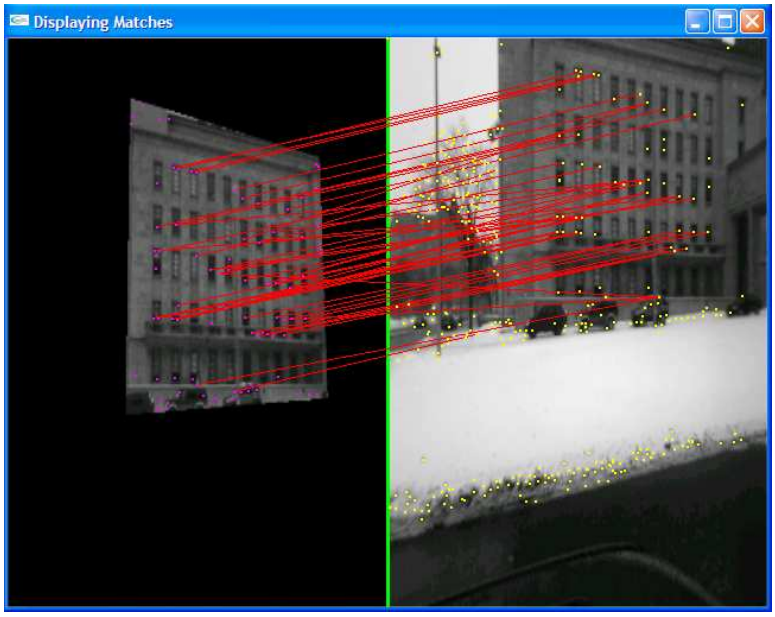


Figure 5.7: Second pass: match versus restricted region

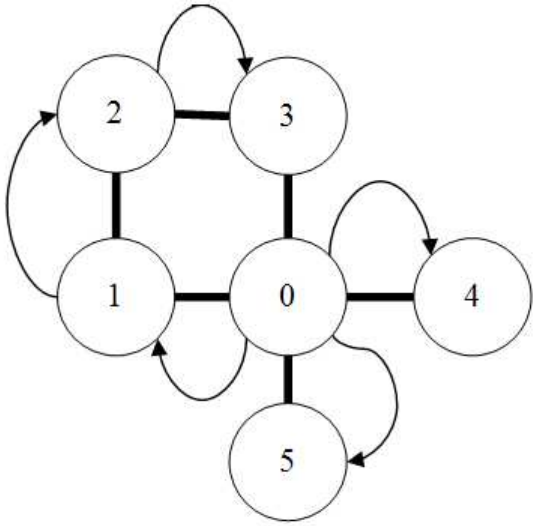


Figure 5.8: Depth-first search pattern

5.3.10 Recursive Propagation

The matching results are propagated outwards from the point of augmentation insertion. The propagation is done in the form of a *depth-first search*, as illustrated in Figure 5.8. In this figure, circles indicate panoramas, and the thick lines indicate connections between panoramas in the map file. The links are established prior to the authoring process. In this thesis, these links are performed manually. The arrows indicate the order that homographies are calculated.

Starting at the center node, node zero, homographies are exhaustively calculated along one direction before searching other directions. This method was chosen, because it is simple to implement in a recursive fashion. A flaw in this approach is visible at node

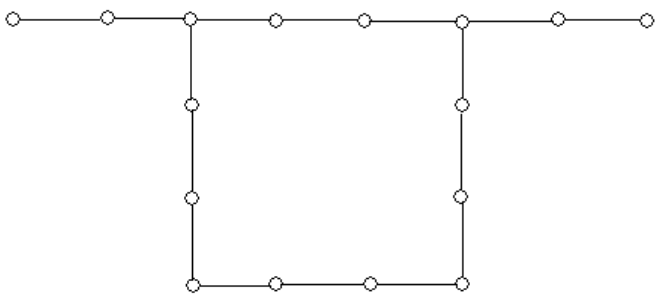


Figure 5.9: Typical panorama graph

three, where we obtain the homography by matching versus node two. As the length of the chained homography increases, the degree of error present also increases, so it would be more effective to match against node zero, switching to a breadth-first search method. This flaw does not appear when dealing with one-dimensional sequences of panoramas (i.e. a linear sequence), but would be more apparent in a grid-style arrangement of panoramas. The data sets used for testing of the system were largely one-dimensional sequential in nature, with graphs resembling the one in Figure 5.9. With sequences of this nature, there is little advantage of switching away from the depth-first search to a breadth-first search.

In the case of a failure to propagate a homography, the system attempts to recalculate the homography against the original insertion panorama. Referring to Figure 5.8, if we failed to calculate a homography at panorama two (using the augmentation computed for panorama one), we would instead attempt to calculate a homography from panorama zero to panorama two. If this homography succeeds, then we would proceed to calculate a homography from panorama two to panorama three. In this case, the homography from the original insertion panorama to panorama three would be $H_0^2 H_2^3$, rather than $H_0^1 H_1^2 H_2^3$.

Chapter 6

Occlusion Detection

During the augmentation process, it is important to properly detect and deal with occlusions. Consider the images in Figure 6.1 which depicts an example of the type of occlusion that the system should handle. In the left image, we have a good view of the building wall. In the right image, the wall is obstructed by a light pole. An augmentation placed on the wall should also be partially obstructed by this pole.

If we place the augmentation with no concern as to occlusion, the planar augmen-

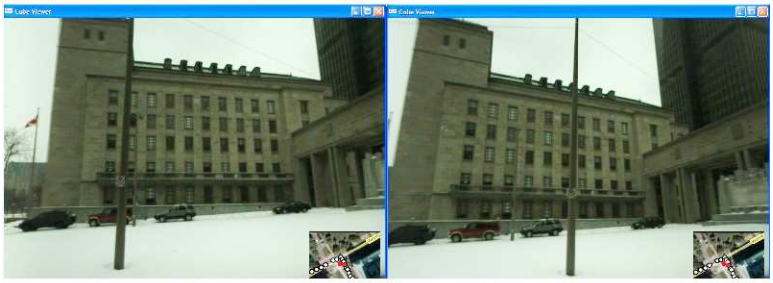


Figure 6.1: Example of occlusion of a planar region.

tation will overlap the pole. This gives an incorrect impression of depth, which greatly diminishes the realism of the scene. The goal of the occlusion detection mechanism is to classify pixels that are occluded by a foreground object; an augmentation should not be rendered on these pixels; allowing the foreground object to remain visible. We wish to generate an *occlusion mask* that specifies which pixels of the augmentation are occluded by a foreground object.

6.1 Outline of Occlusion Detection Algorithm

The algorithm used for occlusion detection follows:

1. De-projection of reference and comparison planes
2. Alignment of de-projected planes
3. Pre-filtering of aligned planes
4. Bitwise thresholding of differences of aligned planes
5. Post-filtering of aligned planes

The assumption is that the rectangular planar region selected by the user contains no occlusions. This is considered a reasonable assumption, since in most cases the user would select the anchoring plane from a panorama where the region is most visible. This assumption simplifies calculation, by giving us the ability to assume that the initial planar region in the insertion view represents an occlusion-free image; deviations from this planar region can be viewed as an occlusion. While it was not explored in this thesis, it would be possible to avoid this assumption by assigning confidence to pixels

or subregions of the planar area in the initial insertion panorama. If subregions recur in nearby panoramas, our confidence that the subregion is not an occlusion would increase.

During the matching process, a homography is identified that relates the user's reference rectangle with equivalent rectangles from nearby panoramas. To compare these planar regions, it is necessary to remove their projections to change them into front-facing rectangular images. The transformation required to do this is precisely the inverse of the homography that is calculated in Section 5.3.3.

$$\mathbf{x_{view}} = \mathbf{H}\mathbf{x_{ref}} \tag{6.1}$$

$$\mathbf{x_{ref}} = \mathbf{H^{-1}}\mathbf{x_{view}} \tag{6.2}$$

The de-projection requires the use of subpixel interpolation, due to the fractional pixel coordinate values that we obtain. The simple and well-known bilinear interpolation technique is used to obtain the RGB values for the de-projected pixels.

Ideally, after having de-projected the rectangular sections, the rectangular images we obtain would represents precisely the same sections of the planar region. Unfortunately, in practice the computed homographies contain some degree of error. The errors in the homographies will result some degree of misalignment between the planar regions, which in turn will introduce errors into the occlusion mask. To reduce the effect of these misalignment errors, a pixel shift is computed to apply to the rectangular regions prior to performing comparison.

6.2 Shift Calculation

The goal is to calculate a motion vector (s,t) to apply to one of the rectangular regions so that they represent the same physical location on the planar region. Geometrically, this would represent horizontal and vertical shifts in the de-projected rectangle.

This technique makes the assumption that homography errors will result only in shifts along the plane of the rectangular regions; in theory, errors in the homography may result in errors that cannot be represented in this fashion. However, calculating a motion vector between the regions does seem to improve the occlusion mask generation.

To calculate the motion vector, a hierarchical method similar to that used in motion stabilization [23] is employed. The goal is to to identify the shift (s,t) that minimizes a sum of the absolute difference in pixel intensities between the rectangular regions. The mathematical formulation of this condition is shown in equation 6.3.

$$(s,t) = \underset{x,y}{\operatorname{argmin}} \sum_{i,j} |I_1(i,j) - I_2(i-x, j-y)| \tag{6.3}$$

For regions of pixel dimension MxN, each shifted position entails the computation of MN differences. We test a certain number, T, of the possible shifts. Assuming T is a significant fraction of the MN possible shifts, we obtain an order $O(M^2N^2)$ for the calculation of possible shifts. For typical values of MN, this is a very time consuming process.

A reduction to the values of M and N has a significant effect on the time for alignment. Therefore a simple approach to reduce the computation time for the absolute differences

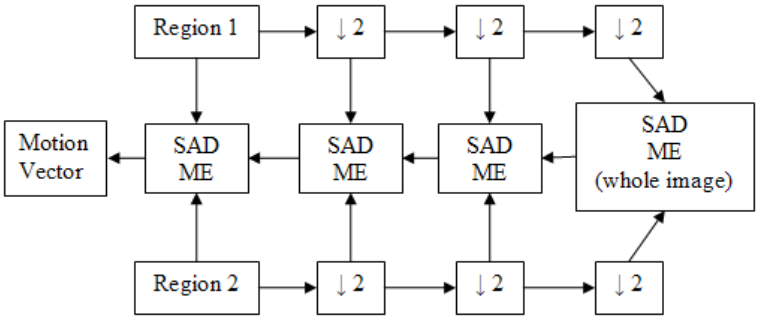


Figure 6.2: Block diagram of 3-stage hierarchical motion estimator

is to downsample the regions, then align the lower-scale images. A shift in the lower-scale image will represent a larger shift in the higher-scale image. For instance, in an image downsampled by a factor of two in both the horizontal and vertical directions, a shift (s,t) in this downsampled image will represent a larger shift (2s,2t) in the original image. The search time on this downsampled image will be therefore reduced by a factor of four. It is possible to use the searches on the lower-scale images to set limits on searches on the higher-scale image. For instance, if we take the regions downsampled by two in both dimensions and locate a shift (2s,2t), we can then refine the search to offsets of 1 in the higher-scale image. For a resolution MN for the original, higher-scale region, we have reduced the computation time from M^2N^2 to $M^2N^2/4 + 9$. If we increase the depth of the downsampling, we can further reduce the computation time.

Figure 6.2 illustrates a 3-stage implementation of the shift determination scheme described above. In the actual implementation, a 4-stage estimator was used. Note that the downsampling blocks use separable Gaussian anti-aliasing filters that are not shown in the diagram. After the two views to compare are fully downsampled, the sum-of-absolute-difference motion estimator (SAD ME) determines the shift generating the lowest sum of absolute differences between the two images. This is fed to the SAD ME from the previous stage, which refines the search to within one pixel of the best match from the lowest-scale stage. This process is repeated, until we obtain a motion estimate

in the original scale.

As an example, consider the case of two 800x600 images. After three downsamplings, the images being compared are 100x75. At the lowest scale (scale 3), we determine a lowest-SAD motion vector of (10,20). At scale 2, this corresponds to a shift of (20,40) so we perform a search for the lowest-SAD at scale 2 on the shift range ([19-21],[39-41]). The process is repeated at scale 1, then at scale 0, and we obtain an estimate on the best shift.

We apply this shift to one of the reference regions, and pass the aligned regions to the pre-filtering stage. The amount of shift varied significantly, with relatively small shifts (under 5 pixels) being typical for the immediately adjacent panoramas. As we move farther from the insertion point, the homography error accumulates and the augmentation drifts, leading to a corresponding increase in the shift.

6.3 Pre-Filtering

It is possible that using a filtered version of the comparison regions might help with the generation of the occlusion mask. Since Gaussian-filtered versions of the reference regions are generated during the downsampling phases of the motion estimation stage, we can apply the alignment to these filtered images rather than the original reference regions.

In practice, this did not to offer a significant improvement, and so no pre-filtering was done on the aligned regions.

6.4 Thresholding of Aligned Regions

To detect an occlusion, we employed a simple global threshold of the pixel intensity differences between the aligned regions. We binarize the absolute difference between the two intensity images into two possible values. Pixels where the difference exceeds the threshold were flagged as *occluded* and those below the threshold are flagged *unoccluded*.

This method of detecting occlusions has some obvious limitations. The planar region can be occluded by an object of similar brightness, in which case the occlusion will be undetected. Any alignment errors that are not handled by the previous stage will appear as a false positive or false negative occlusion detection in this stage. This method is also highly sensitive to lighting changes between frames of the panoramic sequence, which will lead to a large number of false positive occlusion detections. This pixel-by-pixel approach works when there is a large occluding object of very different luminance from the background; for instance, a very bright or dark object on a mid-brightness background. Better results may be attainable by moving to an edge-based approach to define the occluding object.

6.5 Post-Filtering

After we have performed thresholding, there are typically a number of isolated pixels that are falsely identified as occlusions. This will often occur at the edges of objects, where alignment is not exact and will yield small clusters of pixels that are incorrectly identified as occluding objects. To handle this type of error, we employ a *blob filter*. This filter assigns a *size* value to all the connected occluded pixel regions in the occlusion map. Where the size of these regions, or blobs, is below a given threshold, we drop this

region from the occlusion map. This removes a significant number of spuriously detected occlusion regions from the occlusion map.

To further reduce the number of false occlusions, we also use another filtering condition. If the occluding blob does not touch the edge of the occlusion region, then it is rejected as an occlusion. This criterion is based on the assumption that the occluding object will not be "floating" in front of the planar region; in almost all cases we would reasonably expect the occlusion to be touching one of the sides of the region. This edge-touching test also helps to reduce the number of spuriously detected occlusions.

Chapter 7

Results

The following section discusses the results obtained using the augmentation authoring system, some of the difficulties and limitations observed in the use of the system, as well as possibilities for future enhancements.

7.1 Test Sequences

Testing was primarily performed on two sequences. The sequences were generated by Dr. Mark Fiala as part of the NAVIRE project. The sequences were taken using the Ladybug camera setup described in the introduction. The first sequence, referred to as the *indoor* sequence, consists of images taken inside the National Research Council. The scene consists of two principal sections; a series of largely featureless hallways, and several panoramas taken in a lab containing objects such as desks, computers, posters, and other objects. The second sequence, referred to as the *outdoor* sequence, is a sequence taken

Table 7.1: Test sequence comparison

Property	**Indoor**	**Outdoor**
Cube face resolution	512 x 512	1024 x 1024
Quantity of Features	Limited in hallways, higher within rooms	High
Brightness consistency	Good within single rooms	Some variation

by mounting the Ladybug through the sunroof of an automobile and capturing images as it navigated through the city.

The indoor sequence was taken earlier in the project, and uses a lower resolution (512 x 512) for each of the cube faces compared to the outdoor sequence. The frames of the indoor sequence have fairly limited texture in sections within the hallways, which led to a low number of feature points in these regions. Testing concentrated on sequence sections within rooms, or areas with posters or other textured regions to use as an anchor for the augmentations. A typical testing frame and corresponding view from the sequence is shown in Figure 7.1. An example from the outdoor sequence is shown in Figure 7.2.

Additional results are given below. A result taken from inside the University of Ottawa VIVA Lab is given in Figure 7.3. An example of a model augmentation is given in Figure 7.4, where a microphone model is inserted into a scene with a church. The user can click on the microphone to play an audio file associated with the augmentation.

7.2 Augmentation Propagation Time

To test the time to propagate an augmentation, we record the amount of time from the start of the propagation process to the end of the propagation process. This is divided

Figure 7.1: Indoor frame

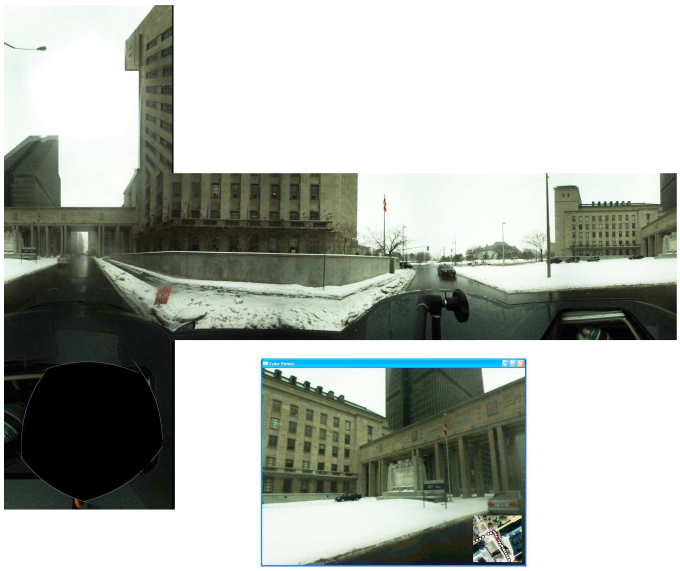
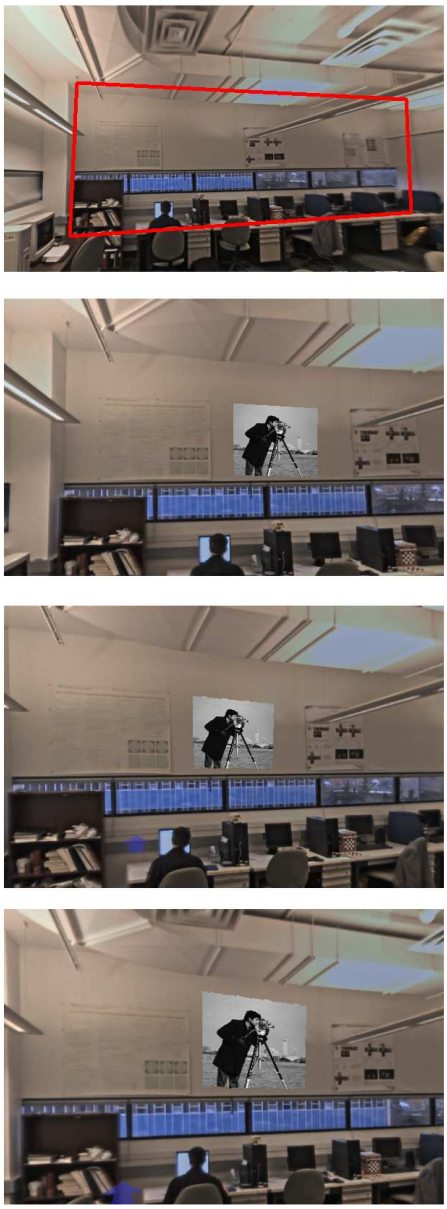


Figure 7.2: Outdoor frame



a) Insertion plane

b) Inserted Planar Augmentation

c) Propagation 1

d) Propagation 2

Figure 7.3: Example planar propagation

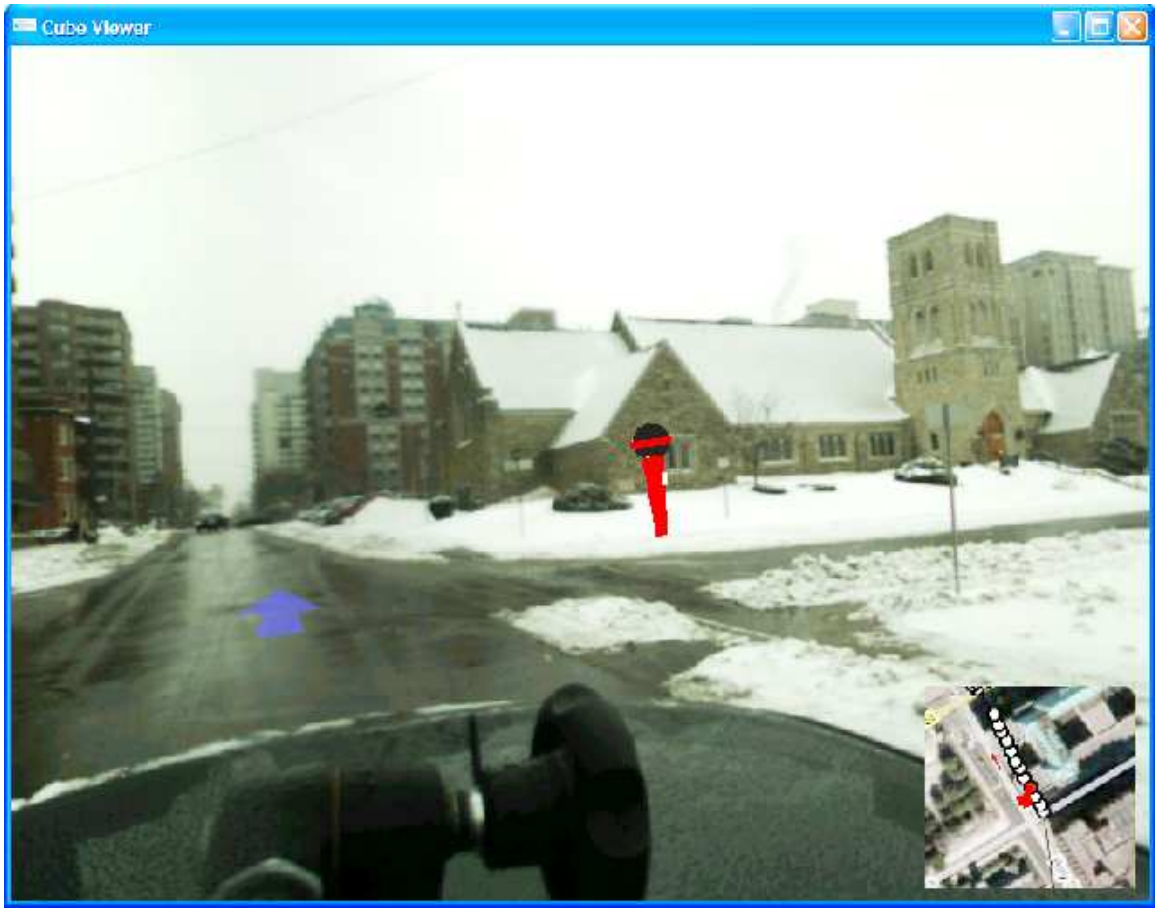


Figure 7.4: Example model augmentation

by the number of panoramas that the augmentation is propagated to, giving an average time per panorama.

The time required to perform propagation is strongly related to the number of features. In typical sequences, the propagation time per panorama ranges between ten to fifteen seconds. This value is strongly affected by the number of panoramas in the scene, as well as the number of features within the anchoring planar area.

7.3 Augmentation Accuracy

"Ground truth" geometric data on the scenes are not available, but subjective measures of augmentation accuracy of the scenes can be used. The following are some subjective evaluations of the augmentation results.

7.3.1 Augmentation Stability

The problem of augmentation stability is manifested in a slightly different fashion than in typical augmented reality applications. Within a single panorama, an augmentation will be completely immobile. This means that while a user is focused on an augmentation, and not transitioning between panoramas, the augmentation will remain stable. The augmentation may be in an inaccurate position with respect to the author's placement position, but augmentations will not exhibit tracking "jitter" that is found in real-time augmented reality systems. In this system, an analogous error to tracking jitter could be observed by transitioning between adjacent panoramas and examining the augmentation for positional consistency, but absolute error is a more applicable form of error than

stability.

7.3.2 Planar versus Model Augmentations

Planar augmentations in general obtain higher perceptual quality compared to model augmentations. Inaccuracies in the homography will result in stretching or skewing of planar augmentations, but this type of distortion does not (within limits) have a dramatic effect on the perceptual quality of the augmentation. Errors in the alignment of model augmentations are less perceptually acceptable, as stretching, skewing, or inaccurate rotation of the three-dimensional model is more obvious to the viewer.

7.4 Qualitative Results

The augmented reality system worked successfully in a number of areas of the test sequences. Areas where the system is weakest are explained later in the section.

Figures 7.5 and 7.6 illustrate a successful planar augmentation propagation in the indoor sequence. In the top-left image of Figure 7.5, observe the planar region that is selected by the user. A virtual poster of a man looking into a camera on a tripod has been placed super-imposing a real poster. The remaining images show the augmentation as automatically placed by the system. Occlusion detection was disabled for this test. Note that the selected planar region contains some non-planar elements, including the foreground computer, but the features lying within this non-planar region will be ignored because of the random-sampling component of the homography generation process. In general, some degree of non-planarity in the selected region will often yield acceptable

results. A small number of foreground objects is permissible, provided that a significant majority of the features still arise from the planar region.

The virtual poster has been inserted onto the wall to the right of the doorway, overlapping a physical poster in the scene used. The augmentation's positional accuracy is evaluated by qualitatively judging how closely the augmentation's position matches the physical poster over which it is placed. For most of the sequence, the augmentation's position is fairly good: the outline of the virtual poster matches well with the physical poster.

Tracking is lost in the middle row, left frame: here too much of the planar region has become occluded, causing a failure in the homography generation. Tracking error begins to accrue as the camera approaches the physical poster, and in the last image shown the positional accuracy is very different from the physical poster position. In the frame following this one, the planar region is entirely out of view and matching fails.

Figure 7.7 shows partial results from a model (a virtual tree) inserted into the outdoor sequence. The model's origin is placed co-planar to the selected planar region. No occlusion-testing is done in this case.

The results shown here suffer from the lack of occlusion detection for model augmentations. Notably, the lower-left image in the figure suffers significantly from the stone sign appearing to lay behind the tree, rather than occluding it. The positional consistency of the tree can be confirmed by examining the windows behind the model.

The matching process is not, however, always successful. The system requires us to anchor the augmentations on regions of high texture in the scene. There are many cases, particularly in indoor sequences, where there is simply not enough texture to reliably

Figure 7.5: Indoor sequence, planar augmentation

Figure 7.6: Indoor sequence, planar augmentation (cont'd)

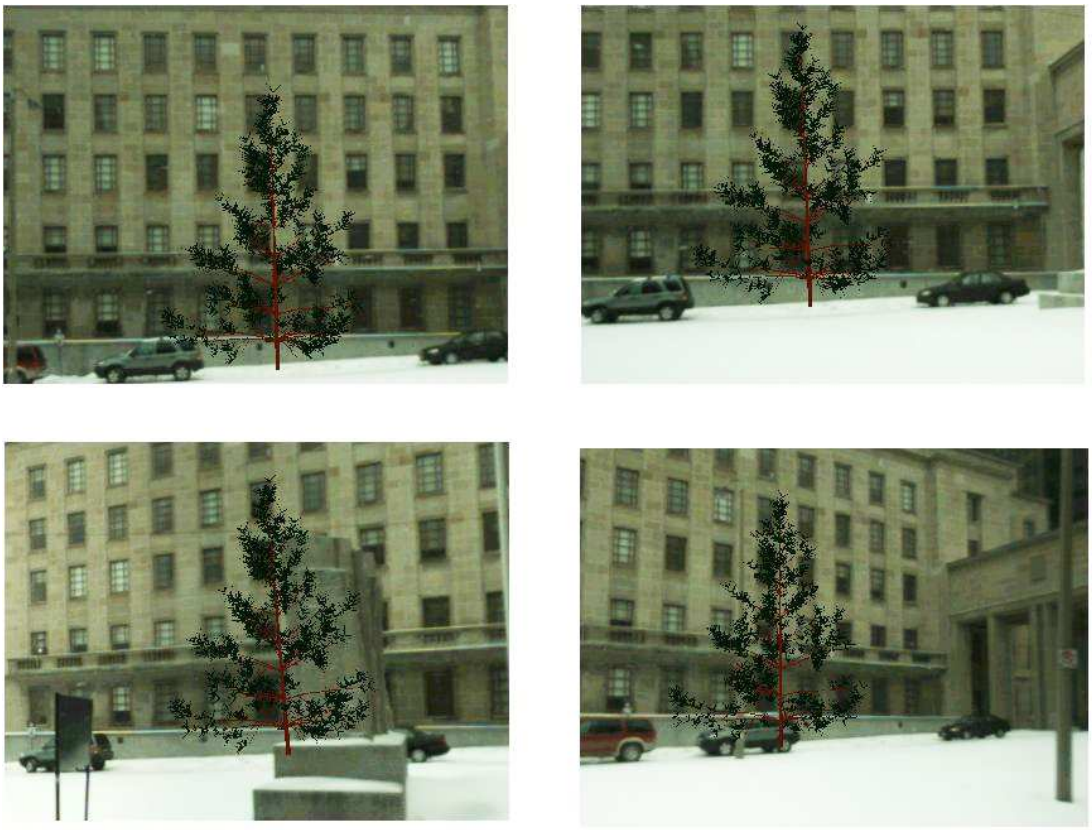


Figure 7.7: Outdoor sequence, model augmentation

ensure matches in the adjacent panoramas. In other cases, textured regions may not be sufficiently distinct to allow for proper matching. For instance, selecting a particular window on the face of a large building will likely be ineffective. The best results are achieved when we can select a large planar area to use as an anchor, for instance the face of an entire building, or a large poster.

7.4.1 Occlusion Detection Results

The occlusion detection scheme worked for certain types of occlusions, with predictable limitations. The occlusion detector works well in an outdoor test sequence, where the typical occlusions were large, dark light poles that would occlude the buildings behind. The poles have a significant difference in intensity from the building walls, which is an ideal situation for the threshold-based occlusion detection. The occluding objects were also relatively large, which avoided erroneous removal by the blob-filtering in the post-filter step. Figure 7.8 shows the results of various steps from this sequence. In the first row, we see the original images. The right image contains the occluding pole. In the second row, we see the binarized threshold images without (left) and with (right) the alignment step. It is clear the alignment significantly improves the occlusion mask. The last row shows the results of applying the blob filter and edge-touching constraint to the aligned occlusion mask.

Further tests revealed some limitations of the occlusion detection method used in this thesis. The system is unable to detect occlusions by objects of similar intensity to the planar background. In this case the threshold will not be exceeded and we obtain a false negative result, ie. pixels are marked as unoccluded where an occlusion is occurring. This could potentially be improved by matching against colour as well as intensity in order to deal with the situation where objects are similar in intensity but differ in colour.

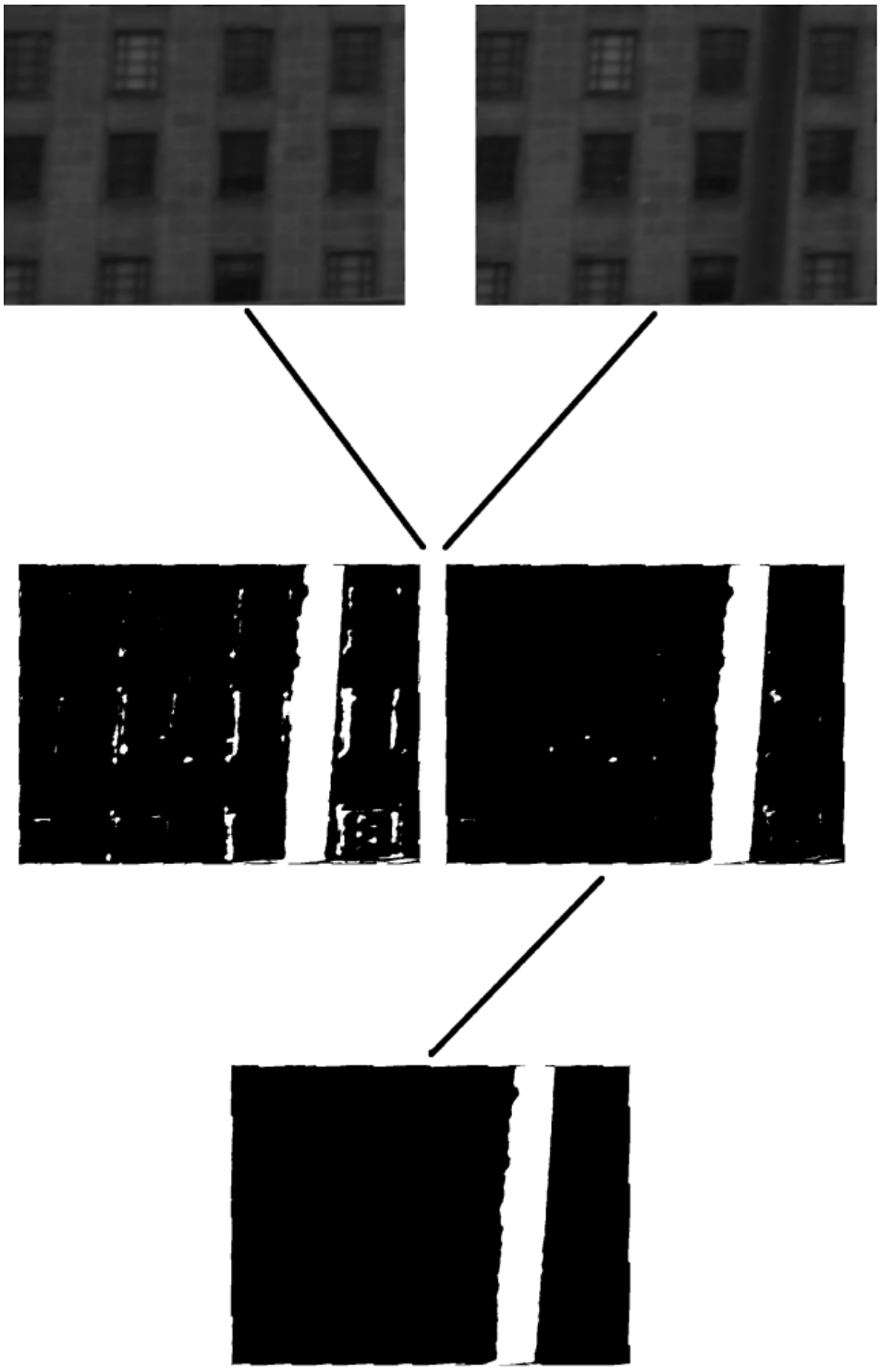


Figure 7.8: Occlusion detection process.

Another error was observed where there was a subtle lighting change between frames of the sequence. In this case, there is a risk of a high number of false positives, ie. locations where pixels are falsely marked as occluded. This occurs because the lighting difference may cause many pixels to exceed the detection threshold, even if the regions are properly matched. To alleviate this problem, it may be advisable to apply normalization of the brightness between frames of the sequence.

7.5 Pre-Warping of Match Region

SIFT and PCA-SIFT features suffer from variability with out-of-camera-plane rotations [18]. Therefore, if the planar region being tracked undergoes a significant rotation of this nature, tracking may be lost between frames. This situation can occur if we attempt to track a planar region close to the panoramic camera as it is moved down a hallway: the out-of-camera-plane rotational component will be significant. To help deal with this issue, we implemented a pre-rotation system similar in concept to an approach from from [24]. After the user selects the region, we perform a set of out-of-camera-plane rotations on the planar region, and match the adjacent panoramas versus all versions of the rotated plane. On the assumption that the selected region is perfectly planar, we are able to generate accurate rotated versions of the region. Figure 7.9 illustrates a use of this technique: the top image is the planar region selected by the user, the two lower images show artificial rotation of the region. Features are generated for the original planar image, as well as the rotated images.

This additional pre-rotation step improved matching in the case where there is significant rotation out of the camera plane, but at the cost of a great increase in the processing time. Generating a relatively small number of additional rotation angles leads to a dra-

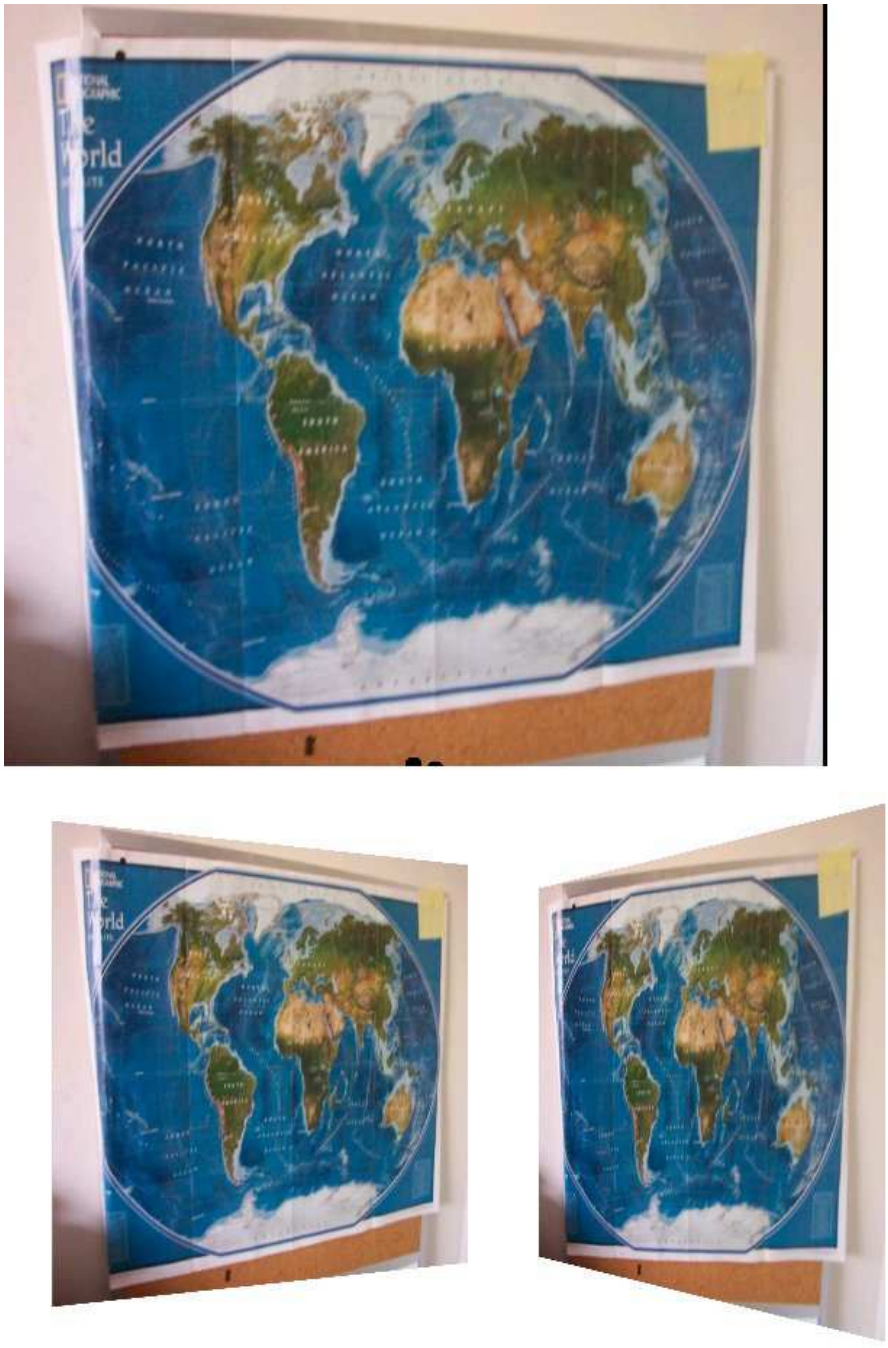


Figure 7.9: Artificial rotation of planar region

matic increase in processing time. Each artificially-generated view must have feature points generated, and these new feature-point sets must be used for comparison during the matching phases.

This pre-rotation step was not used in the final implementation of the system because of the large increase in computation time, but it may be promising for a system using more quickly generated feature descriptors.

7.6 Chapter Summary

This chapter has presents the results obtained using the authoring system from this thesis. Results are given using both an indoor and outdoor panoramic sequence. The results of an extension using pre-warping of the match region is explained.

Chapter 8

Conclusions and Future Work

The purpose of this thesis was to develop a system to insert virtual objects into a cubic panorama image sequence. Augmentations are automatically propagated to other panoramas in the sequence.

The contribution of this thesis is in regards to inserting augmentations into panoramic image environments. Most pre-existing work on augmented reality applies to the insertion of virtual objects into traditional video scenes, rather than the cubic panoramas used here. The thesis uses a unique two-stage authoring process, based on SIFT/PCA-SIFT features and planar homographies. The first stage identifies a coarse match angle towards the matched planar region. The angle from the first stage is used to restrict the match region of the second stage, allowing the generation of a homography with increased accuracy.

The second contribution comes in the form of the occlusion-handling system. We have presented a method that provides an effective way to generate occlusion masks that

can greatly improve the appearance of planar augmentations in sequences with occluding objects. The homography used for augmentation is used to generate front-facing, rectified bitmaps of subregions of the panorama containing the augmentation. These bitmaps can be compared for luminance differences in a pixel-wise manner, with differences being classified as occlusion.

8.1 Future Work

The following are ideas for future work to improve the augmented reality system.

8.1.1 Calculation Speedup: Pre-generated Features

The processing time, while relatively low, could be significantly reduced. The system calculates SIFT/PCA-SIFT features for the planar region after the user selects it. In addition, the system calculates features for the planar regions warped by the homography at each stage of the algorithm. Rather than recalculating the feature points, it may be possible to simply take the feature points from the cube faces (already calculated prior to the authoring process), and use the subset of these lying within the planar region for the matching process. This has a potentially adverse effect on matching success when augmentations are significantly off-plane from the cube faces, but much of the system's processing time is related to computation of the SIFT/PCA-SIFT features. In cases where the augmentation lies parallel to a single cube face, this approach will offer a significant speedup.

8.1.2 Calculation Speedup: Machine Learning Pre-filter

Some of the features within the selected match region may be relatively poor for matching. In some cases, the features within the region will be far too common within the sequence to be useful for distinctive matching. If we can pre-filter some of these "too-common" features prior to the matching calculation and cull them from the list of features to match, this can offer another speed-up for the match calculations.

8.1.3 Multi-Planar Matching

In the current system, we anchor the augmentation against a single plane within the panorama. We could extend this to having the user select multiple planes within the image, and track position relative to each of these planes. This would improve the success of the matching, since there is a higher probability of at least one of the planes remaining visible in frames that are difficult to match for others.

The disadvantages to this approach would be more work for the author to insert the augmentations, as well as additional computational complexity for fusing the poses obtained from each of the individual planes.

Bibliography

[1] D. Kimber, J. Foote, and S. Lertsithichai, "Flyabout: spatially indexed panoramic video," in *ACM Multimedia*, 2001, pp. 339–347.

[2] "NAVIRE Project," Internet:http://www.site.uottawa.ca/research/viva/projects/ibr/f.

[3] "QuickTime VR," Internet:http://www.apple.com/quicktime/resources/tools/qtvr.html.

[4] "CubicConnector," Internet:http://www.clickheredesign.com.au/cubicconnector/.

[5] M. U. et al., "Image-based interactive exploration of real-world environments." *IEEE Computer Graphics and Applications*, vol. 24, no. 3, May/June 2004.

[6] T. Asai, M. Kanbara, and N. Yokoya, "3D reconstruction of outdoor environments from omnidirectional range and color images," *Proceedings of the SPIE*, pp. 579–588, March 2005.

[7] C. Warrington, G. Roth, and E. Dubois, "Markerless augmented reality for cubic panorama sequences," *International Symposium on Mixed and Augmented Reality (ISMAR 2006)*, pp. 255–256, 2006.

[8] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004.

[9] M. A. Fischler and R. C. Bolles, "Random sampling consensus: A paradigm for model fitting with applications to image analysis and automated cartography." *Comm. Assoc. Comp. Mach.*, vol. 24, no. 6, pp. 381–395, 1981.

[10] "OpenGL," Internet:http://www.opengl.org/.

[11] R. T. Azuma, "A survey of augmented reality," *Presence: Teleoperators and Virtual Environments*, vol. 6, no. 4, pp. 355–385, August 1997.

[12] V. Ferrari, T. Tuytelaars, and L. V. Gool, "Markerless augmented reality with a real-time affine region tracker," *Proceedings IEEE and ACM International Symposium on Augmented Reality*, vol. I, pp. 87–96, October 2001.

[13] "ARToolKit," Internet: http://www.hitl.washington.edu/artoolkit/.

[14] H. Kato, M. Billinghurst, I. Poupyrev, K. Imamoto, and K. Tachibana, "Marker Tracking and HMD Calibration for a video-based Augmented Reality Conferencing System," *Proceedings of the 2nd International Workshop on Augmented Reality (IWAR 99)*, 1999.

[15] "ARTag Rev1: marker Detection," Internet: http://www.artag.net/rev1.html.

[16] C. Harris and M. Stephens, "A combined corner and edge detector." *Proceedings 4th Alvey Vision Conference*, pp. 147–151, 1988.

[17] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[18] Y. Ke and R. Sukthankar, "PCA-SIFT: A more distinctive representation for local image descriptors," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 506–513, 2004.

[19] I. Skrypnyk and D.G.Lowe, "Scene modelling, recognition and tracking with invariant image features," *International Symposium on Mixed and Augmented Reality (ISMAR 2004)*, pp. 110–119, 2004.

[20] J. Li, R. Laganiere, and G. Roth, "Online estimation of trifocal tensors for augmenting live video," in *International Symposium on Mixed and Augmented Reality (ISMAR 2004)*, 2004, pp. 182–190.

[21] M. Fiala, "Immersive panoramic imagery," *Proc. Canadian Conf. on Computer and Robot Vision (CRV 2005)*, pp. 386–391, 2005.

[22] A. Brunton, C. Shu, and G. Roth, "Belief propagation on the GPU for stereo vision," *3rd Canadian Conference on Computer and Robot Vision (CRV 2006)*, June 2006.

[23] R. Szeliski, "Image alignment and stitching," *Microsoft Technical Report. Draft. MSR-TR-2004-92*, 2004.

[24] V. Lepetit, P. Lagger, and P. Fua, "Randomized trees for real-time keypoint recognition," *Conference on Computer Vision and Pattern Recognition*, pp. 775–781, June 2005.