3D OBJECT RECONTRUCTION USING MULTIPLE-VIEW GEOMETRY: SIFT DETECTION

LEOW RUEY SHYAN

A project report submitted in partial fulfilment of the requirements for the award of the degree of Bachelor (Hons.) of Mechatronics Engineering

Faculty of Engineering and Science
Universiti Tunku Abdul Rahman

April 2011

DECLARATION

I hereby declare that this project report is based on my original work except for citations and quotations which have been duly acknowledged. I also declare that it has not been previously and concurrently submitted for any other degree or award at UTAR or other institutions.

Signature : _____

Name : Leow Ruey Shyan

ID No. : 07UEB06709

Date : 15/04/2011

APPROVAL FOR SUBMISSION

I certify that this project report entitled **"3D OBJECT RECONSTRUCTION USING MULTIPLE-VIEW GEOMETRY: SIFT DETECTION"** was prepared by **LEOW RUEY SHYAN** has met the required standard for submission in partial fulfilment of the requirements for the award of Bachelor of Engineering (Hons.) Mechatronics Engineering at Universiti Tunku Abdul Rahman.

Approved by,

Signature : _____

Supervisor : Dr. Tay Yong Haur

Date : _____

3D OBJECT RECONSTRUCTION USING MULTIPLE-VIEW GEOMETRY: SIFT DETECTION

ABSTRACT

In the recent years, three-dimensional (3D) model reconstruction has gained attention and become popular mainly because it plays an important role in various engineering applications, especially in computer vision. To be able to carry out 3D reconstruction, there are several important keys that must be kept in mind. First, we must be able to determine the 3D model based on multiple images or video captured using an uncalibrated camera. Then comes the question on how to select only points that are significant and helpful in providing information regarding the shape of the body of 3D object of interest. Besides that, determining corresponding matching points in different images captured from different views and angles is as equally important. In this final year project, we propose to use multiple view geometry to conduct 3D object reconstruction based on the captured images of the object of interest. Multiple view geometry is the case whereby non-rigid arbitrary motions are viewed from several translational cameras with the help of numerous corresponding points. To align with our proposed approach, SIFT (Scale Invariant Feature Transform) feature is used to detect and extract useful corresponding keypoints in a series of desired images captured. Published by David Lowe in 1999, SIFT feature has become among the most popular feature for object recognition and localization based on local point features in the industry. Though SURF (Speeded Up Robust Features) may outperform SIFT in terms of speed, yet if compared to other available methods of local features, SIFT still outperforms them in terms of robustness and distinctiveness. SIFT technique was chosen over SURF to be implemented simply because SIFT is much more popular in current industries. Basically, SIFT features are invariant to image translation, scaling and rotation while partially invariant to illumination changes, and can be said to be robust against noise and partial occlusions.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS / ABBREVIATIONS

c_p	specific heat capacity, J/(kg·K)
h	height, m
K_d	discharge coefficient
M	mass flow rate, kg/s
P	pressure, kPa
P_b	back pressure, kPa
R	mass flow rate ratio
T	temperature, K
v	specific volume, m^3
α	homogeneous void fraction
η	pressure ratio
ρ	density, kg/m^3
ω	compressible flow parameter
ID	inner diameter, m
MAP	maximum allowable pressure, kPa
MAWP	maximum allowable working pressure, kPa
OD	outer diameter, m
RV	relief valve

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Aim

The main purpose or aim of this final year project titled “3D Object Reconstruction Using Multiple-View Geometry: SIFT Detection” is to use multiple view geometry method to reconstruct 3D objects from a series of images of an object of interest by using an uncalibrated camera, with angles remain unknown By implementing SIFT feature algorithm in this project, the 3D models are to be constructed based on the given SIFT keypoints.

1.2 Background and Motivation

Basically this project is divided in two parts - SIFT detection part and reconstruction part. For the first part, SIFT algorithm will be implemented to detect and extract local feature keypoints. For the second part, 3D object reconstruction process will be taken place automatically based on the given SIFT keypoints by using multiple view geometry. For the author’s part, focus will be placed on the first part – SIFT detection. SIFT algorithm will be implemented using Matlab with the help of VL Feat toolbox. Digital camera will be used to conduct this entire project.

Before deciding on this topic, a brief research regarding 3D reconstruction field and SIFT implementations was conducted. There exists many methods to carry

out 3D reconstruction may it be by measuring the distance of the object placed on a turntable (active method) or by using moving range sensors to measure the 3D shape of large objects (passive method).

Unlike 3D data acquisition that has become fast and cheap in recent years, 3D reconstruction is still a huge challenge in its industry. Though various approaches can be applied to it, many of them are still unable to produce fully-automated and detailed 3D models. Besides that, although the costs of sensors, platforms and processing hardware to reconstruct 3D models are becoming cheaper nowadays, it is still costly and complicated to use them.

Furthermore, most of the research available about SIFT algorithm are ways to improve its efficiency and reduce its runtime, while keeping up with its accuracy. Not much research was done on what kind of objects are most suitable for SIFT implementation and what kind of objects are not suitable.

Hence, the motivation of this project is to determine the types of objects suitable for SIFT implementation and at the same time suitable for 3D object reconstruction. Besides that, conditions of image capturing should also be determined in order to obtain optimum results for both parties.

1.3 Objectives

The main objective of this final year project is to implement the understanding of 3D object reconstruction, SIFT implementation and multiple view geometry. By gathering the information and knowledge gained in these three fields, the author will focus mainly in producing a feature extraction approach, namely the SIFT algorithm so that her partner for this project will be able to carry out the reconstruction part with the keypoints obtained from SIFT algorithm implementation.

This SIFT algorithm should be invariant to scale changes, orientation and image translations. When a series of images of the object of interest are taken and

SIFT algorithm is implemented, it should be able to find corresponding matching keypoints of the object. This means that the algorithm is capable of finding points that have the same characteristics and identical positions in the images captured in different views and angles. The implementation of the algorithm should show good consistency.

Besides that, the author will carry out various experiments involving four types of objects which include transparent objects, white-coloured objects, smooth-surfaced objects and heavily-textured objects. These experiments should help to determine which type of object is more suitable for the implementation of SIFT algorithm as well as be useful for 3D object reconstruction.

Upon completing the final year project, the author should be able to master MATLAB software with the help of VL Feat 0.9.9 toolbox while running SIFT algorithm.

1.4 Scope of Work

Ms. Leow Ruey Shyan, which is me, is responsible in implementing SIFT algorithm to detect and extract useful local keypoints of a test object from a series of images captured. I am responsible to ensure that the feature extraction method implemented is efficient, effective and useful for 3D reconstruction. Besides that, I should be able to come out with a conclusion of which type of object is more suitable for 3D object reconstruction.

On the other hand, Ms. Leow Tzyy Shyuan is responsible for the reconstruction of 3D model part. She will use multiple view geometry to reconstruct the object of interest, based on the given SIFT keypoints.

1.5 Report Outline

1.5.1 Chapter 1: Introduction

This chapter consists of the aim, background, motivation, objectives and scope of work of the project. In this chapter, way to carry out this final year project is briefly explained and discussed and at the same time, stating the scope of work between group members.

1.5.2 Chapter 2: Literature Review

In this chapter, brief explanations of SIFT implementations in 3D reconstruction and SIFT technique are included to ensure that readers will understand the contents of this project.

1.5.3 Chapter 3: Methodology

In this chapter, the methods and procedures used to carry out this project are explained in details.

1.5.4 Chapter 4: Results and Discussions

In this chapter, experiments are carried out to determine the type of objects which are suitable to be implemented with SIFT algorithm as well as 3D object reconstruction. Figures and table are included for readers understanding. Besides that, experiment analysis and discussions are included as well.

1.5.5 Chapter 5: Conclusion and Recommendations

In this chapter, a brief conclusion about the entire project is included. Other subtopic like contributions which is a summary of work that has been done to accomplish this project is included as well. The last part of this topic is about future works that can be done in order to improve the system.

CHAPTER 2

LITERATURE REVIEW

2.1 SIFT Implementation in 3D Object Reconstruction

The general flow of idea from image data acquisition to 3D object reconstruction based on the acquired series of images requires the steps stated below:

1) A person with an uncalibrated hand-held digital camera moves around a test object and capture a series of images from different angles.
2) The images obtained are pre-processed. The pre-processing methods may involve noise removal, illumination normalization or frames selection.
3) The series of images are processed and the 3D model of the test object is reconstructed automatically The detailed steps include:
    a) Feature detection and matching
    b) Epipolar geometry estimation
    c) 3D reconstruction

For this project, focus will be placed in **feature detection and matching**, where SIFT feature algorithm is chosen to be applied to detect and match feature points. This step is essential in order to carry on with other remaining steps for 3D object reconstruction. So what is SIFT?

2.1.1 What is SIFT?

SIFT is the abbreviation name of Scale Invariant Feature Transform. Published by David G Lowe in 1999, SIFT is an algorithm widely used in detecting and describing local features in images. Up till today, it remains as one of the most popular feature matching algorithms in the description and matching of 2D image features.

For multiple images of a same object taken under different conditions, it is extremely important that a feature matching algorithm should be able to find correspondence points, which are points that have the same characteristics and identical positions in these images. It is essential that the correspondence point finding is invariant to scale changes, rotation and view point changes.

There are two stages involved in correspondence point matching which the extraction of interesting points and descriptor making. For the first stage, feature point that has high repeatability from changing environment is known as interesting point. These are the points that are required for extraction stage. Usually they are selected at distinctive locations in the test image, such as corners. Then in the second stage, the extracted interesting points are provided with feature description. This also means that neighbourhood of each interesting point is represented by a feature vector. The description is extracted from a training image, which can be used to identify and locate an object in a test image which contains other objects as well. For this descriptor to work well in object recognition, it has to be distinctive, have invariant characteristics and is robust to changes in scale, rotation, illumination, noise and local geometric distortion.

Therefore the main reason for SIFT to be a successful algorithm in field of feature matching is because it can extract stable feature points. Besides that, it is proven that SIFT is more robust compared to other feature matching techniques as a local invariant detector and descriptor with respect to geometrical changes. It is said to be robust against occlusions and scale variance simple because SIFT feature descriptor is invariant to image translation, scale changes and rotation while partially invariant to illumination changes. Hence, SIFT feature points are used to calculate the fundamental matrix and reconstruct 3D objects.

2.1.2 Overview of SIFT in Feature Detection and Extraction

Basically there are four major components in SIFT framework for keypoint detection and extraction:

1) ***Scale space extrema detection:*** This is the first stage of computation that searches all scales and image locations. Difference of Gaussian (DoG) function is implemented to detect local maxima and minima. These form a set of candidate keypoints.

2) ***Keypoint localization:*** Every candidate keypoint is fitted to a detailed model for location and scale determination. Low contrast points and poorly localized edge responses are discarded. Remaining keypoints are stable points.

3) ***Orientation assignment:*** Based on local image gradient direction, one or more orientations are assigned to each keypoint location. Additional keypoints are created for the case of stronger directions.

4) ***Keypoint descriptor:*** The local image gradients are measured at the selected scale in the region around each keypoint. These are transformed into a normalized representation that allows significant levels of local shape distortion and illumination changes

SIFT matching phase is as equally important as SIFT feature detection and extraction phase. Therefore in the next sub section, a thorough explanation of both SIFT feature detection and matching will be included.

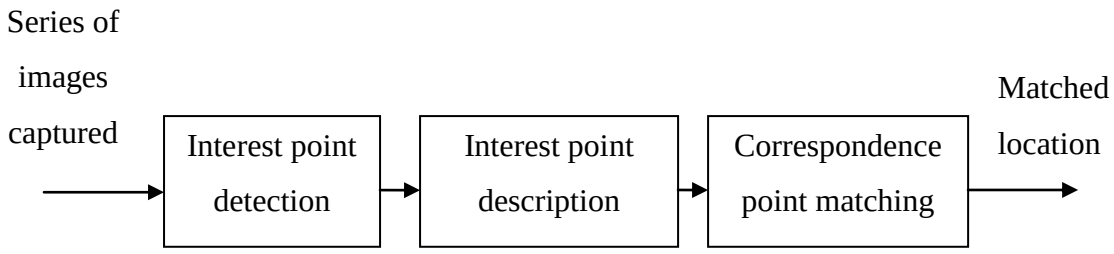

Figure 2.1: Flow chart of SIFT feature algorithm

2.1.3 Overview of the Important Key Stages of SIFT

Besides going through the above stated framework for keypoint detection, SIFT matching algorithm is carried out to identify correspondence matching point.

In order to provide a better understanding regarding the entire SIFT implementation process which consists of keypoint detection, description and matching, a series of detailed steps required are stated below.

1. ***Scale invariant feature detection***

   In this stage, SIFT keypoints of an object of interest are extracted from a series of reference video sequences and stored into a database called training set. This collection of SIFT keypoints from various images are to be invariant to scale changes, rotation and image translations, partially invariant to illumination changes and robust against occlusions. For any geometric distortion, blurring and resampling of local image orientation planes are needed to carry out

   For key localization, difference of Gaussian (DoG) is applied to a series of smoothed and resampled images. The local maxima and minima obtained from the results of application are identified as interest points (SIFT candidate keypoints) in the object. Keypoints with low contrast and responses along edges are discarded. Then dominant orientations are assigned to these localized keypoints. These steps are essential to ensure that the keypoints are more stable for matching and recognition.

2. ***Feature matching and indexing***

   The problem of storing SIFT keypoints and detecting matching keypoints in test images is known as indexing. To overcome this problem, best-bin-search method (nearest neighbour) is applied. It is actually a modified k-d tree algorithm. The advantages of best-bin-search (BBS) include the reduction of computation speed, leading to higher speed and efficiency.

In BBS algorithm, the keypoint found is identified as the nearest neighbour in the training set (database) of keypoints, then it is classified as the best candidate match. Nearest neighbour can be defined as the keypoint with minimum Euclidean distance from the given descriptor vector. For robustness, ratio of distance from the closest neighbour to the distance of second closest neighbour is taken.

For efficiency sake, a threshold of 0.8 is set. Ratio distance more than the threshold is eliminated while the BBS algorithm search is stopped after checking the first two hundred nearest neighbour candidates. This should ensure that database with large number of keypoints, nearest neighbour search can be speeded up while maintaining less than 5% of loss in the number of correct matches.

3. Cluster identification by Hough Transform voting

After computing SIFT features on the input image, these features are compared to the SIFT features stored in database. Each SIFT keypoint specifies four parameters: 2D location, scale and orientation.

To increase the robustness of recognition, Hough Transform is used to identify clusters of matches that vote for the same object pose. Each keypoint votes for the set of object poses that are consistent with the keypoint's location, scale and orientation. When the locations in Hough accumulator accumulate at least three votes, the locations are selected as candidate pose.

4. Model verification by linear least squares

This is a verification step whereby the training image for the hypothesized pose is matched to the image using a linear least-squares fit to the hypothesized location, scale and orientation of the object.

5. Outlier detection

By checking the agreement between each image feature and the model with given parameter solution, outliers can be removed. The remaining points will be resolved using linear least-squares solution and the process is repeated. As usual, if less than three points remain after removing the outliers, the match is rejected.

To make the decision to either accept or reject a model hypothesis, a detailed probabilistic model is computed. First, the expected false matches to the model are computed. Then Bayesian probability analysis is carried out. It presents the probability of the presence of object based on the actual number of matching features found. To be qualified for acceptance, the final probability for a correct interpretation must be greater than 0.98.

Object matches that pass all the above tests are identified as correct with high confidence.

To summarize the entire process, a table is presented below:

Table 2.1: Summary of SIFT implementation process

Problem	Technique	Advantages
Key localization	Difference of Gaussian (DoG)	• Accurate • Stable • Invariant to scale • Invariant to rotation
Scale changes	Scale-space pyramid	
Orientation	Orientation assignment	
Geometric distortion	• Blurring • Resampling of local image orientation planes	Invariant to affine distortion
Indexing and matching	Best-bin search algorithm	• Higher speed • Higher efficiency
Cluster identification	Hough Transform voting	Able to obtain reliable pose models

Model verification / outlier detection	Linear least-squares solution	Better error tolerance with fewer matches
Decision of acceptance / rejection	Bayesian Probability analysis	Outcome is reliable

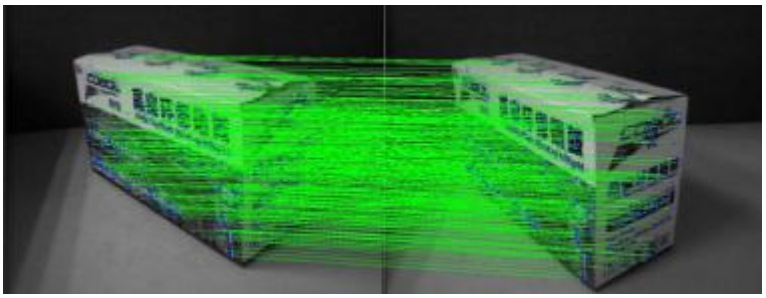

Figure 2.2: SIFT matching points (Keju, Xin, Dongxiang, & Yunhui, 2009)

Hence, once the model of hypothesis is accepted, this indicates that existence of object of interest in the series of images. The SIFT feature points will then be used for another part of this project , which is the 3D object reconstruction part.

2.1.4 Applications of SIFT

Given its invariance characteristics towards scale changes, rotation and image translations and robust to affine transformation, SIFT is very useful in object recognition field. Basically SIFT features can be applied to any tasks that require object location identification and matching between images.

SIFT algorithm is widely used in 2D object recognition, 3D reconstruction, motion tracking and segmentation, robot localization and mapping, panorama stitching and so on. For this project, focus will be placed in using SIFT algorithm in 3D object reconstruction.

CHAPTER 3

METHODOLOGY

3.1 System Overview

This project is a MATLAB based project with the implementation of VL Feat 0.9.9 on standard laptop with a 2.0 GHz processor. The SIFT algorithm developed mainly focuses on keypoint detection and matching. Since object recognition is not part of the concern of the project, hence databases of tested images are not needed. Tools used include:

1. MATLAB
2. VL Feat 0.9.9
3. Digital camera (Sony DSC-T77 with 10.1 megapixels)

The basic system overview is as below:

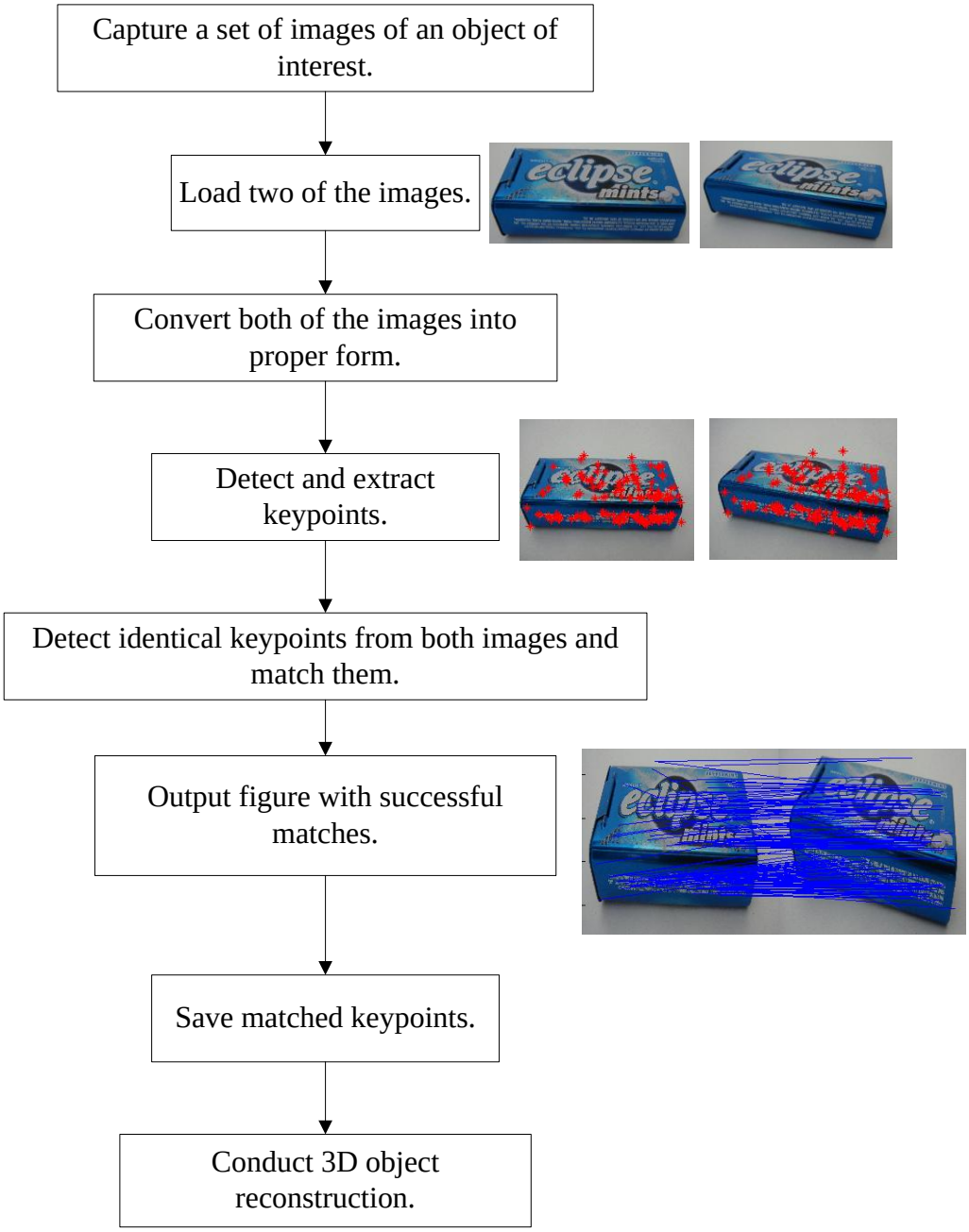

Figure 3.1: Basic overview of processing steps involved in the entire system

Detailed details of the steps taken to carry out SIFT algorithm will be discussed in the following sub-chapter.

3.2 Detailed Description of the Entire System

3.2.1 Pre-processing Steps

1. ***Image capturing***

   After identifying the object of interest, for example a sweet box, a series of images of the object is captured from different angles. At least four different angles should be taken.

Figure 3.2: Example of images captured from different angles

2. ***Resize images***

   Sizes of the images should be resized to smaller sizes. It is advisable that the sizes of pictures captured should be resized to less than 1 Mb to save computation time as well as memory consumption. Sizes used for each set of images should be consistent. For this project, the default width pixel size is set to 1728, before rotating to the correct position.

3. ***Choose reference image***

   Choose an image with a significant view as reference image for matching purposes in the later part of process.

3.2.2 Processing Steps

1. ***Load images***

   Load reference image (example: *image1.jpg*) and any other image (example: *image2.jpg*) captured earlier.

Figure 3.3: (Left) image1.jpg; (Right) image2.jpg

2. ***Convert images into appropriate form***

   To run the SIFT algorithm using VL Feat in MATLAB, the usage of the *vl_sift* command is essential. This particular command requires single-precision grayscaled image. Hence, both images loaded are first converted from true colour RGB to grayscale intensity colour. This is then followed by the single-precision intensity conversion. By default, MATLAB stores data in arrays of class double.

3. ***Compute keypoints and descriptors***

   By running the *vl_sift* command on the two images, keypoints and descriptors for both images will be detected. The keypoints detected will be stored in respective frame matrix while descriptors are stored in respective descriptor matrix. By default, descriptors are 128-dimensional.

   If we visualize the keypoints(frames) and descriptors detected in *image1.jpg* with a random selection of 250 features, the figures should be as below:

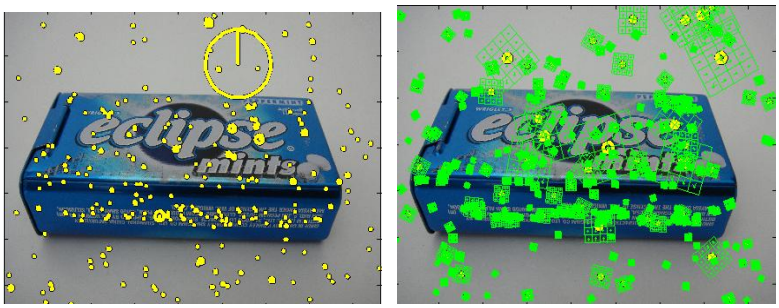

Figure 3.4: (Left) keypoints detected; (Right) descriptors detected

4. Image conversions

To ensure that the algorithm runs faster, the images are converted into integer class, unsigned 8-bit (UINT8) integer array. This means that the elements of each array can range only from 0 to 255.

5. Match descriptors

To match the descriptors detected in both images, the *vl_ubcmatch* command is used. Among the descriptors detected in *image2.jpg*, a closest descriptor to match each descriptor in *image1.jpg* will be searched. When a closest descriptor is found, a match between these two descriptors is formed.

To narrow down the number of matches found to increase the accuracy matching, a specified threshold value can be used. With this threshold, a descriptor in *image1.jpg*, D1, is matched to a descriptor in *image2.jpg*, D2, only if the distance, d, between these two descriptors multiplied by the threshold value is not greater than the distance of D1 to all other descriptors.

d(D1, D2) * threshold value < distance of D1 to all other descriptors

By default, if no threshold value is specified, then the threshold value will be automatically set to the value of 1.5.

6. Plot and display matched descriptors

Matching descriptors are plotted in red on each image. Then these two plotted images are being displayed.

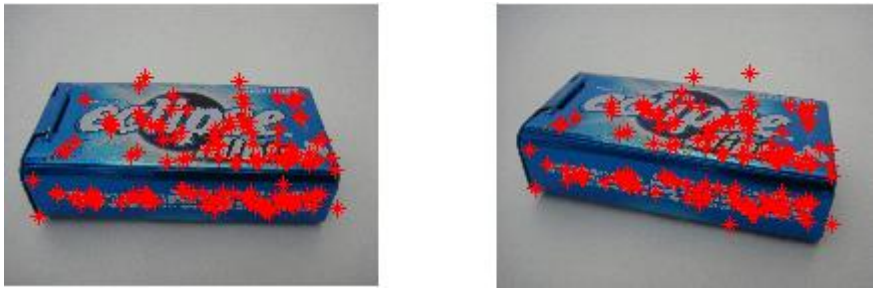

Figure 3.5: Plot of matching descriptors

7. Output figure with successful descriptor matches

Lines are drawn to connect every set of matching descriptors and are displayed.

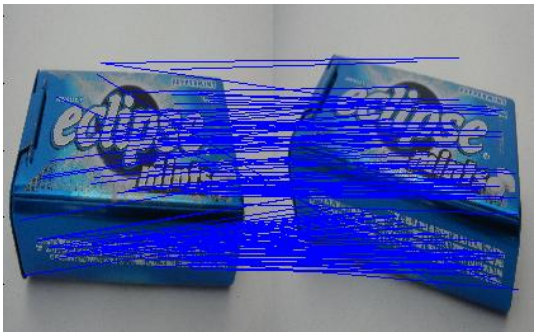

Figure 3.6: The blue lines shown are successful matches between the two images

8. Output details needed

For convenience, details such as number of matches found and time taken to carry out the entire process are being displayed on screen.

9. Save matched keypoints

Coordinates of the successfully matched keypoints in both images are saved and be used in 3D object reconstruction part.

10. Repeat Step 1 to Step 9

Steps above are repeated in a loop until the last image of each set of images is being compared and matched.

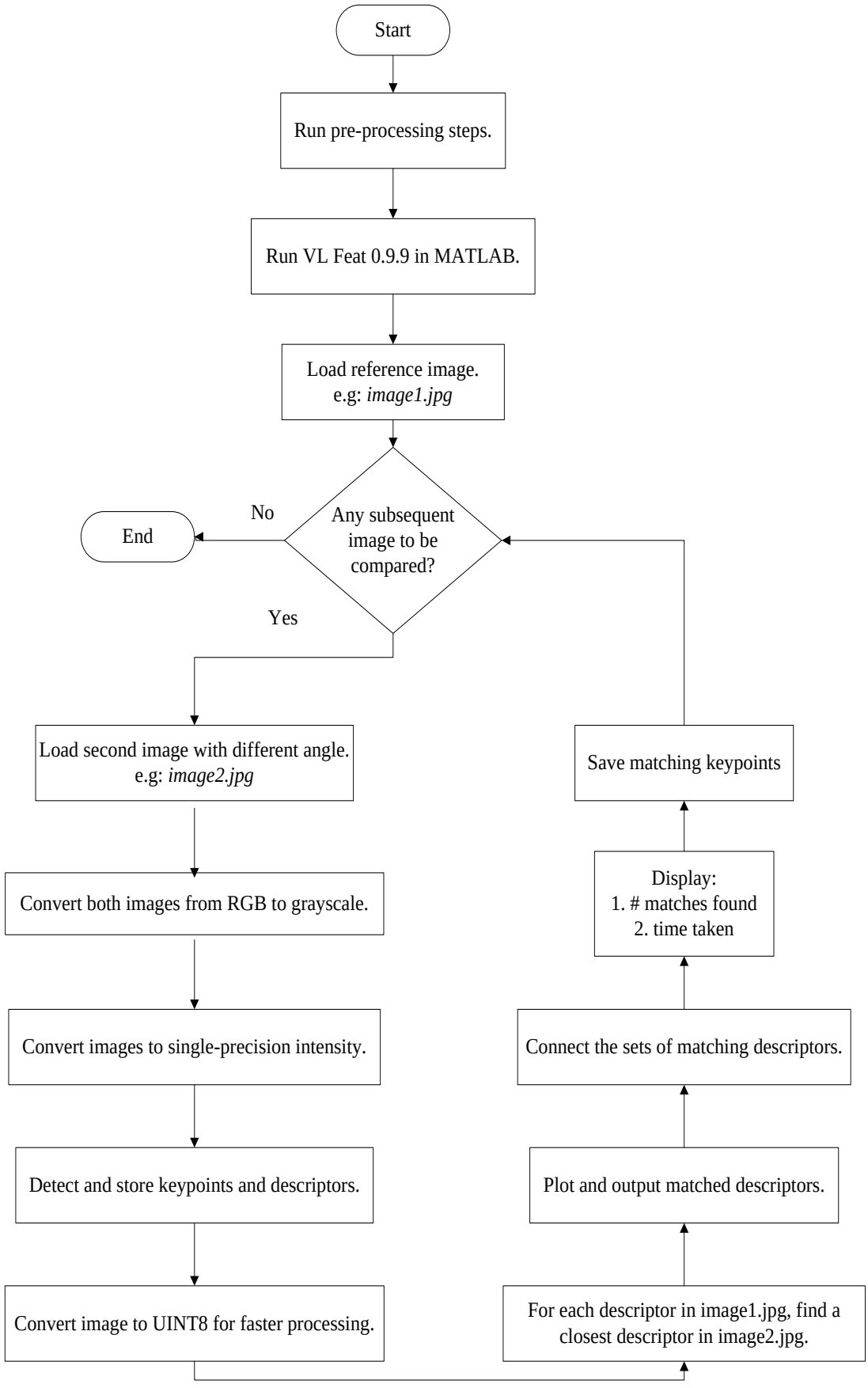

Figure 3.7: Flow chart of detailed steps involved in the entire process

CHAPTER 4

RESULTS AND DISCUSSIONS

4.1 Brief Overview of Experiments Conducted

In this project, focus is mainly placed in three areas:

1. To show that the SIFT algorithm implemented is invariant to rotation, position and scale changes.
2. To determine SIFT algorithm is suitable to be implemented in which type of objects.
3. To determine what kind of result is suitable for 3D object reconstruction.

For the second area mentioned above, four types of object were used. These four objects generally represent four different categories which are:

- Smooth-surfaced objects
- Heavily-textured objects
- White-coloured objects
- Transparent objects

Hence, to achieve the above mentioned objectives, four experiments were conducted. Generally, these experiments include:

- **Experiment 1:** To test the effectiveness of SIFT detection on smooth-surfaced objects.
- **Experiment 2**: To test the effectiveness of SIFT detection on heavily-textured objects.

- **Experiment 3:** To test the effectiveness of SIFT detection on white-coloured objects.
- **Experiment 4:** To test the effectiveness of SIFT detection on transparent objects.

There are a few things about the experiments conducted that need to be noted:

a) For each category (smooth, heavily-textured, white and transparent objects), two datasets were used.
b) For each dataset, at least five images were captured from different angles. As for report purpose, only five images are shown.
c) Plain backgrounds were used while capturing images. This is to reduce unwanted keypoint detections and also to ensure that each dataset used is consistent.
d) All pixel sizes of the images were resized to 1728 in width before rotating to appropriate position.
e) For consistency, threshold value is set to default, which is 1.5.

4.2 Experiment 1: Effectiveness of SIFT Detection on Smooth-Surfaced Objects

4.2.1 Experiment 1(a): Rectangular Container ('Eclipse' Mint Container)

Figure below shows five images of a rectangular container captured from different angles. Starting from the left of first row, the names of images are known as *eclipse1*, *eclipse2*, *eclipse3*, *eclipse4* and *eclipse5*, whereby *eclipse1* is the base image.

Figure 4.1: Images used in Experiment 1(a)

By running the SIFT algorithm, *eclipse1* is first compared with *eclipse 2*, then followed by *eclipse 3* and so on. Below shows the results obtained.

1. SIFT detection on *eclipse1* and *eclipse 2*

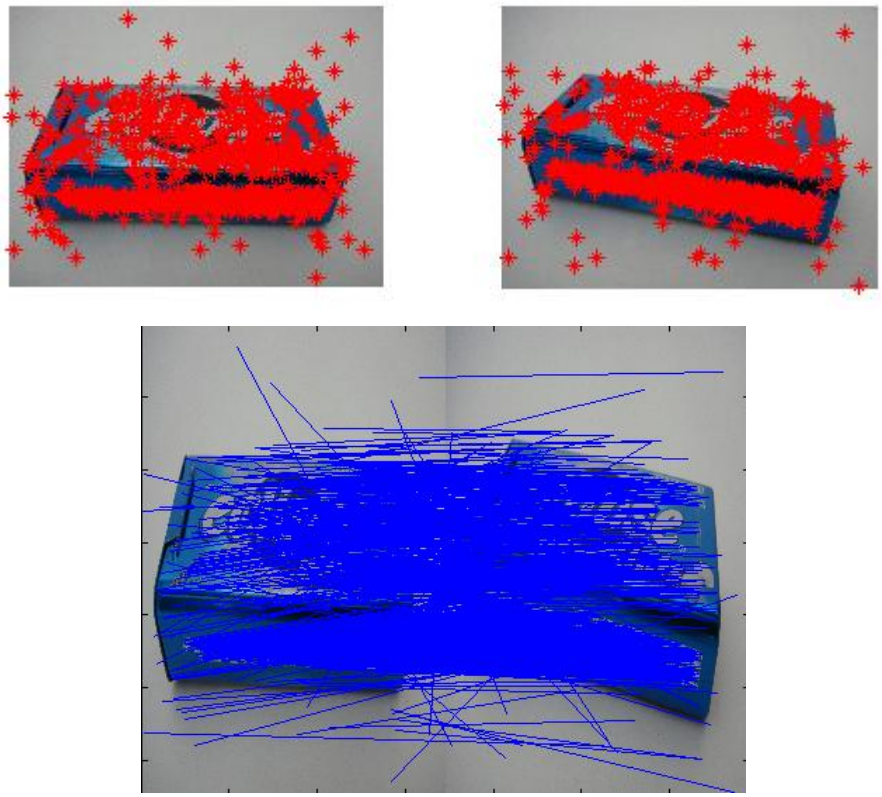

Figure 4.2: Plots of matching keypoints (top) and successful matches (bottom) between the *eclipse1* and *eclipse2*

Table 4.1: Details of results obtained by comparing *eclipse1* and *eclipse2*

	eclipse1	***eclipse2***
Number of keypoints detected	4574	4637
Number of successful matches	1119	
Time taken (s)	17.111	

2. SIFT detection on *eclipse1* and *eclipse3*

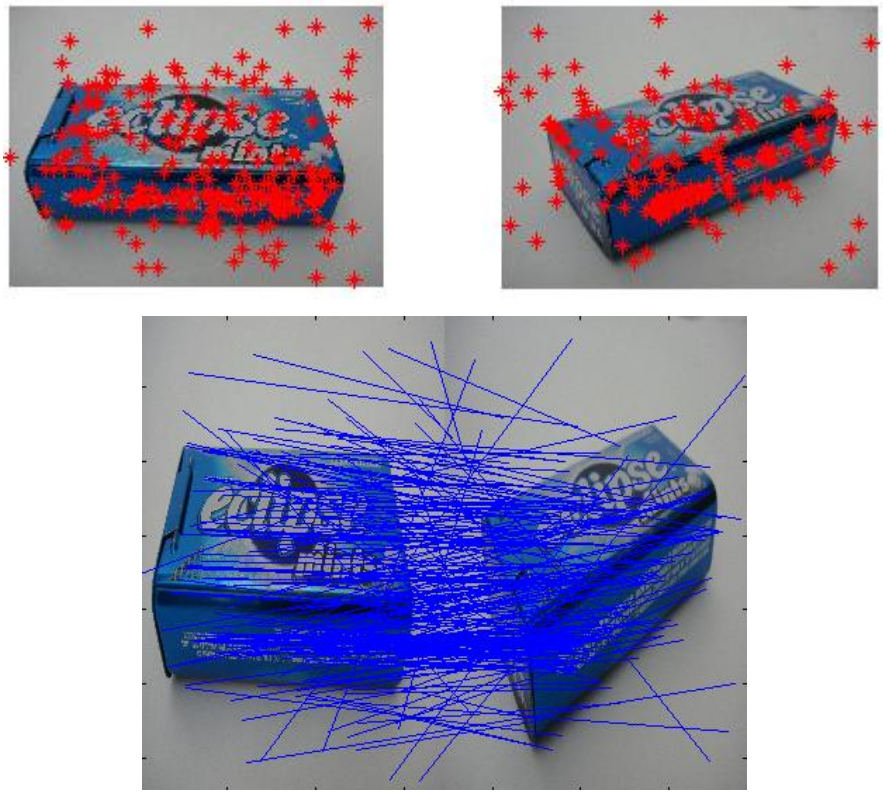

Figure 4.3: Plots of matching keypoints (top) and successful matches (bottom) between *eclipse1* and *eclipse3*

Table 4.2: Details of results obtained by comparing *eclipse1* and *eclipse3*

	eclipse1	***eclipse3***
Number of keypoints detected	4574	3975
Number of successful matches	253	
Time taken (s)	15.026	

3. SIFT detection on *eclipse1* and *eclipse4*

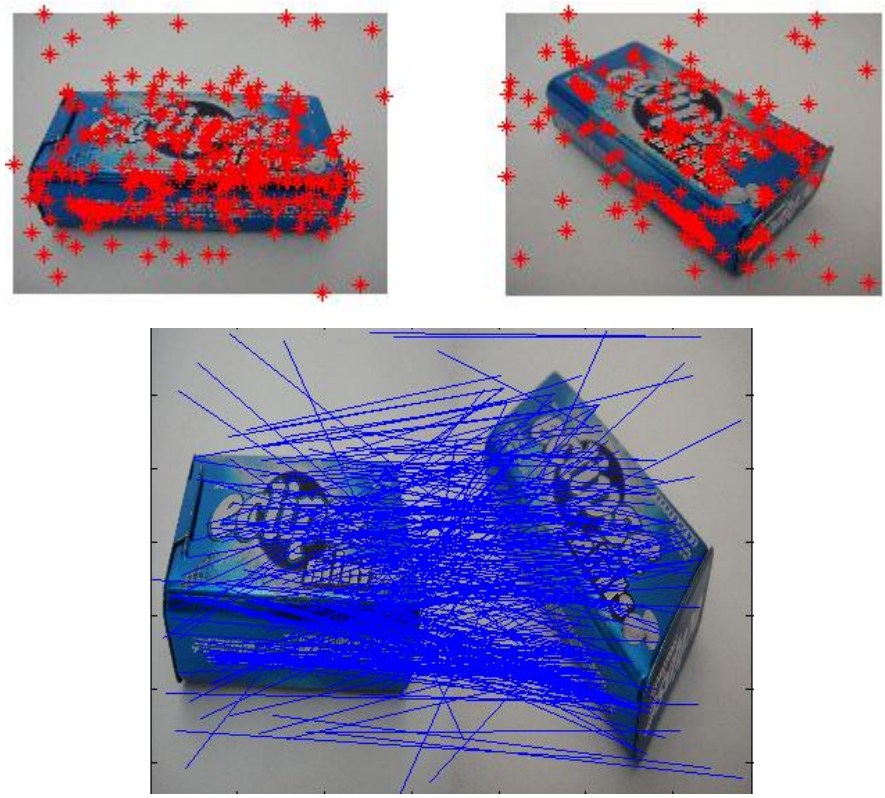

Figure 4.4: Plots of matching keypoints (top) and successful matches (bottom) between *eclipse1* and *eclipse4*

Table 4.3: Details of results obtained by comparing *eclipse1* and *eclipse4*

	eclipse1	***eclipse4***
Number of keypoints detected	4574	3881
Number of successful matches	321	
Time taken (s)	16.637	

4. SIFT detection on *eclipse1* and *eclipse5*

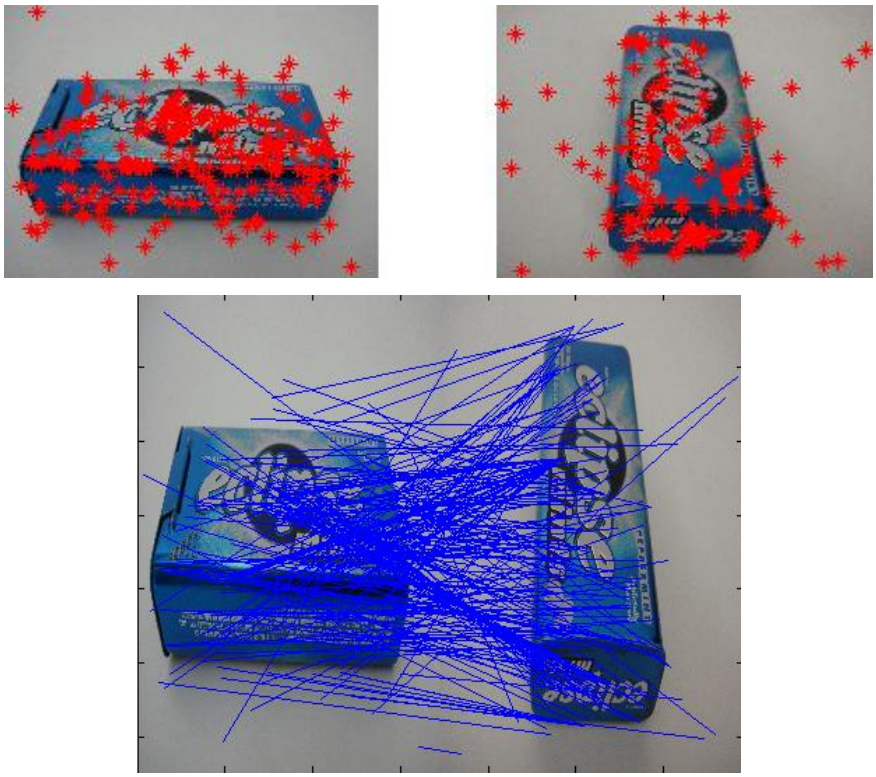

Figure 4.5: Plots of matching keypoints (top) and successful matches (bottom) between *eclipse1* and *eclipse5*

Table 4.4: Details of results obtained by comparing *eclipse1* and *eclipse5*

	eclipse1	***eclipse5***
Number of keypoints detected	4574	3646
Number of successful matches	256	
Time taken (s)	15.686	

4.2.2 Experiment 1(b): Mobile Phone (Nokia 5800 Express Music)

Figure below shows five images of a mobile phone captured from different angles. Starting from the left of first row, the names of images are known as *phone1*, *phone2*, *phone3*, *phone4* and *phone5*, whereby *phone1* is the base image.

Figure 4.6: Images used in Experiment 1(b)

1. SIFT detection on *phone1* and *phone2*

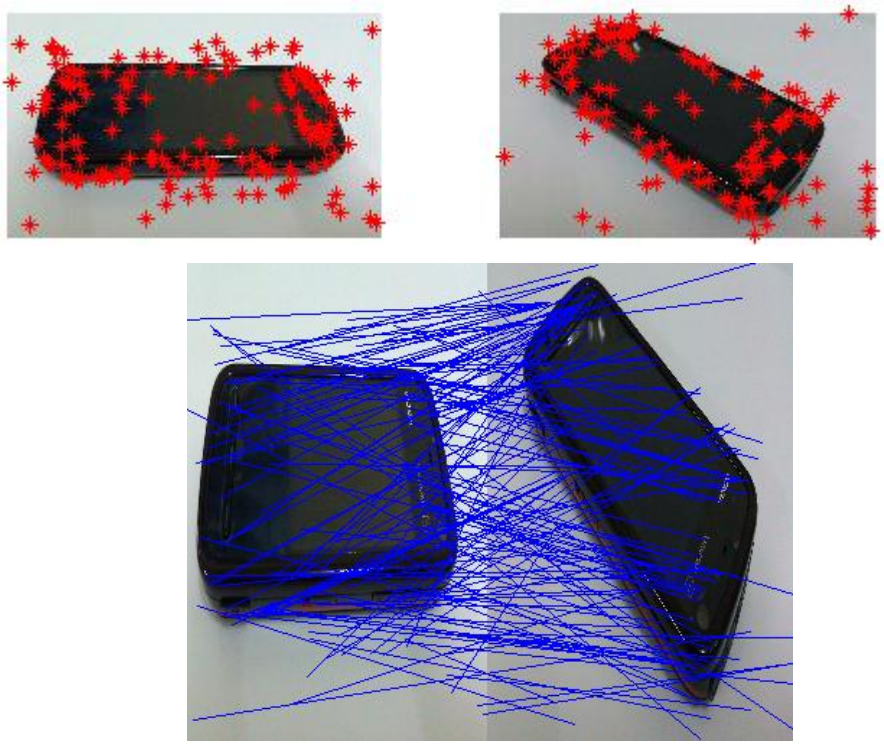

Figure 4.7: Plots of matching keypoints (top) and successful matches (bottom) between *phone1* and *phone2*

Table 4.5: Details of results obtained by comparing *phone1* and *phone2*

	phone1	***phone2***
Number of keypoints detected	3879	3451
Number of successful matches	163	
Time taken (s)	13.716	

2. SIFT detection on *phone1* and *phone3*

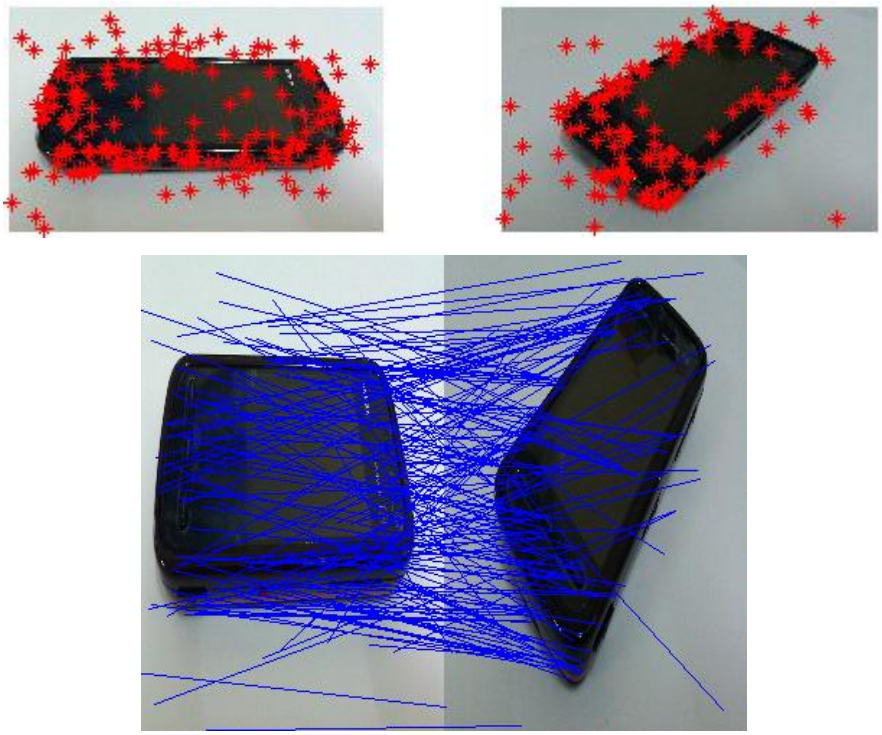

Figure 4.8: Plots of matching keypoints (top) and successful matches (bottom) between *phone1* and *phone3*

Table 4.6: Details of results obtained by comparing *phone1* and *phone3*

	phone1	***phone3***
Number of keypoints detected	3879	3291
Number of successful matches	187	
Time taken (s)	13.636	

3. SIFT detection on *phone1* and *phone4*

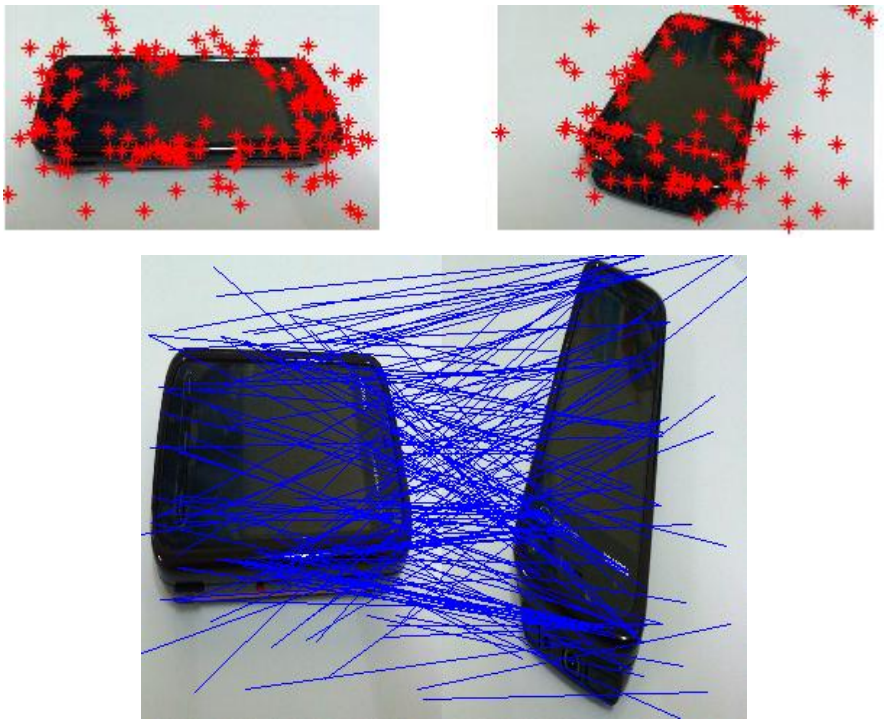

Figure 4.9: Plots of matching keypoints (top) and successful matches (bottom) between *phone1* and *phone4*

Table 4.7: Details of results obtained by comparing *phone1* and *phone4*

	phone1	***phone4***
Number of keypoints detected	3879	3892
Number of successful matches	173	
Time taken (s)	16.021	

4. SIFT detection on *phone1* and *phone5*

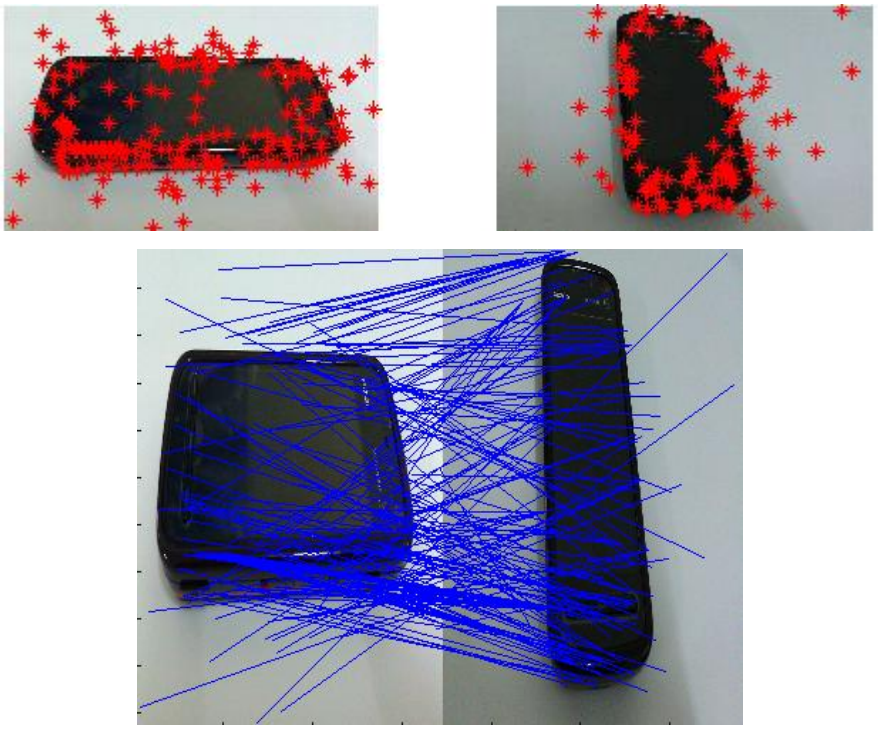

Figure 4.10: Plots of matching keypoints (top) and successful matches (bottom) between *phone1* and *phone5*

Table 4.8: Details of results obtained by comparing *phone1* and *phone5*

	phone1	***phone5***
Number of keypoints detected	3879	2976
Number of successful matches	192	
Time taken (s)	13.038	

4.3 Experiment 2: Effectiveness of SIFT Detection on Heavily-Textured Objects

4.3.1 Experiment 2(a): Pencil Holder

Figure below shows five images of a pencil holder captured from different angles. Starting from the left of first row, the names of images are known as *holder1*, *holder2*, *holder3*, *holder4* and *holder5*, whereby *holder1* is the base image.

Figure 4.11: Images used in Experiment 2(a)

1. SIFT detection on *holder1* and *holder2*

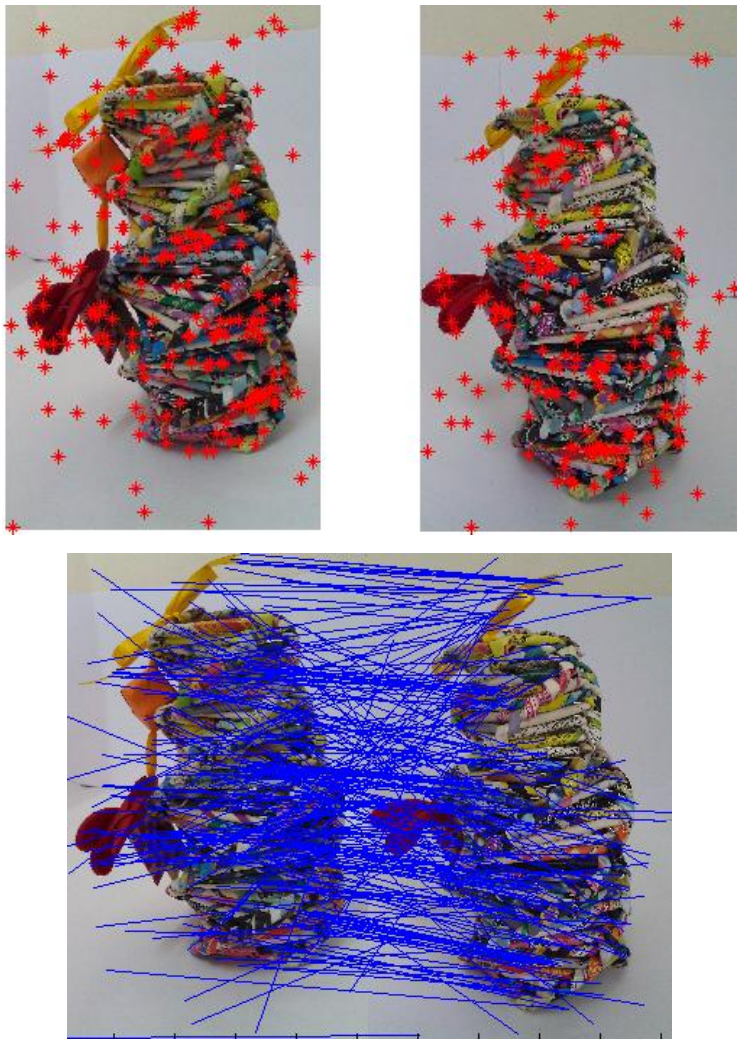

Figure 4.12: Plots of matching keypoints (top) and successful matches (bottom) between *holder1* and *holder2*

Table 4.9: Details of results obtained by comparing *holder1* and *holder2*

	holder1	***holder2***
Number of keypoints detected	4992	4431
Number of successful matches	250	
Time taken (s)	18.709	

2. SIFT detection on *holder1* and *holder3*

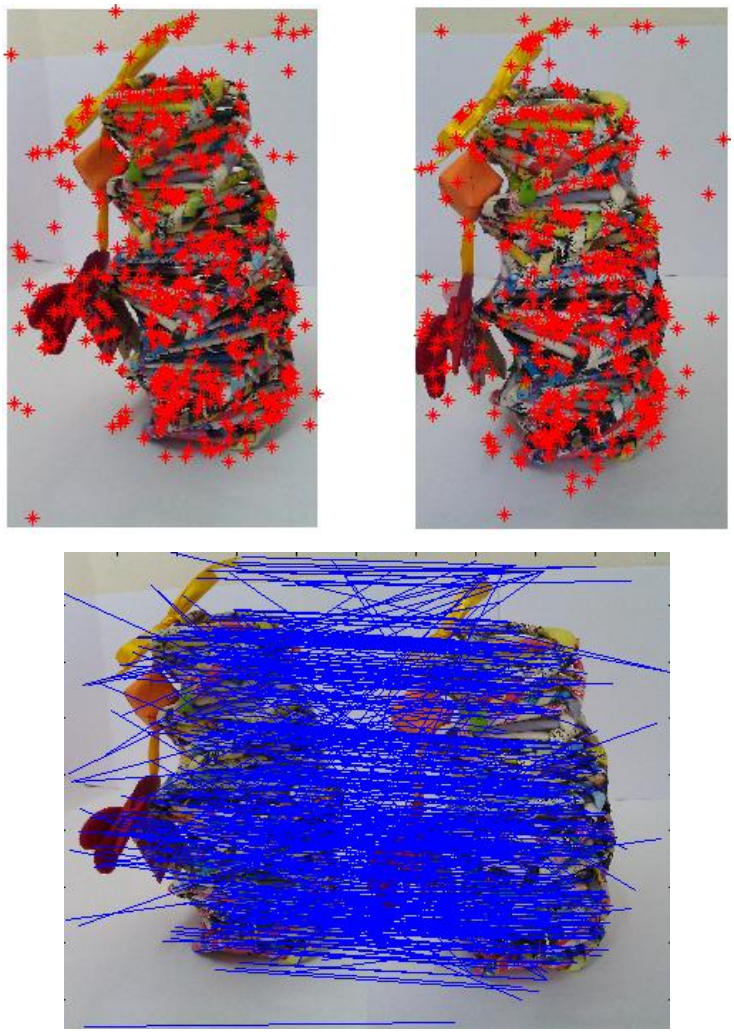

Figure 4.13: Plots of matching keypoints (top) and successful matches (bottom) between *holder1* and *holder3*

Table 4.10: Details of results obtained by comparing *holder1* and *holder3*

	holder1	***holder3***
Number of keypoints detected	4992	4469
Number of successful matches	506	
Time taken (s)	17.961	

3. SIFT detection on *holder1* and *holder4*

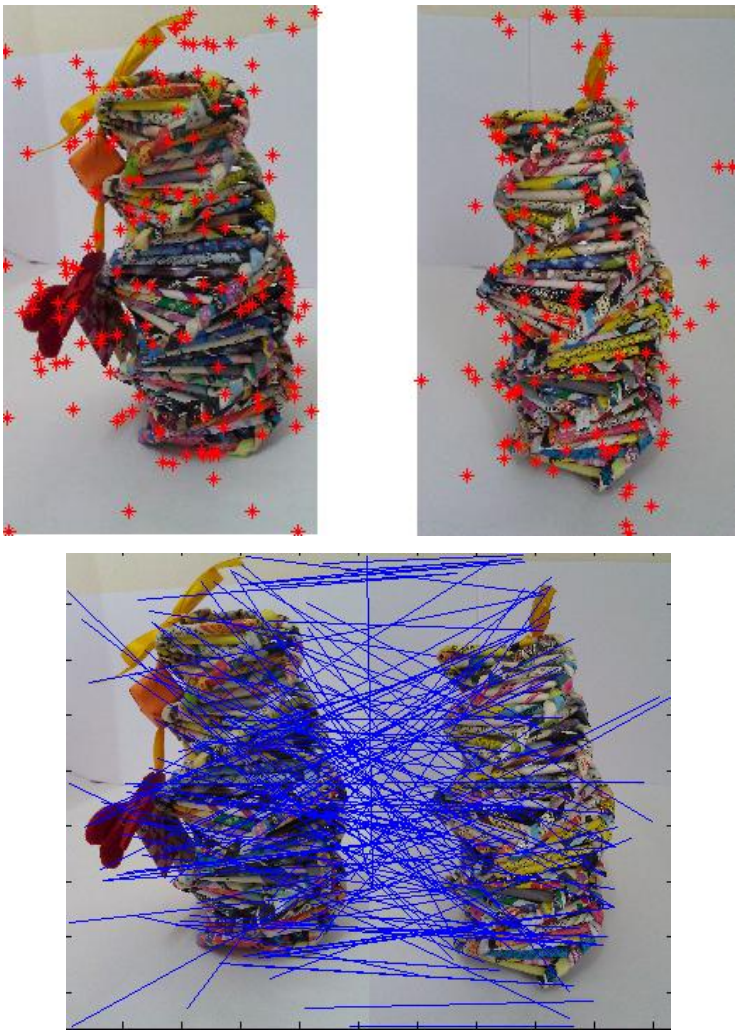

Figure 4.14: Plots of matching keypoints (top) and successful matches (bottom) between *holder1* and *holder4*

Table 4.11: Details of results obtained by comparing *holder1*and *holder4*

	holder1	***holder4***
Number of keypoints detected	4492	4199
Number of successful matches	147	
Time taken (s)	113.444	

4. SIFT detection on *holder1* and *holder5*

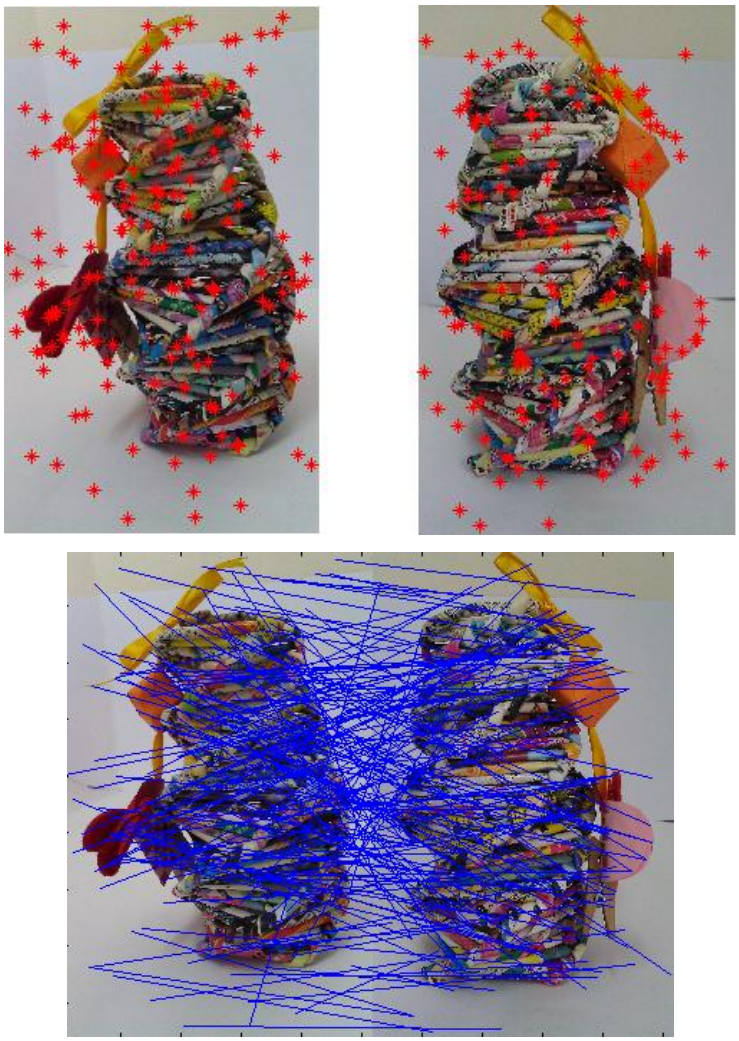

Figure 4.15: Plots of matching keypoints (top) and successful matches (bottom) between *holder1* and *holder5*

Table 4.12: Details of results obtained by comparing *holder1* and *holder5*

	holder1	***holder5***
Number of keypoints detected	4492	4563
Number of successful matches	167	
Time taken (s)	18.661	

4.3.2 Experiment 2(b): Digital Camera (Nikon D3100)

Figure below shows five images of a digital camera captured from different angles. Starting from the left of first row, the names of images are known as *camera1*, *camera2*, *camera3*, *camera4* and *camera5*, whereby *camera1* is the base image.

Figure 4.16: Images used in Experiment 2(b)

1. SIFT detection on *camera1* and *camera2*

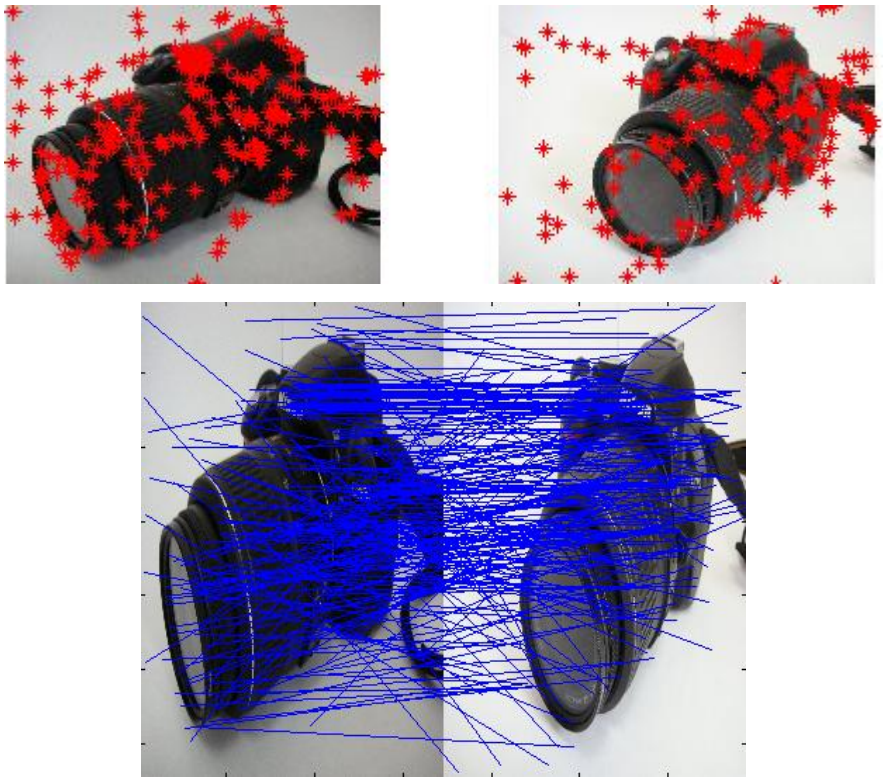

Figure 4.17: Plots of matching keypoints (top) and successful matches (bottom) between *camera1* and *camera2*

Table 4.13: Details of results obtained by comparing *camera1* and *camera2*

	camera1	***camera2***
Number of keypoints detected	3914	4263
Number of successful matches	237	
Time taken (s)	14.475	

2. SIFT detection on *camera1* and *camera3*

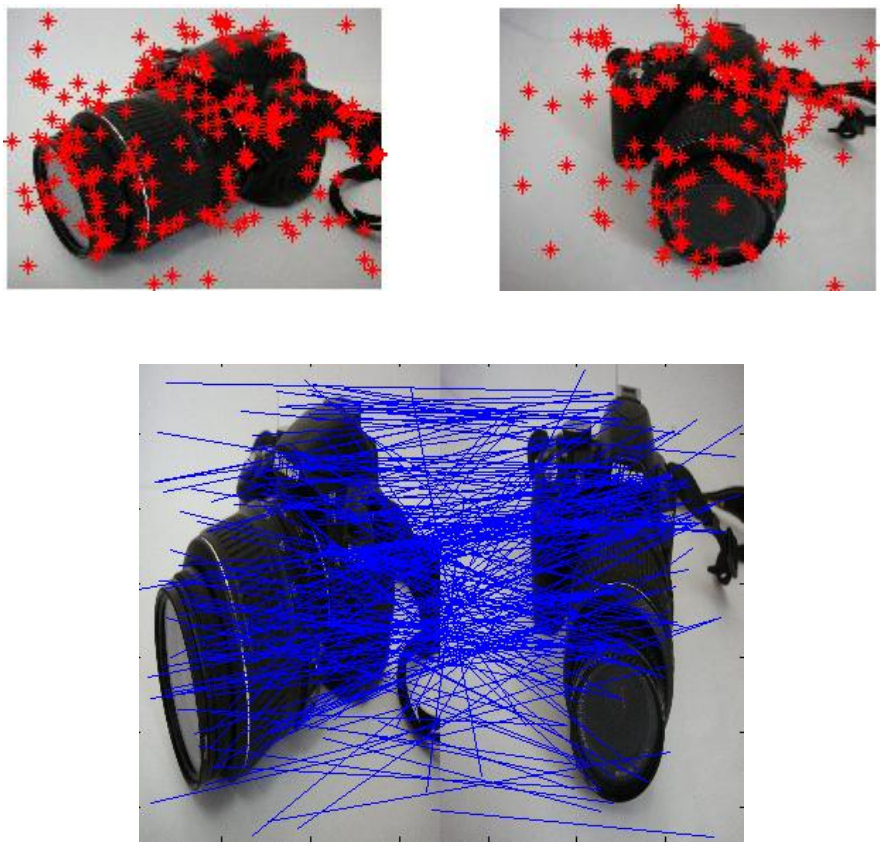

Figure 4.18: Plots of matching keypoints (top) and successful matches (bottom) between *camera1* and *camera3*

Table 4.14: Details of results obtained by comparing *camera1* and *camera3*

	camera1	***camera3***
Number of keypoints detected	3914	3841
Number of successful matches	212	
Time taken (s)	16.597	

3. SIFT detection on *camera1* and *camera4*

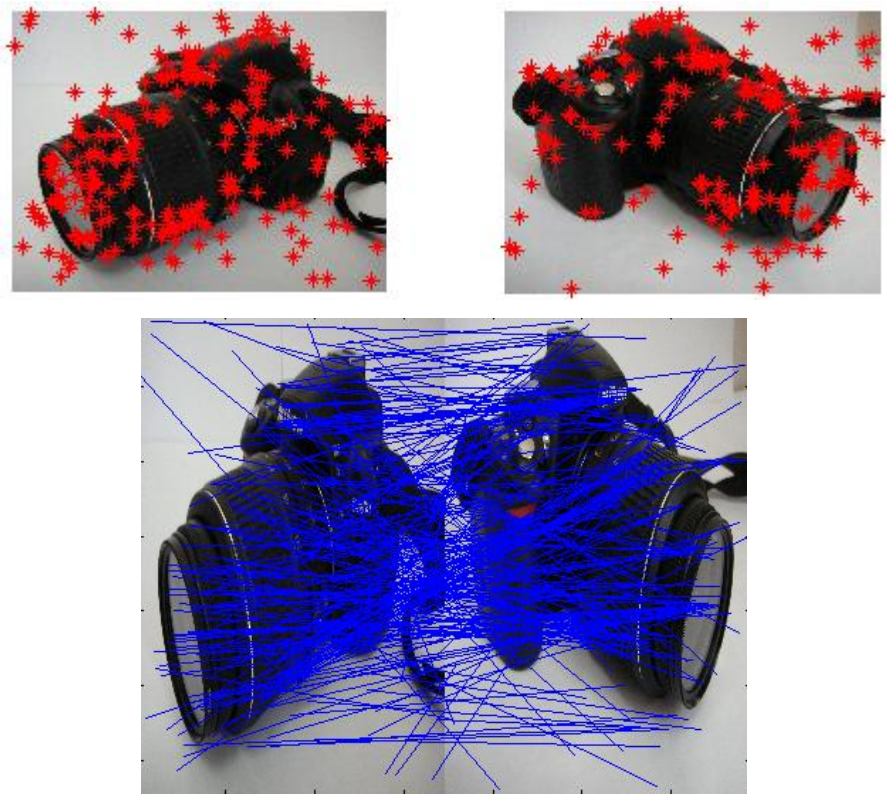

Figure 4.19: Plots of matching keypoints (top) and successful matches (bottom) between *camera1* and *camera4*

Table 4.15: Details of results obtained by comparing *camera1* and *camera4*

	camera1	***camera4***
Number of keypoints detected	3914	4280
Number of successful matches	234	
Time taken (s)	15.365	

4. SIFT detection on *camera1* and *camera5*

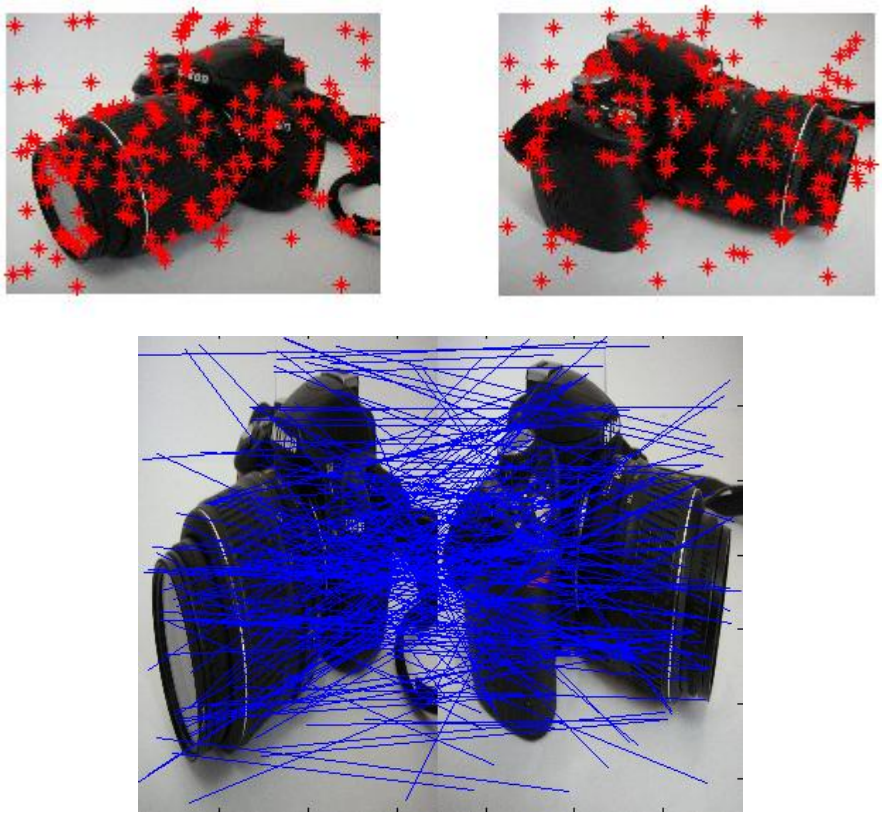

Figure 4.20: Plots of matching keypoints (top) and successful matches (bottom) between *camera1* and *camera5*

Table 4.16: Details of results obtained by comparing *camera1* and *camera5*

	camera1	***camera5***
Number of keypoints detected	3914	4622
Number of successful matches	196	
Time taken (s)	15.022	

4.4 Experiment 3: Effectiveness of SIFT Detection on White-Coloured Objects

4.4.1 Experiment 3(a): Plain White Vase

Figure below shows five images of a plain white vase in different angles captured in white background. Starting from the left of first row, the names of images are known as *vase_white1*, *vase_white2*, *vase_white3*, *vase_white4* and *vase_white5*, whereby *vase_white1* is the base image.

Figure 4.21: Images used in Experiment 3(a) for white background

1. SIFT detection on *vase_white1* and *vase_white2*

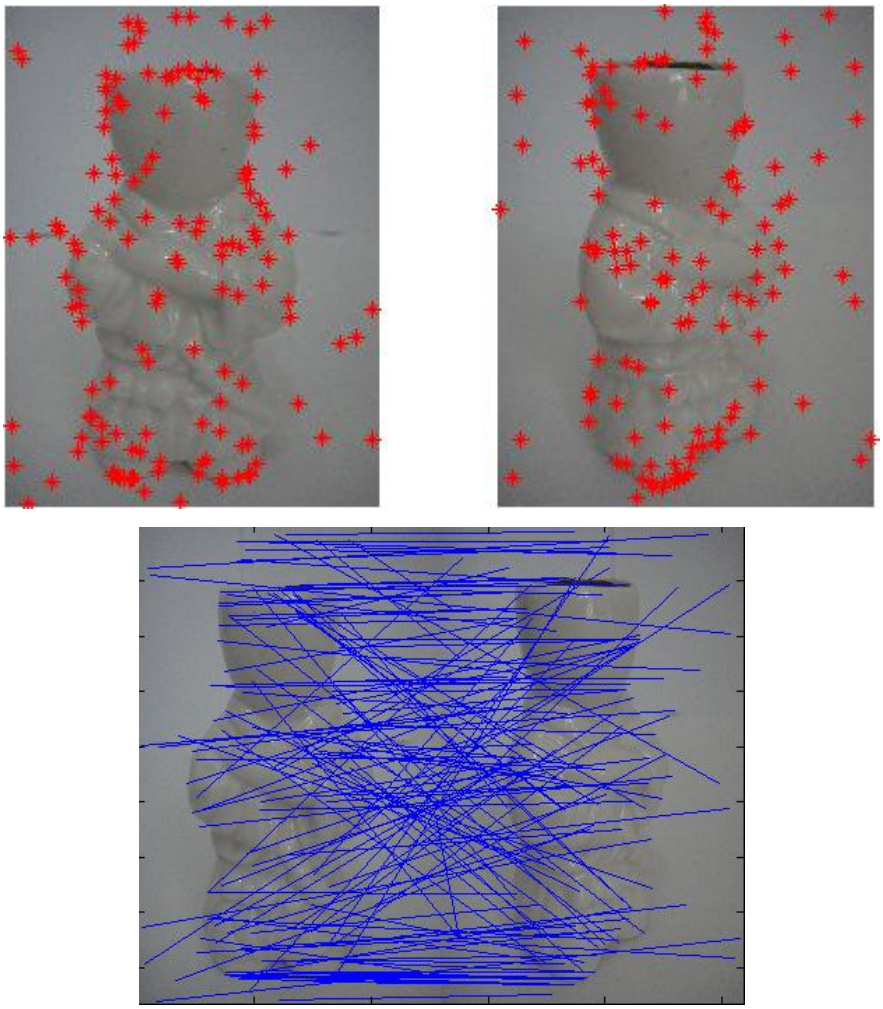

Figure 4.22: Plots of matching keypoints (top) and successful matches (bottom) between *vase_white1* and *vase_white2*

Table 4.17: Details of results obtained by comparing *vase_white1* and *vase_white2*

	vase_white1	***vase_white2***
Number of keypoints detected	3002	2892
Number of successful matches	153	
Time taken (s)	12.421	

2. SIFT detection on *vase_white1* and *vase_white3*

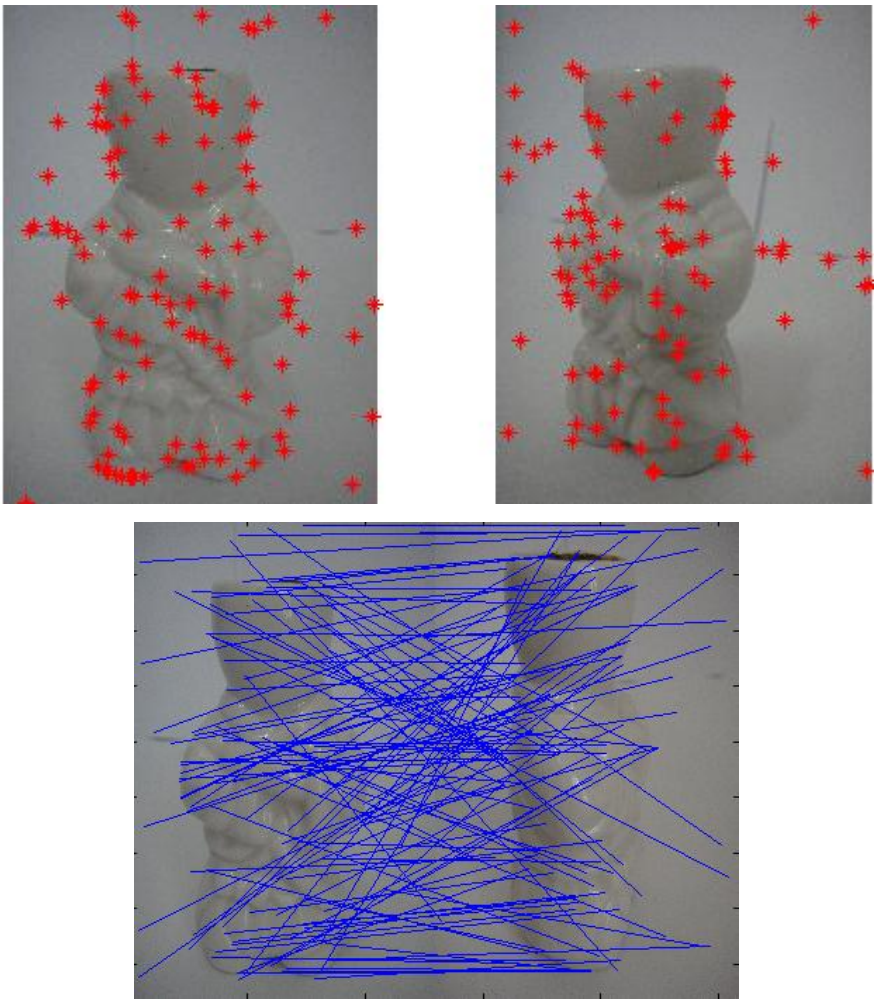

Figure 4.23: Plots of matching keypoints (top) and successful matches (bottom) between *vase_white1* and *vase_white3*

Table 4.18: Details of results obtained by comparing *vase_white1* and *vase_white3*

	vase_white1	***vase_white3***
Number of keypoints detected	3002	2748
Number of successful matches	115	
Time taken (s)	12.054	

3. SIFT detection on *vase_white1* and *vase_white4*

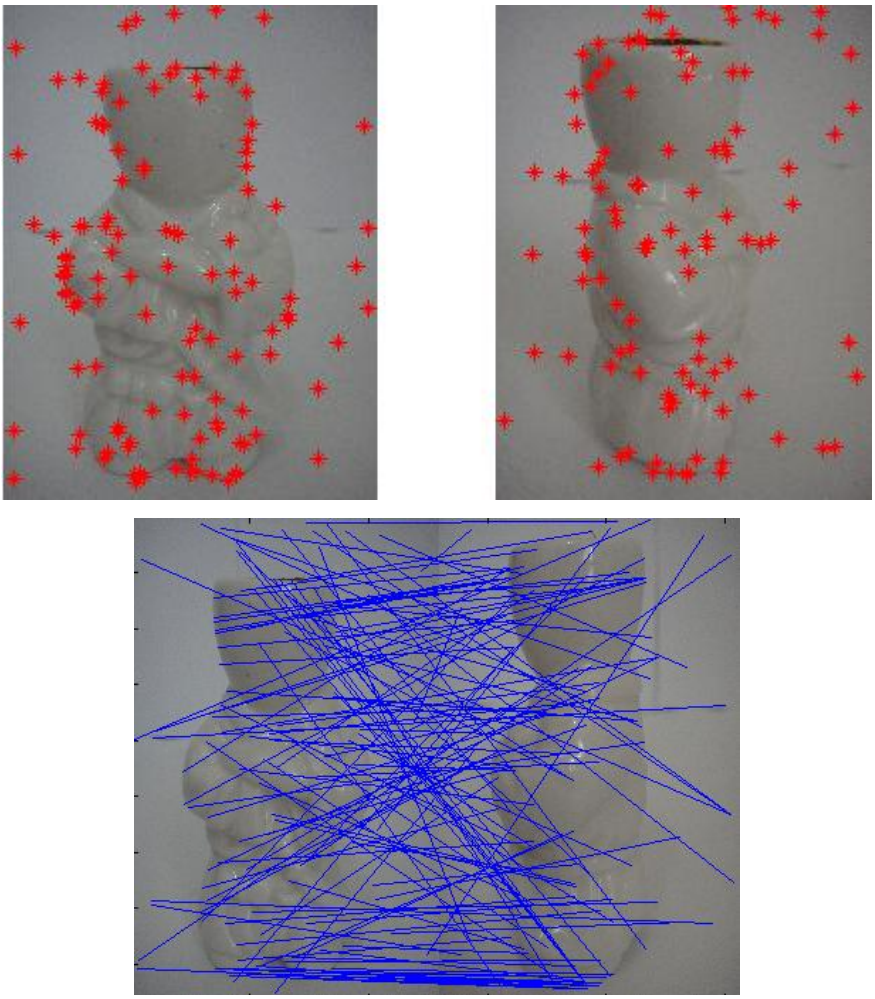

Figure 4.24: Plots of matching keypoints (top) and successful matches (bottom) between *vase_white1* and *vase_white4*

Table 4.19: Details of results obtained by comparing *vase_white1* and *vase_white4*

	vase_white1	***vase_white4***
Number of keypoints detected	3002	2814
Number of successful matches	123	
Time taken (s)	12.755	

4. SIFT detection on *vase_white1* and *vase_white5*

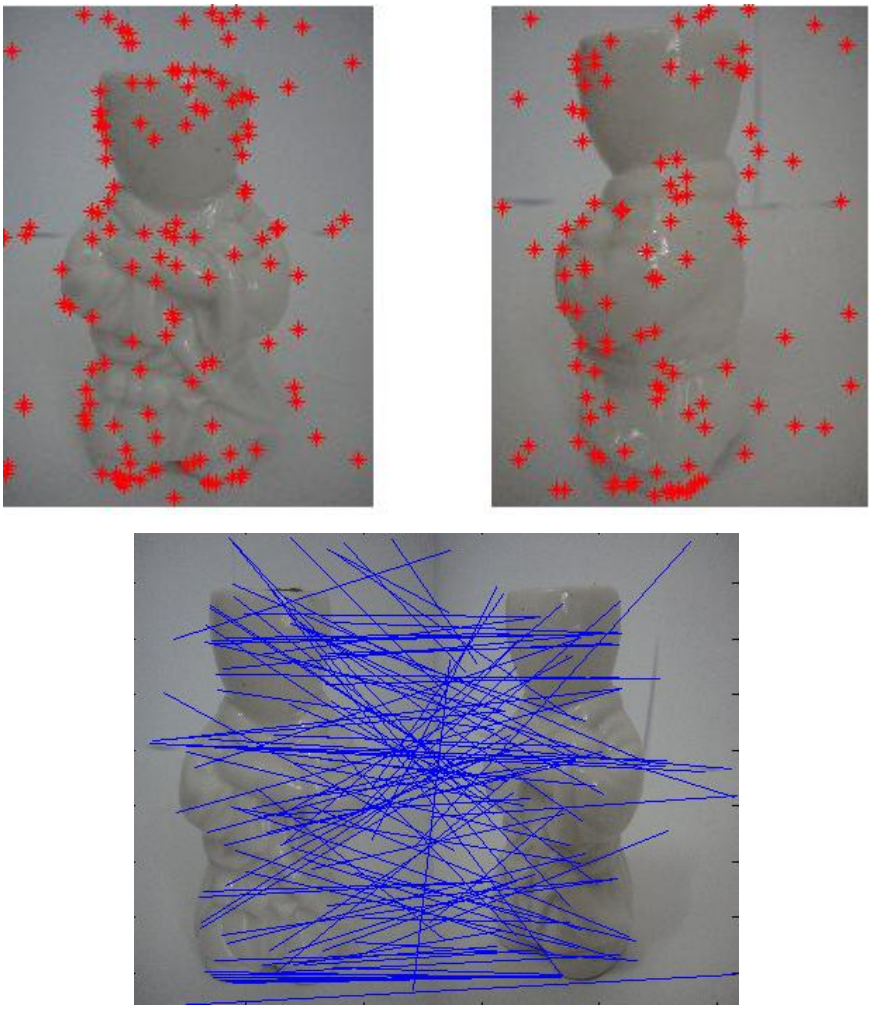

Figure 4.25: Plots of matching keypoints (top) and successful matches (bottom) between *vase_white1* and *vase_white5*

Table 4.20: Details of results obtained by comparing *vase_white1* and *vase_white5*

	vase_white1	***vase_white5***
Number of keypoints detected	3002	2766
Number of successful matches	133	
Time taken (s)	12.460	

Figure below shows five images of the same plain white vase but in black background captured from different angles. Starting from the left of first row, the names of images are known as *vase_black1*, *vase_black2*, *vase_black3*, *vase_black4* and *vase_black5*, whereby *vase_black1* is the base image.

Figure 4.26: Images used in Experiment 3(a) for black background

1. SIFT detection on *vase_black1* and *vase_black2*

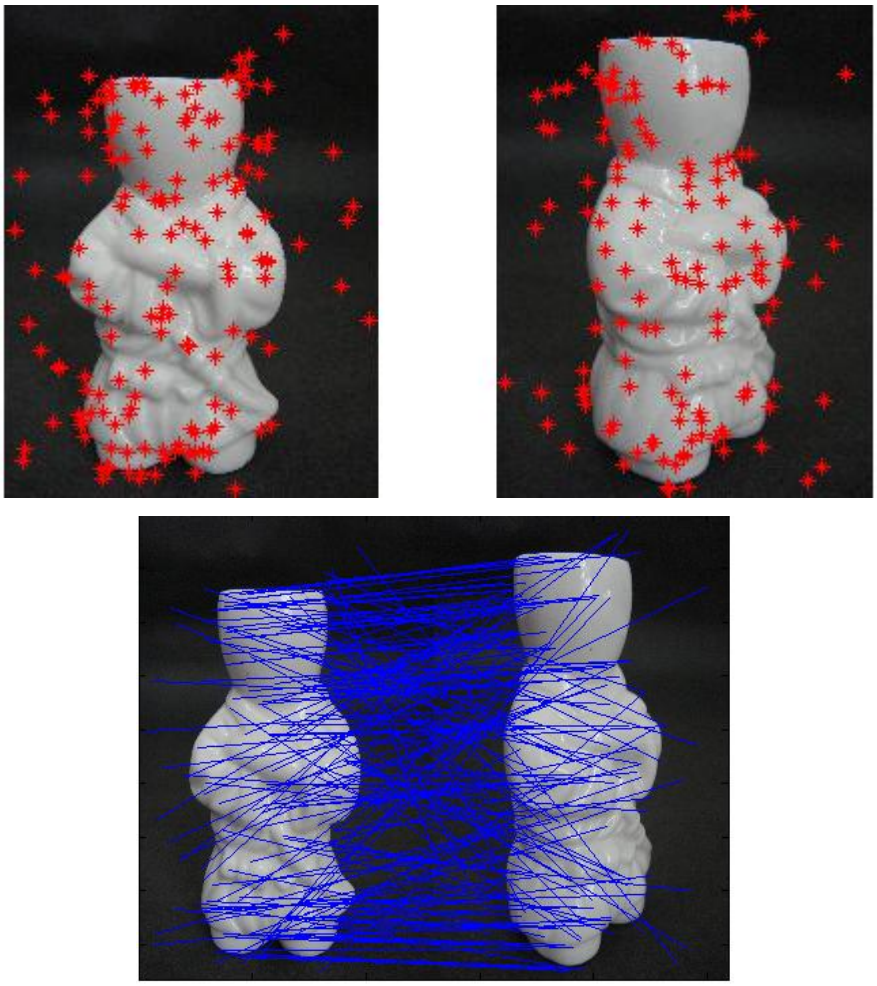

Figure 4.27: Plots of matching keypoints (top) and successful matches (bottom) between *vase_black1* and *vase_black2*

Table 4.21: Details of results obtained by comparing *vase_black1* and *vase_black2*

	vase_black1	***vase_black2***
Number of keypoints detected	3976	3759
Number of successful matches	166	
Time taken (s)	16.207	

2. SIFT detection on *vase_black1* and *vase_black3*

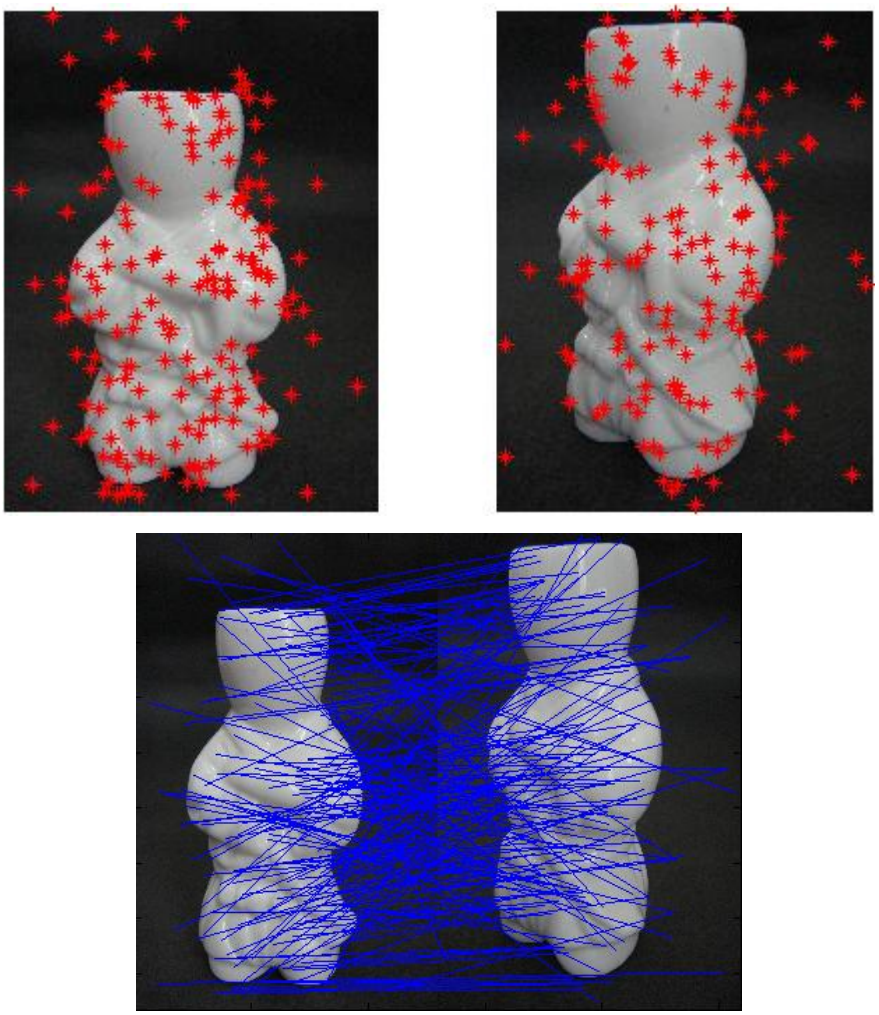

Figure 4.28: Plots of matching keypoints (top) and successful matches (bottom) between *vase_black1* and *vase_black3*

Table 4.22: Details of results obtained by comparing *vase_black1* and *vase_black3*

	vase_black1	***vase_black3***
Number of keypoints detected	3976	4186
Number of successful matches	176	
Time taken (s)	17.469	

3. SIFT detection on *vase_black1* and *vase_black4*

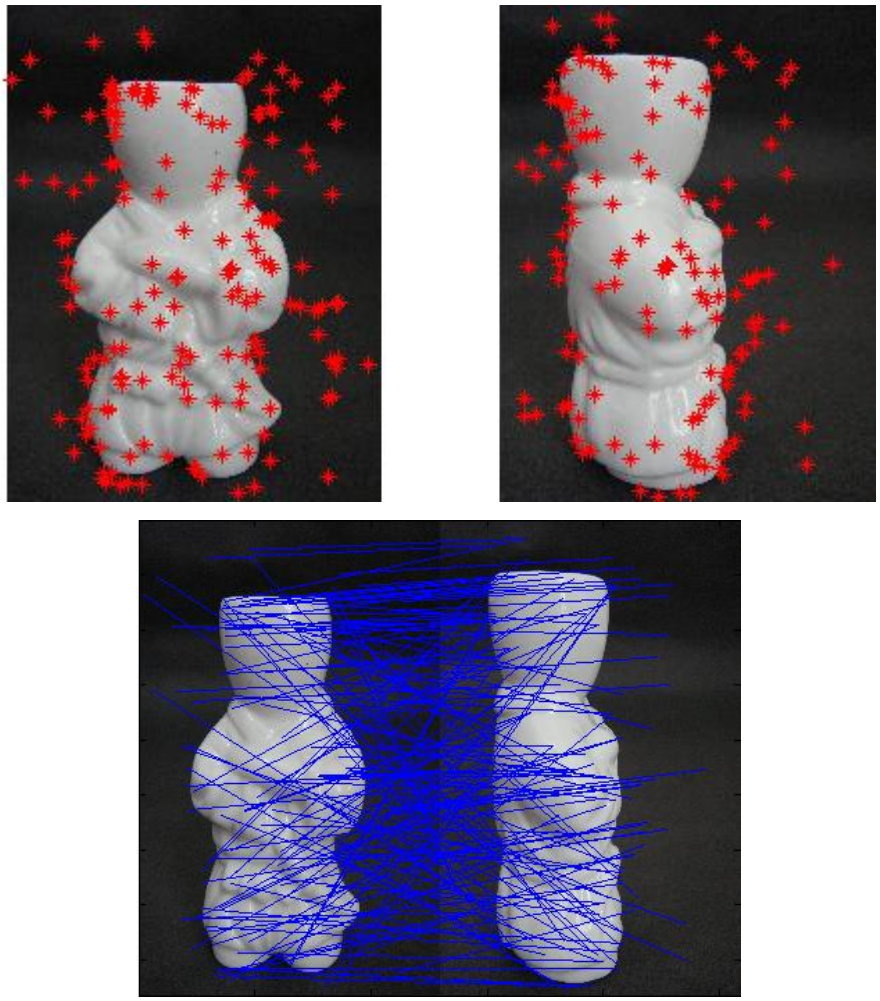

Figure 4.29: Plots of matching keypoints (top) and successful matches (bottom) between *vase_black1* and *vase_black4*

Table 4.23: Details of results obtained by comparing *vase_black1* and *vase_black4*

	vase_black1	***vase_black4***
Number of keypoints detected	3976	4188
Number of successful matches	173	
Time taken (s)	17.166	

4. SIFT detection on *vase_black1* and *vase_black5*

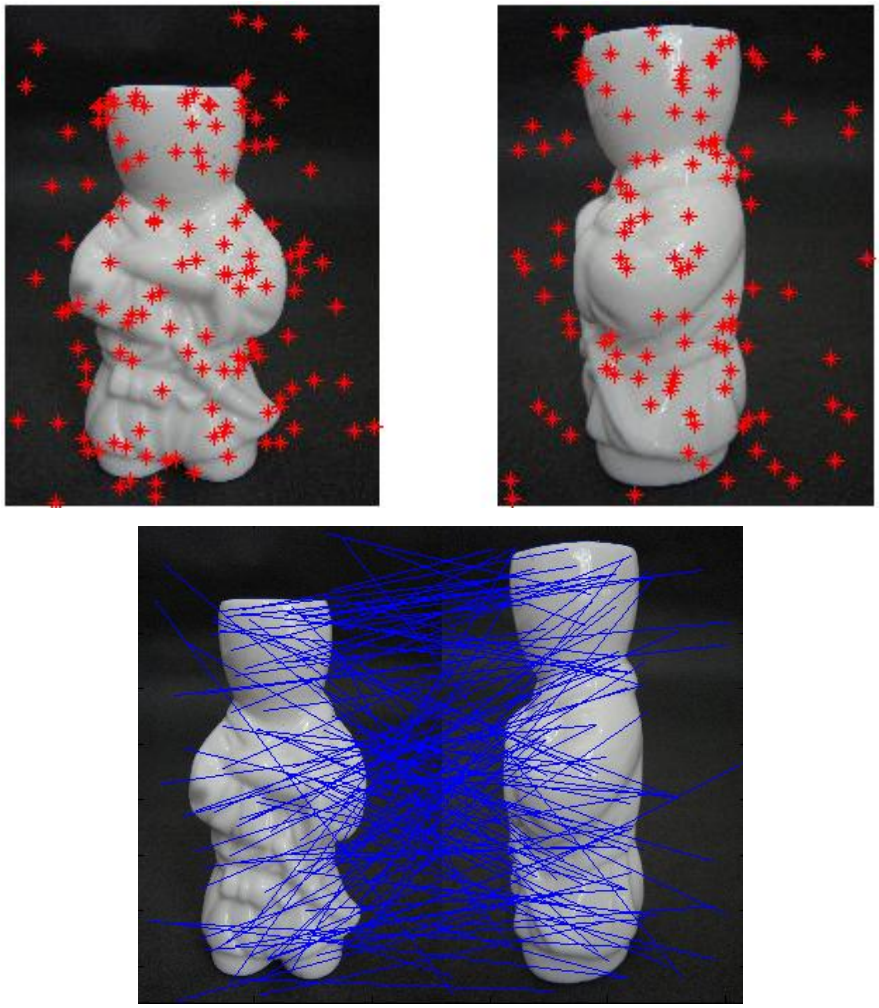

Figure 4.30: Plots of matching keypoints (top) and successful matches (bottom) between *vase_black1* and *vase_black5*

Table 4.24: Details of results obtained by comparing *vase_black1* and *vase_black5*

	vase_black1	***vase_black5***
Number of keypoints detected	3976	3918
Number of successful matches	136	
Time taken (s)	16.325	

4.4.2 Experiment 3(b): Plain White Cup

Figure below shows five images of a plain white cup in different angles captured in white background. Starting from the left of first row, the names of images are known as *cup_white1*, *cup_white2*, *cup_white3*, *cup_white4* and *cup_white5*, whereby *cup_white1* is the base image.

Figure 4.31: Images used in Experiment 3(b) for white background

1. SIFT detection on *cup_white1* and *cup_white2*

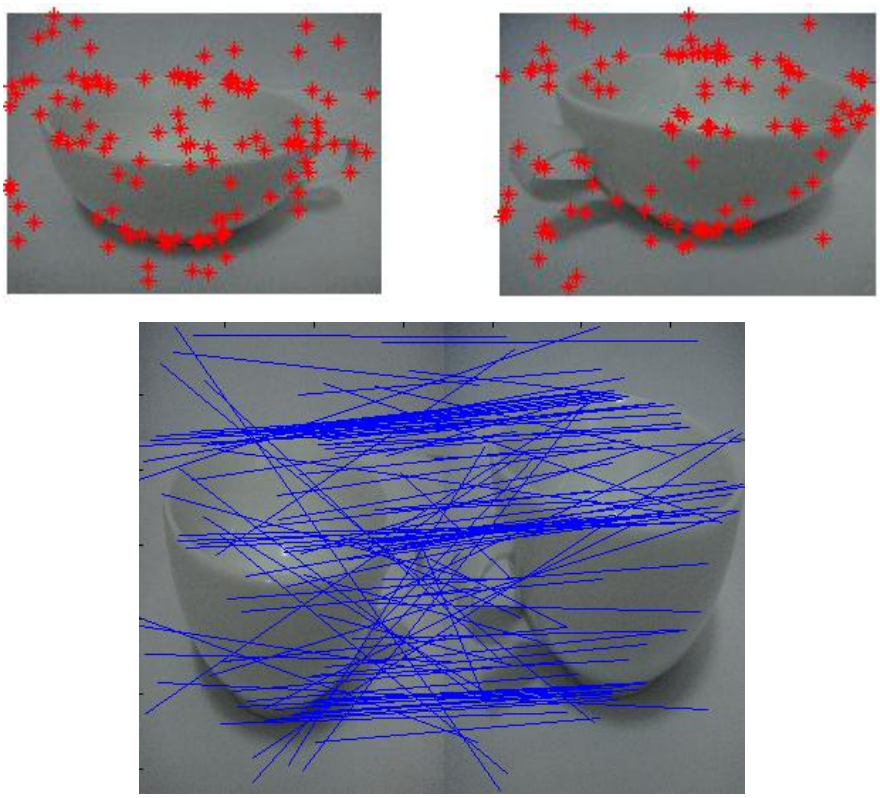

Figure 4.32: Plots of matching keypoints (top) and successful matches (bottom) between *cup_white1* and *cup_white2*

Table 4.25: Details of results obtained by comparing *cup_white1* and *cup_white2*

	cup_white1	***cup_white2***
Number of keypoints detected	2692	2584
Number of successful matches	118	
Time taken (s)	11.852	

2. SIFT detection on *cup_white1* and *cup_white3*

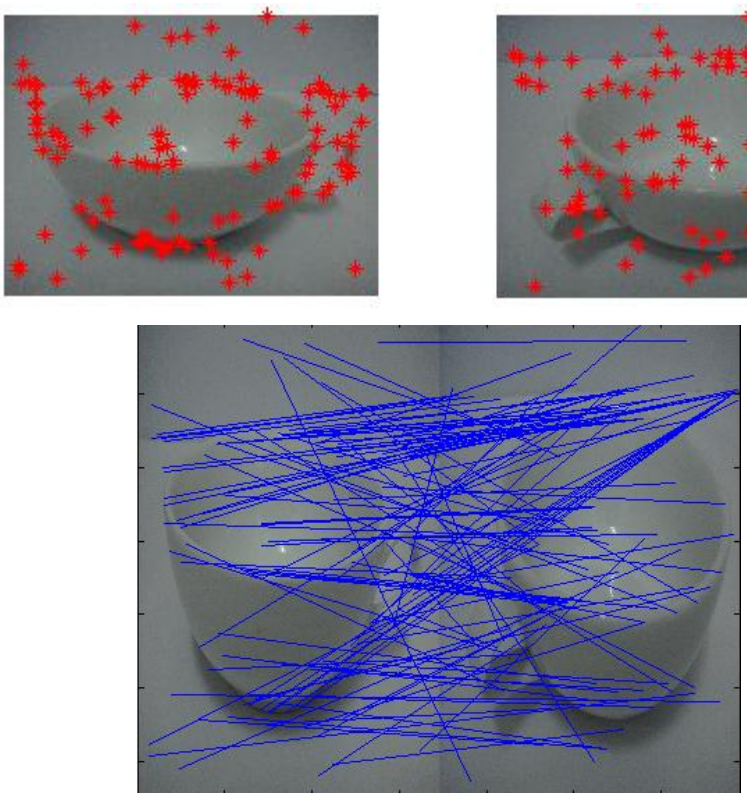

Figure 4.33: Plots of matching keypoints (top) and successful matches (bottom) between *cup_white1* and *cup_white3*

Table 4.26: Details of results obtained by comparing *cup_white1* and *cup_white3*

	cup_white1	***cup_white3***
Number of keypoints detected	2692	2677
Number of successful matches	119	
Time taken (s)	12.023	

3. SIFT detection on *cup_white1* and *cup_white4*

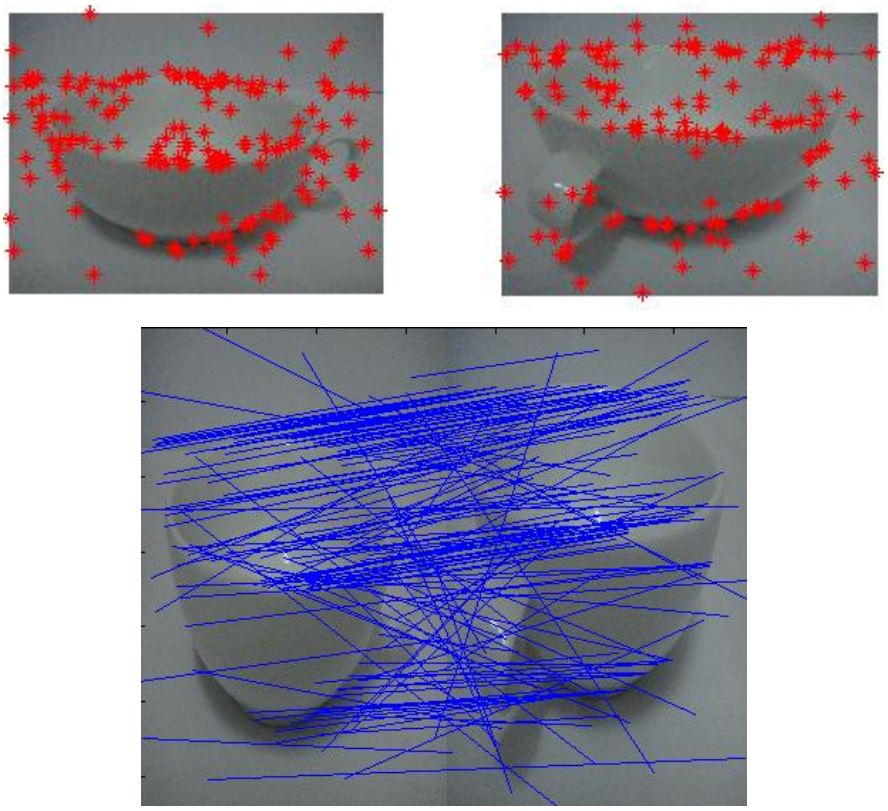

Figure 4.34: Plots of matching keypoints (top) and successful matches (bottom) between *cup_white1* and *cup_white4*

Table 4.27: Details of results obtained by comparing *cup_white1* and *cup_white4*

	cup_white1	***cup_white4***
Number of keypoints detected	2692	2754
Number of successful matches	158	
Time taken (s)	12.408	

4. SIFT detection on *cup_white1* and *cup_white5*

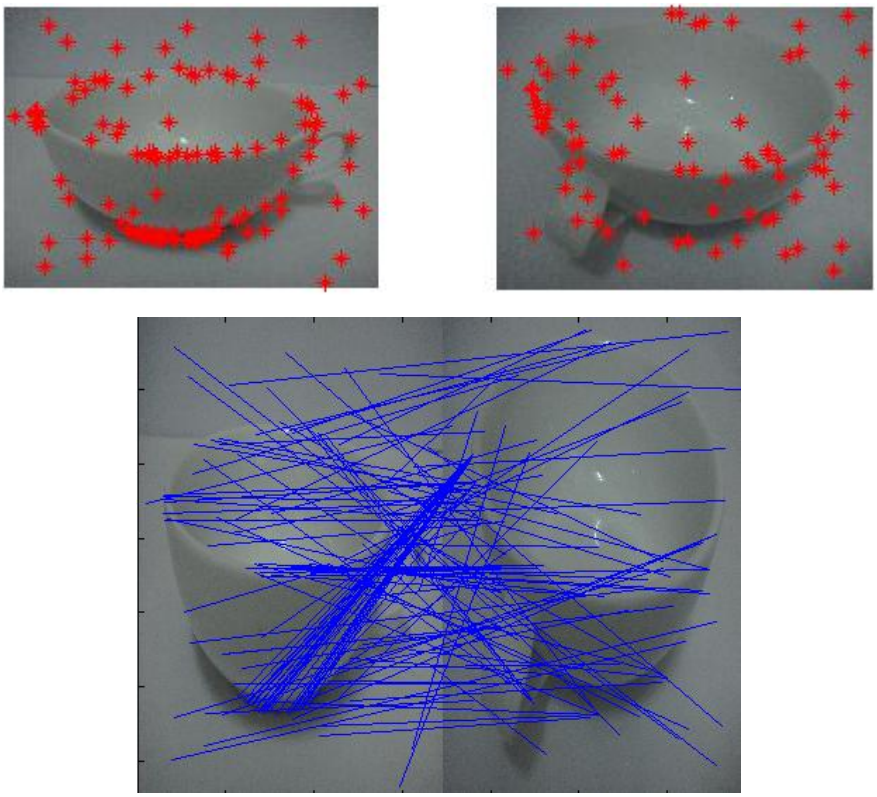

Figure 4.35: Plots of matching keypoints (top) and successful matches (bottom) between *cup_white1* and *cup_white5*

Table 4.28: Details of results obtained by comparing *cup_white1* and *cup_white5*

	cup_white1	***cup_white5***
Number of keypoints detected	2692	2595
Number of successful matches	128	
Time taken (s)	13.118	

Figure below shows five images of the same white cup but in black background captured from different angles. Starting from the left of first row, the names of images are known as *cup_black1*, *cup_black2*, *cup_black3*, *cup_black4* and *cup_black5*, whereby *cup_black1* is the base image.

Figure 4.36: Images used in Experiment 3(b) for black background

1. SIFT detection on *cup_black1* and *cup_black2*

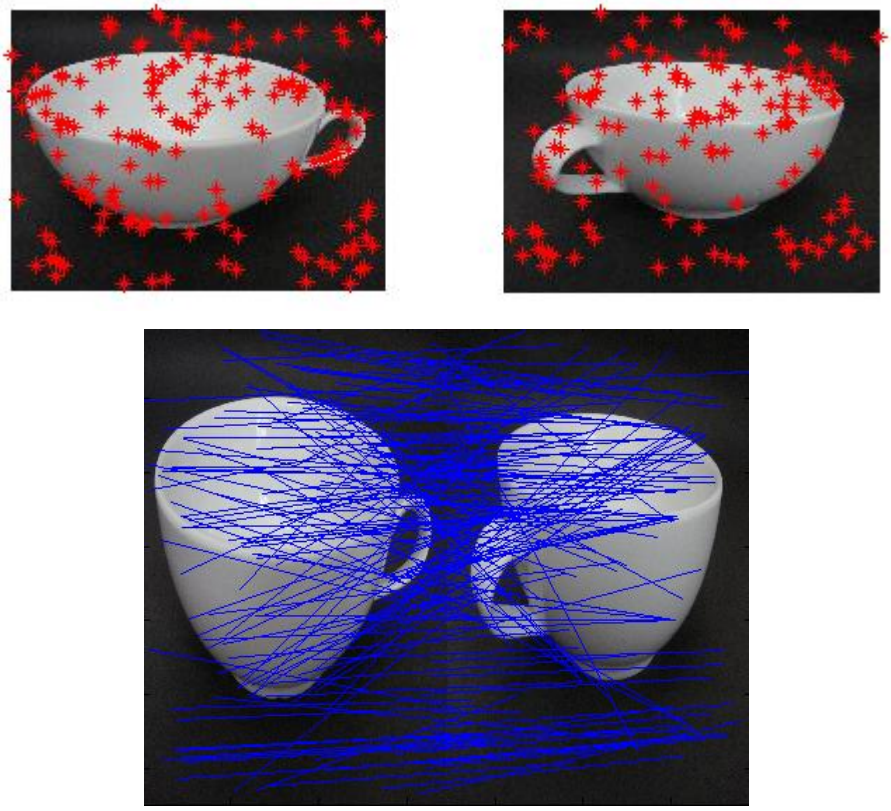

Figure 4.37: Plots of matching keypoints (top) and successful matches (bottom) between *cup_black1* and *cup_black2*

Table 4.29: Details of results obtained by comparing *cup_black1* and *cup_black2*

	cup_black1	***cup_black2***
Number of keypoints detected	4268	4487
Number of successful matches	166	
Time taken (s)	18.682	

2. SIFT detection on *cup_black1* and *cup_black3*

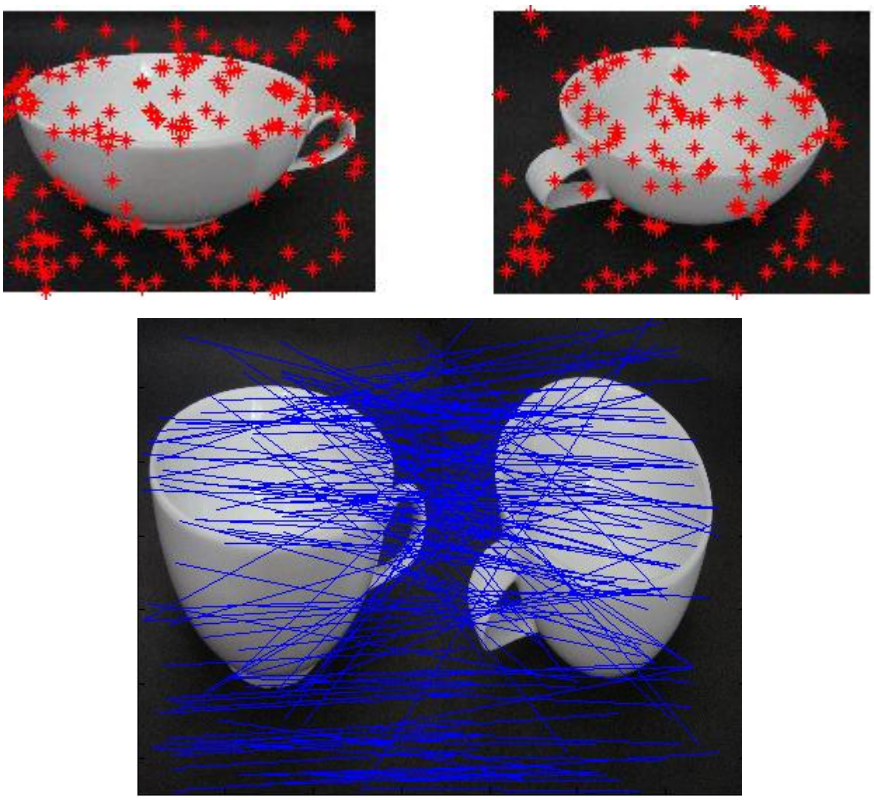

Figure 4.38: Plots of matching keypoints (top) and successful matches (bottom) between *cup_black1* and *cup_black3*

Table 4.30: Details of results obtained by comparing *cup_black1* and *cup_black3*

	cup_black1	***cup_black3***
Number of keypoints detected	4268	3941
Number of successful matches	161	
Time taken (s)	16.939	

3. SIFT detection on *cup_black1* and *cup_black4*

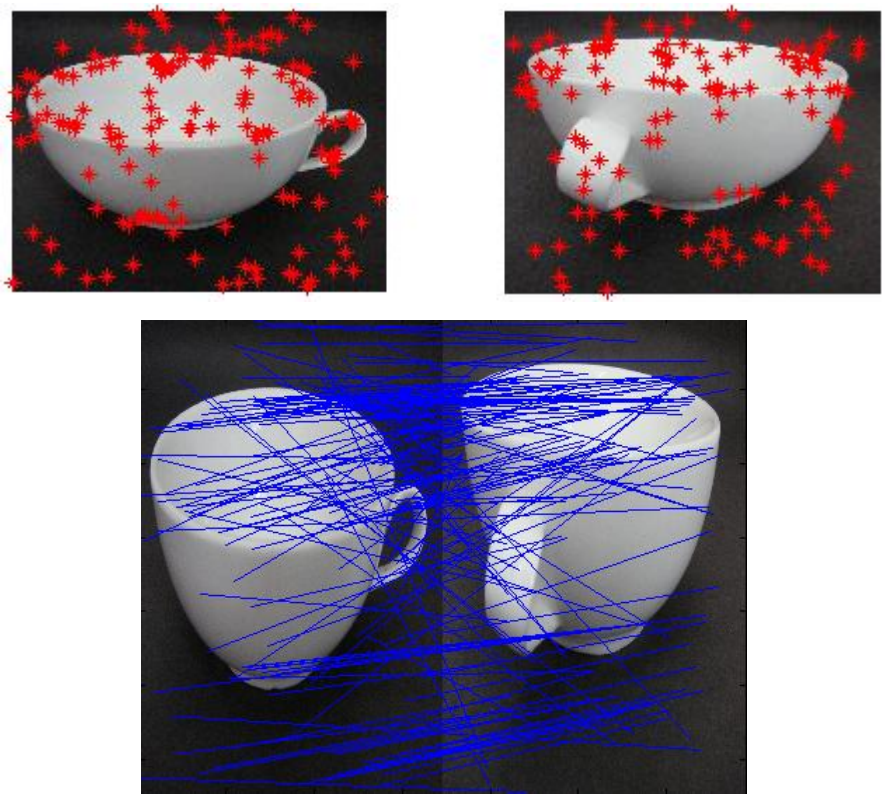

Figure 4.39: Plots of matching keypoints (top) and successful matches (bottom) between *cup_black1* and *cup_black4*

Table 4.31: Details of results obtained by comparing *cup_black1* and *cup_black4*

	cup_black1	***cup_black4***
Number of keypoints detected	4268	5192
Number of successful matches	146	
Time taken (s)	18.873	

4. SIFT detection on *cup_black1* and *cup_black5*

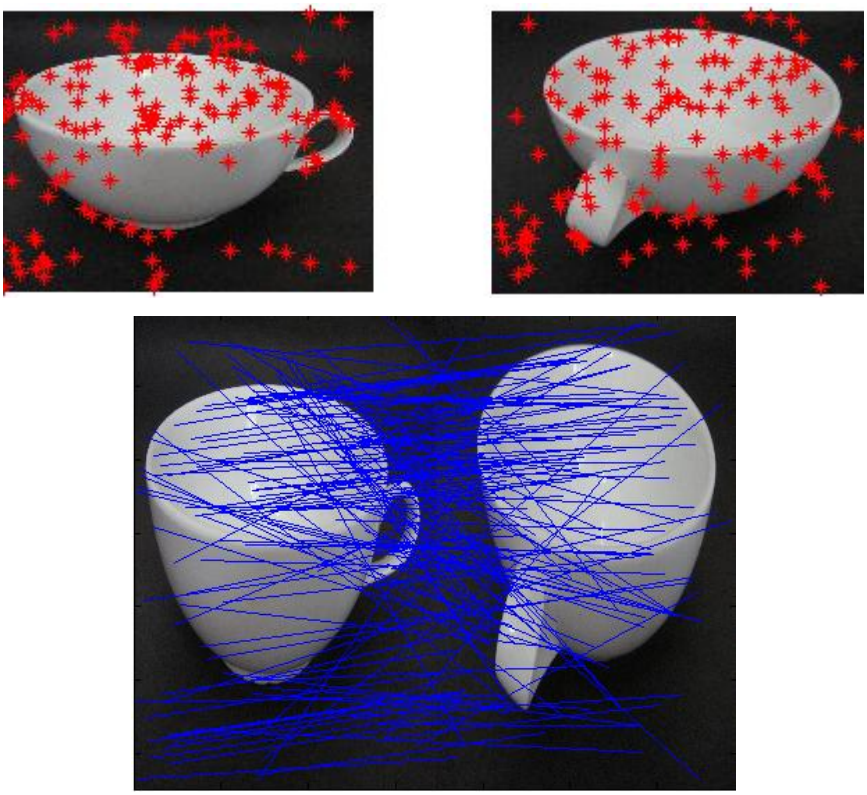

Figure 4.40: Plots of matching keypoints (top) and successful matches (bottom) between *cup_black1* and *cup_black5*

Table 4.32: Details of results obtained by comparing *cup_black1* and *cup_black5*

	cup_black1	***cup_black5***
Number of keypoints detected	4268	3708
Number of successful matches	149	
Time taken (s)	15.618	

4.5 Experiment 4: Effectiveness of SIFT Detection on Transparent Objects

4.5.1 Experiment 4(a): Transparent Coca-Cola Glass

Figure below shows five images of a glass captured from different angles in white background. Starting from the left of first row, the names of images are known as *glass_white1*, *glass_white2*, *glass_white3*, *glass_white4* and *glass_white5*, whereby *glass_white1* is the base image.

Figure 4.41: Images used in Experiment 4(a) for white background

1. SIFT detection on *glass_white1* and *glass_white2*

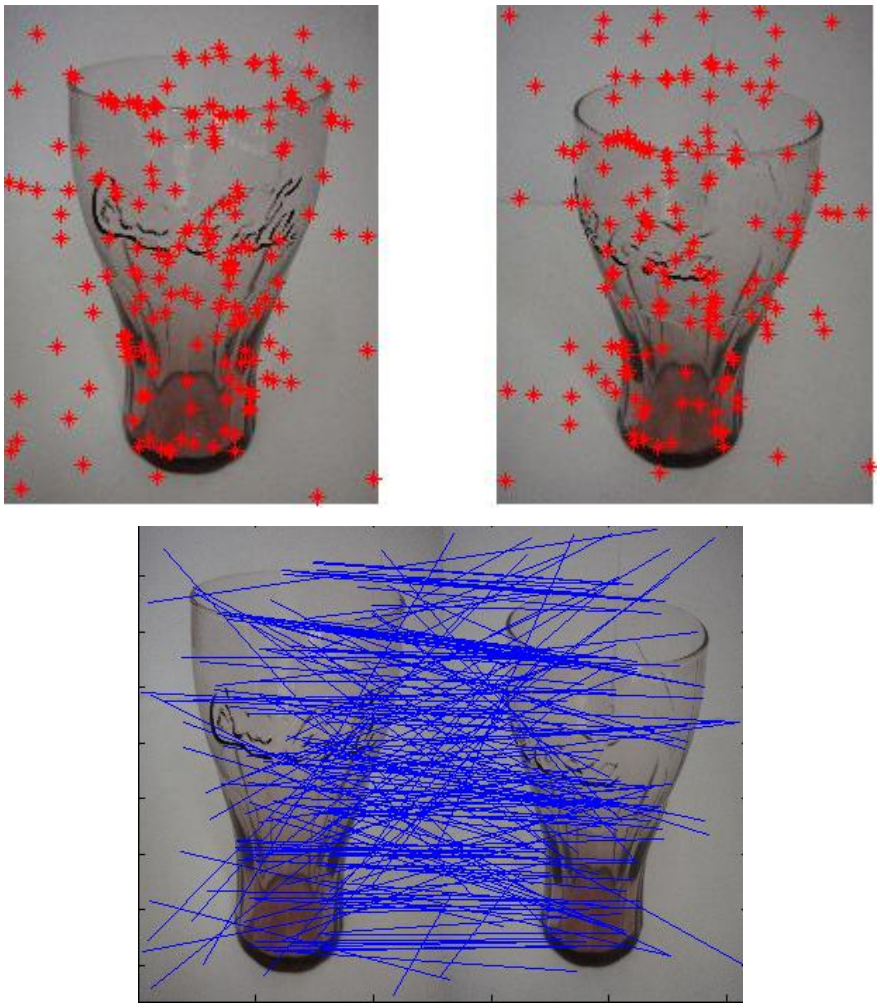

Figure 4.42: Plots of matching keypoints (top) and successful matches (bottom) between *glass_white1*and *glass_white2*

Table 4.33: Details of results obtained by comparing *glass_white1* and *glass_white2*

	glass_white1	***glass_white2***
Number of keypoints detected	3522	3529
Number of successful matches	197	
Time taken (s)	12.439	

2. SIFT detection on *glass_white1* and *glass_white3*

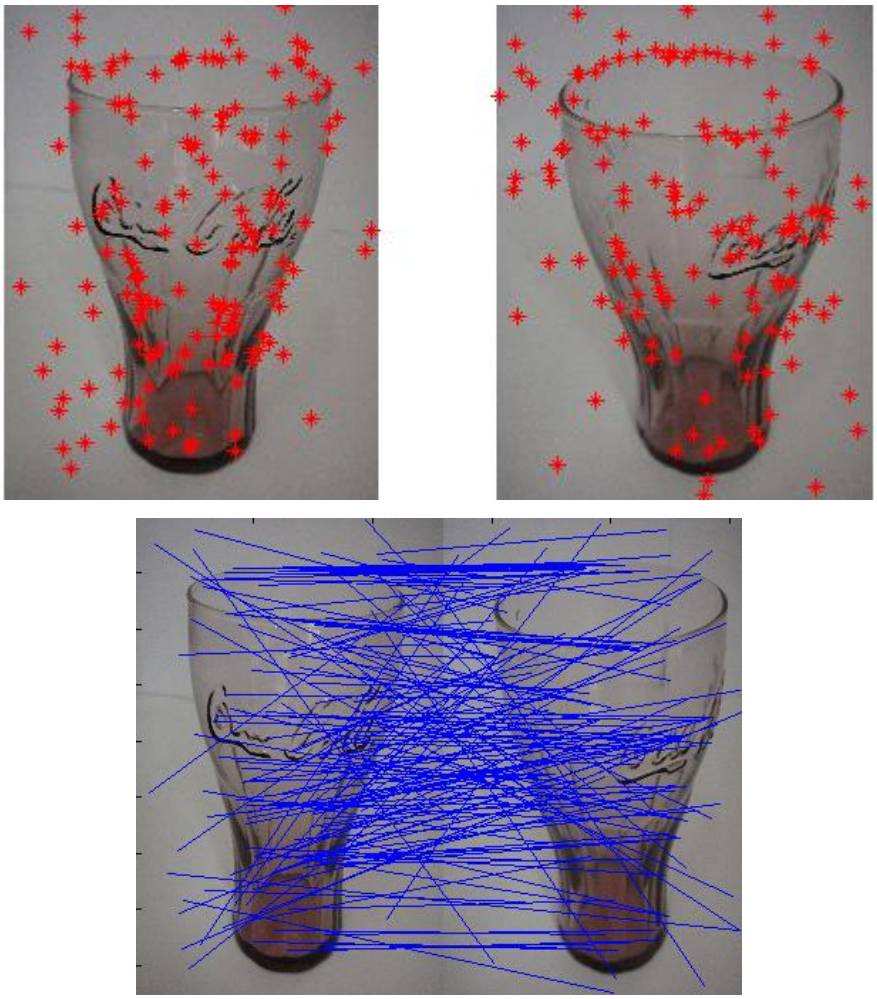

Figure 4.43: Plots of matching keypoints (top) and successful matches (bottom) between *glass_white1*and *glass_white3*

Table 4.34: Details of results obtained by comparing *glass_white1* and *glass_white3*

	glass_white1	***glass_white3***
Number of keypoints detected	3522	3176
Number of successful matches	167	
Time taken (s)	12.684	

3. SIFT detection on *glass_white1* and *glass_white4*

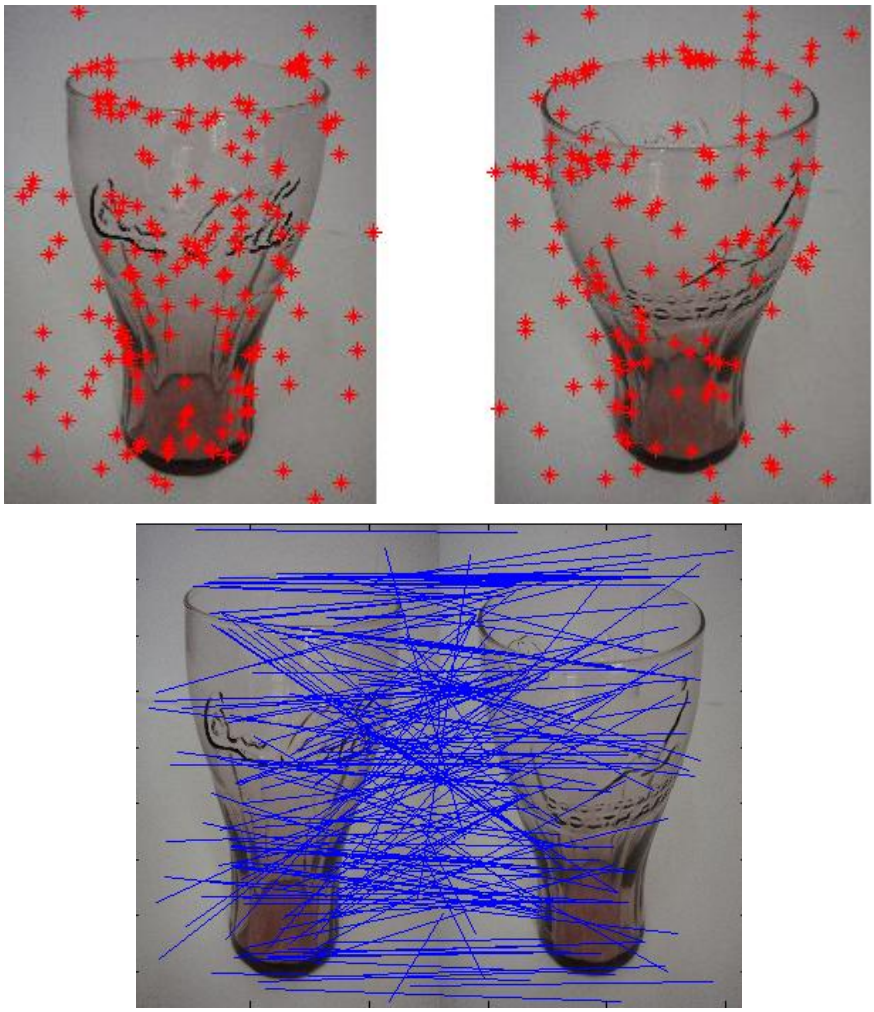

Figure 4.44: Plots of matching keypoints (top) and successful matches (bottom) between *glass_white1*and *glass_white4*

Table 4.35: Details of results obtained by comparing *glass_white1* and *glass_white4*

	glass_white1	***glass_white4***
Number of keypoints detected	3522	3598
Number of successful matches	187	
Time taken (s)	13.089	

4. SIFT detection on *glass_white1* and *glass_white5*

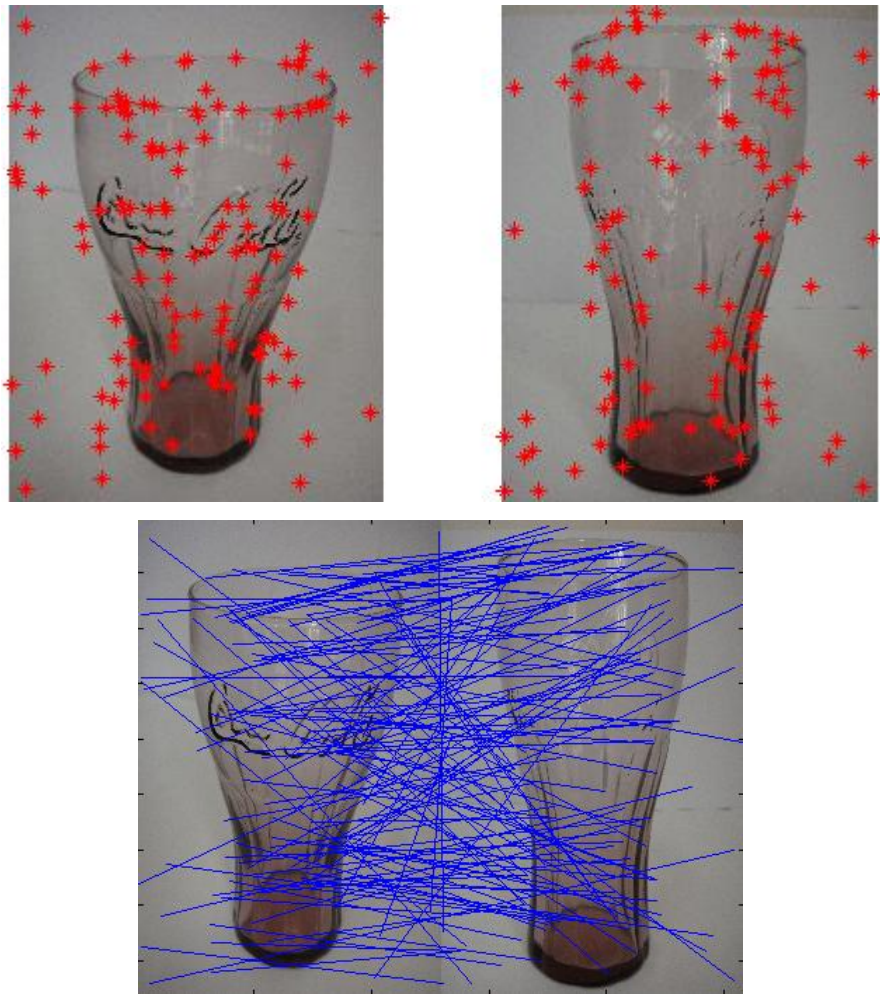

Figure 4.45: Plots of matching keypoints (top) and successful matches (bottom) between *glass_white1*and *glass_white5*

Table 4.36: Details of results obtained by comparing *glass_white1* and *glass_white5*

	glass_white1	***glass_white5***
Number of keypoints detected	3522	3766
Number of successful matches	135	
Time taken (s)	14.011	

Figure below shows five images of the same glass in different angles but captured in black background. Starting from the left of first row, the names of images are known as *glass_black1*, *glass_black2*, *glass_black3*, *glass_black4* and *glass_black5*, whereby *glass_black1* is the base image.

Figure 4.46: Images used in Experiment 4(a) for black background

1. SIFT detection on *glass_black1* and *glass_black2*

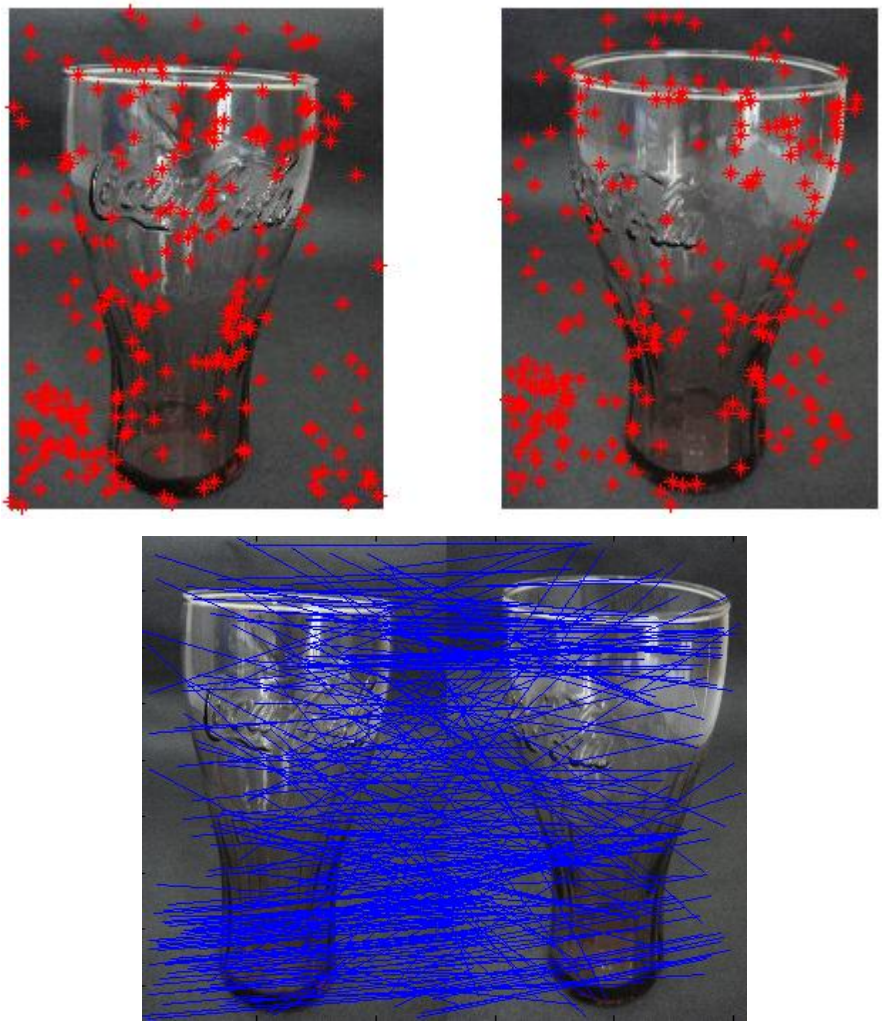

Figure 4.47: Plots of matching keypoints (top) and successful matches (bottom) between *glass_black1*and *glass_black2*

Table 4.37: Details of results obtained by comparing *glass_black1* and *glass_black2*

	glass_black1	***glass_black2***
Number of keypoints detected	7043	6623
Number of successful matches	273	
Time taken (s)	28.334	

2. SIFT detection on *glass_black1* and *glass_black3*

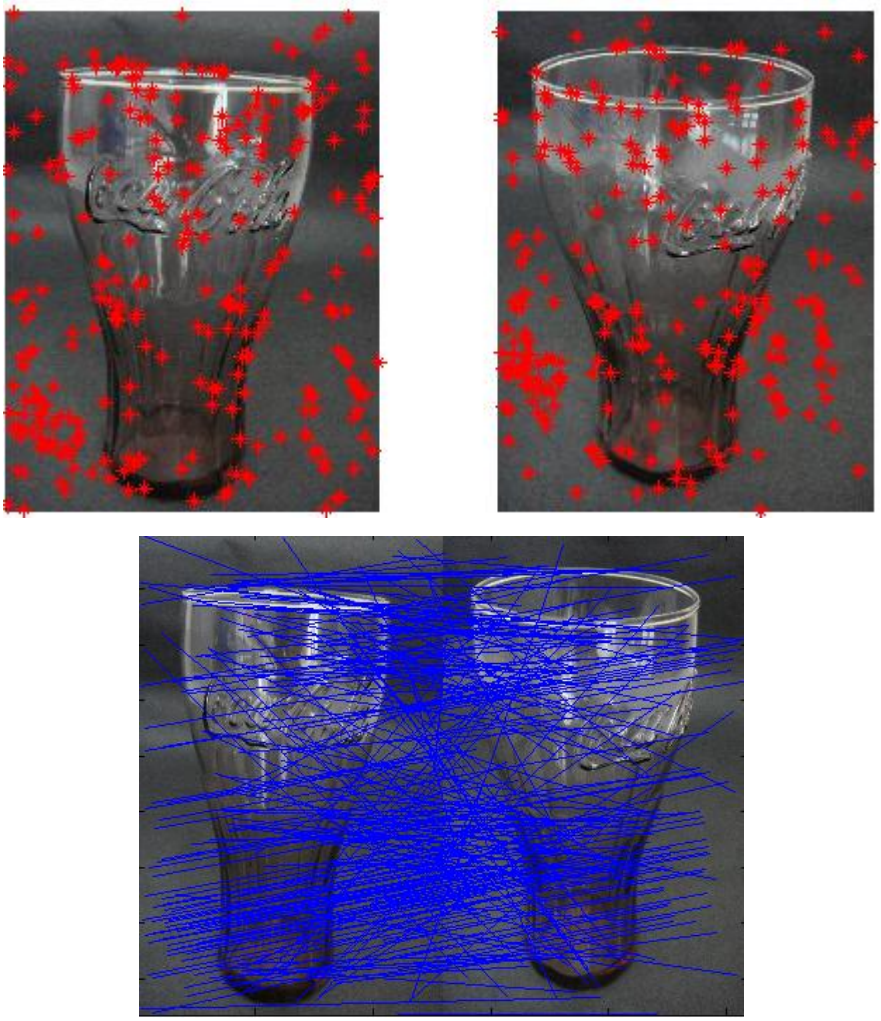

Figure 4.48: Plots of matching keypoints (top) and successful matches (bottom) between *glass_black1*and *glass_black3*

Table 4.38: Details of results obtained by comparing *glass_black1* and *glass_black3*

	glass_black1	***glass_black3***
Number of keypoints detected	7043	7529
Number of successful matches	270	
Time taken (s)	30.016	

3. SIFT detection on *glass_black1* and *glass_black4*

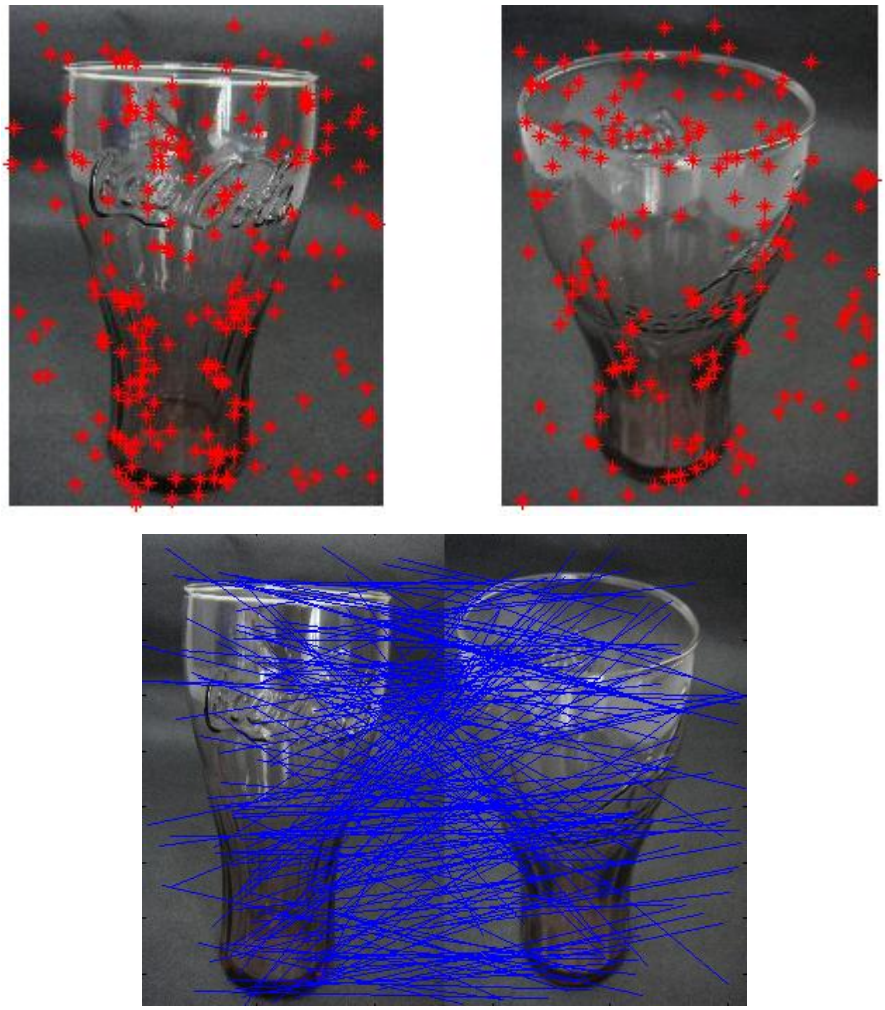

Figure 4.49: Plots of matching keypoints (top) and successful matches (bottom) between *glass_black1*and *glass_black4*

Table 4.39: Details of results obtained by comparing *glass_black1* and *glass_black4*

	glass_black1	***glass_black4***
Number of keypoints detected	7043	5004
Number of successful matches	214	
Time taken (s)	23.473	

4. SIFT detection on *glass_black1* and *glass_black5*

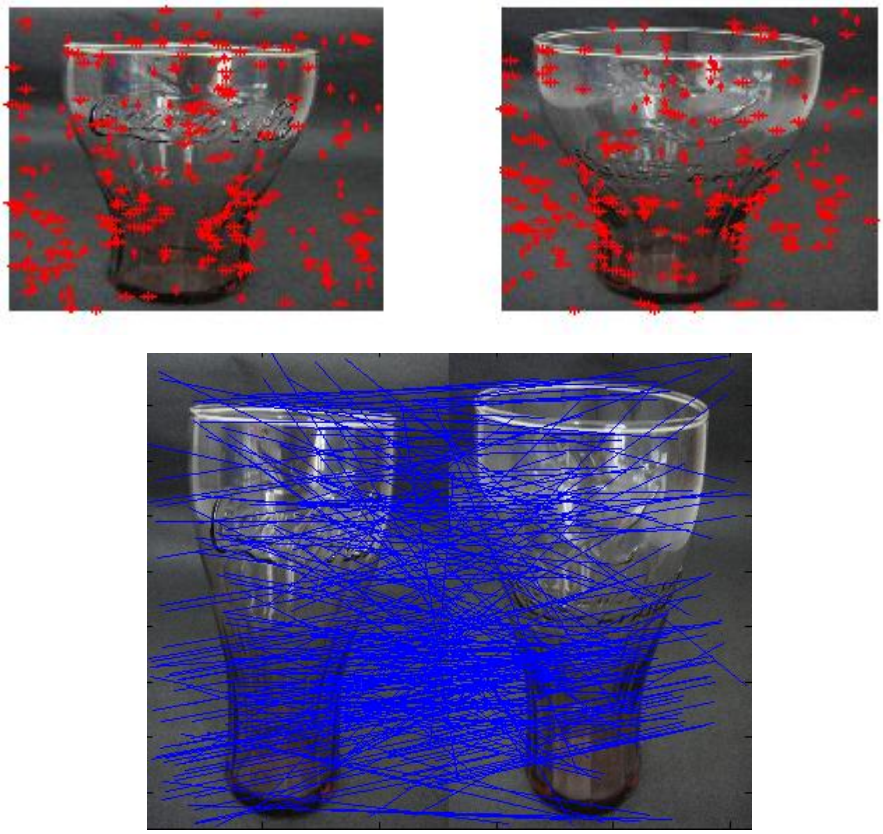

Figure 4.50: Plots of matching keypoints (top) and successful matches (bottom) between *glass_black1*and *glass_black5*

Table 4.40: Details of results obtained by comparing *glass_black1* and *glass_black5*

	glass_black1	***glass_black5***
Number of keypoints detected	7043	6537
Number of successful matches	260	
Time taken (s)	27.304	

4.5.2 Experiment 4(b): Transparent Plate

Figure below shows five images of a transparent plate captured from different angles in white background. Starting from the left of first row, the names of images are known as *plate_white1*, *plate_white2, plate_white3, plate_white4* and *plate_white5,* whereby *plate_white1* is the base image.

Figure 4.51: Images used in Experiment 4(b) for white background

1. SIFT detection on *plate_white1* and *plate_white2*

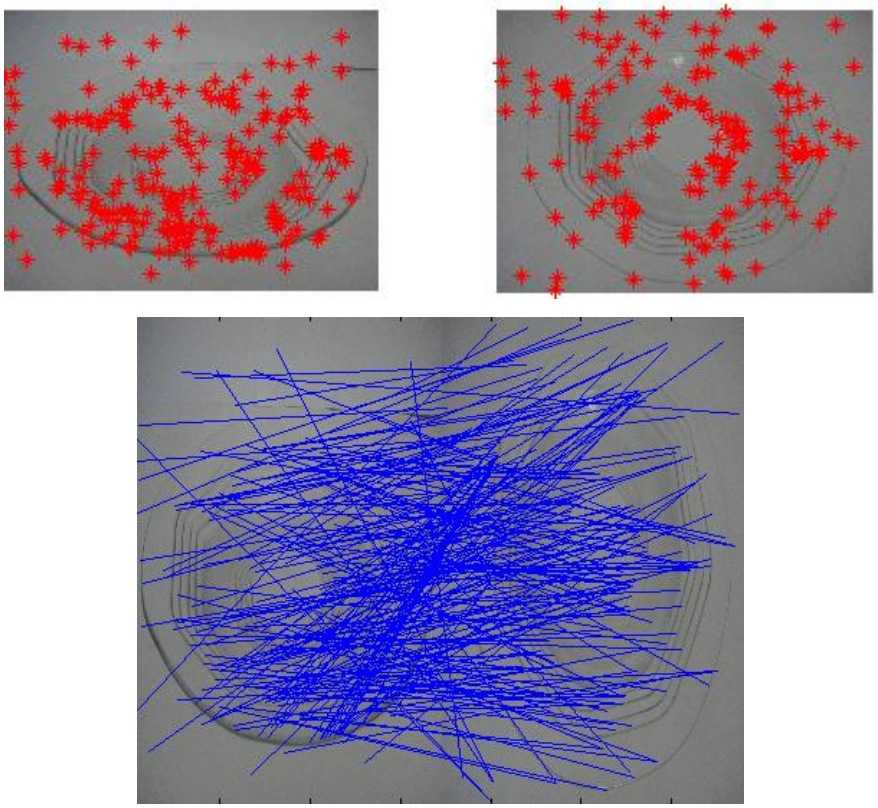

Figure 4.52: Plots of matching keypoints (top) and successful matches (bottom) between *plate_white1*and *plate_white2*

Table 4.41: Details of results obtained by comparing *plate_white1* and *plate_white2*

	plate_white1	***plate_white2***
Number of keypoints detected	4111	4522
Number of successful matches	223	
Time taken (s)	15.688	

2. SIFT detection on *plate_white1* and *plate_white3*

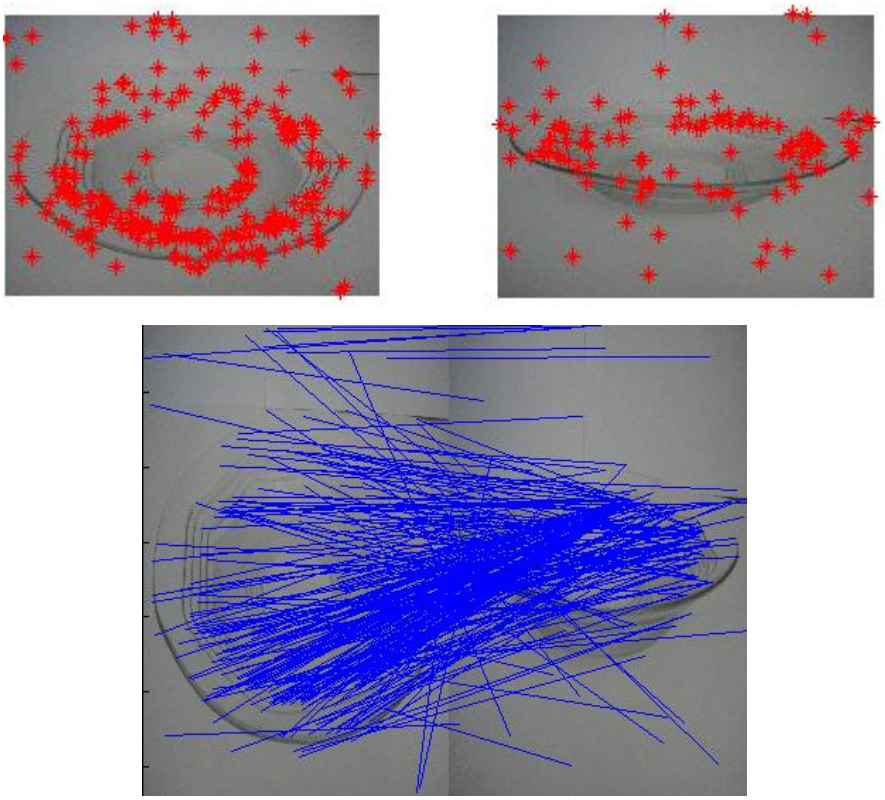

Figure 4.53: Plots of matching keypoints (top) and successful matches (bottom) between *plate_white1*and *plate_white3*

Table 4.42: Details of results obtained by comparing *plate_white1* and *plate_white3*

	plate_white1	***plate_white3***
Number of keypoints detected	4111	3391
Number of successful matches	272	
Time taken (s)	13.597	

3. SIFT detection on *plate_white1* and *plate_white4*

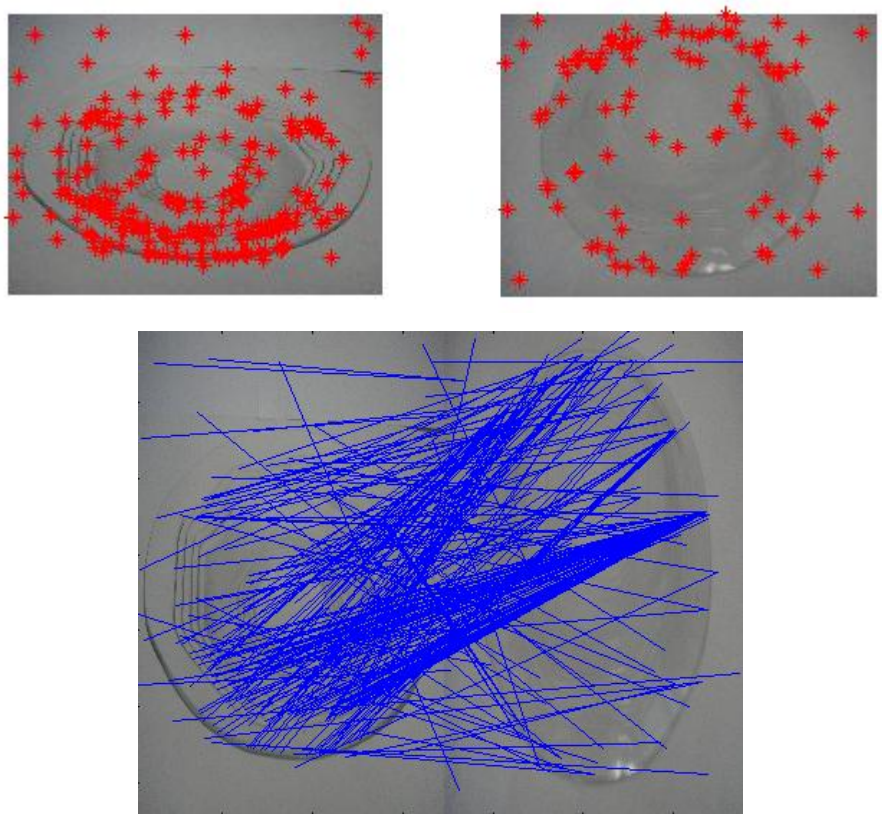

Figure 4.54: Plots of matching keypoints (top) and successful matches (bottom) between *plate_white1*and *plate_white4*

Table 4.43: Details of results obtained by comparing *plate_white1* and *plate_white4*

	plate_white1	***plate_white4***
Number of keypoints detected	4111	4005
Number of successful matches	245	
Time taken (s)	15.268	

4. SIFT detection on *plate_white1* and *plate_white5*

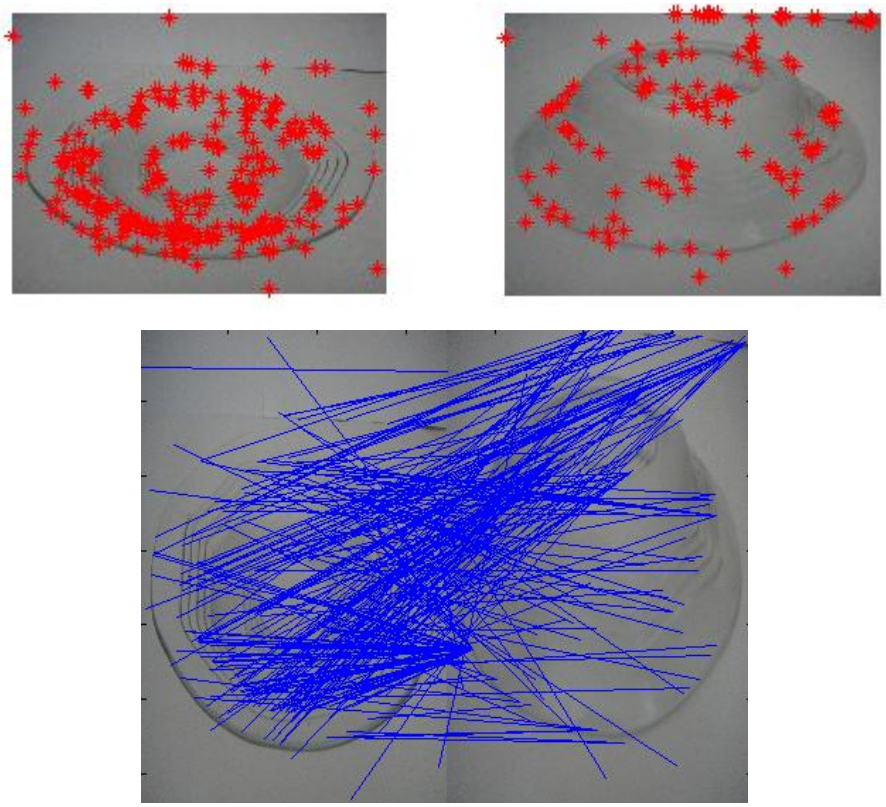

Figure 4.55: Plots of matching keypoints (top) and successful matches (bottom) between *plate_white1*and *plate_white5*

Table 4.44: Details of results obtained by comparing *plate_white1* and *plate_white5*

	plate_white1	***plate_white5***
Number of keypoints detected	4111	2986
Number of successful matches	272	
Time taken (s)	12.525	

Figure below shows five images of the same plate in different angles but captured in black background. Starting from the left of first row, the names of images are known as *plate_black1*, *plate_black2*, *plate_black3*, *plate_black4* and *plate_black5*, whereby *plate_black1* is the base image.

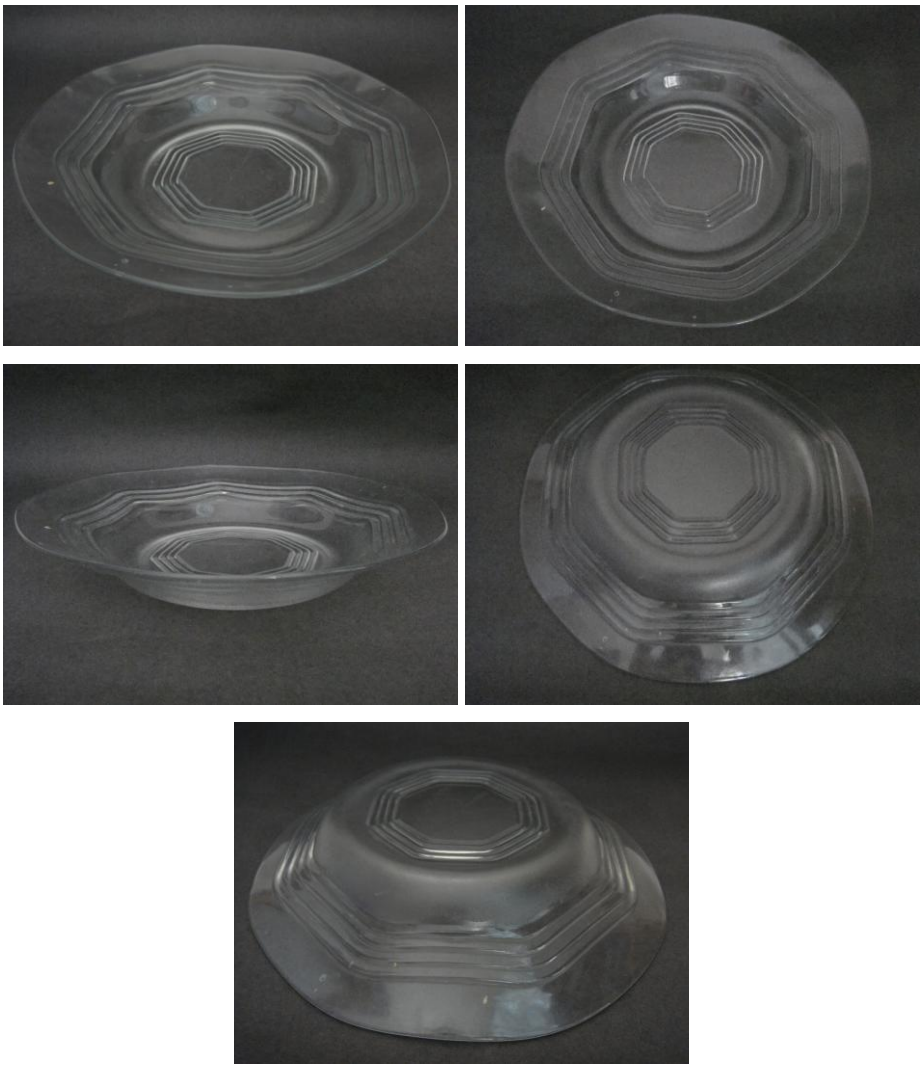

Figure 4.56: Images used in Experiment 4(b) for black background

1. SIFT detection on *plate_black1* and *plate_black2*

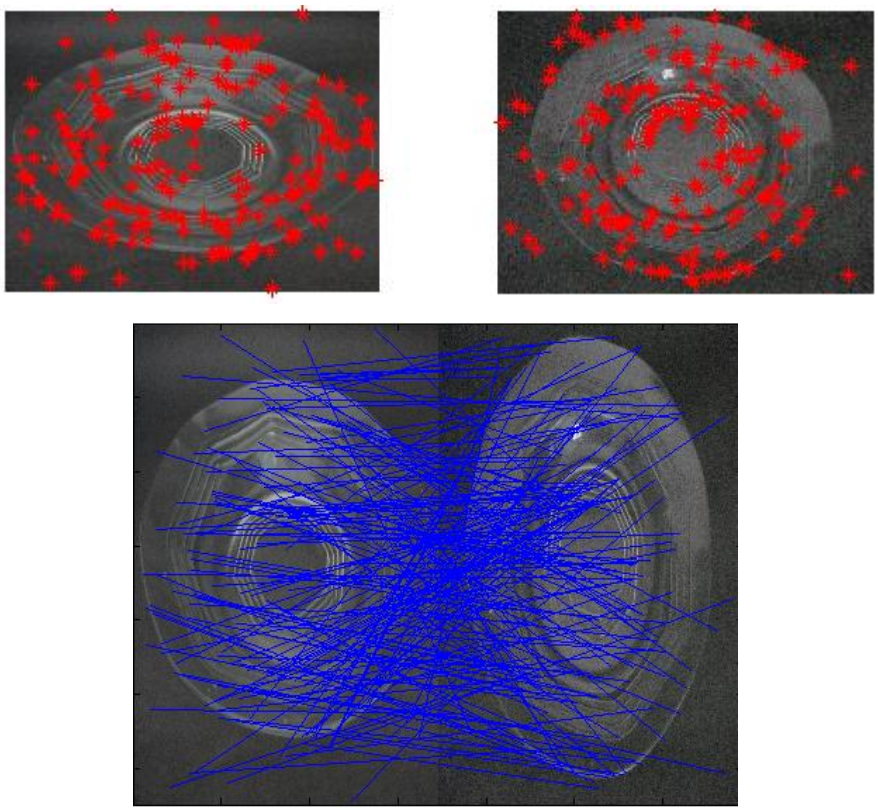

Figure 4.57: Plots of matching keypoints (top) and successful matches (bottom) between *plate_black1*and *plate_black2*

Table 4.45: Details of results obtained by comparing *plate_black1* and *plate_black2*

	plate_black1	***plate_black2***
Number of keypoints detected	7857	5284
Number of successful matches	199	
Time taken (s)	24.218	

2. SIFT detection on *plate_black1* and *plate_black3*

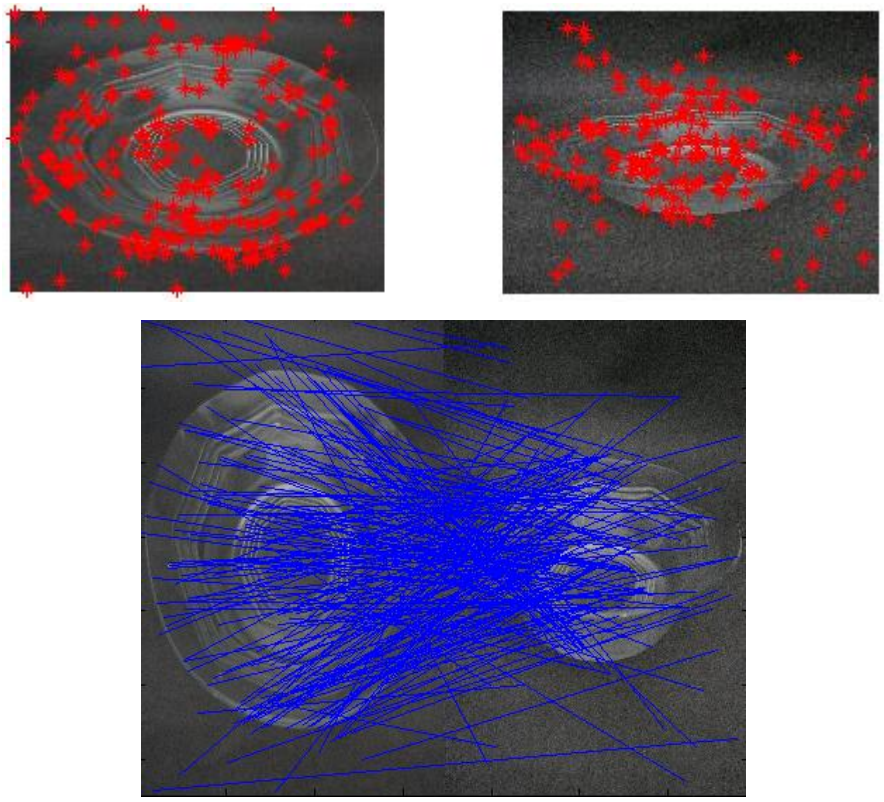

Figure 4.58: Plots of matching keypoints (top) and successful matches (bottom) between *plate_black1*and *plate_black3*

Table 4.46: Details of results obtained by comparing *plate_black1* and *plate_black3*

	plate_black1	***plate_black3***
Number of keypoints detected	7857	4715
Number of successful matches	220	
Time taken (s)	22.909	

3. SIFT detection on *plate_black1* and *plate_black4*

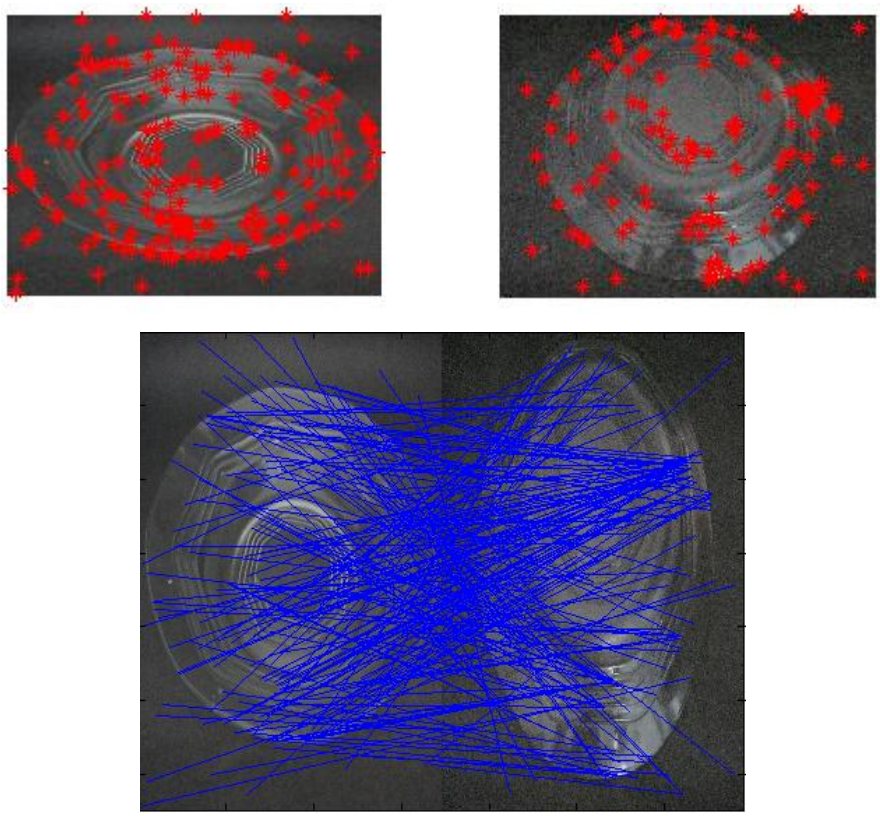

Figure 4.59: Plots of matching keypoints (top) and successful matches (bottom) between *plate_black1*and *plate_black4*

Table 4.47: Details of results obtained by comparing *plate_black1* and *plate_black4*

	plate_black1	***plate_black4***
Number of keypoints detected	7857	5168
Number of successful matches	217	
Time taken (s)	24.357	

4. SIFT detection on *plate_black1* and *plate_black5*

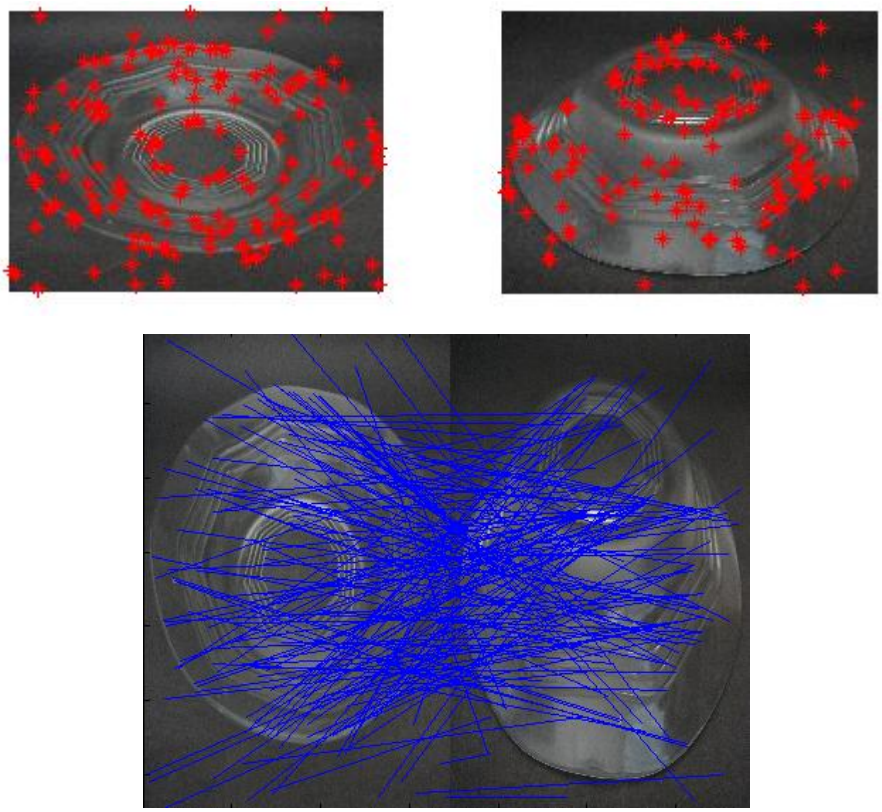

Figure 4.60: Plots of matching keypoints (top) and successful matches (bottom) between *plate_black1*and *plate_black5*

Table 4.48: Details of results obtained by comparing *plate_black1* and *plate_black5*

	plate_black1	***plate_black5***
Number of keypoints detected	7857	7050
Number of successful matches	173	
Time taken (s)	31.759	

4.6 Experiment Analysis and Discussion

Generally, every experiment involves images of an object taken from five different views and angles. This is to prove that the SIFT algorithm implemented in this project is able to detect and match identical features of the object in a series of images captured. Based on the results obtained from each image dataset, it can be said that the SIFT algorithm implemented has the characteristic of rotation, position and scale invariance. This is because images captured in each set of dataset are different in terms of position, rotation and scale.

4.6.1 Experiment 1(smooth-surfaced objects)

For this experiment, a patterned rectangular-shaped container and a non-patterned black-coloured rectangular-shaped mobile phone were used.

Both objects tested are smooth-surfaced rectangular object. However it is noticed that for container, number of features detected averages about 4000 with number of successful matches averages about 300. The matching keypoints in images of container mostly concentrate in printed wordings on the surface.

Whereas for mobile phone, the plots in red colour have clearly shown that the matching keypoints of mobile phone mostly concentrate in the phone edges and the wording ‘NOKIA’. Not many plots can be seen on the middle of the screen of the phone. Furthermore, the number of features detected averages only about 3500 with the number of successful matches averages about 200.

The reason that plots are concentrated on the pattern of the container rather than edges is because if compared to mobile phone, the edges of container are more rounded. Sharper edges will act as more distinctive locations whereby the chances of getting keypoints extracted from there are higher. And this explains as well why wordings on the container have so many extracted keypoints. Wordings have many edges and are distinctive.

Hence, based on the results obtained in Experiment 1, it can be said that SIFT algorithm is suitable to be applied on smooth object to detect and extract keypoints as long as the edges of the object are less rounded and have patterns on its surface. This is to ensure that more useful keypoints can be extracted.

4.6.2 Experiment 2 (heavily-textured objects)

For this experiment, a colourful and spirally-shaped pencil holder and a black colour Digital SLR (professional camera) were used. Both of these objects are considered as textured objects.

At first sight, it was thought that the pencil holder will have more successful matches since it has many edges and is colourful. However the results obtained shows that with average more than 4000 keypoints detected for both pencil holder and camera, the number of successful matches for camera exceeds 200 whereas for pencil holder, number of matches is less than 200.

Furthermore, the matching keypoints of the images of camera are plotted more consistently. It can be seen that approximate same areas of the camera are plotted in each set of images. This may be due to the consistency of angle changing while capturing the images. On the other hand, the plots in the set of pencil holder images are not so consistent. Some of the matches shown for pencil holder are very inaccurate.

Hence, for heavily-textured object, if want to obtain optimum results from SIFT implementation, it is advisable to choose objects that have different types of edges like the camera used in this experiment. Besides that, if possible do not choose objects that are too colourful.

4.6.3 Experiment 3 (white-coloured objects)

In this experiment, a plain white vase and a plain white cup were used. The vase has a textured surface whereas for the cup, its surface is smooth.

Besides that, this experiment was carried out in two different backgrounds, white and black. Since the objects used are white, therefore white background was used to test whether SIFT feature extractions on pure white objects are suitable to be carried out in white background. As for the black background, it represents coloured backgrounds. Position of objects and angle of images captured in both backgrounds were structured to be similar for the same object used. For example, the series of vase angles and positions captured in white background are similar to the series of vase angles and positions captured in black background.

Based on the results obtained, it seems that for white background, number of detected keypoints for both the vase and the cup averages about 2700, with the number of matches ranging from 110 to 160. The computation time is between 12 to 13 seconds. Whereas for black background, the number of detected keypoints for both objects averages about 4000 with the number of matches ranging from 130 to 180. The computation time ranges from 16 to 19 seconds. Though computation time for white object in black background may be longer, however time taken to complete the matching process is not a main concern in this project.

It is evident that if white objects were to be placed in white background for SIFT detection, the detection rate will not be as high as the detection rate in black background. Furthermore, matches in figures of white object in white background do not look accurate if compared to sets of images in black background. For example in Figure 4.35, it is obvious that most of the matches are wrong. Besides that, if Figure 4.25 is to be compared with Figure 4.30 (similar position and angle but with different backgrounds), it can be seen that matches in Figure 4.25 are not as good as the matches shown in Figure 4.30.

However, if only datasets of white objects in white background were to be compared, it is clearly shown that textured object like vase shows better matches.

Therefore, it can be said that white objects are actually not really suitable to be used in SIFT detection especially when the background is white. However, if features of white object are to be detected by SIFT algorithm, then the surface of the object should at least be textured moderately to obtain reasonable results.

4.6.4 Experiment 4 (transparent objects)

In this experiment, a heavily-textured transparent glass and a mildly-textured transparent plate were used.

Like Experiment 3, this experiment was also carried in two different backgrounds, white and black. Basically, white background represents plain and light-coloured background. On the other hand, black background represents dark-coloured background. These two types of background were used to determine which type of background is suitable for transparent objects while implementing SIFT feature extraction.

From the results stated in Experiment 4, it is shown that the glass (heavily-textured) image dataset in white background only has an average of 3500 keypoints detected with an average of 170 matches. For the glass in black background, results shows that the number of detected keypoints can reach up to nearly 7600, though the range is from 5000 to 7500 keypoints. As for the number of matches, it can reach up to an average of 260 matches. This shows that for glass image datasets, the outcome of carrying SIFT algorithm out in black background is more desirable. Furthermore, by comparing the matches of each dataset of the glass (similar position and angle but different background), it can be seen that matches obtained in black background are more consistent and accurate, for example Figure 4.43 and Figure 4.48.

However, for the plate image datasets, the results for white and black background are a bit weird. The number of keypoints detected in black background is slightly higher than the number of keypoints detected in white background. But, the number of matches for the plate in black background is slightly lower than the

number of matches obtained in white background. This is a very unusual sight. However, in terms of matches, the results can be considered acceptable. This is because the pattern on the surface of plate is a repetitive pattern. Hence when a match is joined between two identical keypoints on the pattern, the match can never go wrong.

Besides that, based on the plots of matching keypoints for the glass and plate image datasets, most points are concentrated in the pattern and texture of the object. Therefore if SIFT feature extraction is to be implemented on transparent objects, it is better to choose transparent objects with at least a mildly textured surface in order to obtain more desirable results. An object like the Coca-Cola glass will be a very good test object especially when tested in dark backgrounds.

4.7 Problems Encountered with Obtained SIFT Keypoints and 3D Object Reconstruction and Solution Carried Out

For this project, although focus is placed in determining suitable type of object for SIFT implementation, there is still another concern which is to ensure that the obtained SIFT keypoints are usable in 3D object reconstruction part.

Basically the problem faced when combining both parts together is that my partner who is responsible for the the 3D object reconstruction part, could not output a desirable 3D object with the saved SIFT keypoints. Several types of images were tried out, be it smooth, textured and colourful objects. However, still the same problem persisted.

After other attempts of trial and error, it was found out that keypoints given to my partner were not sufficient enough and angle variations actually do affect the output of 3D object reconstruction part. Hence to solve the problem, the following steps were carried out:

1. Choose an object that can output many matching keypoints after implementing SIFT algorithm.

2. Change the threshold value to default, which is 1.5. This is to ensure that maximum number of points extracted can be obtained from each image. Initially, the threshold value was set a bit higher than 1.5.
3. Series of images of object were captured, each with slight angle variation. This step is also to ensure that more matching points can be obtained.
4. Capture at least 8 images for each object. The more images captured the better results can be obtained.

After carrying out the steps stated above, my partner could at least construct something more satisfying than before. At least patterns on the surfaces of the object of interest were more visible.

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

In conclusion, the three main objectives of this project have been achieved. The SIFT algorithm developed is able to find corresponding matching keypoints of the object of interest. It is capable of finding points that have the same characteristics and identical positions in the images captured in different views and angles. The implementation of the algorithm has shown reasonable consistency. In short, it is invariant to scale, position and rotation changes.

Besides that, suitable type of object for SIFT implementation has been determined. To obtain optimum results, it is best to choose objects that are heavily textured yet not too colourful. Smooth-surfaced objects are suitable as long as there are patterns on the surface and edges are not too rounded. White-coloured objects are not recommended especially is the background is white colour. If needed to use white-coloured objects, background should be dark colour and the object should be at least mildly textured. Transparent objects should also be textured in order to obtain optimum results. By getting optimum results, 3D object reconstruction part can be carried out easily.

Furthermore, problems regarding saved SIFT keypoints and 3D object reconstruction were partially solved. Ms. Leow Tzyy Shyuan is able to output something partially similar to the object of interest with the saved SIFT keypoints.

5.2 Contributions

To develop a workable SIFT algorithm, effort has been put in finding relevant sources about SIFT and its respective coding methods in MATLAB form. After understanding what SIFT is all about and how MATLAB works, tutorials available on websites were being tried out before starting developing the complete program. While undergoing this process, many problems were encountered initially as knowledge in MATLAB programming was very minimal. However by using trial and error method, most of the problems could successfully be solved and understood.

To determine which type of object is most suitable for SIFT implementation as well as 3D object reconstruction, four main experiments mentioned in Chapter 4 were conducted. In the middle of this process, many other types of objects have been tried out but for report purpose, only two image datasets were included for each subpart of experiment.

Combining the SIFT keypoints obtained with the developed 3D object reconstruction algorithm, several problems were encountered. Once again, trial and error method was implemented to solve the problem. Although it took quite some time to figure out what went wrong, but in the end at least part of the problem could be settled. However more future works should be done to improve the detection rate in order to obtain more number and accurate keypoints for 3D object reconstruction part.

5.3 Future Works

Since 3D object reconstruction relies a lot on the SIFT keypoints obtained, hence the more number and accurate of matched keypoints, the better the results will be, especially when combine with 3D object reconstruction part. Therefore it is recommended that in future, SIFT algorithm can be implemented with other feature extraction operators such as Harris Corner Detector. Maybe other algorithms such as SURF (Speeded UP Robust Features) which is a variation of SIFT algorithm can be

implemented to test the outcome difference when combining with 3D object reconstruction part. This can further enhance the consistency and accuracy of the keypoints detected.

REFERENCES

Azad, P., Asfour, T., & Dillmann, R. (2009, 10-15 Oct. 2009). *Combining Harris interest points and the SIFT descriptor for fast scale-invariant object recognition.* Paper presented at the Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on.

Cheng, W., & Sato, J. (2008, 8-11 Dec. 2008). *Computing multiple view geometry in space-time from mutual projections of multiple cameras.* Paper presented at the Pattern Recognition, 2008. ICPR 2008. 19th International Conference on.

Keju, P., Xin, C., Dongxiang, Z., & Yunhui, L. (2009). *3D reconstruction based on SIFT and Harris feature points.* Paper presented at the Robotics and Biomimetics (ROBIO), 2009 IEEE International Conference on.

Kozuka, K., & Jun, S. (2007, 10-14 Sept. 2007). *Rectification of 3D Data Obtained from Moving Range Sensors by using Multiple View Geometry*. Paper presented at the Image Analysis and Processing, 2007. ICIAP 2007. 14th International Conference on.

Mark, F., & Chang, S. (2006). *3D Model Creation Using Self-Identifying Markers and SIFT Keypoints.* Paper presented at the Haptic Audio Visual Environments and their Applications, 2006. HAVE 2006. IEEE International Workshop on.

Tomono, M. (2008). *3D object modeling and segmentation using edge points with SIFT descriptors.* Paper presented at the Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on.

Vural, E., & Alatan, A. A. (2008). *Outlier Removal for Sparse 3D Reconstruction from Video.* Paper presented at the 3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video, 2008.

APPENDICES

APPENDIX A: Coding for SIFT Algorithm Developed

```
%% Clear off any functions that were done before.

clc;

%% Run VL Feat 0.9.9 before running the program.

run('C:\Program Files\vlfeat-0.9.9\toolbox\vl_setup.m');


%% Load the base image of a particular image dataset.

% Base image is camera1.jpg, which is stored in the 'test images'
folder in vl-feat.
pfx = fullfile(vl_root,'sherril','phone1.jpg') ;
figure;
img1 = imread(pfx);
image(img1)

%% Continue on the rest of the steps in a loop.


for(i=1:1)

    %show the current process going on on the screen of Matlab
    fprintf('\n\n Extracting keypopints from Image 1 and
Image %d\n\n', i+1);

    %load the next image to be compared

pic=fullfile(strcat(vl_root,'\sherril\phone',int2str(i+1),'.jpg'));
    %figure;
    eval('img=imread(pic);');
    image(img)

    %start counting from here (computation time)
    tic;

    %1. extract keypoints and descriptors from each picture
    %2. f=frame
```

```
    %3. d=descriptor
    %4. rgb2gray = convert RGB image to grayscale image
    %5. im2single = converts intensity image I to single
    [f1, d1] = vl_sift(im2single(rgb2gray(img1))) ;
    [f{i+1},d{i+1}] = vl_sift(im2single(rgb2gray(img)));


    %match the extracted descriptors, where vl_ubcmatch finds the
closest
    %descriptor in d{i+1} (as measured by the L2 norm of the
difference between them)
    [matches, scores] = vl_ubcmatch(d1, d{i+1}, 5.5);

    %change image to unsigned 8-bit integer
    img1=uint8(img1);
    img=uint8(img);

    fprintf('\n\n Figure shows extracted keypoints. \n\n');
    figure;

    %output subplots of matching keypoints between the 2 images
    subplot(1,2,1);
    imshow(img1);
    hold on;
    plot(f1(1,matches(1,:)),f1(2,matches(1,:)),'r*');


    subplot(1,2,2);
    imshow(img);
    hold on;
    plot(f{i+1}(1,matches(2,:)),f{i+1}(2,matches(2,:)),'r*');


    [drop, perm] = sort(scores, 'descend') ;
    %index of original match and closest distance stored in each
column of
    %matches
    matches = matches(:, perm) ;
    %distance between pair is stored in scores
    scores  = scores(perm) ;

    %display the 2 images side by side
    figure ; clf;
    imagesc(cat(2, img1, img)) ;

    %find and save the matching keypoints
    x1 = f1(1,matches(1,:)) ;
    x{i+1} = f{i+1}(1,matches(2,:)) + size(img1,2) ;
    y1 = f1(2,matches(1,:)) ;
    y{i+1} = f{i+1}(2,matches(2,:)) ;

    hold on ;

    %set the linewidth and colour of the matching line
    h = line([x1 ; x{i+1}], [y1 ; y{i+1}]) ;
    set(h,'linewidth', 1, 'color', 'b') ;

    %save the number of matches
```

```
    numberOfMatches = size(matches,2);

    %save number of keypoints detected in base image
    base_keypoints = size(d1,2);

    %save number of keypoints detected in the compared image
    compared_keypoints = size(d{i+1},2);


    %display number of matches, # of keypoints detected in base and
    %comapred images
    fprintf('Matches found:%d\n',numberOfMatches);
    fprintf('Keypoints detected (base image):%d\n', base_keypoints);
    fprintf('Keypoints detected (compared image):%d\n',
compared_keypoints);

    %display total time used to compute matches
    fprintf('Total time used = %.3f s\n', toc);
end
```