VIEW-BASED 3D OBJECT RECOGNITION

Chan Wai Keong

NATIONAL UNIVERSITY OF SINGAPORE
2002

Name: Chan Wai Keong
Degree: Master of Science
Dept: Computer Science
Thesis Title: View-Based 3D Object Recognition

Abstract

Mobile robots require vision capabilities to sense and adapt to the environment. The task of 3D object recognition is an important component of visual sensing. This thesis describes a view-based 3D object recognition technique that recognizes 3D objects from 2D images. We use view morphing to generate novel images of the 3D objects to compare with the 2D query image using the concept of mutual information. As view morphing requires a small set of images of the 3D objects to generate novel images, the amount of space needed for the image database is significantly reduced. Our matching technique makes use of the distribution of image attributes like gray-level and color. This avoids the difficult problem of extracting and matching corresponding features like contours and corners. We tested our system in recognizing 3D objects from 2D images and encouraging results were obtained.

Keywords:
view morphing
object recognition
mutual information
image warping

VIEW-BASED 3D OBJECT RECOGNITION

Chan Wai Keong
(B. Sc. (Hon.) in Computer and Information Sciences, NUS)

A THESIS SUBMITTED
FOR THE DEGREE OF MASTER OF SCIENCE
DEPARTMENT OF COMPUTER SCIENCE
SCHOOL OF COMPUTING
NATIONAL UNIVERSITY OF SINGAPORE
2002

Acknowledgments

I would like to express my gratitude to my project supervisors Dr Ng Teck Khim and A/P Leow Wee Kheng. They have offered me invaluable advice and constructive ideas for my research. They gave me their patience and support whenever I encountered problems during my research. I have benefit tremendously both technically and personally from their guidance and supervision.

My gratitude extends to my friends Li Rui, Tai Peng, Indri, Bryan, Fun Siong, Cher Min, Patrick, Chee Wee, Chuan Hoo and Keng Teck. They encourage me and bring laughter to me when I am feeling down. I would always remember the stories that we shared during our lunch.

My family is my most important source of support. My parent and my brother have given me the encouragement needed during difficult times in the research. Last but not least, Lee Neng for her continuous support and encouragement in my study.

Contents

List of Figures

List of Tables

Summary

Mobile robots require vision capabilities to sense and adapt to the environment. The task of 3D object recognition is an important component of visual sensing. 3D object recognition is however a very difficult task because of the infinite amount of variations possible in the 3D real-life scene.

In this thesis, we propose a view-based 3D object recognition system that recognizes 3D objects from 2D images. This is to cope with changes in pose. We use view morphing to generate novel images of the 3D objects to compare with the 2D query image using the concept of mutual information. As view morphing requires a small set of images of the 3D objects to generate novel views, the amount of space needed for the image database is significantly reduced. Our matching technique makes use of the distribution of image attributes like gray-level and color. This avoids the difficult problem of extracting and matching corresponding features like contours and corners.

We implemented a 3D object recognition system to recognize cars. Our system consists of a recognition module and a database creation module. The database creation module stores the correspondence points needed for view morphing. The database is

created off-line. The recognition module compares the query image with the generated views of every 3D object, using the concept of mutual information.

We tested our system to recognize different types of cars. The mutual information concept allows us to correctly match cars of the same model but different color. Our system is designed to recognize 3D objects with unknown pose that result in images that are predictable by the database. Encouraging results were obtained. We had also tested our system's robustness against changes in image brightness and misalignment of images during matching.

Chapter 1

Introduction

Advances in robotics technology has resulted in the invention of robots that can sense and adapt to their environment. For example, Sony has recently developed the prototype for small biped entertainment robots. In order for the robot to sense and adapt to its environment, the robot must be able to "see" and recognize objects like a human. This is where 3D object recognition systems play a role.

3D object recognition systems try to simulate human's ability to recognize objects in the real world based on a priori knowledge of the appearance of different objects. This is a very difficult task because of the infinite amount of variations possible in the 3D real-life scene. Therefore most of the current object recognition systems are used in industrial applications where the degree of uncertainty can be easily controlled.

In this research, our goal is to develop a 3D object recognition system that can recognize 3D objects in 2D images of a real-life scene under less stringent constraints. We shall begin the discussion in Section 1.1 on the challenges generally faced in 3D object

recognition systems. In Section 1.2, we examine the existing techniques for 3D object recognition. This is followed by a description of our view-based system in Section 1.3. We outline the organization of the thesis in Section 1.4.

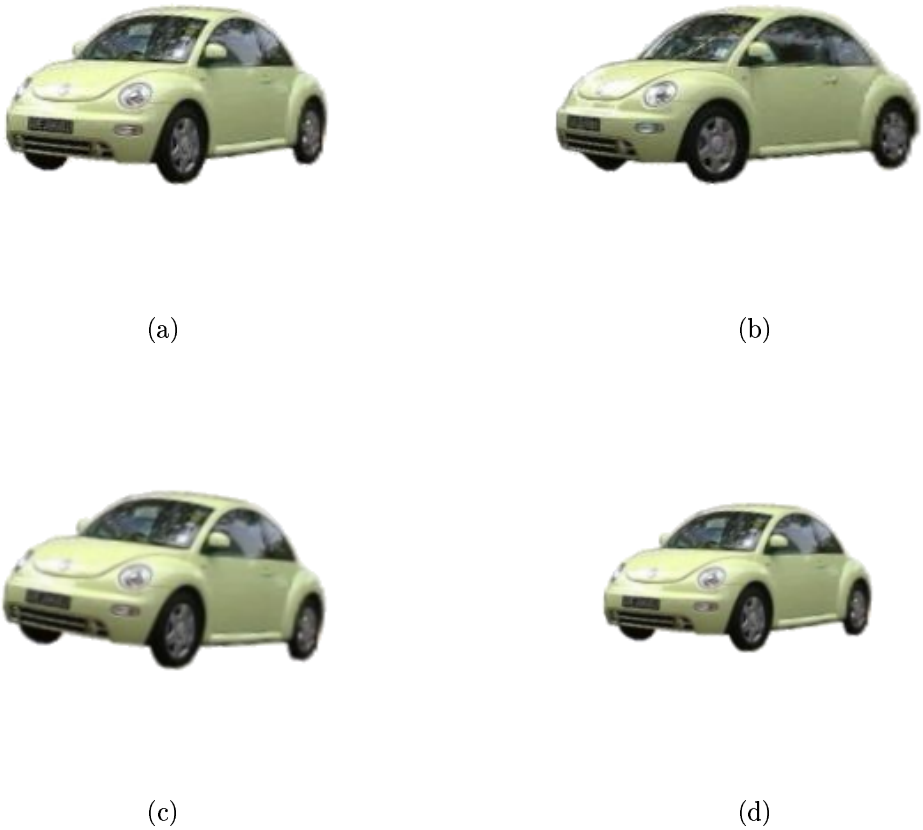
1.1 Challenges in 3D Object Recognition

In this section, we discuss the challenges faced by most object recognition systems. These challenges come from the variation in pose, illumination, scene, shape of 3D objects and their size.

1.1.1 Variation in Pose

In the object recognition scenario that we are considering, the query is a 2D image of an unknown object in the 3D world. The appearance of the unknown object depends on the pose of the camera (Figure 1.1a,b). If the camera is far away from the unknown 3D object, the object will appear smaller in the 2D image (Figure 1.1a,d). If the camera is tilted, the object may appear rotated in the image (Figure 1.1c). Since it is a transformation from 3D to 2D space, the depth information is lost and the object suffers from self occlusion. A 3D object therefore can have an infinite number of 2D views depending on where the viewer is standing and how the camera is oriented.

1.1.1 Variation in Pose



(a) (b)

(c) (d)

Figure 1.1: These are images of the same Volkswagen Beetle. (a) Volkswagen Beetle. (b) Volkswagen Beetle seen from a different location. (c) Volkswagen Beetle seen with a rotation to the right. (d) Volkswagen Beetle seen from a further distance.

(a) (b)

Figure 1.2: (a) Uniformly illuminated image. (b) Non-uniformly illuminated image [15].

1.1.2 Variation in Scene

Depending on applications, the object recognition system needs to take care of varying degrees of scene variations. For example, a recognition system deployed in a production line to locate defective products will have fewer variations in scene compared to a system mounted on an airplane to recognize objects on the ground. There is also the problem of occlusion when the object to be recognized can be blocked by other objects.

1.1.3 Variation in Illumination

Depending on where and when the images are taken, two images of the same 3D scene taken using the same camera may have different illumination. Illumination can be uniform or non-uniform (Figure 1.2). Illumination affects the appearance of the 3D objects. Objects with smooth surface like cars are usually more affected by illumination changes

due to specular reflection.

1.1.4 Variation in Shape and Size of 3D Objects

3D objects in a real-life scene come in a variety of shapes and sizes. Even for objects that are in the same category (e.g. different types of chairs), it is possible that they appear in different forms (Figure 1.2b). The texture of the same class of object can be different too. For example the same type of arm chair may either have a cushioned arm rest or a plastic arm rest. Depending on the type of recognition that is desired, we need to represent the 3D objects accordingly [15].

The variation in pose, scene, illumination and object appearance make the process of recognizing 3D objects from 2D images very difficult. In Section 1.2, we examine some of the existing techniques for 3D object recognition.

1.2 Related Work

Recognition of 3D objects from 2D images has attracted intense research interest in computer vision. There have been a number of literature surveys in this area [2, 20, 22, 33, 39]. In this section, we review some of the existing techniques. We classify the techniques into two categories: model-based and view-based.

1.2.1 Model-Based Object Recognition

Model-based object recognition techniques describe the shape of the 3D object using explicit mathematical or logical descriptions. In general, the shape description of the 3D object is transformed and aligned with the unknown object to find a match that gives the least error. We will discuss three classes of methods that are used to create the shape description.

The first class of methods represents the 3D shape using a computer aided design (CAD) model. The CAD model can be used to generate different views of the 3D objects. The generated view is matched with the unknown object in the query image. While this method is robust to pose changes, the creation of CAD model is however difficult.

The second class of methods represents the 3D shape using the relationship between a collection of simple volumetric primitives like cubes, wedges and cylinders [3, 5, 24, 37]. In the recognition process, the unknown object is segmented into different parts. The relationship between the different parts of the unknown object in the query is compared with those recorded in the database. The disadvantage of this class of methods is that it is difficult for a computer to segment the object automatically.

The third class of methods models the 3D to 2D relationship of lines and corners of the 3D objects [11]. In general, the recognition process transforms the shape representation from the 3D space to the 2D image space and aligns with the 2D features extracted from the 2D query image. The 3D shape that produces the least alignment error gives the best match.

Model-based object recognition is usually limited to 3D objects that can be described using simple geometrical primitives. For 3D objects that are more complicated, another class of techniques called the view-based methods may be more suitable. In the next section, we will discuss these methods.

1.2.2 View-Based Object Recognition

Instead of describing the shape of the 3D object explicitly, view-based object recognition techniques describe the appearance of the 3D objects based on a collection of images taken from different camera pose. The 3D structure of the object is implicitly described by having enough 2D images of the object [21].

The matching for view-based approaches is usually more straight forward than model-based object recognition techniques. It usually only involves comparison between 2D images, either in their raw image form, or in the space of transformed images [4]. We can of course store every possible view of the 3D object in the database and use correlation techniques to find the most similar image in the database. Such approach is however too time consuming and requires large amount of storage space. Researchers have come out with techniques to alleviate these problems.

An approach to reduce the space requirement is to compress the images. A well known image compression technique is Karhunen-Loeve transform which is based on principal component analysis. The images are compressed into a low dimensional subspace called eigenspace. This approach is used in face recognition applications where they are called

the eigenface technique [19, 34]. A query image of an unknown face is projected into a eigenspace that spans the significant variations among the known face images in the database. The class that is the closest to the query in the feature space identifies the query image. This approach is also extended to recognize general objects across pose by Murase et al. [17].

Besides reducing the storage space requirement by compression, there are techniques [14] that are based on features extracted from each view of a 3D object. Statistical approaches have also been used to model some probabilistic properties of the images like gray-level distribution [1, 9].

The approach that is closest to our work is the class of techniques that make use of novel views of the 3D objects generated using a small set of images of the objects in different pose [2, 35]. The novel views of the 3D objects are generated by a linear combination of the small set of images. Recognition is achieved by aligning the corresponding features between the query image and the novel views of the 3D objects, and searching for the one with the least alignment error. There is however a disadvantage of this class of techniques: it requires the extraction of features. Features are generally easier to extract in a controlled environment where the scene variation is limited. In the real world, the scene is almost always cluttered with many other objects, making feature extraction difficult.

Our proposed technique does not require feature extraction. In the next section, we give an overview of our object recognition system.

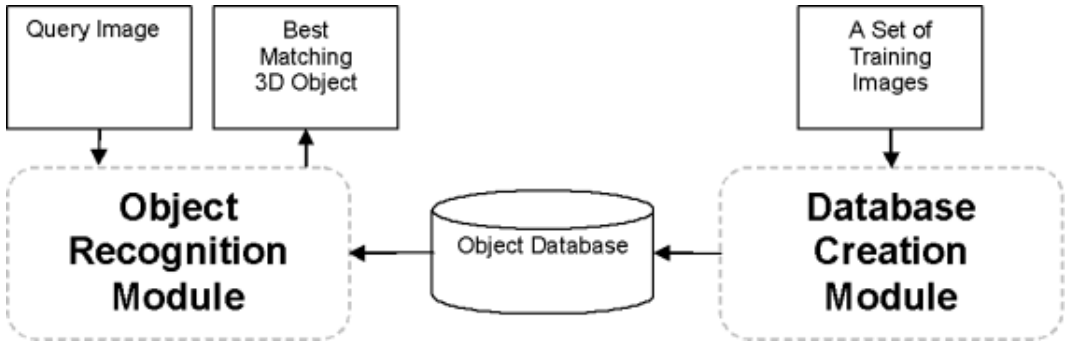


Figure 1.3: General flow of the proposed object recognition system.

1.3 The Proposed Object Recognition System

In this research, we propose a view-based 3D object recognition system that recognizes 3D objects from 2D images. Our system requires only a small set of images of 3D objects in different pose in the database. We use view morphing [28] to generate novel images of the 3D objects, thus predicting views that are not captured in the database. The generated novel images are compared with the 2D query image using the concept of mutual information [36]. Our system performs recognition using the distribution of image attributes like gray-level and color. Hence, we do not need to extract features like edges and corners.

In this research we focus our attention in developing an object recognition system for recognition of cars. Recognition of cars is an interesting and challenging task. Potential applications of car recognition include automated toll system at road gantry, and air

surveillance systems.

Our 3D object recognition system concept is as shown in Figure 1.3. The technique developed is generic and so it can be applied to the recognition of other objects by changing the set of images in the database.

1.4 Thesis Organization

The remainder of this thesis is organized as follows. We give an overview of the different processes in our object recognition system in Chapter 2. In Chapter 3, we discuss view morphing which is a view generation technique that we used to generate novel views of the 3D object. The generated view is compared with the query image using mutual information as a similarity measure. The implementation detail of mutual information is discussed in Chapter 4. In Chapter 5, we show the test results of the experiments conducted. We conclude in Chapter 6 with a discussion of the contributions of this research, followed by future work.

Chapter 2

View-Based 3D Object Recognition System Overview

In this chapter, we discuss the framework of our object recognition system. In Figure 2.1, we show the general flow. There are two modules in our system. One module handles the recognition task and the other handles the database creation.

2.1 The Object Recognition Module

The role of the object recognition module is to recognize the 3D object in the query image. It preprocesses the query image and matches the query image with all the images generated from the 3D object database using view morphing [28]. The similarity between the query and generated image is measured using the concept of mutual information. The object recognition module does not require user involvement in contrast to the database

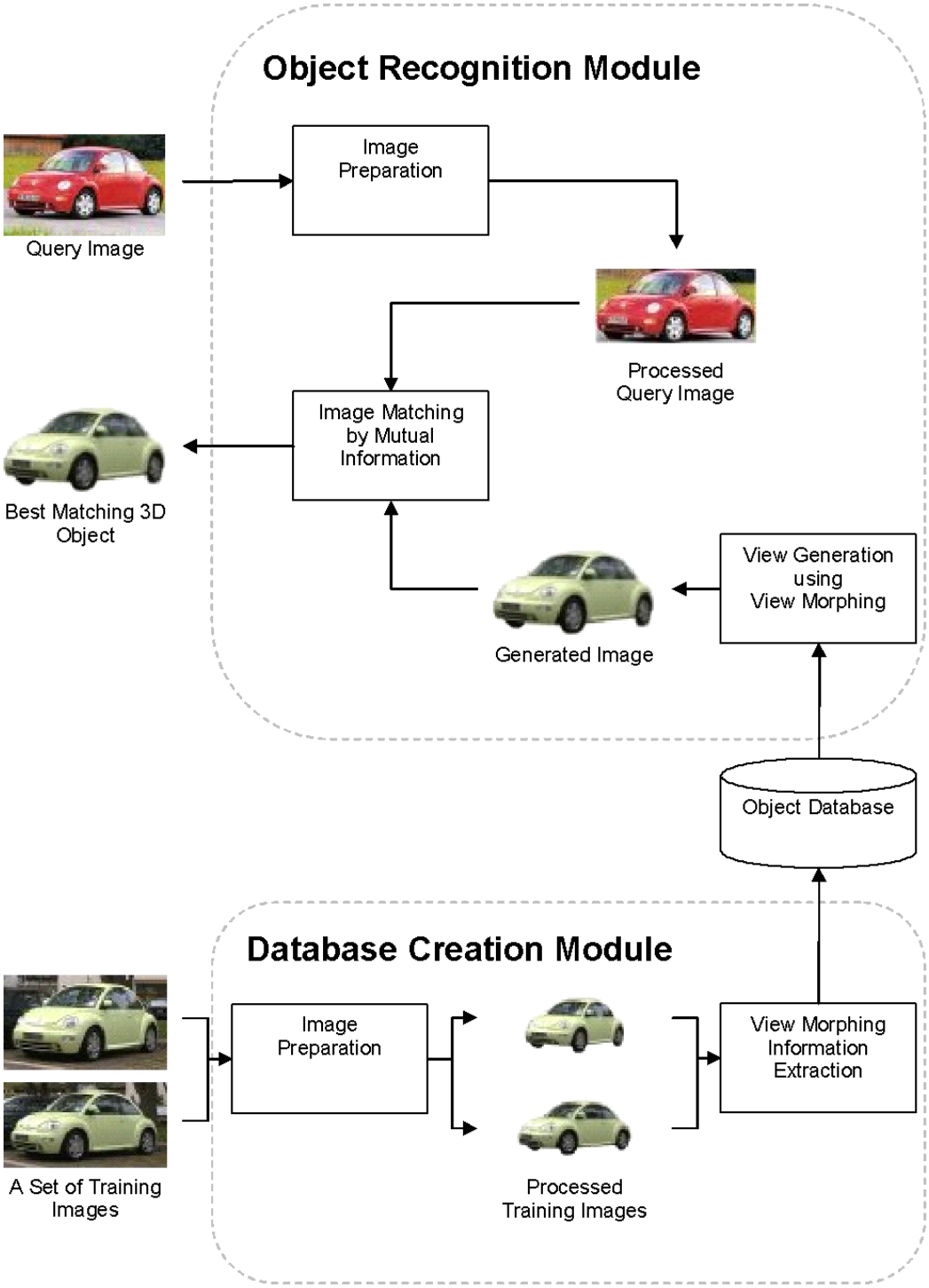


Figure 2.1: View-based 3D object recognition system overview.

creation module. Therefore the recognition process can be run in real time.

The first task in the recognition module is to preprocess the image. Query image comes in different sizes and the car to be recognized can be located anywhere in the image. The preprocessing localizes the car and crops it out from the image. In this research, our focus is on the recognition of the car. The localization of the car is assumed to have been done by motion detection or some other method.

We are using a view-based approach for our object recognition system. Every 3D object is represented by a set of images of the object in different pose. We can of course take many pictures of the 3D object and store them in the database. This will however increase the storage space required. Therefore, we propose to use view morphing [28] to generate novel views of a 3D object on the fly. This technique requires at least two images of the same 3D object taken at different viewing positions. Based on these images, view morphing is able to generate all the geometrically valid views between the original viewing positions. We shall discuss more about view morphing in Chapter 3.

Since view morphing allows us to generate infinite number of views of a 3D object in different pose, the problem of object recognition can be tackled by finding a generated view of the 3D object that best matches the query image. In our object recognition system, we use mutual information to measure the similarity between the query image and the generated views. Mutual information measures the amount of information the query image and the generated image knows about each other. We therefore find the generated view that has the highest mutual information content with the query image.

The object in this generated view will be identified as the object in the query image. We shall discuss more about mutual information in Chapter 4.

2.2 The Database Creation Module

The role of the database creation module is to create the 3D object database for the recognition process. As mentioned in the previous section, we use view morphing to generate views for the matching process. Therefore, we need to store only a small set of images for every 3D object.

The small set of images used to model the 3D objects can be obtained from either digital cameras or video cameras. Since the 3D objects come in different sizes in the image and the background is usually cluttered with many other objects, image preprocessing is needed.

The first step is to segment the object from the background as the background is not relevant to object recognition. This is a difficult process without user involvement. Since our purpose is to create the database, complete automation is not required. Therefore, we remove the background manually using image editing tools. After the background is removed, we scale the size of the 3D object proportionally to the same height.

The last step is to extract the information necessary for view morphing. View morphing needs a number of correspondence points between the basis images to generate new views. The correspondence points are manually selected. It will be inefficient if these corresponding points are to be selected every time view generation is needed. Therefore

we store the corresponding points together with the basis images in the database.

2.3 Summary

We have shown the framework for our view-based 3D object recognition system. Our system represents the 3D objects by a set of images in different pose. Since it is impractical to store every possible views of a 3D object in the database, we propose to use view morphing to generate novel views of the 3D object on the fly. Our recognition problem therefore becomes an image matching problem where our goal is to find a generated view that matches the query image most closely. The best-matching generated view thus gives the type and pose of the object in the query image. We use the concept of mutual information to measure the similarity between the generated image and the query image. A high mutual information measure means the generated image and the query image know a lot about each other, thus they are similar. We discuss view morphing in detail in Chapter 3, followed by a discussion on mutual information as well as some matching results in Chapter 4.

Chapter 3

View Morphing

In our recognition system, we match a query image against a database that contains a set of images showing the 3D objects in different pose. To simply store the images of every pose of a 3D object in the database will require excessive storage space. We propose to use a more intelligent method by storing a small number of images and using the view morphing technique to generate novel views of the 3D object [26, 27, 28, 29].

View morphing relies on measurable attributes that can be computed from a set of basis images to generate physically valid views without the need to explicitly recover the 3D structure. Basically, view morphing can be summarized into three steps: prewarping, interpolating and postwarping. The first step warps the basis images to align the epipolar lines. The warped basis images are used to generate an intermediate version of the new view by a simple linear interpolation in the second step. The last step transforms the image plane of the intermediate version of the new view so that the generated view corresponds to the desired camera pose.

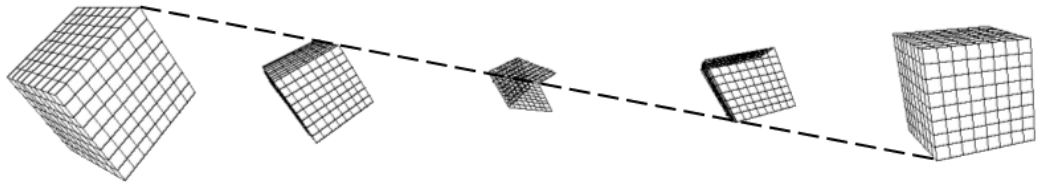


Figure 3.1: Result of a simple linear interpolation.

In Section 3.1, we discuss in detail the concept of view morphing. We discuss the mathematics of view morphing in Section 3.2. After that, we show an example of view morphing in Section 3.3. At the end of the chapter, we give a summary of the view morphing technique.

3.1 Concept of View Morphing

View morphing is different from simple image morphing that uses linear interpolation to generate novel views. A simple linear interpolation between images of the same 3D object viewed from different angles is not sufficient to generate physically valid views because it fails to account for the change in viewpoint between the two given images [28].

Let us illustrate this problem with an example. The leftmost and the rightmost images in Figure 3.1 are the views of a cube taken from different angles. The left cube is tilted anti-clockwise at an angle with respect to the right cube.

The three middle images are obtained by a simple linear interpolation. We can see that the cube is distorted. The dashed line shows the linear path of one feature point

during the transformation. The distortion is most severe if the left cube is rotated 180° with respect to the right cube. In this situation, the morph may collapse into a point at a particular instant of the transformation.

This problem can be solved by aligning the image planes properly before the interpolation process. This is the trick used by Steve Seitz in his seminal paper on view morphing [28]. The image planes are aligned to be parallel to each other before the actual morphing.

We shall begin the discussion in Section 3.1.1 with the simplest case where the image planes are parallel. After that, we proceed to Section 3.1.2 to discuss the more common case where the image planes are not parallel to each other.

3.1.1 Parallel Views

Let us assume we take a photograph I_0 of a 3D object, move the camera in a direction that is parallel to the image plane of I_0, zoom and take another photograph I_1 of the same object (Figure 3.2).

Let us also assume that the optical center C_0 of image I_0 is located at the world origin, the optical center C_1 of image I_1 is moved to position $(C_X, C_Y, 0)$ and the focal length is changed from f_0 to f_1. In this situation, the projection matrices Π_0 of image I_0 and Π_1 of image I_1 are:

$$\Pi_0 = \begin{bmatrix} f_0 & 0 & 0 & 0 \\ 0 & f_0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{3.1}$$

3.1.1 Parallel Views

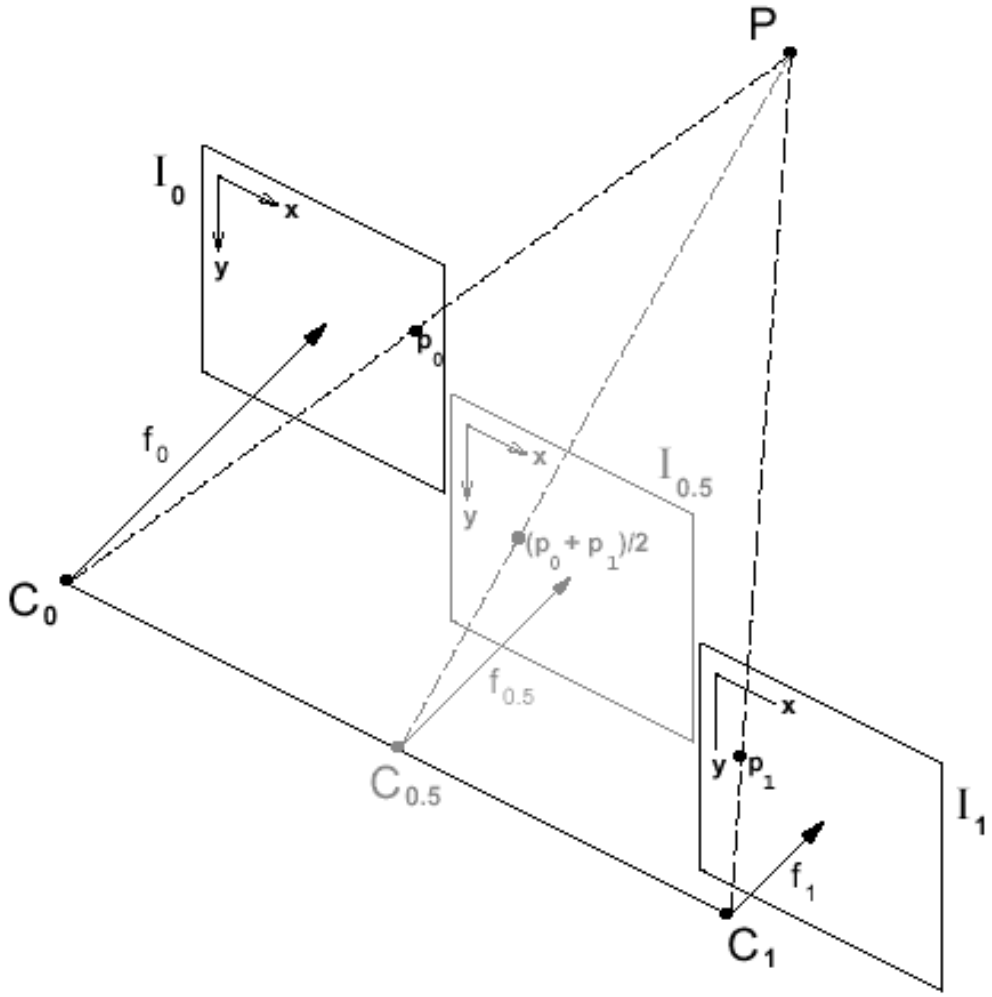


Figure 3.2: Interpolating parallel views.

3.1.1 Parallel Views

$$\Pi_1 = \begin{bmatrix} f_1 & 0 & 0 & -f_1 C_X \\ 0 & f_1 & 0 & -f_1 C_Y \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{3.2}$$

Cameras and views with this form of projection matrices are referred to as parallel cameras and parallel views respectively.

Let $p_0 \in I_0$ and $p_1 \in I_1$ be the projections of a scene point $P = [X\ Y\ Z\ 1]^T$ on image I_0 and image I_1 respectively. A linear interpolation of p_0 and p_1 gives:

$$(1-s)p_0 + sp_1 = (1-s)\frac{1}{Z}\Pi_0 P + s\frac{1}{Z}\Pi_1 P = \frac{1}{Z}\Pi_s P \tag{3.3}$$

where

$$\Pi_s = (1-s)\Pi_0 + s\Pi_1 \tag{3.4}$$

and s is a weighting factor.

Therefore, the linear interpolation corresponds to a new projection matrix Π_s. Π_s is written as the interpolation of the projection matrices Π_0 and Π_1 . The new projection matrix represents a camera with optical center C_s and focal length f_s given by

$$C_s = (sC_X, sC_Y, 0) \tag{3.5}$$

$$f_s = (1-s)f_0 + sf_1 \tag{3.6}$$

Using linear interpolation on the parallel camera configuration, we produced an illusion of moving the camera along the line $\overline{C_0 C_1}$ with continuous zoom. The result of this interpolation conforms to real physical views as it is producing new views of the same object using an interpolated but valid projection matrix.

3.1.2 Non-parallel Views

Very often however, a given pair of photographs is not taken using the parallel cameras configuration. This is because when we take a photograph of a 3D object from different positions, we tend to rotate the camera to keep the object within view. In order to interpolate two non-parallel images, we have to process the images to make them parallel (Figure 3.3).

Suppose the images have projection matrices $\Pi_0 = [H_0|-H_0C_0]$ and $\Pi_1 = [H_1|-H_1C_1]$. We can first establish the homographies H_0 and H_1 between I_0 and I_1 respectively to get the prewarped images $\hat{I}_0$ and $\hat{I}_1$.

This set of prewarped images $\hat{I}_0$ and $\hat{I}_1$ represents views with projection matrices $\hat{\Pi}_0 = [I|-C_0]$ and $\hat{\Pi}_1 = [I|-C_1]$, where I is a 3 x 3 identity matrix. $\hat{\Pi}_0$ and $\hat{\Pi}_1$ indicate that the prewarped images have the property that corresponding points in the two images appear on the same scanline. Therefore, this set of prewarped images is equivalent to the parallel views configuration and we can apply linear interpolation to generate the new view $\hat{I}_s$ without introducing distortion. To convert $\hat{I}_s$ into I_s, we apply the homography H_s to $\hat{I}_s$. The matrix H_s can be easily computed by doing a linear interpolation on H_0 and H_1.

In the next section, we discuss the 3-step algorithm for view morphing based on the concept we have discussed.

3.1.2 Non-parallel Views

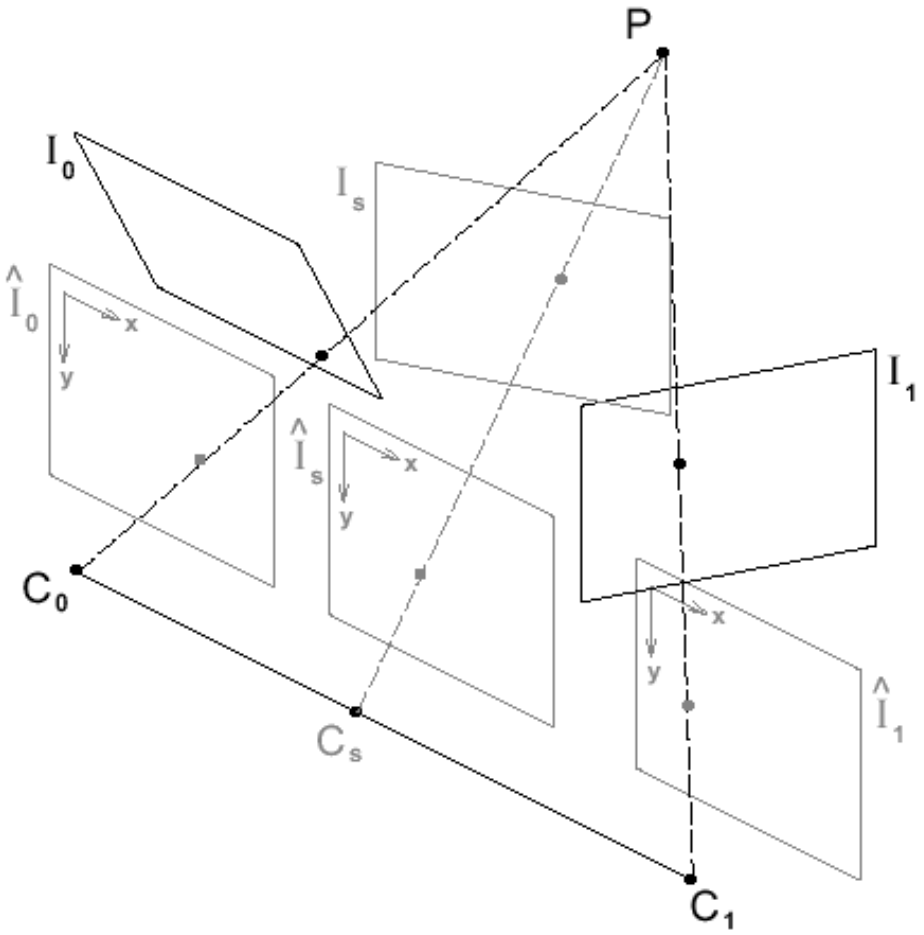


Figure 3.3: Interpolation for a non-parallel configuration.

3.2 The 3-Step Algorithm

The algorithm for view morphing consists of three steps:

1. Prewarp
2. Interpolation
3. Postwarp

The objective of the first step is to transform a non-parallel view configuration to a parallel view configuration so that view generation (step 2 and 3) can be done correctly using a simple linear interpolation. Among these three steps, the prewarp step is the most crucial one as it affects the accuracy of the subsequent steps.

3.2.1 Prewarp

The prewarp step warps the given images so that their image planes are parallel to each other with the scanlines of the images aligned. The prewarp transform is computed from the images without a priori knowledge of the camera matrices. The whole process can be divided into the following steps:

1. Compute the fundamental matrix
2. Compute the epipole
3. Align the image planes
4. Align the scanlines

3.2.1 Prewarp

In order to warp the images correctly, we must find the epipolar geometry that describes the relationship between the pair of images of a 3D scene [12, 31]. In Figure 3.4, we show the epipolar geometry of a pair of images. The epipolar geometry is described by a 3 x 3 rank two matrix F which is called the fundamental matrix [38]. The matrix has this property:

$$p_1^T F p_0 = 0 \tag{3.7}$$

for any pair of image points $p_o \in I_0$ and $p_1 \in I_1$ corresponding to the same 3D scene point.

Fundamental matrix can be computed when at least 7 corresponding points are known [16]. If the matrix is to be computed using a linear algorithm, 8 points are required [8, 13].

Fundamental matrix has this interesting property with the epipoles:

$$F e_0 = F^T e_1 = 0 \tag{3.8}$$

where e_0 is the epipole in the first image and e_1 is the epipole in the second image. A brief explanation of the epipoles is given in Figure 3.4.

The epipoles play an important role in aligning the images in the prewarp step. Suppose the given image planes are parallel to each other and the scanlines are properly aligned, the epipoles e_0 and e_1 will take the form $e_0 = [e_0^x \ 0 \ 0]^T$ and $e_1 = [e_1^x \ 0 \ 0]^T$ respectively for some unknown constants e_0^x and e_1^x.

With this property, we can find a rotation matrix to warp the images so that their image planes are parallel to each other.

3.2.1 Prewarp

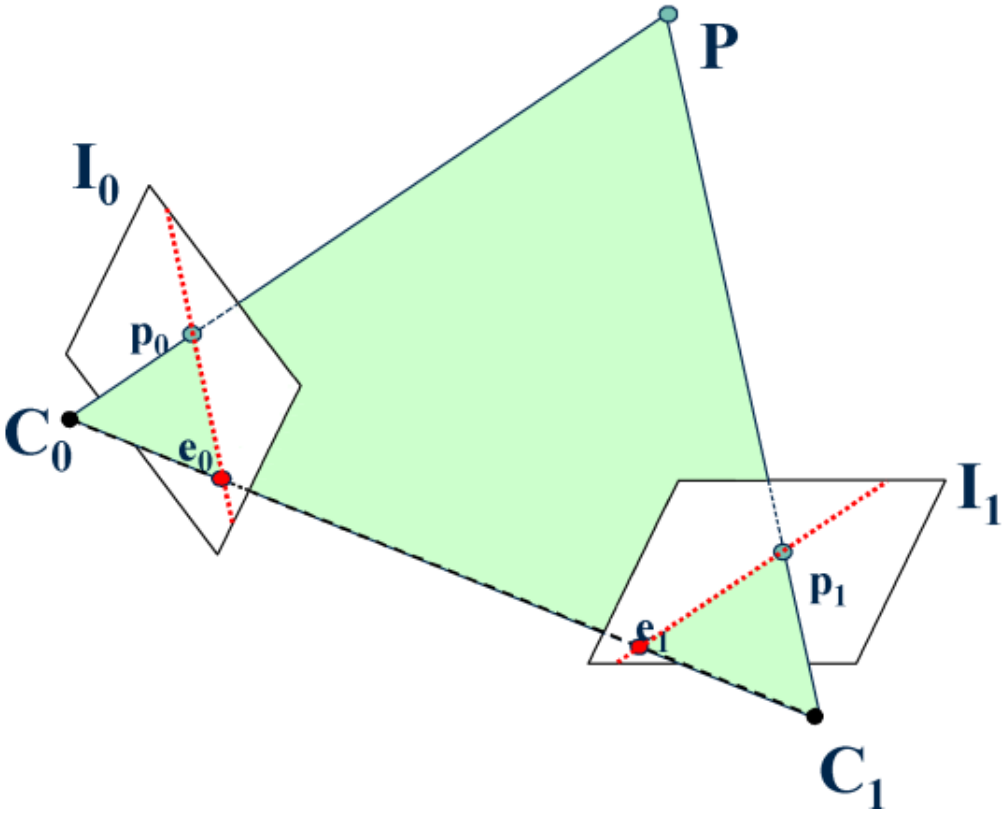


Figure 3.4: Point P in 3D world projects to a point p_0 on image plane I_0 and a point p_1 on image plane I_1. The point of intersection between the line formed by the two optical centers C_0 and C_1 and the image plane forms the epipole. e_0 is the epipole in image plane I_0 and e_1 is the epipole in image plane I_1. The plane that contains the 3D point P, the two optical centers C_0 and C_1 and the two image points p_0 and p_1 forms an epipolar plane. The intersection of the epipolar plane and the two image planes I_0 and I_1 forms two lines called the epipolar lines. These are the lines where the 3D point P will lie in the images.

3.2.1 Prewarp

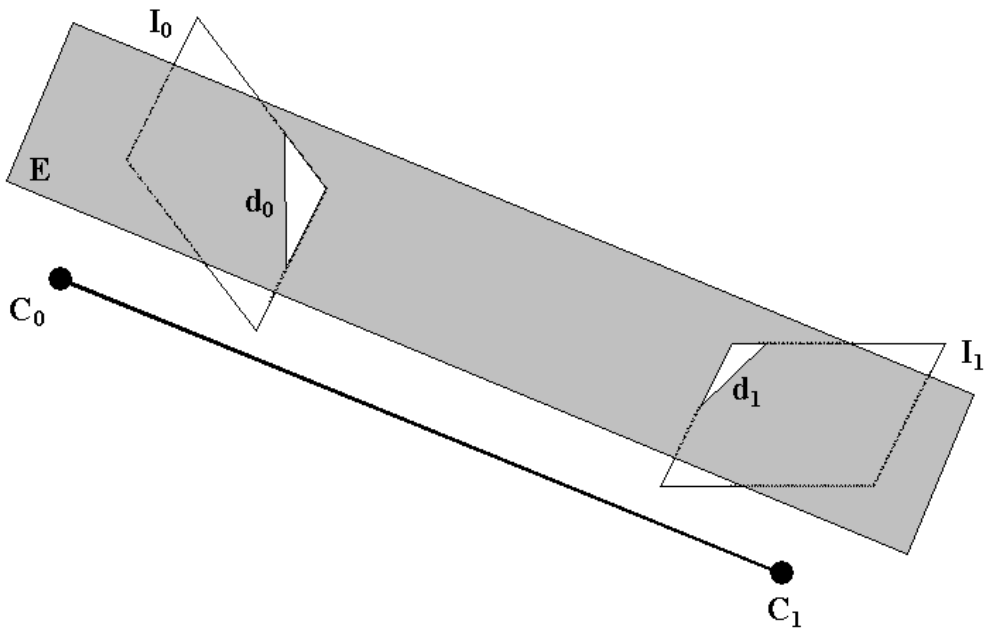


Figure 3.5: Selection of rotation axis d_0 and d_1.

Let E be a plane parallel to the line $\overline{C_0C_1}$ which connects the two optical centers, as shown in Figure 3.5. We rotate the image planes I_0 and I_1 such that they are parallel to the plane E. This is achieved by rotating each image plane I_i about a line d_i, where d_i is the intersection of the plane E and the image I_i.

We do not need to explicitly select the plane E. Instead we only need to select the appropriate d_i for the image I_i. For convenience, this line is assumed to pass through the image origin.

We know that the epipole e_0 is the point where the line $\overline{C_0C_1}$ intersects the image plane I_0. By rotating about the line d_0, the image plane I_0 is made parallel to the line $\overline{C_0C_1}$ and the epipole is at infinity. Therefore, we should find a rotation matrix such that

the new epipole $\hat{e}_0 = R^{d_0}_{\theta_o} e_0$ has the form $\hat{e}_0 = [\hat{e}^x_0 \ \hat{e}^y_0 \ 0]^T$.

The rotation matrix derived using the Rodrigues Formula [18] is given by:

$$R^{d_0}_{\theta_0} = \begin{bmatrix} 1-(d^y_0)^2(1-\cos\theta_0) & d^x_0 d^y_0(1-\cos\theta_0) & d^y_0 \sin\theta_0 \\ d^x_0 d^y_0(1-\cos\theta_0) & 1-(d^x_0)^2(1-\cos\theta_0) & -d^x_0 \sin\theta_0 \\ -d^y_0 \sin\theta_0 & d^x_0 \sin\theta_0 & 1-((d^y_0)^2+(d^x_0)^2)(1-\cos\theta_0) \end{bmatrix} \tag{3.9}$$

By fixing $(d^x_0)^2 + (d^y_0)^2 = 1$, we can simplify $R^{d_0}_{\theta_0}$ to:

$$R^{d_0}_{\theta_0} = \begin{bmatrix} (d^x_0)^2 + (1-(d^x_0)^2)\cos\theta_0 & d^x_0 d^y_0(1-\cos\theta_0) & d^y_0 \sin\theta_0 \\ d^x_0 d^y_0(1-\cos\theta_0) & (d^y_0)^2 + (1-(d^y_0)^2)\cos\theta_0 & -d^x_0 \sin\theta_0 \\ -d^y_0 \sin\theta_0 & d^x_0 \sin\theta_0 & \cos\theta_0 \end{bmatrix} \tag{3.10}$$

Using the requirement that $[\hat{e}^x_0 \ \hat{e}^y_0 \ 0]^T = R^{d_0}_{\theta_o} e_0$, the desired rotation angle is [28]:

$$\theta_0 = tan^{-1}\left(\frac{e_z}{d^y_0 e^x_0 - d^x_0 e^y_0}\right) \tag{3.11}$$

We should choose an axis where the non-linear distortion in the prewarped images is minimized. One approach is to minimis the $|\theta_0|$. To minimize $|\theta_0|$, we are actually minimizing the following expression:

$$\left|\frac{e_z}{d^y_0 e^x_0 - d^x_0 e^y_0}\right| \tag{3.12}$$

Therefore, the optimal value of $|\theta_0|$ is obtained when $d^x_0 = \alpha e^y_0$ and $d^y_0 = \alpha e^x_0$, where $\alpha = \frac{1}{\sqrt{(e^x_0)^2+(e^y_0)^2}}$.

Once the rotation axis d_0 for image I_0 is determined, we can use the fundamental matrix F to determine the corresponding rotation axis d_1 for the image I_1 using the

following equation:

$$i_1 = Fd_0 \tag{3.13}$$

where i_1 is the corresponding point on image I_1 corresponding to the rotation axis d_0. If $[x\ y\ z]^T = Fd_0$, then $d_1^x = \alpha y$ and $d_1^y = -\alpha x$, where $\alpha = \frac{1}{\sqrt{x^2+y^2}}$.

Having determined the rotation axis d_1 for the image I_1, the rotation matrix for image I_1 can be easily determined in the same way as that of the image I_0.

After prewarping the images by the rotation matrices, we have made the image planes parallel to the plane E and the new epipoles take the form $[\hat{e}_i^x\ \hat{e}_i^y\ 0]^T$. In order to make the scanlines horizontal to each other, we perform a rotation with respect to the z-axis by:

$$R_{\phi_i} = \begin{bmatrix} \cos\phi_i & -\sin\phi_i & 0 \\ \sin\phi_i & \cos\phi_i & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.14}$$

where the rotation angle ϕ_i is computed using:

$$\phi_i = -\tan^{-1}(\hat{e}_i^y / \hat{e}_i^x) \tag{3.15}$$

After applying these image plane rotations, the new fundamental matrix will have the following form up to a scale factor:

$$\tilde{F} = R_{\phi_1} R_{\theta_1}^{d_1} F R_{-\theta_0}^{d_0} R_{-\phi_0} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & a \\ 0 & 1 & b \end{bmatrix} \tag{3.16}$$

3.2.1 Prewarp

By applying the rotation matrix $R_{\phi_i} R_{\theta_i}^{d_i}$ to I_i, we obtain a set of images with horizontal epipolar lines. It is however possible that the epipolar lines appear in row reversed order. In this case, an additional 180° rotation to the matrix R_{ϕ_i} is needed.

The final step to complete the prewarp is to convert the fundamental matrix $\tilde{F}$ to the form $\hat{F}$ defined up to a scale factor [28]. Fundamental matrix in the form $\hat{F}$ indicates that the two image planes are parallel to each other.

$$\hat{F} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \tag{3.17}$$

This can be easily done by applying the following matrix T to scale the image I_1 vertically, followed by a translation:

$$T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -a & -b \\ 0 & 0 & 1 \end{bmatrix} \tag{3.18}$$

We can easily verify that $(T^{-1})^T \tilde{F} = \hat{F}$. After performing all the steps above, we have computed the prewarp transforms H_0 and H_1:

$$H_0 = R_{\phi_0} R_{\theta_0}^{d_0} \tag{3.19}$$

$$H_1 = T R_{\phi_1} R_{\theta_1}^{d_1} \tag{3.20}$$

3.2.2 Interpolation

After the prewarping process, the image planes are parallel to each other and all the corresponding points between the two images have been aligned on the same scanlines and in the same order. This condition makes the generation of new views a lot simpler, and a simple linear interpolation will not introduce distortion in views.

We need to determine the correspondence between the images. Having determined the correspondences, we can linearly interpolate the position and color of corresponding points accordingly.

There are many ways to determine the correspondence between the images. One very simple method is to create a simple user interface to indicate the correspondence. Since we do not really need all the correspondent points to generate a new view, the boundary of the regions in the image is sufficient for generating a new view.

3.2.3 Postwarp

The final step of the 3-step algorithm is to put the generated view into the correct image plane so that the view corresponds to the desired camera pose. For every view that we generate, we have to provide the corresponding prewarp transform matrix H_s. The simplest method of computing H_s is to linearly interpolate the prewarp transform matrices H_0 and H_1 of the images I_0 and I_1 respectively. Therefore, the prewarp transform of the image I_s is:

$$H_s = (1 - s)H_0 + sH_1 \tag{3.21}$$

(a) (b)

Figure 3.6: These are two different views of the same Volkswagen Beetle. We shall refer to (a) as left image and (b) as right image in future reference.

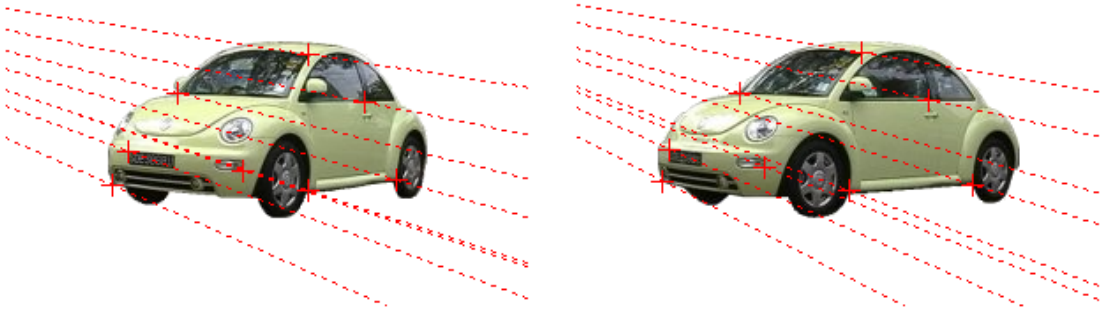
where s is a scale factor.

By applying this prewarp transform H_s, we can project the generated image to correspond to the desired camera pose.

We have completed our discussion of the 3-step algorithm. In the next section, we show an example of view generation using view morphing.

3.3 Example of View Morphing

In this section, we show an example of view morphing based on the algorithm described in Section 3.2. We use two images of the same Volkswagen Beetle taken at different viewing positions (Figure 3.6). The background is not required for view morphing, therefore it is removed.



(a) (b)

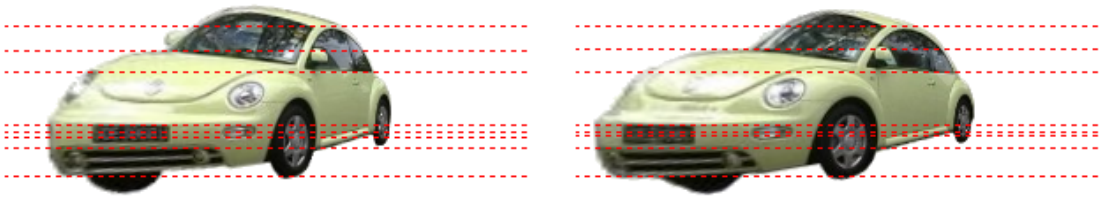
Figure 3.7: These images show the epipolar lines in red. These lines will converge to a point when they are extended. The convergence point is the epipole.

In Figure 3.7, we show the epipolar lines of the 8 corresponding points as red dotted lines. These 8 corresponding points are used to compute the fundamental matrix using a linear algorithm [8, 13]. If we extend the epipolar lines, they will converge to a point which is the epipole.

The first step of the 3-step algorithm is to warp the images so that the image planes are parallel to each other and the scanlines are horizontal and aligned. In Figure 3.8, we show the result of the prewarp step. The epipolar lines are aligned though the Volkswagen Beetle looks deformed after the warping.

The image planes of the left and right Volkswagen Beetle are now in the parallel views configuration that we have mentioned in Section 3.1.1. We can now generate novel views that conformed to the real physical world.

In Figure 3.9, we generated a series of novel views of the Volkswagen Beetle. Fig-



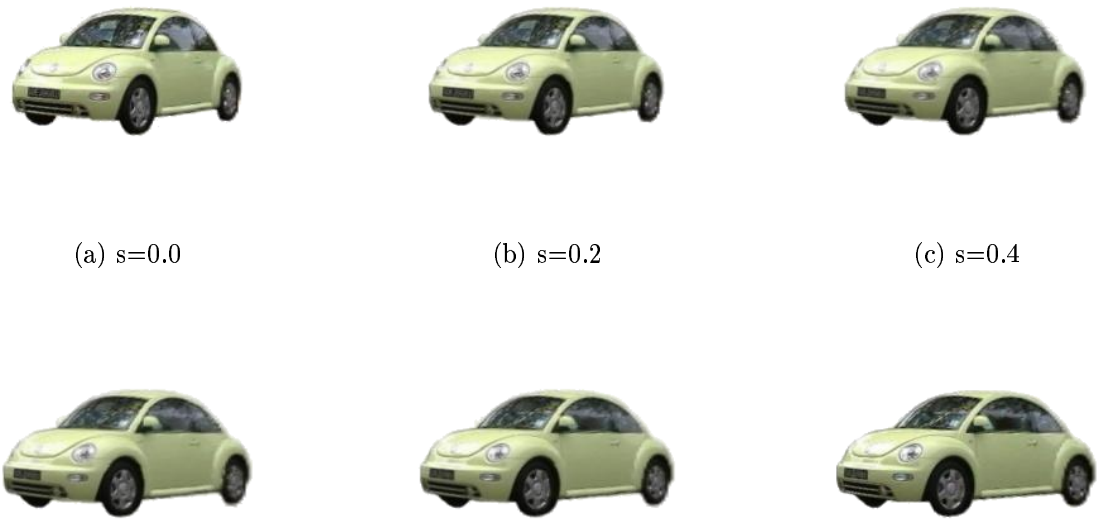
(a) (b)

Figure 3.8: The epipolar lines are shown as red dotted lines. These lines are all parallel to each other and the corresponding lines between the left and the right images are aligned.

ure 3.9a is the original left image of the Volkswagen Beetle. Figure 3.9f is the original right image of the Volkswagen Beetle. The parameter s under every image is the relative distance with respect to the left image. The left image has an s value of 0 and the right image has an s value of 1. We see a smooth transition from the left image to the right image, depicting a rotation of the Volkswagen Beetle.

3.4 Summary

View morphing is a view generation technique proposed by Steve Seitz in SIGGRAPH 1996 [28]. This technique consists of three steps: prewarp, interpolation, and postwarp. The main idea is to convert the given images into a parallel views configuration, perform simple linear interpolation in the parallel views configuration to generate a new view



(a) s=0.0 (b) s=0.2 (c) s=0.4

(d) s=0.6 (e) s=0.8 (f) s=1.0

Figure 3.9: Parameter s is the relative viewing point with respect to the left image. The views corresponding to s=0.0 and s=1.0 are the original views that are input to the view generation system.

and then project the generated view into the desired viewing position. It is indeed because of this property of generating views that conformed to the real physical world that we propose to use view morphing for object recognition. By using view morphing, we can generate infinite number of views of a 3D object. Thus, the problem of 3D object recognition has become a problem of image matching. In the next chapter, we will discuss mutual information, which is the proposed similarity measure for our object recognition system.

Chapter 4

Matching By Mutual Information

As mentioned in Chapter 2, the problem of 3D object recognition can be transformed to a 2D image matching problem by using the view morphing technique to generate novel and physically valid views of a 3D object. In this chapter, we discuss the measure that we use to determine the amount of similarity between the query image and the generated views of the 3D objects. The similarity measure is mutual information.

Mutual information measures the amount of information one random variable knows about another random variable [7]. In our context of image matching, the random variables are the query image and the generated images. Our goal is to maximize the mutual information between the query image and the generated image [6, 9, 10, 36]. If mutual information between a query image and a generated image is high, it implies that the generated image knows a lot about the query image, thus it is likely that the unknown 3D object in the query image matches the 3D object in the generated image.

We shall begin the discussion with some basic concepts of stochastic processing in

Section 4.1. After that we discuss how view morphing and mutual information are integrated in our 3D object recognition system in Section 4.2. In Section 4.3, we show some matching results using mutual information.

4.1 Random Variable, Entropy and Mutual Information

In this section, we discuss some basic concepts of stochastic processing. After that, we will discuss the use of a combination of stochastic techniques and view morphing for our 3D object recognition system.

A random variable is a function that associates a unique value for every outcome of an experiment [23]. A discrete random variable, as its name implies, can only be assigned discrete numbers. The assigned value varies from trial to trial as the experiment is repeated.

Let Ω_X be the set of values that a discrete random variable X can be assigned. The probability of an event x_i happening is written as $P(X = x_i)$ or $P(x_i)$. The sum of all event probabilities must be equal to 1:

$$\sum_{x_i \in \Omega_X} P(X = x_i) = 1 \tag{4.1}$$

Entropy measures the randomness of a random variable. The entropy for a discrete random variable is defined as:

$$H(X) = -\sum_{x_i \in \Omega_X} P(X = x_i) log(P(X = x_i)) \tag{4.2}$$

For the case where $P(X = x_i) = 0$, its entropy is defined to be 0.

In the case where there are two random discrete variables, X and Y, we can measure the joint distribution, $P(X, Y)$, which summarizes the co-occurrence of events between the variables. Using the joint distribution, the joint entropy of two discrete random variables is defined as:

$$H(X,Y) = -\sum_{x_i \in \Omega_X} \sum_{y_j \in \Omega_Y} P(X = x_i, Y = y_j) log(P(X = x_i, Y = y_j)) \tag{4.3}$$

Joint entropy is used to calculate the mutual information between two discrete random variables. Mutual information is defined as [7, 30, 36]:

$$I(X,Y) = H(X) + H(Y) - H(X,Y) \tag{4.4}$$

where $H(X)$ is the entropy of the discrete random variable X, $H(Y)$ is the entropy of the discrete random variable Y, and $H(X, Y)$ is the joint entropy of the discrete random variables X and Y.

When the mutual information is high, it indicates that the first random variable contains a lot of information about the second variable. Thus the two random variables have high similarity. If the mutual information is low, the first random variable contains little information about the second random variable. Therefore, they have low similarity.

4.2 Integrating Mutual Information and View Morphing for 3D Object Recognition

In Section 3.3 of Chapter 3, we have demonstrated view morphing's ability to generate novel views that conformed to the physical world. This allows us to transform a 3D object recognition problem into an image matching problem. In this section, we explain how mutual information and view morphing are integrated into our 3D object recognition system. We will also explain the method used to model the distribution of image attributes.

4.2.1 The Matching Algorithm

We begin the discussion with the matching algorithm for our recognition system.

Let Q be a query image that contains an unknown object in an unknown pose. Q will be compared with the different views DB of every database object generated using the view morphing technique. The similarity between the query image and the generated view is determined using mutual information:

$$I(DB, Q) = H(DB) + H(Q) - H(DB, Q) \tag{4.5}$$

where $H(DB)$ is the entropy of the generated view, $H(Q)$ is the entropy of the query image, and $H(DB, Q)$ is the joint entropy between the generated view and the query image.

Basically, our object recognition system generates different images of the 3D objects

in the database and performs a sequential comparison with the query image. The best matching view for every object in the database is recorded together with its corresponding mutual information. At the end of the matching process, the result list is returned, sorted based on mutual information content.

4.2.2 Image Attributes

In our recognition system, we make use of the distribution of image attributes like gray-level and color [32].

When using the gray-level attribute, we perform quantization to group the pixels into 11 bins. The goal of the quantization is to group pixels that are close in intensity so that minor perturbation due to noise in the original 256 intensity levels will not cause a mismatch.

When using the color attribute, the color image is converted from the RGB (Red, Green, Blue) color space into the HSI (Hue, Saturation, Intensity) color space. We group the pixels together based on their color. We have 11 different color bins, namely red, yellow, green, cyan, blue, magenta, black, dark gray, medium gray, light gray and white.

4.2.3 Histogram and Co-occurrence Matrix

We model the distribution of image attributes using a histogram. A histogram is easy to construct and is used in many other applications [25, 32]. In our implementation, we count the occurrence of each attribute value in the image and increment the correct bin

4.2.3 Histogram and Co-occurrence Matrix

1	1	1
1	2	2
3	3	2

(a)

1	1	2
2	3	3
2	1	2

(b)

	1	2	3
1	2/9	2/9	0
2	0	1/9	2/9
3	1/9	1/9	0

(c)

Figure 4.1: a) Color distribution of an image, X, b) Color distribution of an image, Y, and c) Co-occurrence matrix for the images X and Y.

accordingly. The histogram is then divided by the total number of pixels in the image to normalize its value to fall between 0 and 1.

The joint distribution is modelled using a co-occurrence matrix in our system. Co-occurrence matrix has been used in statistical methods of texture analysis [12, 31]. It captures the spatial dependency of gray-level values which forms the perception of texture. In our recognition system, we model the co-occurrence of gray-level or color between two images (Figure 4.1). The co-occurrence matrix can be expressed as [10]:

$$CM_{XY}(i,j) = \frac{N_{i,j}}{N_S} \tag{4.6}$$

where X and Y are two images, $N_{i,j}$ is the frequency of pairs of pixels where the pixel in image X has a value i and the pixel in image Y has a value j, and N_S is the total number of pixels in image X.

It is observed that the gray-level or color distribution of a 3D object forms a spatial

relationship in the image. By using the co-occurrence matrix to capture the joint distribution of gray-level or color between two images, we are determining whether these images share the same spatial relationship in their gray-level or color distribution. The joint distribution will be more random when the spatial relationship is different. In this case, the mutual information will be decreased.

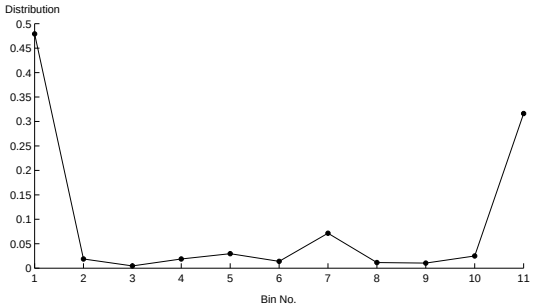
We have discussed the matching algorithm for our 3D object recognition system. We have also discussed the representation of the distribution of image attributes like gray-level or color using histogram and co-occurrence matrix. In the next section, we show some matching results using mutual information.

4.3 Matching Result Using Mutual Information

In this section, we show some matching results using mutual information. We use a histogram to model the distribution of color in an image and a co-occurrence matrix to model the joint distribution of color between the images. In this section, as we are only showing the matching results between the images, there is no view generation nor searching in these examples.

4.3.1 Matching a red Volkswagen Beetle with itself

We begin the discussion with the simplest case: matching an image of a red Volkswagen Beetle with itself (Figure 4.2). The entropy of the image is 2.0569. The peaks formed a diagonal in the joint distribution plot (Figure 4.3). The joint entropy is 2.0569. The



(a) (b)

Figure 4.2: (a) Red Volkswagen Beetle. (b) Its corresponding color distribution with entropy=2.0569.

mutual information computed using the Equation 4.5 is 2.0569. This is the maximum mutual information that is possible for the red Volkswagen Beetle.

4.3.2 Matching a red Volkswagen Beetle with a uniformly colored image

We match an image of the red Volkswagen Beetle (Figure 4.2) with a uniformly colored image (Figure 4.4). The entropy of image of the red Volkswagen Beetle is 2.0722. The entropy of the uniformly colored image is zero. We observed that the peaks formed a row in the joint distribution plot (Figure 4.5). The joint entropy between these images is 2.0722. The mutual information is zero. This implies that the images know nothing about each other. Thus, they are dissimilar.

4.3.2 Matching a red Volkswagen Beetle with a uniformly colored image

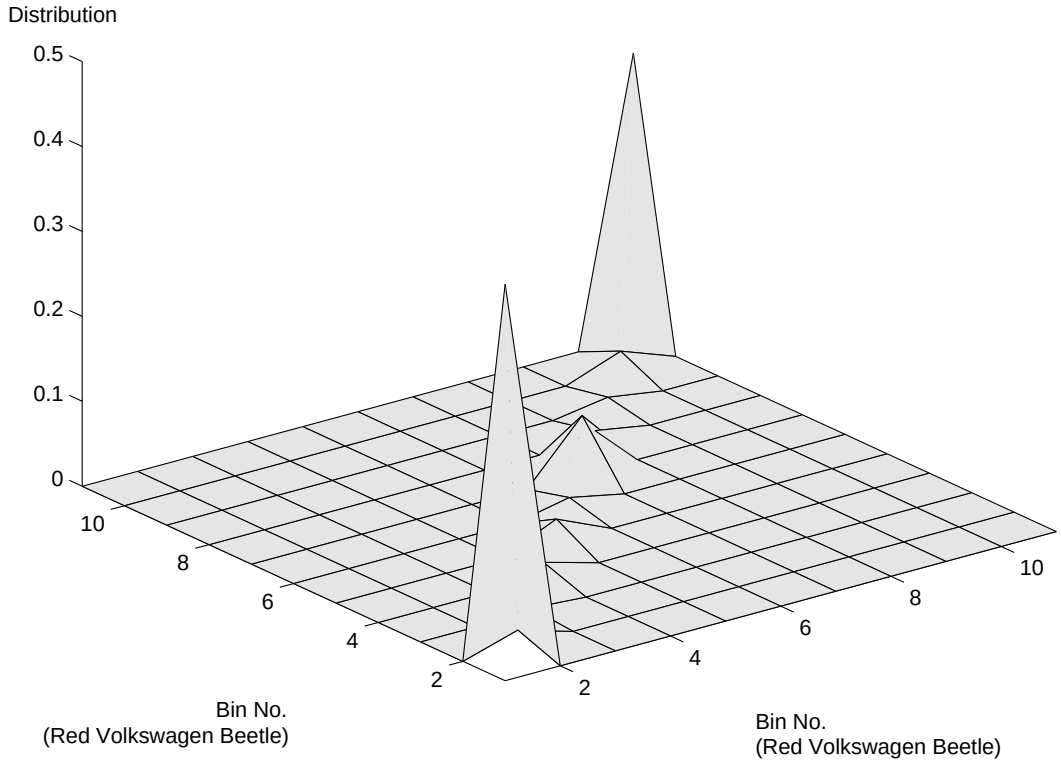
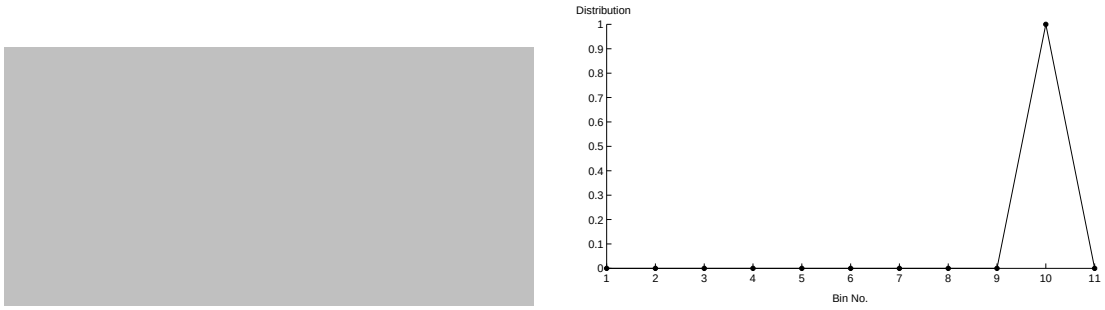


Figure 4.3: Joint distribution between the red Volkswagen Beetle and itself with joint entropy=2.0569.



(a) (b)

Figure 4.4: (a) Uniformly-colored image. (b) Its corresponding color distribution with entropy = 0.

4.3.3 Matching a red Volkswagen Beetle with a noisy image

We will now match the image of the red Volkswagen Beetle (Figure 4.2) with a very noisy image of randomly distributed color (Figure 4.6). The entropy of the noisy image is 3.0336. The joint distribution between the image of the red Volkswagen Beetle and the noisy image is shown in Figure 4.7. The peaks are scattered. The joint entropy is 5.0989. The mutual information is 0.0068. Though this value is not zero, we can still conclude that the image of the red Volkswagen Beetle and the noisy image is very dissimilar.

4.3.3 Matching a red Volkswagen Beetle with a noisy image

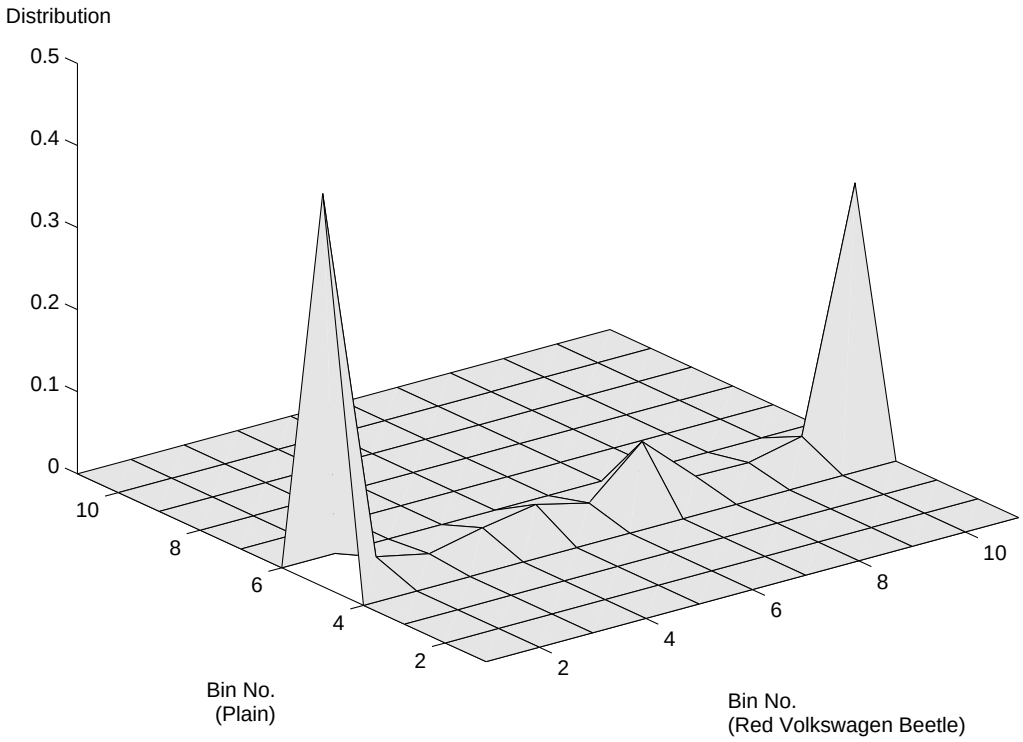
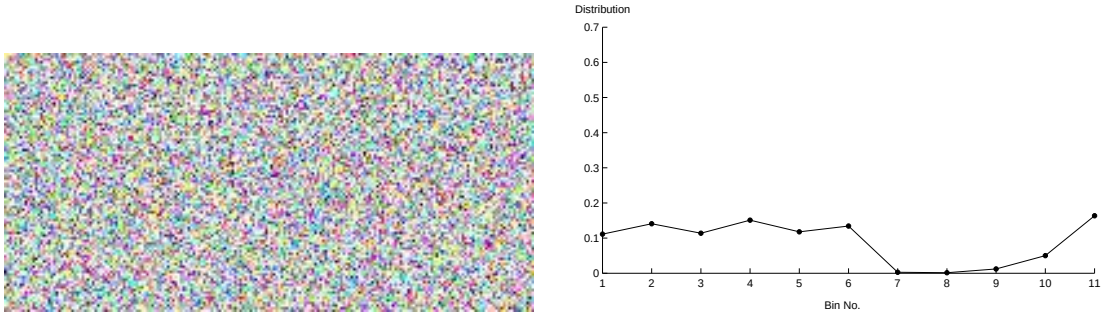


Figure 4.5: Joint distribution between the red Volkswagen Beetle and the uniformly-colored image with joint entropy=2.0722.



(a) (b)

Figure 4.6: (a) A noisy image. (b) Its corresponding color distribution with entropy=3.0336.

4.3.4 Matching a red Volkswagen Beetle with a green Volkswagen Beetle

In the next example, we are matching the image of the red Volkswagen Beetle (Figure 4.2) with an image of a green Volkswagen Beetle (Figure 4.8). The entropy for the red Volkswagen Beetle image and green Volkswagen Beetle image is 2.0625 and 2.1845 respectively. The joint distribution is shown in Figure 4.9. We observed that the peaks are less scattered and lie mostly along the diagonal. The joint entropy is 3.5509. The mutual information is 0.6961. The mutual information is higher than the previous two examples. Therefore, we conclude that in comparison with the uniformly-colored image and the noisy image, the image of the green Volkswagen Beetle is more similar to the image of the red Volkswagen Beetle.

4.3.4 Matching a red Volkswagen Beetle with a green Volkswagen Beetle

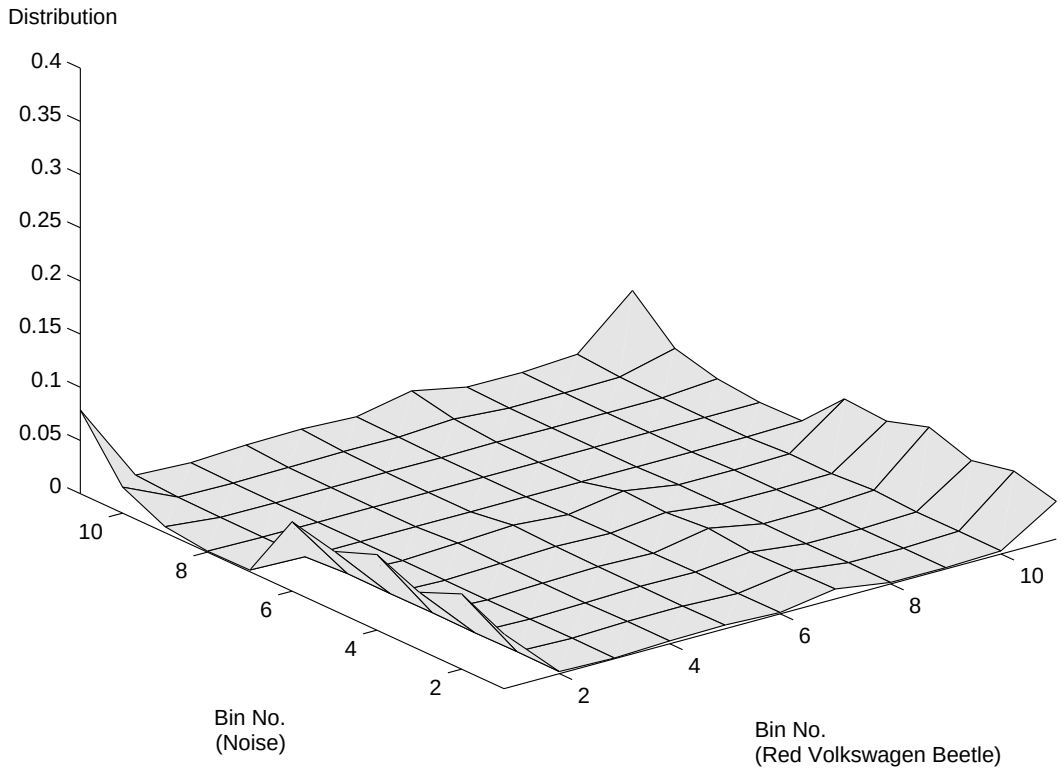
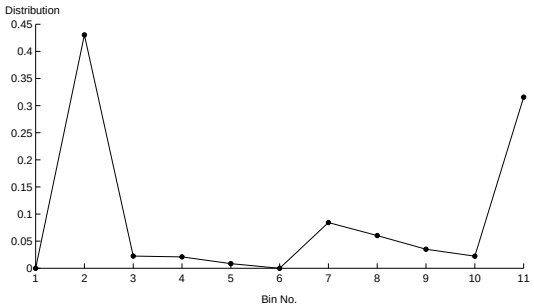


Figure 4.7: Joint distribution between the red Volkswagen Beetle and the randomly-colored image with joint entropy=5.0989.



(a) (b)

Figure 4.8: (a) Green Volkswagen Beetle. (b) Its corresponding color distribution with entropy=2.3437.

4.3.5 Matching a red Volkswagen Beetle with a bronze Mazda

In this example, we will show the result of matching the image of the red Volkswagen Beetle (Figure 4.2) with an image of a bronze Mazda (Figure 4.10). The entropy for the red Volkswagen Beetle image and the bronze Mazda image is 2.0572 and 2.8036 respectively. The joint distribution is shown in Figure 4.11. The peaks are also very scattered. The joint entropy is 4.5061. The mutual information is 0.3547. Since it is a different car, the mutual information naturally is lower than the mutual information between the red and green Volkswagen Beetles.

4.3.5 Matching a red Volkswagen Beetle with a bronze Mazda

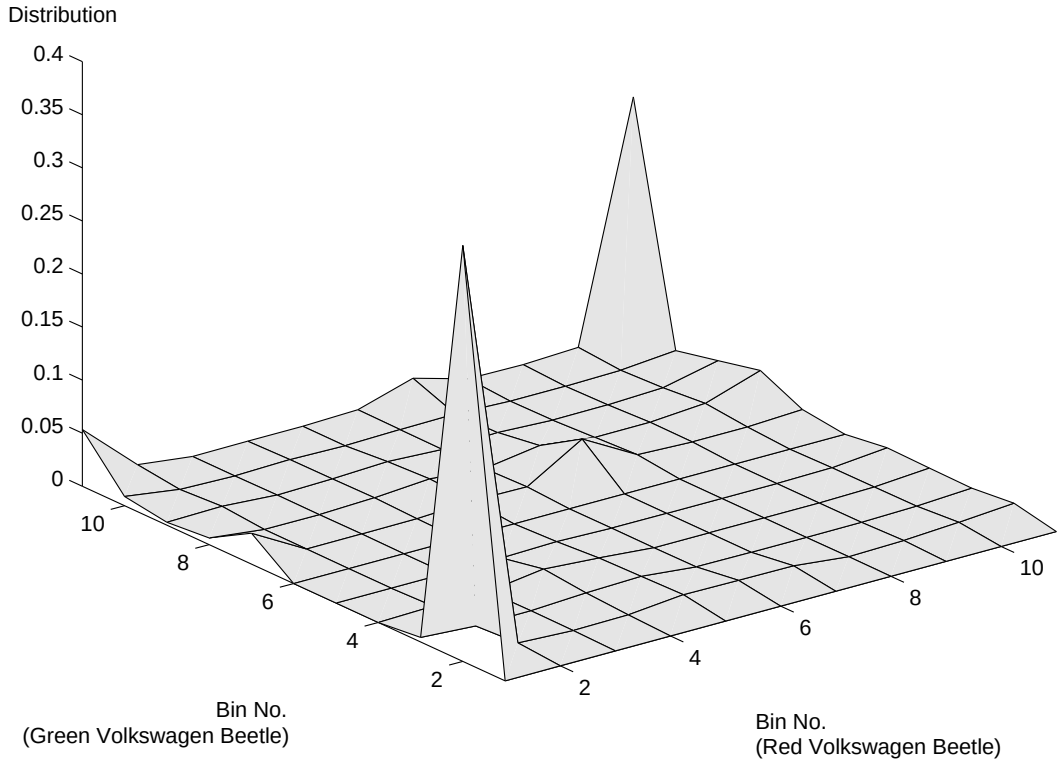
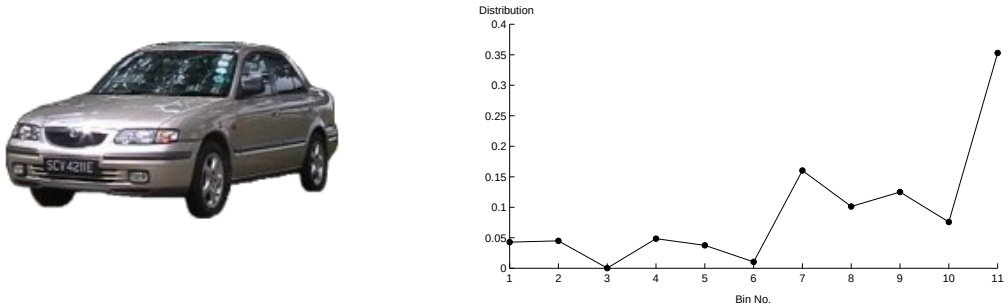


Figure 4.9: Joint distribution between the red Volkswagen Beetle and the green Volkswagen Beetle with joint entropy=3.5509.



(a) (b)

Figure 4.10: (a) Bronze Mazda. (b) Its corresponding color distribution with entropy=2.8036.

4.3.6 Matching a red Volkswagen Beetle with a distorted red Volkswagen Beetle

In the final example, we compare the image of the red Volkswagen Beetle (Figure 4.2) with an image of a distorted red Volkswagen Beetle (Figure 4.12). The entropy of the red Volkswagen Beetle image and the distorted red Volkswagen Beetle image is 2.0569 and 2.0177 respectively. The joint distribution is shown in Figure 4.13. Although both images have almost the same color distribution, the peaks are scattered in the joint distribution. The joint entropy is 3.8489. The mutual information is 0.2478. This example shows that the method we used to compute the joint distribution describes the similarity between the spatial relationship in the color distribution of the images.

4.3.6 Matching a red Volkswagen Beetle with a distorted red Volkswagen Beetle

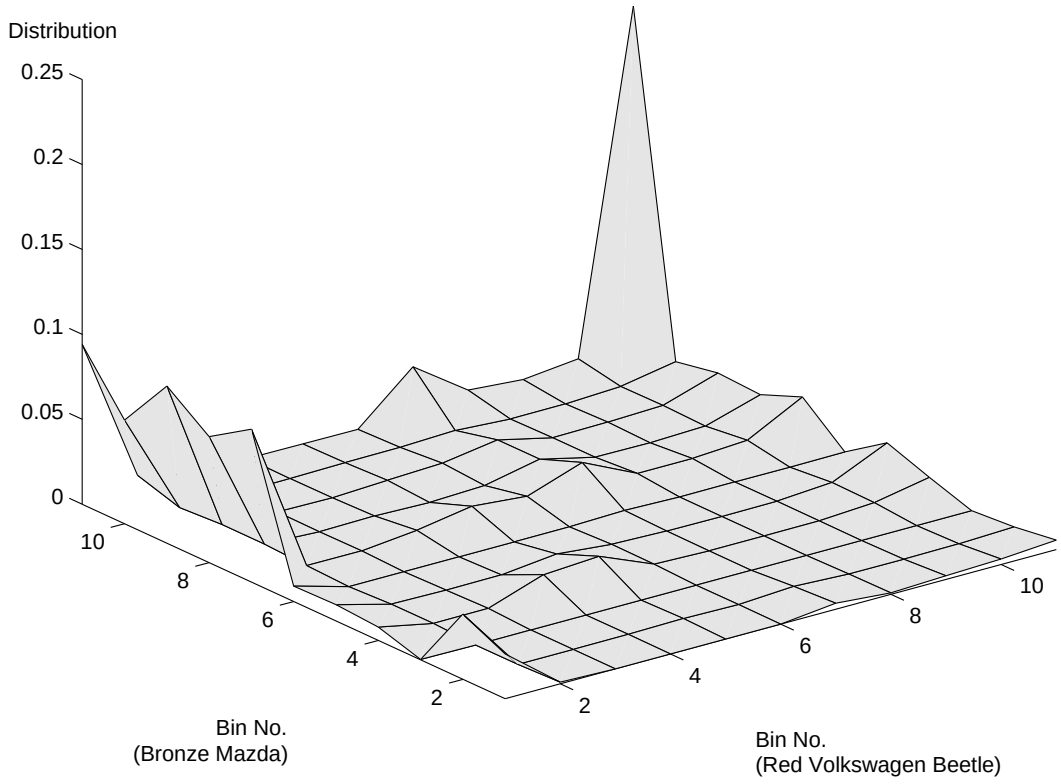
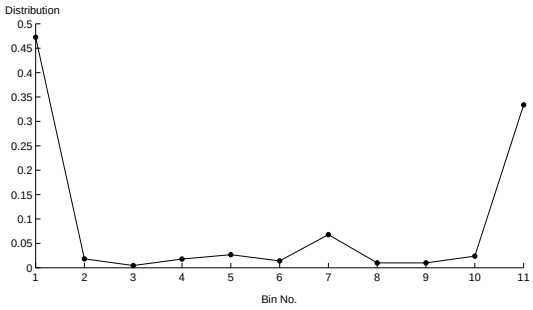


Figure 4.11: Joint distribution between the red Volkswagen Beetle and the bronze Mazda with joint entropy=4.5061.

4.3.6 Matching a red Volkswagen Beetle with a distorted red Volkswagen Beetle



(a) (b)

Figure 4.12: (a) Distorted red Volkswagen Beetle. (b) Its corresponding color distribution with entropy=2.0177.

Image	RB	UI	NI	GB	BM	DRB
RB	2.0569	0	0.0071	0.6961	0.3547	0.2478
UI	0	0	0	0	0	0
NI	0.0068	0	3.0336	0.0052	0.0083	0.0096
GB	0.6961	0	0.0052	2.1845	0.5365	0.2217
BM	0.3547	0	0.0083	0.5365	2.8036	0.2057
DRB	0.2478	0	0.0096	0.2217	0.2057	2.0386

Table 4.1: This table consolidates the mutual information of the matching results. RB represents red Volkswagen Beetle, UI represents uniformly-colored image, NI represents noisy image, GB represents green Volkswagen Beetle, BM represents bronze Mazda, and DRB represents distorted red Volkswagen Beetle.

4.3.6 Matching a red Volkswagen Beetle with a distorted red Volkswagen Beetle

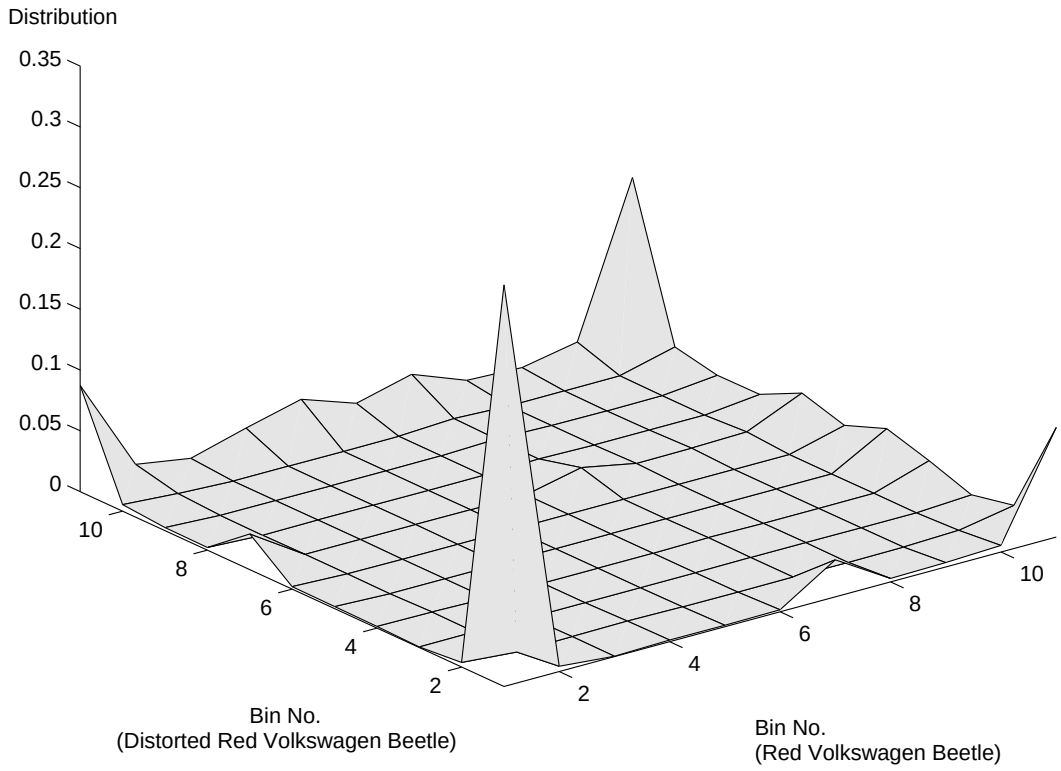


Figure 4.13: Joint distribution between the red Volkswagen Beetle and the distorted Red Volkswagen Beetle with joint entropy=3.8268.

We summarize our experiments in Table 4.1, and include additional matching results. From the table, we find that the highest mutual information is achieved when an image is compared with an identical copy of itself. Any comparisons with a uniformly-colored image give a zero. A comparison with the noisy image gives a very low mutual information measure. It is also interesting to observe the location of the peaks in the plot of the joint distribution. When the two images are identical, the peaks are positioned diagonally in the plot. When one of the images is a uniformly-colored image, the peaks formed a row. When the images are unrelated, the peaks are scattered. **Most importantly, we have shown that spatial relationship formed by the color distribution is important in the matching process**.

4.4 Summary

Mutual information measures the amount of information one random variable knows about the other random variable [7]. In our context, the images are the random variables. If the mutual information between a generated view and a query image is large, it means that the generated view and the query image know a lot about each other [6, 9, 10, 36]. Thus, the generated view is similar to the query image. It is indeed because of this property that mutual information is used in our object recognition system to measure the similarity between the query image and the generated views from view morphing. In the next chapter, we discuss the experiments conducted to test and demonstrate the performance of our 3D object recognition system.

Chapter 5

Experiments And Discussions

In this chapter, we discuss the experiments conducted to test and demonstrate the performance of our view-based 3D object recognition system.

5.1 Experimental Details

We have implemented a prototype of the view-based 3D object recognition system discussed in Chapter 2. This prototype which is implemented using MATLAB, consists of two modules - the database creation module and the object recognition module.

A database of 30 different cars is created for these experiments. Half of the database images are obtained using a video camcorder (Figure 5.1). The other half of the database images are taken from QuickTime movie from the Internet. Each database item is represented by a pair of images showing two views of the object at different pose. Correspondence points required for view morphing are captured off-line using the database

(a) (b) (c)

Figure 5.1: This is an example of an image sequence captured using a video camera. (a) and (c) are used as database images for view morphing. (b) is used as a test image.

creation module.

We collected 40 query images for our experiment. These images are also obtained using a video camcorder and from the Internet. We create a query file to store the coordinates of the minimum bounding box for the unknown car in the query image. In our experiments, we assume that the minimum bounding box can be obtained by motion analysis using a series of images of the car.

In our experiments, we defined these criteria for a correct recognition:

1. Mutual information is greater than or equal to 0.45. This value is empirically selected.

2. Pose of the recognized object must match the pose of object in the query image by visual inspection.

We also defined a recognition rate to measure the performance of our system:

$$Recognition\ Rate = \frac{Total\ no.\ of\ correct\ recognitions}{Total\ no.\ of\ queries} \tag{5.1}$$

We conduct three experiments to measure the performance of our system. The first experiment will test our system's ability to recognize the unknown object in the 2D query image with background. The second and the third experiments will measure the effect of misalignment of the bounding box and varying image intensity respectively.

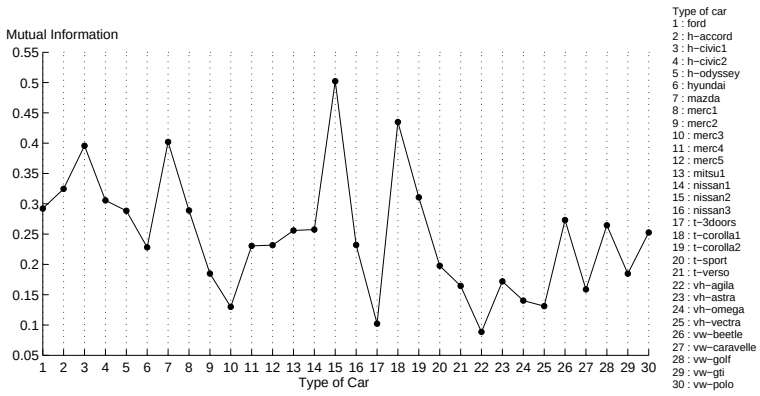
5.2 Recognition with Unsegmented Images

In this experiment, we tested our system with a set of 40 images with background. Our objective in this experiment is to find the recognition rate of the system given query images with noisy background. Among the 40 query images, our system is able to recognize 30 of them correctly. For those query images that failed, we found that the pose of the unknown car in the query images actually falls outside the range that is captured in the database. This small misalignment in pose, though not very obvious to our eyes, is detected by our system. Therefore, the correct car is not returned as the result.

We showed three examples from the successful queries. In Figure 5.2a, we have a query image containing a gray Nissan Sunny. The mutual information between the query image and all the database objects are shown in Figure 5.2c. The peak with the highest mutual information gives the identity of the unknown object in the query image. In this case, the result is a red Nissan Sunny (Figure 5.2b).

(a)

(b)

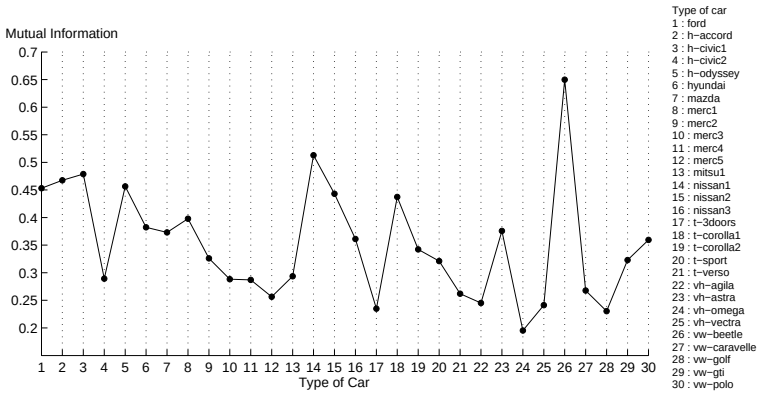


(c)

Figure 5.2: (a) Query image of a gray Nissan Sunny. (b) Generated image of a red Nissan Sunny. (c) Distribution of the mutual information in the comparison.

(a)

(b)



(c)

Figure 5.3: (a) Query image of a blue Volkswagen Beetle. (b) Generated image of a green Volkswagen Beetle. (c) Distribution of the mutual information in the comparison.

In Figure 5.3a, we have a query image of a blue Volkswagen Beetle. Figure 5.3c shows the distribution of the mutual information of all the comparison between the query image and the database objects. The highest peak is located at the green Volkswagen Beetle (Figure 5.3b).

In Figure 5.3a, we have a gold Honda Odyssey for query. The highest peak in the graph is located at the red Honda Odyssey (Figure 5.3c). The generated image shows the correct car in the correct pose (Figure 5.3b).

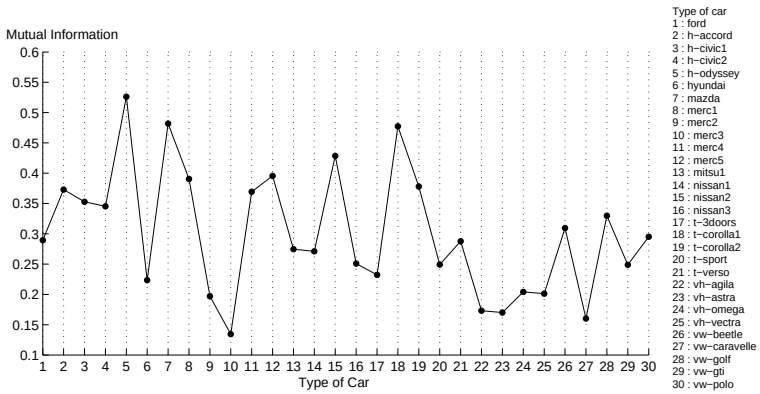
We proceed to show two examples from the unsuccessful query images. In Figure 5.5a, we have an image containing a blue Ford Festiva as the query. We show the expected result in Figure 5.5b and the result returned by the system in Figure 5.5c. We can see that the query image and the generated image of the expected result are taken at different pose. The query image is taken at a greater height with respect to the ground compared to the generated image.

In Figure 5.6, we have an image of a white Nissan as the query. We show the expected result in Figure 5.6b and the actual result returned by the system in Figure 5.6c. As in the previous example, the query image is taken from a greater height with respect to the ground compared to the generated image.

From this experiment, we show that our system is able to recognize the unknown object in the query images without the need to remove the background. This implies that our system can avoid performing object segmentation which by itself is a very difficult task. We also show that our system is color-invariant. For query images with car pose

(a)

(b)



(c)

Figure 5.4: (a) Query image of a gold Honda Odyssey. (b) Generated image of a red Honda Odyssey. (c) Distribution of the mutual information in the comparison.

(a) (b) (c)

Figure 5.5: (a) Query image of a blue Ford Festiva. (b) Generated image of a green Ford Festiva that has the closest pose with the query. (c) Generated image of a red Honda Civic wrongly returned as a match by the system.

(a) (b) (c)

Figure 5.6: (a) Query image of a white Nissan. (b) Generated image of a silver Nissan Sunny that has the closest pose with the query. (c) Generated image of a silver Toyota Corolla Sport wrongly returned as a match by the system.

falling in-between the pose of the database images, the recognition was successful. For query images with car pose falling out of the pose of the database images, the recognition failed. This problem can be solved by extending the system to include more images of the 3D object in different pose into the database.

5.3 Effect of Varying Bounding Box Size

In this experiment, we want to find the effect of varying bounding box size on the recognition rate. We use a set of 25 query images that are recognized successfully in the previous experiment. The bounding box size will be increased by a step size of 2 from -4 to 10. The object to be recognized is kept in the middle of the bounding box. The recognition rate for every bounding box size is plotted in Figure 5.7. This experiment shows that our object recognition system is able to handle a small amount of variation in the bounding box size.

In most cases, the unknown object is not at the center of the bounding box. We therefore test our system with varying misalignment of the unknown object in the bounding box. The same set of images is used for this test. The bounding box size is incremented by a step size of 2. The test is conducted and the result is tabulated in Table 5.1. From the table, we observe that the recognition rate decreases faster when the object is misplaced to the right. The reason for this observation is that the frontal of a car usually contains a lot of details. When the object is misplaced to the right, the frontal area of the car in one image does not match the frontal area of the car in the other image. This

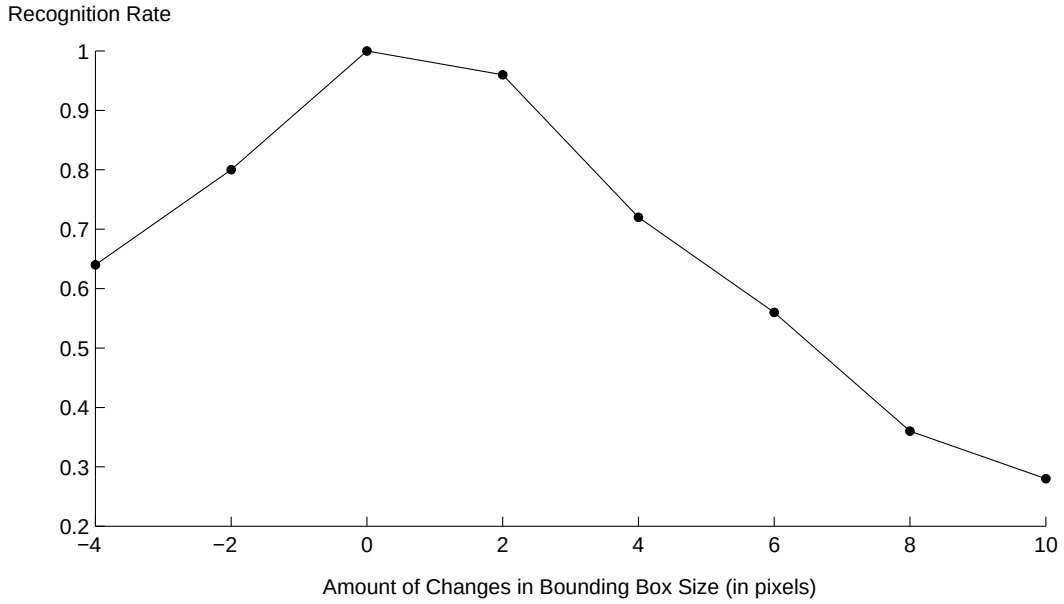


Figure 5.7: The plot of recognition rate versus the amount of changes in the bounding box size shows that our system is able to handle small amount of variation in the bounding box size.

	2	4	6	8	10
Left	0.92	0.92	0.80	0.76	0.64
Right	0.96	0.80	0.68	0.64	0.52
Top	0.92	0.92	0.84	0.76	0.60
Bottom	0.92	0.80	0.76	0.68	0.56

Table 5.1: This table shows the changes in the recognition rate when the unknown object is misaligned in the bounding box. The first column shows the position of the 3D object in the bounding box. The first row shows that amount of misalignment in pixels.

will reduce the mutual information.

5.4 Effect of Varying Image Intensity

In this last experiment, we want to find out how our object recognition system performs under varying image intensity. We use an image editing software to vary the image intensity uniformly. The test is conducted and the result is shown in Figure 5.8. From the figure, we conclude that our object recognition system is robust against image intensity changes as long as the change is uniform throughout the image. In the extreme case where the intensity change is so drastic that the query image becomes white or black, the mutual information will become zero as expected.

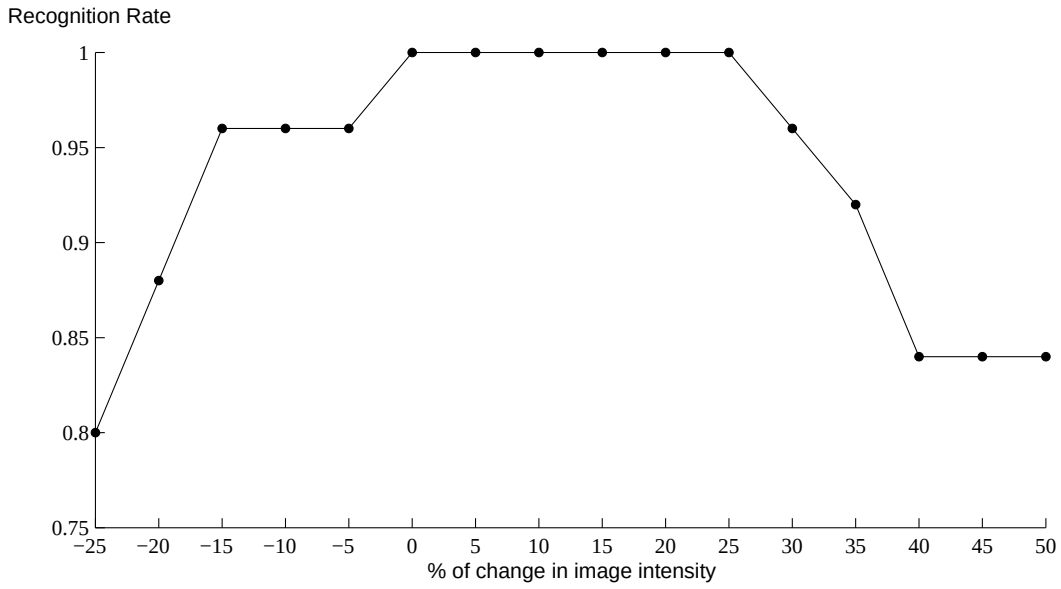


Figure 5.8: The plot of recognition rate versus the percentage of change in image intensity shows that our system is able to handle query image under different image intensity changes, as long as the change is uniform throughout the image.

5.5 Summary

The above experiments conclude that our system works well when the car pose in the query image is in-between the pose of the database images. Our system is also able to tolerate small amount of misalignment in the minimum bounding box and changes in image intensity.

The experiments were run on a PC with Intel Pentium 4 with 1.6GHz processor and 256MB of memory. The system was developed using MATLAB. The time taken to match a query image with a database object containing 11 images ranges between 20 seconds to 25 seconds (including the time required to perform view morphing and calculate the mutual information). The amount of time needed to generate a view of a 3D object takes about 1.5 seconds. The generated images are of size 300 by 200 pixels.

Chapter 6

Conclusion and Future Work

In this chapter, we discuss the contribution of this research. At the end of this chapter, we present some suggestions to improve the present system.

6.1 Contribution

In our method, we used the concept of mutual information to identify the unknown object in the query image. Our technique made use of the distribution of image attributes like gray-level and color. The query image was matched with different views of the 3D objects. The view that had the highest mutual information content was returned as a match. Our method thus avoided the difficult problem of extracting and matching corresponding features like contours and corners.

It would be too space consuming to store every possible view of the 3D objects in the database. Our system alleviated this problem by using view morphing to generate novel

images of the 3D objects to compare with the query image. View morphing required only a small set of images of the 3D objects to generate novel images.

Our system was very flexible in database creation. The images could be added incrementally to increase the range of pose. More importantly, the image need not be obtained from the same physical object. For example, a scanned image of a blue Volkswagen Beetle could be added to the database constructed using images of a blue Volkswagen Beetle captured using a digital camera. The relationship between the images had to be established through correspondence points.

6.2 Conclusion

In this thesis, we described our view-based 3D object recognition system. Our system comprised a database creation module and a recognition module. Several experiments were conducted to test the performance of our system.

From the experiments, our system was shown to be able to recognize 3D objects from 2D images without requiring the unknown object to be segmented from the background. This implied that our system could avoid the difficult problem of object segmentation. The system was also shown to tolerate some misalignment in the minimum bounding box and changes in image brightness. The technique developed is generic and so can be applied to the recognition of other objects. There are however rooms for improvement in our system. In Section 6.3, we discuss some future work for improving our system.

6.3 Future Work

Although our system is able to recognize 3D objects from 2D images, future improvements are desirable. In the following sections, we present some suggestions for improving the present system.

6.3.1 Complete representation of the 3D object

In our current system, we only use two images to represent a 3D object. The system is unable to recognize any object that lies outside the range of pose captured by these two images. View morphing has to be extended to handle more than two images. The relationship between the images must be established in a way that can facilitate view generation. For example, we will have six image pairs if we are given four images of the objects. Each image pair forms a camera baseline. It is relatively easy to generate novel images on the camera baseline. The difficulty arises when we want to generate novel images that lie between two camera baselines.

6.3.2 Tracking of the unknown object in the query image

In the current system, the minimum bounding box is manually specified by the user. A possible enhancement is to integrate the system with a video camera to track the moving object, find the minimum bounding box automatically and perform recognition immediately.

6.3.3 Self-learning object recognition system

The current system requires users' involvement to increase the number of objects that the system can recognize. A good enhancement is to make the system learn as it tries to recognize an object. This will allow the database to grow automatically.

Bibliography

[1] T. Arbel and F. P. Ferrie. Viewpoint selection by navigation through entropy maps. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, pages 248–254, 1999.

[2] R. Basri. Viewer-centered representations in object recognition: A computational approach. In C. H. Chen, L. F. Pau, and P. S. P. Wang, editors, *Handbook of Pattern Recognition and Computer Vision*, chapter 5.4, pages 863–882. World Scientific Publishing Company, 1993.

[3] I. Biederman. Recognition-by-components: A theory of human image understanding. *The Bell System Technical Journal*, 94:115–147, 1987.

[4] T. M. Breuel. View-Based Recognition. Technical Report 93-09, Dalle Molle Institute for Perceptual Artificial Intelligence, 1993.

[5] S. Dickinson, R. Bergevin, I. Biederman, J. Eklundh, R. Munck-Fairwood, A. Jain, and A. Pentland. Panel report: The potential of geons for generic 3-D object recognition, 1997.

[6] S. Gilles. Description and experimentation of image matching using mutual information. Technical report, Department of Engineering Science, Oxford University, UK, 1996.

[7] R. M. Gray. *Entropy and Information Theory.* Springer-Verlag, 1990.

[8] R. I. Hartley. In defence of the 8-point algorithm. In *5th International Conference on Computer Vision*, pages 1064–1070, 1995.

[9] J. Hornegger and D. Paulus. Bayesian Vision: From Intensity Marginals to Mutual Information and Entropic Object Recognition. Technical report, Lehrstuhl fr Mustererkennung (Informatik 5), Universitt Erlangen, March 1997.

[10] H.-W. Hseu, A. Bhalerao, and R. G. Wilson. Image matching based on the co-occurrence matrix. Technical Report CS-RR-358, Department of Computer Science, University of Warwick, 1999.

[11] D. P. Huttenlocher and S. Ullman. Recognizing solid objects by alignment with an image. *International Journal of Computer Vision*, 5(2):195–212, 1990.

[12] R. Jain, R. Kasturi, and B. G. Schunck. *Machine Vision.* McGraw-Hill, Inc., 1998.

[13] K. Kanatani. Optimal fundamental matrix computation: Algorithm and reliability analysis. In *6th Symposium on Sensing via Image Information (SII 2000)*, June 2000.

[14] S. Kovacic, A. Leonardis, and F. Pernu. Planning sequences of views for 3-D object recognition and pose determination. *Pattern Recognition*, 31(10):1407–1417, 1998.

[15] J. C. Liter and H. H. Blthoff. An introduction to object recognition. Technical Report No.(43), Max Planck Institute for Biological Cybernetics, Tubingen, Germany, 1996.

[16] Q.-T. Luong and O. D. Faugeras. The fundamental matrix: Theory, algorithms, and stability analysis. *International Journal of Computer Vision*, 17(1):43–75, 1996.

[17] H. Murase and S. K. Nayar. Visual learning and recognition of 3-D objects from appearance. *International Journal of Computer Vision*, 14(1):5–24, 1995.

[18] R. M. Murray, Z. Li, and S. S. Sastry. *A Mathematical Introduction to Robotic Manipulation.* CRC Press, 1994.

[19] A. Pentland, B. Moghaddam, and T. Starner. View-based and modular eigenspaces for face recognition. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR'94)*, June 1994.

[20] C. G. Perrott and L. G. C. Hamey. Object Recognition—A Survey of the Literature. Technical Report 91-0065C, Department of Computing, Macquarie University, NSW 2109 Australia, 1991.

[21] T. Poggio and S. Edelman. A network that learns to recognize 3D objects. *Nature*, pages 263–266, 1990.

[22] A. R. Pope. Model-based object recognition - a survey of recent research. Technical Report TR-94-04, Department of Computer Science, The University of British Columbia, 1994.

[23] F. M. Reza. *An Inroduction To Information Theory.* McGraw-Hill, Inc, 1994.

[24] E. Rivlin, S. J. Dickinson, and A. Rosenfeld. Recognition by functional parts. *Computer Vision and Image Understanding: CVIU*, 62(2):164–176, 1995.

[25] B. Schiele and A. Pentland. Probabilistic object recognition and localization. In *7th IEEE International Conference on Computer Vision*, pages 177–182, 1999.

[26] S. M. Seitz. Bringing photographs to life with view morphing. In *Proc. Imagina 97 Conf.*, pages 153–158, 1997.

[27] S. M. Seitz and C. R. Dyer. Physically-Valid View Synthesis by Image Interpolation. In *Proc. Workshop on Representation of Visual Scenes.* IEEE Computer Society Press, June 1995.

[28] S. M. Seitz and C. R. Dyer. View morphing. In *Proc. SIGGRAPH 96*, pages 21–30, 1996.

[29] S. M. Seitz and C. R. Dyer. View morphing: Uniquely predicting scene appearance from basis images. In *Proc. Image Understanding Workshop*, pages 881–887, 1997.

[30] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(10):379–423, 623–656, 1948.

[31] L. G. Shapiro and G. C. Stockman. *Computer Vision.* Prentice Hall, 2001.

[32] M. J. Swain and D. H. Ballard. Color indexing. *International Journal of Computer Vision*, 7(1):11–32, 1991.

[33] M. J. Tarr. Visual object recognition: Can a single mechanism suffice? Essay submitted to the James S. McDonnell Centennial Fellowship Competition, 1997.

[34] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

[35] S. Ullman and R. Basri. Recognition by linear combinations of models. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(10):992–1006, 1991.

[36] P. Viola and W. M. Wells III. Alignment by maximization of mutual information. *International Journal of Computer Vision*, 24(2):137–154, 1997.

[37] K. Wu and M. Levine. 3D object representation using parametric geons. Technical Report TR-CIM-93-13, Center for Intelligent Machines, McGill University, 1993.

[38] Z. Zhang, R. Deriche, O. D. Faugeras, and Q.-T. Luong. A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. *Artificial Intelligence*, 78(1-2):87–119, 1995.

[39] W. Zhao, R. Chellappa, A. Rosenfeld, and P. Phillips. Face recognition: A literature survey. Technical Report Technical Report CAR-TR-948, Center for Automation Research, University of Maryland, 2000.