Senior Project Report

Dimension Reduction of the SIFT descriptor and its Object Recognition Application

Danfei Xu

Dickinson College

Department of Computer Science and Mathematics

May 14th, 2013

Table of Contents

Abstract

One of the most challenging tasks in visual object recognition is to reduce the amount of computation in feature matching. Scale Invariant Feature Transformation (SIFT) is a strong algorithm in terms of extracting accurate feature points and descriptors. However, each SIFT descriptors contains a 128-dimensional vector, which causes heavy computational burden when comparing two or more SIFT descriptors. There have been extensive works on solving the curse of dimensionality. In this study, we apply a dimension reduction technique named t-Distributed Stochastic Neighborhood Embedding (t-SNE) to generate lower dimension representations of SIFT descriptors. We evaluated the lower dimension descriptors in a matching application to examine if they maintain the power of distinction that belong to the original high dimensional descriptors.

Introduction

Visual perception is the most important and informative way for human beings to obtain information from the surrounding environments. We use visual perceptions to recognize objects and faces and then convert this perceptual information to semantic information for further high-level processing. The interest of using computer to simulate the human perception ability induces extensive study on computer vision, specifically, object and facial recognition.

In order to endow computer program with the ability of recognize an object, essential and quantifiable visual features of the object needed to be extracted from the visual representations of the object to uniquely characterize an object. Two of the most well-developed 2-dimensional feature extraction algorithms are SIFT (Scale Invariant Feature Transform) and SURF (Speeded Up Robust Features). The concept of SIFT was first introduced by David G. Lowe. The algorithm first detects visual features that are invariant of image scale change and illumination change and other noises. According to Lowe's study, these feature points are often located on the high-contrast region, such as object corners and edges (Lowe, 1999). Then the algorithm uses neighboring pixels to describe the features detected. SURF, partly inspired by the idea of scale-invariant feature was first published by Herbert Bay in 2006. The main contribution of SURF is that it outperforms SIFT in terms of efficiency and robustness (Bay, 2006).

When SIFT was first introduced, the corresponding feature matching algorithm is Best-Bin-First (BBF). It can index the nearest neighbors of feature points in a high-dimensional feature space by their probabilities, and the nearest neighbor is defined as the feature points that have minimum Euclidean distance from the given descriptor vector (Lowe, 1999). Machine learning is another approach to feature matching. In Lior Elazary and Laurent Itti's study, the feature detector on scene image is guided by the prior knowledgebase constructed from computing Bayesian probabilities on visual features of the objects used for training. According to the paper, this approach is more efficient in indexing the plausible candidate lists and faster in converging towards an individual target than other traditional approaches such as HMAX and SIFTS (Elazary, L, and L Itti, 2011). However, because the grounding structure of SIFT feature matching algorithm is computing Euclidean distance between descriptors, and SIFT feature descriptor is composed of 128-dimensional vectors, searching for nearest neighbor in high dimensional space would cause considerable amount of computational burden for real-time application.

The main purpose of our study is to tackle the problem of curse of dimensionality occurs in SIFT feature matching. This study consists of two stages: In the first stage, we studied SIFT and SURF feature extraction algorithms and the properties of their feature descriptors. Based on the properties of SIFT feature descriptor, we designed a program to compute the "average visual representation" of each training objects. However, we discovered a few defects of this approach, which were the potential cost of in unsatisfying results of object classification. In the second stage, we improved the algorithm by introducing the SIFT feature ranking system, which is an algorithm that quantifies the power of distinguishing of individual SIFT features. Then we used a dimension reduction technique named t-SNE (t-Distributed Stochastic Neighborhood Embedding) to reduce the dimension of SIFT feature descriptors. Finally we created a simple object classifier program based on the feature ranking system and dimension reduction. In recognizing objects among four objects selected from Columbia Object Image Library (COIL), we achieved 100% percent of accuracy (4 trials in total). However, its effectiveness is still subject to further experiment. Due to the particular nature of this report, the descriptions and results of both stages are incorporated. At the end of the document, an Appendix is added to show the original software requirement proposed at beginning of the project.

Stage (I)

Overview:

In this stage, we explored the basic properties of SIFT feature extraction strategy, and introduced an approach of computing “average represenatation” of an object, which can potentially be used in calculating the similarity between two object. However, experiments resulted in unsatisfying results. Possible causes is listed at the end of the this section.

(1.1) Brief introduction to OpenCV

This section gives a brief introduction to OpenCV, the computer vision library we used in our study, and the SIFT feature extractor functions provided by OpenCV.

OpenCV

OpenCV is a library of functions for real time computer vision (Bradski, 2008). It is free for both commercial and academic use, and come with C, C++, Python, and Java version running on Linux, Windows, Android, and Mac. The library has over 2500 optimized computer vision algorithm and is maintained by multiple groups worldwide. In this project, we used implementation of SIFT feature extraction algorithm, image displayer, and function that draws keypoints on images provided by OpenCV library (OpenCV Documentation, 2012).

SIFT in OpenCV is included in OpenCV under Feature2D module. It is implemented according to Lowe’s work. Two functions of this implementation are used in our study:

- SIFT::SiftFeatureDetector.detect()
- SIFT::SiftFeatureDetector.compute()

detect() takes an image in cv::Mat data structure and produces a list of cv::Keypoints object, which is the detected location of keypoints (represented as two-dimensional vector). compute() takes the image and keypoints as parameter, and produces a list of keypoints descriptors in cv::Mat datastructure. Each keypoint descriptor is a 128-dimensional vector. Thus in this report, the SIFT keypoint descriptor is specifically referred as such vector.

(1.2) A simple SIFT feature matching Experiment using brute force approach

Purpose:

In this section, we will introduce a brute feature matching strategy for SIFT features. This feature matching strategy is the grounding structure of the feature matching algorithms we used in this study. Throughout this report, "feature matching" is specifically referring to this strategy.

Method:

Two sample images were obtained from OpenCV library package. The first image represents an object, containing no background noise. The second image contains multiple objects and scene including the object in the first image. SIFT feature descriptors were extracted from both images, and produced a set of SIFT features S_1 for object image and S_2 for scene image. Descriptors were matched using nearest neighbor algorithm. The final output of the program is a set M containing paired keypoints. M is defined as:

$$M = \{(p_i, p_j) | \, p_i \in S_1, p_j \in S_2, and \, p_{j1} \in S_2 \; and \; j1! = j, \}$$

, such that

$$argmin\left(distance\,(p_i, p_j)\right) < argmin(distance(p_i, p_{j1})) * k$$

In English, the algorithm output is defined as: For each keypoint p_i in S_1, the algorithm will find a keypoint p_j in S_2 such that the Euclidean distance between p_i and p_j is minimum among all combination of p_i and keypoints in S_2. The algorithm will also find a keypoint p_{j1} in S_2 such that the Euclidean distance between p_i and p_{j1} is the minimum among all combination of p_i and keypoints in S_2 except for p_j. The pair (p_i, p_j) will be in M if and only if the distance between p_i and p_j is smaller than the distance between p_i and p_{j1} multiplied by some constant factor k.

10 trials of feature matching were performed with k values ranging from 0.1 to 1. Then we manually counted and recorded resulting matched keypoints (The keypoints locating on the target object).

Results:

Object to be recognized:

Object recognition result (k = 0.6):

Object recognition result (k = 0.8):

Object recognition result (k = 0.4):

Figure 1.2.1: Scene image with SIFT feature points matched using different k value.

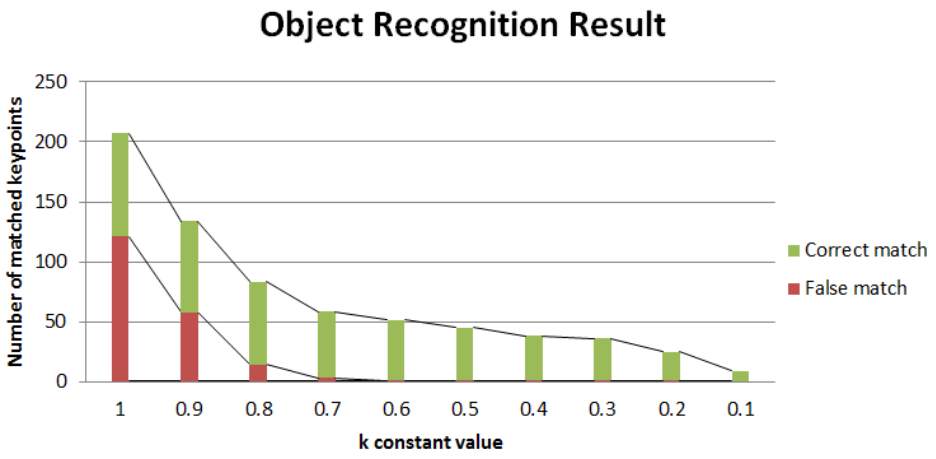


Figure 1.2.2 Number of matched keypoints with different k values.

Discussion:

By observing figure 1.2.2, we find the maximum k value to be approximately 0.7, because with k = 0.7, the ratio between the number of correctly matched keypoints and number of all matched keypoints is maximized. However, the result is limited to this particular pair of images. Hence, the choice of k is still subject to further experiment when this feature matching strategy is used on other images.

(1.3) Searching for "Strong" SIFT Keypoints

Purpose:

Multiple pictures of one object are taken from different angles to represent all possible views of the object. In order to further analyze the distribution of the feature points in these images, we need to find the feature points that are robust against change of views. In other words, we want to find the SIFT keypoints that appear in multiples views of the object. We call such keypoints strong SIFT keypoints.

Method:

We use three different objects in our experiment: A silver-colored battery, a yellow-colored battery, and a green-colored glue stick. The two batteries have the exact same contour but different patterns and colors. 12 pictures are taken for each of the three objects to represent different views.

Yellow battery Silver Battery Green gluestick

SIFT feature keypoints and descriptors are extracted from these images using OpenCV's SIFT feature detector. Then for each object, an image is chosen to be the normal view, whose SIFT keypoints will be compared against the SIFT feature points of other views of this object. For each keypoints in the normal view, we want to find a similar keypoint in each of other views. To find all the keypoints that is similar to the j^{th} keypoint in the normal view, SIFT feature matching algorithm was ran between j^{th} keypoints and all keypoints found in images other than the normal view (with different *k* values). A keypoint in the normal view is considered to be a strong keypoint if the ratio between the number of similar keypoints found in other views and the total number of views is greater than a constant r.

To observe the influence of r and k values on the number of strong SIFT keypoints, we ran the program that finds the number of strong SIFT keypoints with parameter k ranging from 0.1 to 8.0 with a increment step 0.1, and parameter r ranging from 0.1 to 1 with a increment step 0.1. The same experiment is conducted on each of the three objects.

Results:

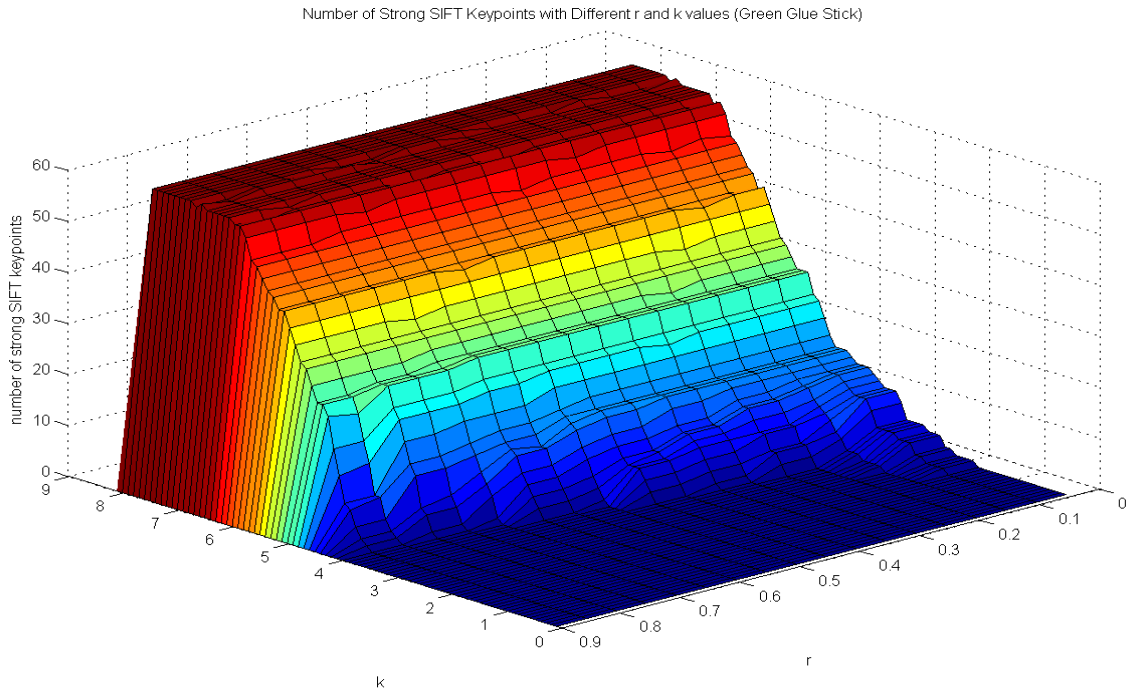


Figure 1.3.1 Number of strong SIFT keypoints found with different r and k values in the images of green-colored glue stick.

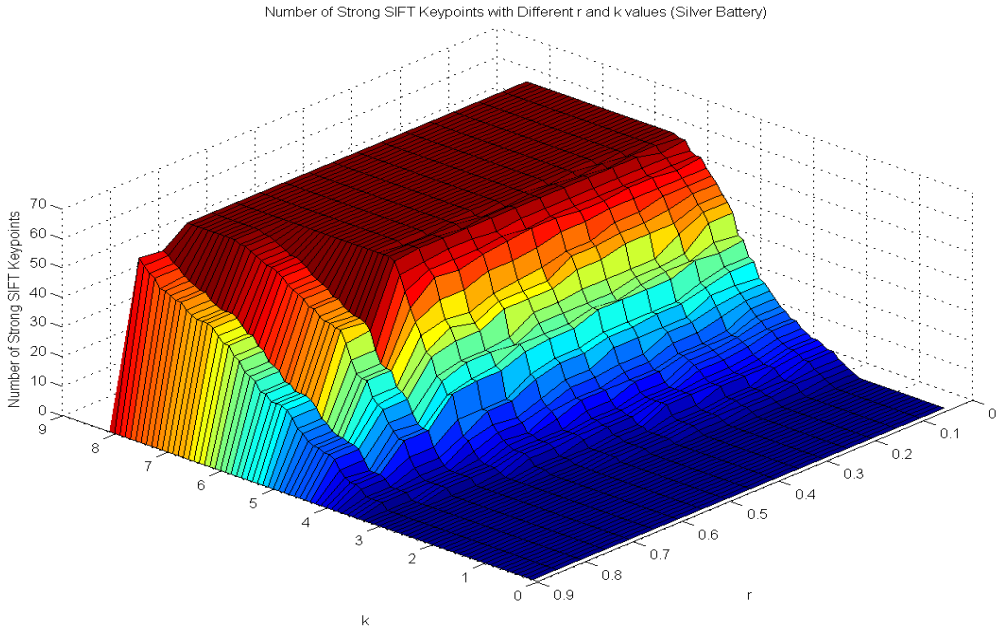


Figure 1.3.2 Number of strong SIFT keypoints found with different r and k values in the images of silver-colored battery.

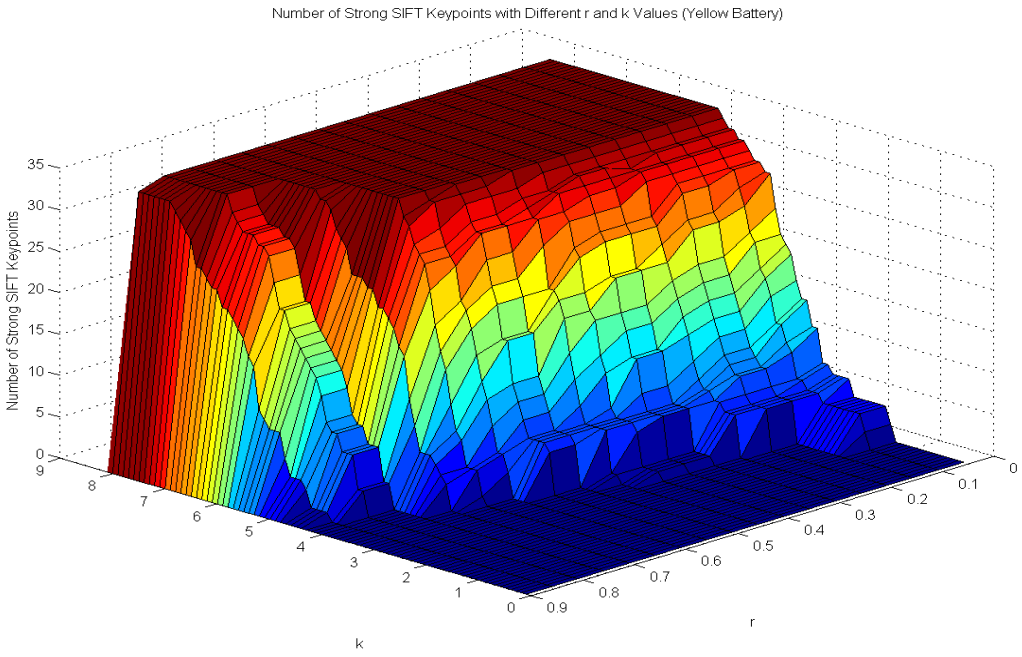


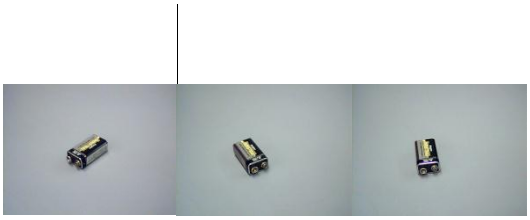
Figure 1.3.3 Number of strong SIFT keypoints found with different r and k values in the images of yellow-colored battery.

Discussion:

By observing Figure 1.3.1, Figure 1.3.2, and Figure 1.3.3, we found that the number of strong SIFT keypoints decreases drastically as the k value, which represents the minimum quality of the strong SIFT keypoints, reduce from 7 to 3.

r value, which represents the minimum generality of the Strong SIFT keypoints, only has very small effects on the number of strong SIFT keypoints when increasing from 0.0 to 0.7. However, in figure 1.3.2 and 1.3.3, we found that at k = 6, there is a sharp decrease in number of strong SIFT keypoints as the r value increase from 0.7 to 1.0 , which may indicate that certain views of battery does not have the features that appears in most of the views of the battery. For example, from the pictures presented below, we can see that view 10 does not have the metal convex part that appears in other views (view 1, 2, 3, and other views that are not presented).

View 10



View 1 View 2 View 3

From this observation, we can conclude that to find the SIFT features that are robust against variation of views, the program needs to use highest possible r value while keeping adequate amount of strong SIFT keypoints produced.

(1.4) Compare Average Models

Purpose:

With the strong SIFT keypoints that obtained in the previous experiment, we can compute an average model for each of the three objects. By visual observation and exercising human intuition, we can safely claim that the similarity between the two batteries is higher than the similarity between a battery and the glue stick. Since short Euclidean distance means similar SIFT descriptor, it is hypothesized that if we perform feature matching between the average model of the two batteries and between a battery and the glue stick, the average Euclidean distance between the matched keypoints of two batteries would be significantly shorter than the average Euclidean distance between the matched keypoints of a battery and the glue stick.

Method:

In this experiment, we used the strong SIFT keypoints obtained in the last experiment. The average model of an object consists a set of averaged strong SIFT descriptors. An average strong SIFT descriptor was computed by summing up its similar descriptors found in different views using vector summation operation and dividing the sum by the number of views that the similar descriptors appears. This calculation can be expressed as:

$$\mathrm{D}_{\mathrm{x,y}} = \frac{1}{n}\sum_{i=1}^{n} SD_i$$

, where n is the number of keypoints similar to the y^{th} descriptor of x^{th} image found in other views, and SD_i being the i^{th} keypoint that is similar to the subject keypoint. Both SD and $D_{x,y}$ are 128-dimension vectors.

After computing the average models for all three objects, feature matching by calculating minimum Euclidean distance was performed between two batteries and between silver battery and glue stick. The setup of feature matching algorithm is similar to the previous experiment. Different r and k values were used in the experiment.

Results:

k	r	B&B(average distance)	B&G(average distance)	Error Ratio	Difference
4	0.7	1708	2438	0.299426	730
5	0.7	1601	1843	0.131308	242
6	0.7	1480	1676	0.116945	196
7	0.7	1470	1739	0.154687	269
5	0.8	1634	1816	0.10022	182
6	0.8	1443	1651	0.125984	208
7	0.8	1463	1737	0.157743	274
8	0.8	1469	1729	0.150376	260
6	0.9	1532	1647	0.069824	115
7	0.9	1340	1719	0.220477	379
8	0.9	1431	1753	0.183685	322

Table 1.4.1 Results of comparing the average models of two battery and average models of silver battery and glue stick.

Discussion:

By observing table 1.4.1, we can easily find a consistent and relative significant difference (about 20%) between the average distance between the average model of two batteries and the average distance between the average model of the silver battery and the glue stick, which confirms our hypothesis. However, we cannot conclude that this is an effect way of comparing the object represented by SIFT features due to limited amount of training objects.

(1.5) Compute Probability Density Function and Confusion Probabilities

Purpose:

After the successful experiment of comparing average models, an experiment was then conducted to test if the average model can be used to compute the confusion probabilities (Bhattacharyya Distance) among a set of objects using the equation introduced in the last update.

Method:

The Bhattacharyya Distance is used to quantify the similarity between two discrete probability distributions (Bhattacharyya, 1943). Greater value of Bhattacharyya Distance indicates small overlap between two distributions, or in our case, less confusion between measurements of two objects.

$$C_{ij,m} = \int \sqrt{p(x|O_i, M_m)p\left(x\middle|O_j, M_m\right)}\, dx$$

Assuming the normal distribution of SIFT feature descriptors within a set of training object, the confusion probability between the probability density function of object i and of object j can be expressed as:

$$C_{ij} = \sqrt{\frac{2\sigma_i\sigma_j}{\sigma_i^2 + \sigma_j^2}}\; e^{-\frac{(\mu_i-\mu_j)^2}{4\left(\sigma_i^2+\sigma_j^2\right)}}$$

C is a number range in 0 and 1. σ_i can be seen as the variance of the Euclidean distance of SIFT feature to its "average model," thus can be computed by the equation

$$\sigma_i = \sqrt{\frac{1}{m}\sum_{k=0}^{m} EuclideanDistance\left(d_{i,k}, d_{i_0,k}\right)^2}$$

, where m is the total number of descriptors of object i, $d_{i,k}$ is the descriptor of kth strong SIFT keypoint of object i, and $d_{i_0,k}$ is the kth descriptor of average model of object i. $\mu_i - \mu_j$ can be expressed as the Euclidean distance between the average model of object i and object j.

In the experiment, confusion probability is computed between two batteries and between silver battery and glue stick. It is hypothesized that the confusion probability is higher for two batteries than for silver battery and glue stick.

Results:

Confusion probability between		Confusion probability
silver battery	yellow battery	0.945521
silver Battery	green glue stick	0.918075
green glue stick	yellow battery	0.943517

Table 1.5.1: results of computing confusion probabilities between two pairs of objects.

Discussion:

The results (Table 1.5.1) suggest that the confusion probabilities between a pair of visually similar objects are not sufficiently greater than the confusion probabilities between a pair of visually distinctive objects.

After further experiments and analysis, we have found some possible causes of the unsatisfying results:

1. The arithmetic mean of a set of descriptors might have significantly less power of distinction than individual descriptors because the power of distinction of a SIFT descriptor is essentially endowed by the unique combination of 128 float numbers.
2. In computing “strong” SIFT keypoints, a training image was arbitrarily chosen to be the reference image, and all the “strong” SIFT keypoints were selected from the keypoints of the reference image. Consequently, the “strong” SIFT keypoints existed in other images were highly likely to be ignored.
3. In the method, there are two constant values that can be adjusted to control the quality of the “strong” SIFT keypoints: one is the ratio between the minimum Euclidean distance and the second minimum Euclidean distance when matching two descriptors,

and the other is the ratio between the number of similar keypoints found in other images and the total number of images. In the experiment, adjusting these two numbers yielded drastic change of confusion probabilities, and we have no way to determine the optimized combination of these two factors.

Stage (2)

Overview

In Stage (1), we explored the approach of computing "average representations" of objects in order to quantify the similarity between objects pairwise. However, the experiments yielded unsatisfying results and we have found some potential causes of the problem. In Stage (2), we took another approach of computing strong SIFT keypoints, and we mainly focused on solving the problem of reducing dimensionality of SIFT descriptors. We devised a method of computing strong SIFT keypoints using SIFT feature ranking system, and reduced the dimensionality of SIFT feature descriptors using a dimension reduction technique called t-SNE. At the end of this section, we presented the results of recognizing object using an object classifier, which is designed basing on dimension reduction technique and SIFT feature ranking system.

(2.1) SIFT keypoints ranking system

Purpose:

To cope with the defects discovered in Stage (I), we devised a new approach to find strong SIFT keypoints based on multiple training images. The main strategy is to assigning a score to each individual SIFT keypoints. Two factors would affect the score of a keypoint: the Euclidean distances between the descriptor to the similar descriptors in other training images, and the number of similar SIFT keypoints found in total. Higher score yields greater power of distinguishing.

Method:

An experiment was conducted to test the effectiveness of this keypoints scoring system.

The experiment uses total of 26 different views of two similar batteries as training images. Score is assigned to each (qualified) keypoints detected by the OpenCV SIFT detector. The score is computed as:

$$Score_{i,j} = \frac{\left(\#OfSimilarKeypoints_{i,j}\right)^2}{\sum_{k=0}^{n} Euclidean\_Distance(d_{i,j}, d_{k,x})}$$

, where i is the index of the training image, j is the index of the descriptor, n is the total number of training images, x is determined by the descriptor matching algorithm. Thus this equation shows that the score of jth keypoint in ith image is equal to the square of number of similar

keypoints found in other images divided by the sum of Euclidean distance from the jth keypoint in ith image to its similar keypoints.

Then all the scored keypoints are added into a list and sorted based on their score (highest scores first). Then the keypoints that are among the top 20% are drawn on the original image (in greyscale).

Results:

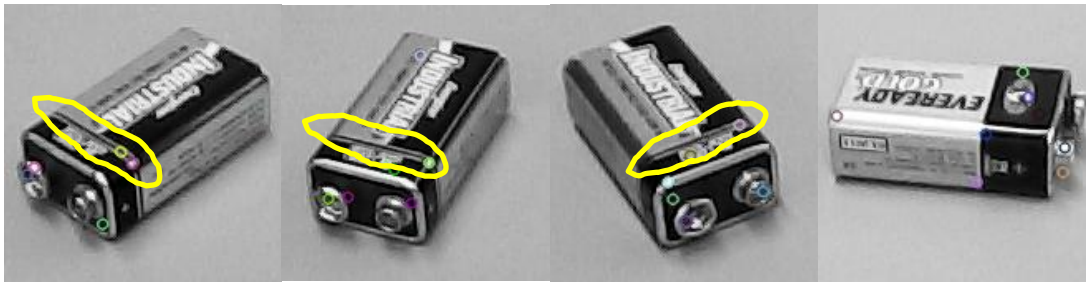


Figure 2.1.1: Some views in which battery terminals are visible. The yellow circle indicates a unique feature which is a line.

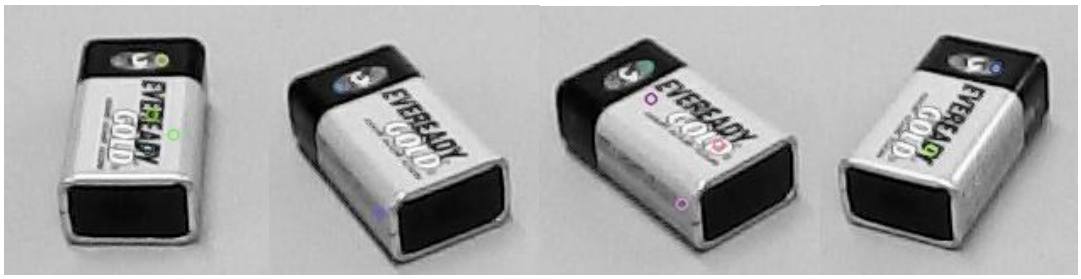


Figure 2.1.2: Some views in which battery terminals are not visible

Discussion:

From figure 2.1.1 we can see that in the views in which battery terminals are visible, a significant portion of the highly ranked keypoints is located on the terminals, which means terminal is a very distinctive feature of this battery. There are also some keypoints located on the stripe in the area circled with yellow, which suggests that the stripe is also a unique feature of this battery.

From figure 2.1.2, we can see that in the views in which battery terminals are not visible, there are less highly ranked points present than the views in figure 2.1.1. We can tell the most

distinctive feature of the battery present in this view is the oval shape located on the black part of the body because in each view, there is at least one highly ranked feature point located on the oval.

Advantage of this approach over the "average model" approach introduced in Stage (I):

1. No arithmetic computation is performed directly to the components of descriptors. Thus the power of distinction can be preserved in average models.
2. No arbitrary reference images are used. Instead, all the keypoints are evaluated individually with the exact same process.
3. Only one factor is used, which is the ratio between the minimum Euclidean distance and the second minimum Euclidean distance when performing descriptor matching. This factor is used to set a minimum quality threshold for keypoints.
4. Instead of manually determined "strong" SIFT keypoints, each individual keypoints has its own score and its influence over the average model is determined by the score.

(2.2) Reduce dimensionality of SIFT feature descriptors using t-SNE

Purpose:

The keypoint descriptor of a SIFT feature point is computed by dividing the surrounding region of the keypoint to 4x4 boxes, and assigning an 8-bins histogram to each box. Based on the total of 16 orientation histograms, a SIFT keypoint descriptor is represented as a 128 dimensional vector. 128 dimensional vectors are difficult to visualize, and would result in heavy computational burden when computing Euclidean distance. Therefore, in this part of the project, we utilized a dimension-reduction technique named t-SNE (t-Distribution Stochastic Neighborhood Embedding) invented by L. Maaten and G. Hinton to reduce the dimensionality of SIFT keypoint descriptor. t-SNE aims to find lower dimensional representations of high dimensional data that best preserve the relative Euclidean distance from each data point to the rest of dataset. t-SNE has been proved to having superior performance over other dimension reduction techniques such as Isomap, LLE, and Sammon mapping in visualizing image datasets such as Olivetti faces library and COIL-20 object library (L. Maaten and G. Hinton, 2592). In this

part of the project, we want to examine if t-SNE can reduce the dimensionality of SIFT keypoints without losing their power of distinguishing.

Method:

In this experiment, we used the COIL-20 object image library to prevent background noise. We chose four visually distinctive objects from the image library, and name them “duck”, “shape1,” “shape2,” and “car” (Figure 2.2.1). Each object has 71 images that represent different views of the object.

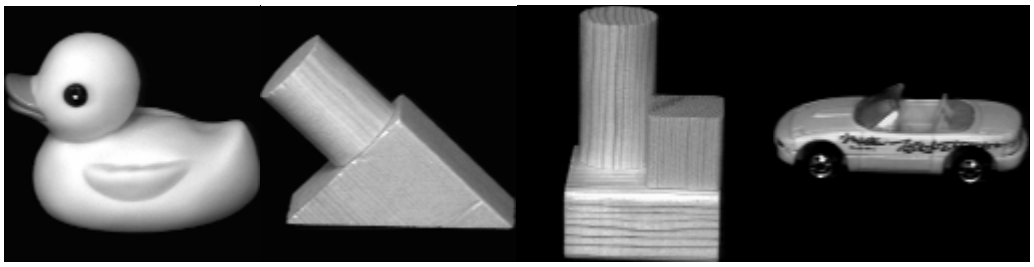


Figure 2.2.1: Sample images of the objects selected for the t-SNE experiment (duck, shape1, shape2, toy car).

Then we used the SIFT keypoints ranking system introduced in the previous section to extract and rank SIFT keypoints. For each object, 100 keypoints that scored the highest among all extracted keypoints were selected to represent the object. In addition to the 100 keypoints from each object, we selected 15 keypoints that has relatively short Euclidean distance to each other. It was hypothesized that if t-SNE can preserve relative Euclidean distance among SIFT keypoints, the lower dimensional representation of these keypoints should form a cluster on the map. After the keypoints were chosen, we used t-SNE to reduce the dimensionality of the selected descriptors to 2, and plotted the resulting data points on a 2-dimensional map.

Results:

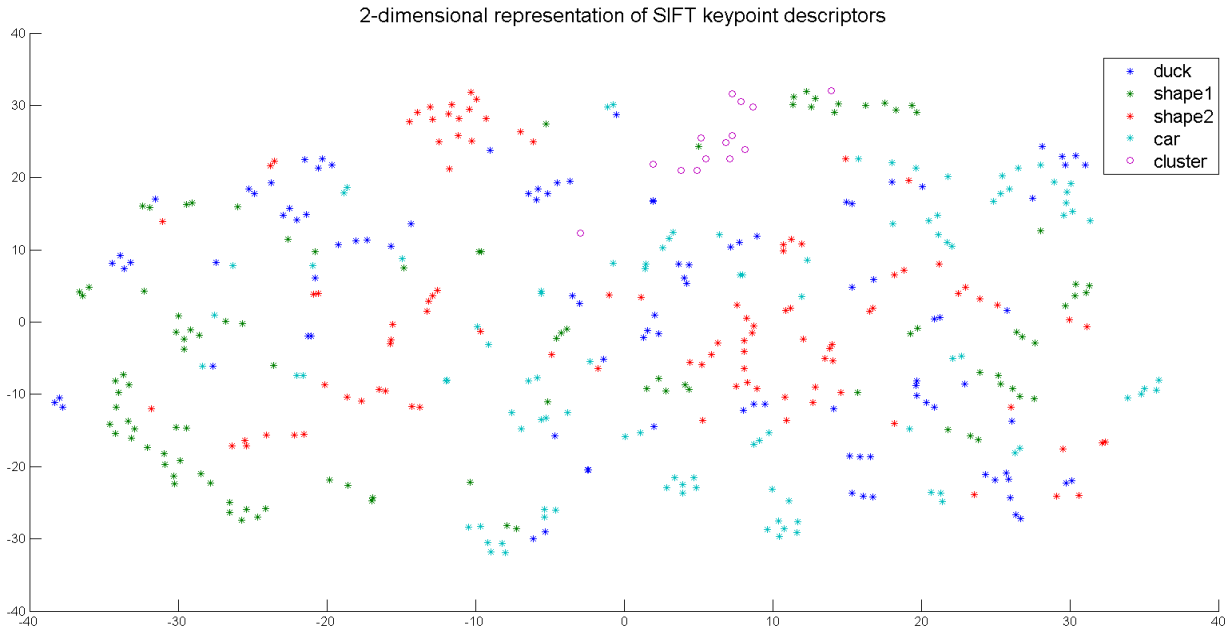


Figure 2.2.2 results of reducing SIFT descriptors to 2-dimensional data points using t-SNE.

Discussion:

From the 2-dimensional map (Figure 2.2.2), we observed that the majority of the data points belong to certain clustering groups. This confirmed the effectiveness of this dimension reduction technique because according to the mechanism of our SIFT keypoint ranking system, keypoints that have many similar keypoints in other views tend to score higher than the keypoints that exist only in certain views; consequently, a selected keypoint should have multiple similar keypoints in the selected keypoints set. Therefore, a data point cluster in Figure 2.2.2 is highly likely to be formed by a group of similar keypoints. Since keypoint similarity is defined by short Euclidean distance, this observation proved that t-SNE did preserve the relative Euclidean distance between keypoints in lower dimension representation. This finding was also confirmed by the relative tight cluster formed by the deliberately selected 15 keypoints that were hypothesized to be located close to each other on a lower dimension map (purple circles in Figure 2.2.2).

(2.3) A simple object classifier

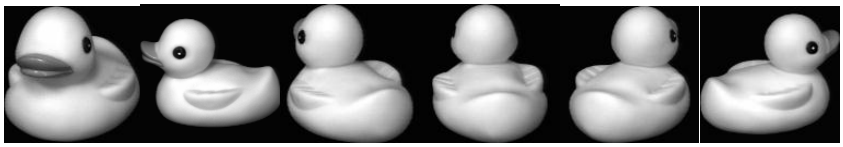
Purpose:

In the previous part of the study, we confirmed the effectiveness of t-SNE in reducing dimensionality of SIFT keypoint descriptor while preserving the relative Euclidean distance among SIFT keypoints. Based on this finding, we want to create an object classifier using lower dimensional representation of SIFT keypoints.

Method:

We selected the same set of objects as the previous experiment to be the training objects (Figure 2.2.1). First, we randomly chose number of (<10) images from image library of one object, and use these images to represent the object to be recognized. Then we used SIFT keypoint ranking system to obtain 25 sample keypoints that scored the highest from these images. After sample keypoints were obtained, 100 strong SIFT keypoints were extracted from each object using the same procedure as described in the previous section, except that the images used as sample images are excluded from being training images. Then we combined these keypoints to be a keypoints set and used t-SNE to generate a 2-dimensional representation of these keypoints. The classification procedure was to find the nearest neighbor of each sample keypoint in the lower dimension map. Each sample keypoint would vote for the object that contains its nearest neighbor keypoint. The object that got highest vote would be selected to be the final decision.

In this experiment, we want the program to recognize some images belong to object “duck,” so we chose the following images as the program input:



These images were first transformed to SIFT features and ranked using SIFT feature ranking system. We chose 25 descriptors that has highest scores to be the representation of this object.

Results:

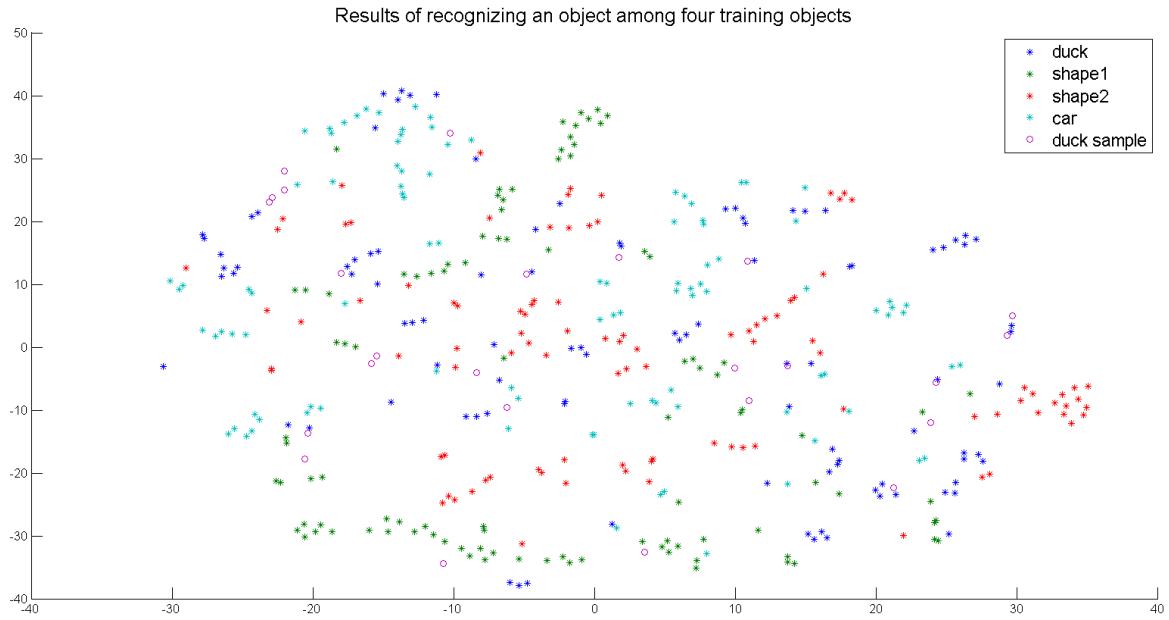


Figure 2.3.1: 2-dimensional representation of identifying object "duck" from total of four training objects.

Object name:	Duck	Shape1	Shape2	Toy car
Number of votes:	13	5	3	3

Table 2.3.1: Results of identifying some images belong to object "duck".

Discussion:

The results showed an object recognition demo that results in correct decision (Table 2.3.1). Because this is the only experiment we performed on the classifier so far, apparently more extensive tests are needed in order to confirm the effectiveness of this classifier. Yet we have identified some potential problems of this classification approach. The first problem was the input that represents the object to be identified consists of keypoints from multiple images, but most of the existing visual object recognition program use single image as input. The reason for using only strong keypoints as input was that the training dataset only contains the strong SIFT keypoint; if we use single image as input and most of the keypoints extracted from that image are not strong SIFT keypoints, the program is likely to produce incorrect identification results. The second problem was that the t-SNE procedure needs to be executed every time the main program accepts a new input, which would cost huge amount of computation that is unaffordable for real-time object recognition system. Some possible improvements will be addressed in the next section.

Summary and Possible Future Works

We have achieved the following goals:

- Comprehended the concept of visual feature and studied properties of SIFT feature extraction strategy.
- Designed the SIFT-keypoint ranking system, which can rank SIFT features by their power of distinguishing.
- Managed to reduce dimensionality of SIFT keypoint descriptors using t-SNE.
- Created a simple object recognition program and tested the program with ideal images.

In Stage (1), two problems are identified: restriction of multiple image input and heavy computational burden. There are two possible solutions to the first problem. The first solution is to have the existing system taking single image as input. Then we would conduct substantial amount of experiments to draw the conclusion if it can produce correct results. The second solution is to incorporate all keypoints, regardless of their ranking, as training data, and use single image as input. In voting procedure, instead of each sample keypoints having equal amount of votes, their votes would be weighed by the ranking scores of their nearest neighbor: higher scores would results in greater effect in the final classification decision, and vice versa. However, this solution is guaranteed to cause heavy computational burden, which raises our second problem. There is no obvious solution to the second problem so far, we might need to further investigate the mechanism behind t-SNE. In addition to solving these two problems, we will test this system in larger object sets and even with real world object images.

Reference

1. Bradski, Gary R, and Adrian Kaehler. *Learning Opencv: Computer Vision with the Opencv Library*. Sebastopol, CA: O'Reilly, 2008. Print.
2. OpenCV 2.4.2 documentation/tutorial site. 2012. OpenCV development group. May 12th , 2013 < http://docs.opencv.org/>
3. Lowe, David G. "Distinctive Image Features from Scale-Invariant Keypoints." *International Journal of Computer Vision*. 60.2 (2004): 91-110. Print.

4. Bhattacharyya, A. "On a measure of divergence between two statistical populations defined by their probability distributions". *Bulletin of the Calcutta Mathematical Society* **35 (**1943): 99–109. Print.
5. Bay, H, T Tuytelaars, and Gool L. Van. "Surf: Speeded Up Robust Features." *Lecture Notes in Computer Science*. (2006): 404-417. Print.
6. van der Maaten and G.E. Hinton. "Visualizing Data Using t-SNE." *Journal of Machine Learning Research : Jmlr*. 9.2 (2009): 2579-2606. Print.

Appendix

Functional Requirements:

Object Identification:

Result representation:

The program shall be able to produce a list of candidate locations represented as two-dimensional Cartesian coordinates with respect to the scene image frame. The size of area that the object may occupy shall be represented by width and height of a rectangular shape with the corresponding location coordinates being its geometric central point. The candidate locations must be sorted by the possibilities of the object presenting at that location.

Identification decision making:

User shall be able to specify a convergence rate with which the program can produce a Boolean decision: object identification succeeded/failed.

Program Input:

The program shall allow at least two types of input for the visual representation of object to be recognized: images in file system and video frames from digital camera

> **Images**: The program should accept most of widely used image format. Specifically, it should be able to process: windows bitmaps (bmp), portable image format (pbm, pgm, ppm), and Sun raster (sr, ras), and other image formats such as JPEG, JPEG 2000, PNG, and TIFF.
>
> **Video frames**: the program shall allow users to specify a rectangular area by cursor in a lively playing video streaming from camera. The image of selected rectangular area should be able to be saved upon user request.

The program shall allow at least two types of input for the visual representation of the scene: image files in file system and video frames from digital camera

> **Images**: (same as the object image)
>
> **Video frames**: the program must be able to query image frames from a digital camera in real time and use the image frames as the visual representation of the scene. (Requirements about camera to be specified in the non-functional requirement section)

User shall be able to select input sources for both object and scene through a menu which should be brought up upon program starts.

Program Output:

Object identification in a scene image:

As stated in the previous section, object identification results are represented as lists of candidate locations. When given a scene image as input, the program should produce an image of given format that has the object location drawn as points/rectangular shapes. Other candidate locations shall be printed in console ranked by their possibilities.

Real-time object tracking:

Result of real-time object tracking shall be represented as a rectangular shape drawn on each single frame of the output video to show the approximate size and location of the object being tracked. Lower output video refresh rate is allowed when using high-resolution camera to perform object recognition.

Non-functional Requirements:

Object recognition

Time efficiency

Keypoints and descriptors extraction: As stated in the introduction section, program shall use feature extraction functions provided by SIFT and SURF classes in OpenCV. Modifications on these functions in improving time efficiency are encouraged but not required.

Space Efficiency

The majority of program's space usage should depend on the size and amount of images/video frames used for training the program.

Development/Usage platform

Development environment

To guarantee the expandability of the program in case of further development, the program shall be implemented in C++ and/or python in order to support most of standard OpenCV 2.0 functions.

Camera interface

The program shall use the OpenCV digital camera interface in order to support most of the standard cameras that feeds images in real-time via USB.

Operating systems

The program must be able to function normally on both Unix-based operating system and Windows operating system that installed with OpenCV 2.0 or newer. The source code of the program shall be able to be compiled by any standard C++ compiler on any Unix-based or Windows operating system installed with OpenCV 2.0 or newer

Patent/non-free

SIFT and SURF are protected by patent, thus the program is prohibited from commercial use.