

ASSET IDENTIFICATION USING IMAGE DESCRIPTORS

by

Reena Ursula Friedel

A Thesis Submitted to the Faculty of
The College of Engineering and Computer Science
in Partial Fulfillment of the Requirements for the Degree of
Master of Science

Florida Atlantic University

Boca Raton, Florida

May 2012

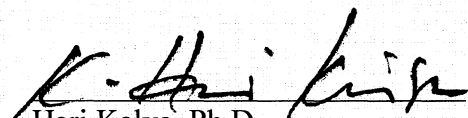
ASSET IDENTIFICATION USING IMAGE DESCRIPTORS

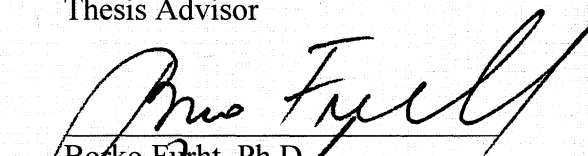
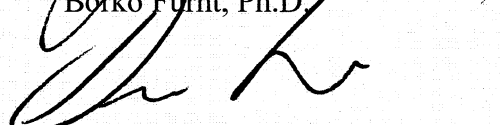
by

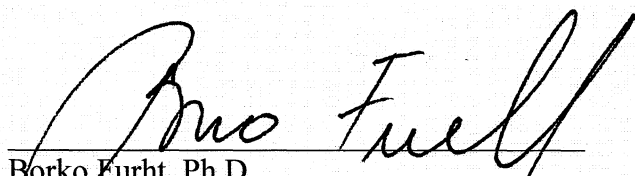
Reena Ursula Friedel

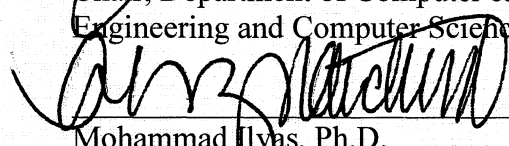
This thesis was prepared under the direction of the candidate's advisor, Dr. Hari Kalva, Department of Computer & Electrical Engineering and Computer Science, and has been approved by the members of her supervisory committee. It was submitted to the faculty of the College of Engineering and Computer Science and was accepted in partial fulfillment of the requirements for the degree of Master of Science.

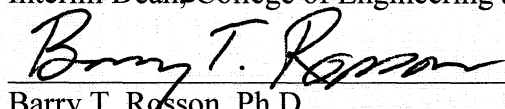
SUPERVISORY COMMITTEE:

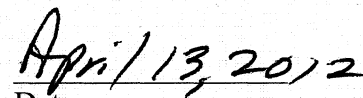

Hari Kalva, Ph.D.
Thesis Advisor


Borko Furht, Ph.D.

Daniel Raviv, Ph.D.


Borko Furht, Ph.D.
Chair, Department of Computer & Electrical
Engineering and Computer Science


Mohammad Ilyas, Ph.D.
Interim Dean, College of Engineering and Computer Science


Barry T. Ross, Ph.D.
Dean, Graduate College


Date

ACKNOWLEDGEMENTS

I would like to express gratitude to my advisor, Dr. Hari Kalva, for his constant guidance and support in helping me complete this thesis successfully. His enthusiasm, inspiration and his efforts to explain all aspects clearly and simply helped me throughout.

My sincere thanks also go to my committee members Dr. Borko Furht and Dr. Daniel Raviv for their valuable comments, suggestions and inputs to the thesis.

I would also like to thank the Southeast National Marine Renewable Energy Center (SNMREC) for making this study possible by providing its funding and Gabriel Alsenas, project manager for FAU's Center for Ocean Energy Technology for having given me the opportunity to work on this project and various other projects during my Masters. Thanks to Robert Miller, Senior Software Engineer, from Emerson Network Power, for the initial idea and the motivation for the project.

Many thanks to my fellow lab mates Oscar (for the iPad implementation), Velibor and others in the Multimedia lab at FAU for the stimulating discussions, enthusiastic support and interesting times we have had during the past two years. Last but not the least I would like to thank my family, my parents and brother for being a constant source of support and encouragement during my Masters.

ABSTRACT

Author: Reena Ursula Friedel
Title: Asset Identification Using Image Descriptors
Institution: Florida Atlantic University
Thesis Advisor: Dr. Hari Kalva
Degree: Master of Science
Year: 2012

Asset management is a time consuming and error prone process. Information Technology (IT) personnel typically perform this task manually by visually inspecting assets to identify misplaced assets. If this process is automated and provided to IT personnel it would prove very useful in keeping track of assets in a server rack. A mobile based solution is proposed to automate this process. The asset management application on the tablet captures images of assets and searches an annotated database to identify the asset. We evaluate the matching performance and speed of asset matching using three different image feature descriptors. Methods to reduce feature extraction and matching complexity were developed. Performance and accuracy tradeoffs were studied, domain specific problems were identified, and optimizations for mobile platforms were made. The results show that the proposed methods reduce complexity of asset matching by 67% when compared to the matching process using unmodified image feature descriptors.

ASSET IDENTIFICATION USING IMAGE DESCRIPTORS

LIST OF FIGURES	vii
LIST OF TABLES	viii
INTRODUCTION	1
1.1 Motivation	2
1.2 Main Contributions	4
1.4 Overview Of Thesis	5
1.5 Glossary Of Terms	6
BACKGROUND AND RELATED WORK	8
2.1 Traditional Methods	8
2.3 Asset Identification	11
2.4 Applications Using A Similar Approach	12
2.5 Image Processing In Asset Identification.....	14
2.5.1 Feature Detection (Extraction)	14
2.5.2 Feature Description.....	15
2.5.3 Feature Matching.....	16
2.6 Descriptors	17
2.6.1 Scale-Invariant Feature Transform (SIFT)	17
2.6.2 Speeded Up Robust Feature (SURF).....	17
2.6.3 Features From Accelerated Segment Test (FAST).....	18
PROBLEM DESCRIPTION.....	19
3.1 Asset Management Problem.....	19
3.2 Data Center Layout	19
3.3 Proposed Solution	21
3.4 System Considerations	23

PERFORMANCE EVALUATION	25
4.1 Dataset	25
4.2 Choice Of Descriptor	26
4.3 Choice For Resolution Of Images In The Database.....	29
4.4 Keypoint Reduction.....	32
4.4.1 Assessing the keypoints based on feature strength.....	32
4.4.2 Removing Overlaps Among Keypoints.....	34
4.5 Evaluation Of The Proposed Methods	36
4.6 Limitations Of The Proposed Solution	37
4.6 Evaluation Of Our Method Using The Stanford Dataset	38
CONCLUSION AND FUTURE WORK	41
5.1 Conclusion.....	41
5.2 Other Applications	42
5.3 Future Work	42
BIBLIOGRAPHY	43

LIST OF FIGURES

Figure 1.1 An estimate of the number of servers in the world. Numbers are taken from (“Google’s insane number of servers visualized”, n.d.).....	2
Figure 1.2 Asset tracking system	3
Figure 1.3 Life cycle of an asset	4
Figure 2.1 Sample Barcode.....	8
Figure 2.2 An RFID chip	10
Figure 2.3 Subset of features detected in both the query image and the database image (R. Szeliski, 2010).....	15
Figure 2.4 Lines indicate common matches between the query image and the database image (R. Szeliski, 2010)	16
Figure 3.1 A typical section of rack rail, showing rack unit distribution in comparison with a filled data center rack	20
Figure 3.2 Asset Identification Procedure	22
Figure 4.1: Image taken at full resolution with a Nikon D80 camera. Example of 1U asset.....	25
Figure 4.2: Image taken by an iPad and used as a query image. Example of 1U asset	26
Figure 4.3 Number of keypoints detected for the asset	27
Figure 4.4 Impact of the number of keypoints on the image resolutions	29
Figure 4.5 Impact of the keypoint extraction time on the image resolutions	30
Figure 4.6 Images to show the elimination of the keypoints	34
Figure 4.7 Different stages of the keypoint reduction steps	36

LIST OF TABLES

Table 4.1 Comparison of descriptors where time is reported in milliseconds.....	28
Table 4.2 Comparison of the query image against different resolutions of the same asset in the database	31
Table 4.3 Impact of using reduced keypoints	32
Table 4.4 Impact of using reduced keypoints	33
Table 4.5 Results of different threshold values chosen	35
Table 4.6 Comparison of implemented methods (all time in milliseconds)	37
Table 4.7 Comparison of time taken for the SMVS dataset	40

CHAPTER 1

INTRODUCTION

A data center is a facility that hosts computer systems, servers, power supplies storage systems, and other related computing equipment, referred to as assets. The size and number of these data centers are continuously increasing to accommodate the need and demand for data services. Assets are mounted in racks and a typical rack can accommodate up to 42 assets depending on the asset size. Large data centers have thousands of racks and keeping track of these large numbers of assets manually makes it very tedious and highly prone to errors. Google alone accounts for close to a million servers in their data center and that is only 2 percent of the servers existing all over the world, as seen in figure 1.1. The other big companies have not revealed their numbers but it is estimated that each of them have well over 100,000 servers.

Keeping track of such a large number of servers is difficult and gives room for a lot of manual errors. Human errors continue to be the greatest cause of unplanned downtime in data centers these days. The probability of human error causing points of failure within the data center is much greater in such companies.

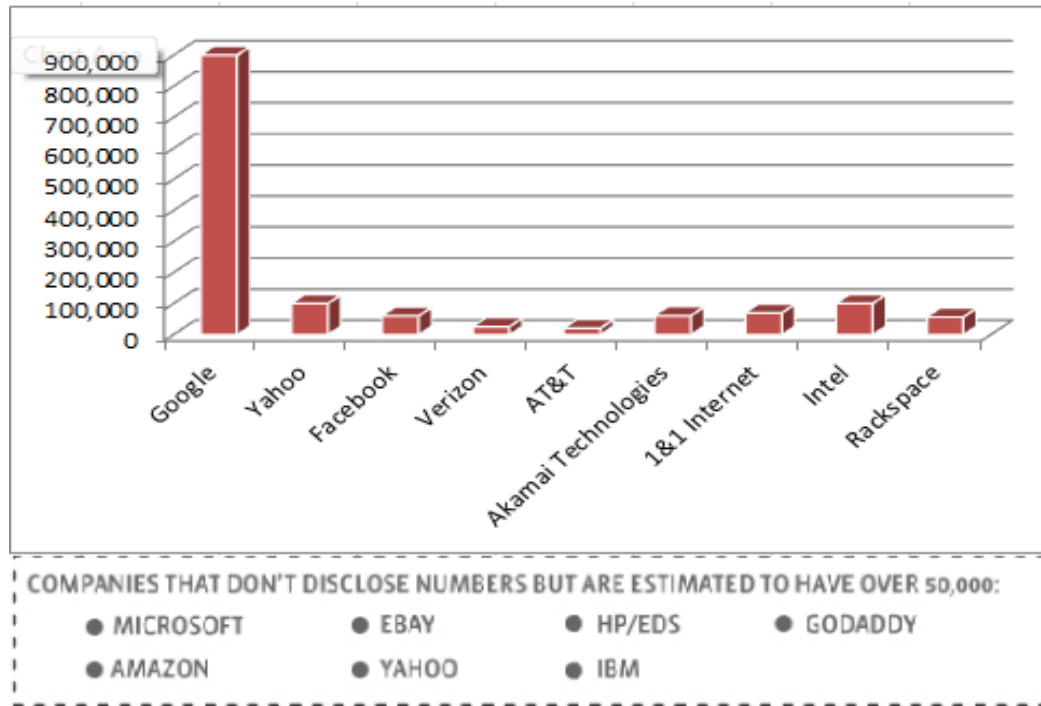


Figure 1.1 An estimate of the number of servers in the world. Numbers are taken from (“Google’s insane number of servers visualized”, n.d.).

1.1 Motivation

With increasingly complex environments, mass virtualization to the cloud and disparate tools used by traditional facilities and IT managers; it is more costly and difficult than ever to manage data centers. The result is expensive downtime, budget overruns, poorly planned changes and inability to meet business demands. Since most of the operations done in data centers today are manual, it is expensive in terms of human labor and is highly error-prone. Data center personnel can no longer haphazardly place a server in any open rack, without understanding how it will affect the overall data center from a holistic perspective.

Overlooking this impact analysis can result in severe disruptions in service. Thus with mobile devices and tablets becoming common and inexpensive, image processing applications can be exploited to cater to the needs of asset tracking and management in the data centers.

This thesis aims at presenting an image processing application for identifying assets in a data center using mobile devices, which can be used to track various details about the servers easily. An example where the user stands in front of a rack and uses the mobile device for asset identification is seen in Figure 1.2. Factors such as differences in size, color, background, scale, low light conditions, and camera resolution make the task of identification more complex.

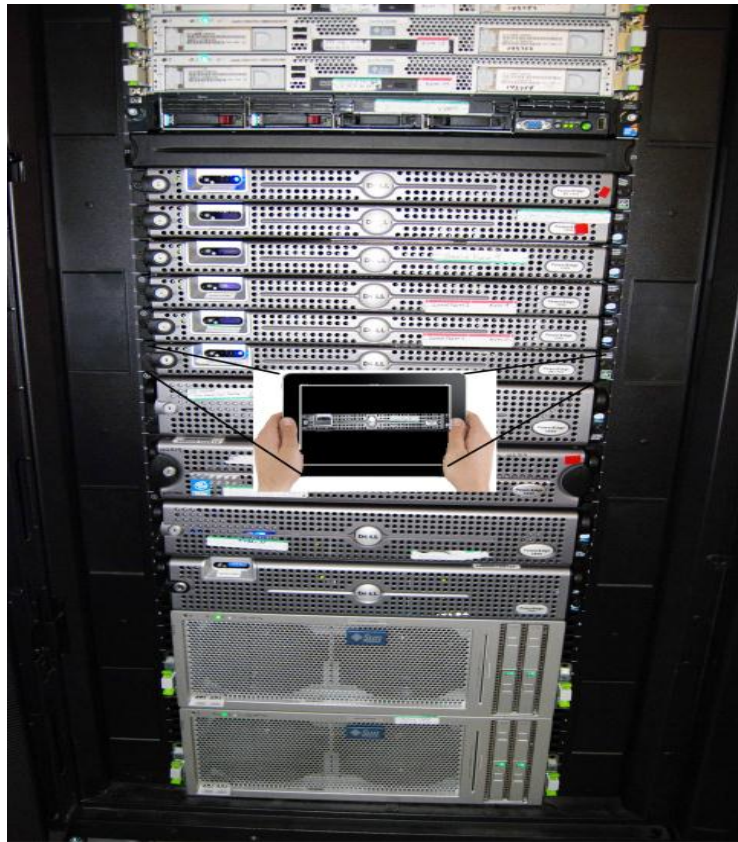


Figure 1.2 Asset tracking system

1.2 Main Contributions

The development of image matching algorithms by using a set of local interest points has been done over many years but applying this to the data center problem with constraints posed by the use of mobile devices makes this problem challenging.

The key contributions to this thesis include:

- 1) Identifying the challenges in asset tracking using image search.
- 2) Performance evaluation of image descriptors in the problem domain.
- 3) Complexity reduction and optimization for use on mobile devices.

1.3 Life cycle of an Asset:

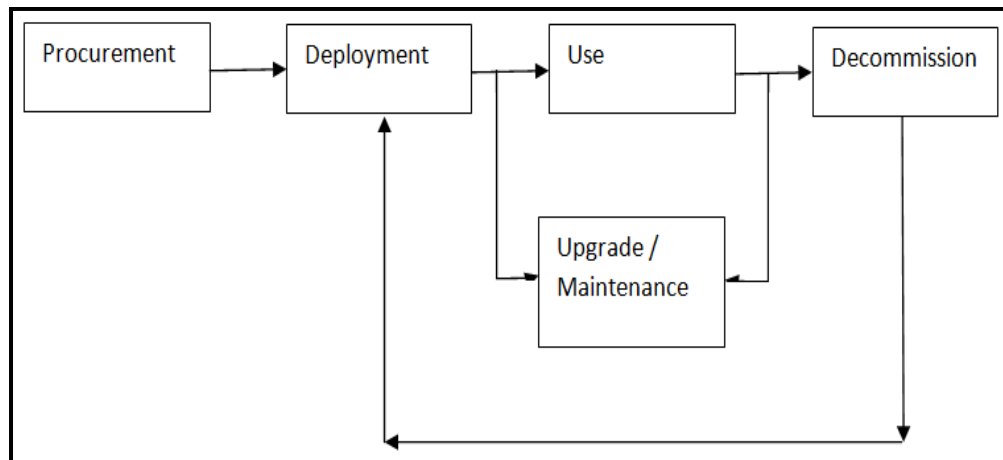


Figure 1.3 Life cycle of an asset

As seen from figure 1.3, the first phase in the asset lifecycle is its procurement. When an asset is bought its details are entered into an asset management system and it begins to be managed.

The second phase of the asset lifecycle is deployment. When an asset is deployed, the system is updated with relevant data such as location, responsible person for the asset, configuration details, vendor, warranty information, and any other data that will be useful in managing the asset.

The third phase of the asset management cycle is usage. During this phase measures about asset usage is tracked and it can be tracked to see which assets have been misplaced, which assets are not being used or which assets have been replaced. The majority of our application of asset identification happens during the Usage and the Upgradation phases.

As time progresses, the asset might be upgraded. For example, the software version may be changed, or a new hard drive may be added. When this occurs the configuration information for the asset requires an updation.

When an asset is no longer being used, it is decommissioned. Decommissioned assets may be useful to an organization, in which case they can be redeployed. Else, they most probably have some salvage value for which the asset management system should be capable of tracking it.

1.4 Overview Of Thesis

This thesis is structured as follows: Chapter 2 presents a background and related work in computer vision and image processing problems. Chapter 3 describes the problem pertaining to asset tracking and identification. Chapter 4 shows results of experiments carried out to address the thesis's goal, and their interpretation. Finally, Chapter 5 contains conclusions and suggestions for future work.

1.5 Glossary Of Terms

Object recognition: In computer vision, it deals with the task of detecting and classifying a given object in an image or a video sequence.

Asset: It refers to computer systems with its associated components, including telecommunications, storage systems, backup power supplies, security devices etc.

Feature: Any “interesting” part of an image like edges, corners etc.

Feature Detection: It deals with finding interesting points (i.e. features) in an image

Feature Extraction: It is the process of identifying the relevant and representative features (feature vectors) from an image to reduce the amount of data to process.

Keypoint: Every pixel of the image is searched to try and locate a feature in the feature detection phase. The keypoint is characteristic of such a feature and is described with (x,y) coordinates and has a response and size.

Response: Every keypoint in OpenCV is given a value that is representative of how strong the feature is. For a SURF detector this variable is populated by the HAAR Wavelets response value.

Descriptor: N-dimensional vectors that describe a feature point obtained during the feature description stage of image processing. A descriptor has to be distinctive, robust to noise, and geometric and photometric deformations.

SIFT: Scale Invariant Feature Transform (SIFT) transforms image data into scale-invariant coordinates.

SURF: Speeded Up Robust Feature (SURF) is a feature detector and descriptor. It is used in applications like object recognition. It is partly inspired by the SIFT descriptor.

FAST: Features from Accelerated Segment Test (FAST) is another feature detector which is based on identifying corners.

Query images: Images used to initiate search

Reference images: Images containing views of objects intended to be retrieved

Dataset: Collection of images, including reference and query images

Database: Collection of image descriptors and associated information.

CHAPTER 2

BACKGROUND AND RELATED WORK

Asset tracking and management is essential for the smooth and continuous operations for big companies. It also helps big retailers like Wal-Mart to identify and deal with problems in their supply chain, reduce overstocking and also locate goods. Several of these big companies, including government and military organizations are always looking out for low cost and efficient ways to track their assets and equipment.

2.1 Traditional Methods

In earlier times most of the asset management and tracking was done manually by people (maintaining a manual paper inventory). This was later replaced by the barcode systems and RFIDs. Gathering information manually was time consuming as the information had to be recorded first, then transcribed and later fed into a computer system. Every stage increased the already high chance of errors.



Figure 2.1 Sample Barcode

Barcode systems need a person to affix a barcode, (example shown in Figure 2.1), to each asset and then use a barcode reader to retrieve information. The barcode and its reader require no prior knowledge to use it and are fairly accurate under controlled settings. Although it is much faster and less error prone than a manual process, it is still a human intensive task which consumes time as each asset has to be manually labeled with a barcode, thereby increasing deployment cost.

Additionally, (McCathie & Michael, 2005), speak of barcodes requiring a line of sight technology which makes it necessary to place labels so that they are clearly visible. Barcodes are also susceptible to damage, and hence they have to be kept clean, handled gently in abrasion free environments, and cannot be exposed to extreme temperatures and harsh surroundings. Assets with non-flat surfaces and small edges additionally hinder barcodes to be placed correctly. All this makes it unfavorable to use barcodes for asset tracking and identification.

Radio Frequency IDentification (RFIDs), on the other hand, aims to address the above concerns, and in time replaces barcodes altogether. (Ouertani et al., 2008), describe an RFID system to consist of a “tag” which has a small integrated circuit (Silicon chip), memory and an antenna onboard, and a reader which can interrogate the tag through the antenna to retrieve information contained in the tag memory (seen in Figure 2.2).

RFID’s allow non-line-of-sight scanning, but they have a limited scanning range which is based on the necessary frequencies required for the application. To deal with the range limitations, (Patil et al., 2008) proposed to use a robot with an attached RFID reader which periodically sweeps along some programmed routes within a warehouse.

Due to their need to conserve power, RFID's have limited processing power and memory. (Ibach et al. 2005) proposed a system to attach a more powerful and a more expensive embedded system to each asset for accurate asset tracking. These embedded systems continuously measure the signal strength of multiple nearby Wi-Fi access points, and exchange the measurement results among themselves in order to derive their own locations. All these systems still require the use of specialized hardware and the additional cost of manually attaching external tags to all the assets. This leads to high cost of deployment, thereby discouraging many small and medium-sized institutions or companies from using them.



Figure 2.2 An RFID chip

An RFID tag can store a lot more information when compared to a barcode, such as delivery date, or even the last maintenance action and date. Although passive RFID tags can be used as an information carrier, their use to determine precise location is limited, as stated by (Ouertani et al., 2008).

2.2 Image Based Methods

Due to the limitations of barcodes and RFIDs, discussed in Section 2.1, there is a demand for cheaper and more feasible solutions. This leads to the solutions based on image processing. Modern mobile phones and tablets are well equipped with cameras, displays and have good graphics capabilities that it makes it possible to exploit image and video processing applications on them. This gives rise to different kinds of mobile visual search applications like identifying assets, paintings, landmarks, books, CDs, DVDs, printed documents etc. ("Google goggles," 2011), ("Nokia Point and Find", 2012), ("Kooaba," 2011) and ("SnapTell," 2007-2009) were the initial systems to deploy such mobile visual search systems. All these are commercialized image recognition applications which make use of the basic image processing methods and are used for searches based on pictures taken by handheld mobile devices.

2.3 Asset Identification

The asset tracking system developed as part of this thesis has the following features which make it desirable for use.

- Firstly, it uses readily available smartphones and tablets which users own.
- Secondly, the system does not require any tags or markers, thus eliminating the time and effort needed to physically attach them to individual assets. RFIDs and barcodes cannot be used here since a data center could have a large number of servers which increases cost, and if an RFID tag were to be attached to each of the servers there would be a frequency issue that would arise in reading information from them.

- Thirdly, since mobile phones are being used, we can easily display detailed information about the assets on the mobile phone's display.

2.4 Applications Using A Similar Approach

In the past many mobile visual search systems have been developed. Applications include landmark recognition where a mobile application serves as a virtual tour guide; product cover recognition where users take pictures of products to automatically retrieve a list of current prices from different vendors and additionally media samples for CD and DVD covers, a video snapshot recognition where visual search seamlessly links what people watch on their TV's at home and on their mobile devices when away from home.

(Quoc and Choi, 2009) considered a system with three components which included book region extraction, book segmentation, and title extraction. OCR was used to identify the book title. OCR may not be robust, against photometric and geometric distortions often encountered in photos taken by smartphones.

The book recognition system developed by (Chen et al., 2010a) makes use of a system where they perform edge detection to identify the book splines and then identify longer edges to determine the book boundaries. Such a model cannot be used by our system, as an asset could have a long line drawn in the center through the logo and assets stacked over each other could have labels over them which make it difficult to identify one long and horizontal edge.

In the mobile product recognition system, built by (Chen et al., 2010b), which is based on a low bit-rate image retrieval system with a client-server architecture.

In our asset recognition system this cannot be used as most data centers are not well equipped with Wi-Fi access which forces the images to be stored and dealt with locally on the mobile device or the tablet device.

Additionally work has been done by (Burkhard et al., 2011), to identify logos on vehicles, where the SIFT descriptor was ruled out, since variations in brightness, size, and color of the logos significantly reduce the robustness of feature detection. Vehicle features in the foreground, and specular reflections hindered the process of identification. They use the Fourier shape descriptors, which naturally utilizes the varying shapes of the logos in determining the most likely manufacturers. Two assumptions: the logo is seen head-on (no perspective change or rotation), and that the logo is the largest object in the image. In our system we cannot use such assumptions as the logo on the asset need not be the largest object in the image.

The CD, DVD and Book cover image retrieval system, developed by (Chen et al. 2011), is another such application that has been developed which uses the client server architecture.

Technologies for image based search have matured sufficiently enough and the MPEG committee has begun standardization efforts as seen in (ISO/MPEG, 2011). It is required that this standard for compact descriptors will simplify descriptor extraction and matching for visual search related applications, also provide hardware support for such applications on mobile devices, make sure that visual search applications and databases are inter operable, and thus enable high performance of applications conformant to the standard.

2.5 Image Processing In Asset Identification

The main aim of Computer Vision is to perceive the information within a picture. A computer vision system processes images captured with a camera, and extracts meaningful information from the acquired images. It tries to imitate human vision where the brain processes images captured by the eyes. Computer vision is a vast topic for study and research and its applications are rapidly growing over the years.

A computer vision system is a complex system involving various stages that include acquiring an image, pre-processing, feature extraction, segmentation, accomplishing visual tasks like object detection and recognition, and many others. In this thesis, the main focus is on object recognition, which consists of identifying an asset in data center server racks.

In this section, the image processing steps which help in identifying the asset is discussed. These include the steps of feature detection, feature extraction and feature matching.

2.5.1 Feature Detection (Extraction)

A feature is defined as an "interesting" part of an image. Feature detection is one of the basic steps and is usually done at the beginning of any image processing application. Every pixel of the image is searched to try and locate a feature. Examples of such features include, edges, corners, region of points, and ridges. A subset of the features for a given image is shown in Figure 2.3.

For robust image matching, we desire interest points to be repeatable under perspective transformations (or, at least, scale changes, rotation, and translation) and real-world lighting variations. Current generation smartphones have limited compute power, typically only a tenth of what a desktop personal computer provides. Thus the interest points are required to be fast to compute and highly repeatable, as referred to in the paper by (Girod et al., 2011).

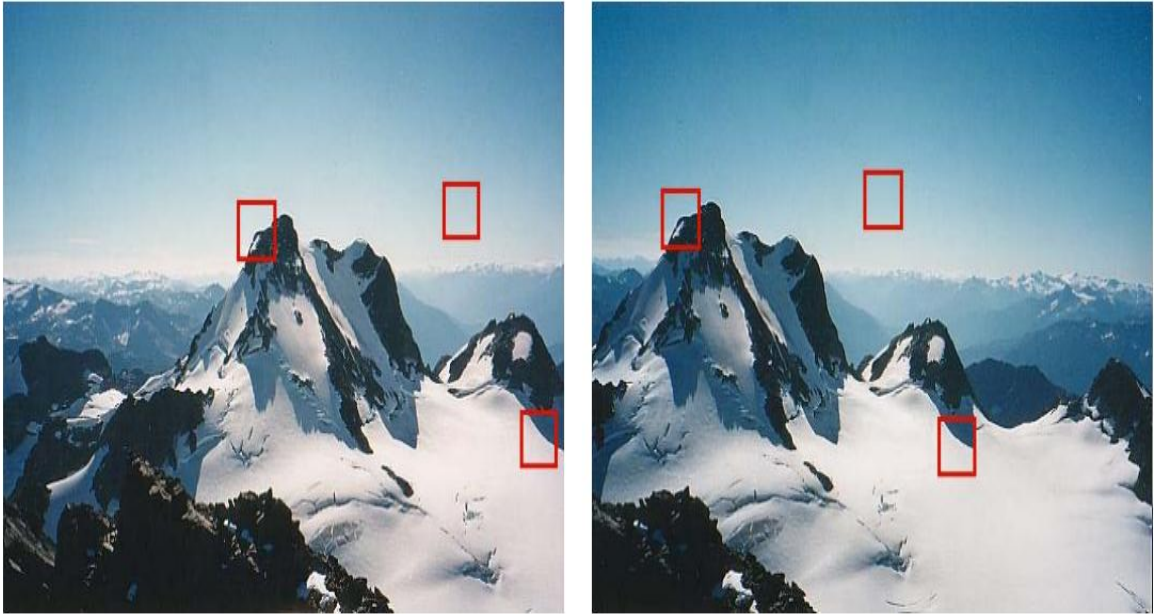


Figure 2.3 Subset of features detected in both the query image and the database image (R. Szeliski, 2010)

2.5.2 Feature Description

Once the feature detection is complete, a local image region around the features is extracted. Each region around detected keypoint locations is converted into a more compact and stable (invariant) descriptor that can be matched against other descriptors. This extraction takes up most of the computations in image processing applications. This gives rise to a feature descriptor (or feature vector).

Feature descriptors are usually N-dimensional vectors that describe a feature point, ideally in a way that is invariant to change in lighting and to small perspective deformations.

Descriptors should be discriminative, i.e., characteristic of an image or a small set of images. Descriptors that occur in almost every image (the equivalent of the word ‘and’ in text documents) would not be useful for retrieval.

2.5.3 Feature Matching

This stage deals with searching for the most likely matching points among the other images. Features are extracted from a set of reference images and stored in a database in the form of feature descriptors. A new image is matched by individually comparing each feature from the query image to this database and finding candidate matching features based on matching methods like the Euclidean distance, nearest-neighbors etc.

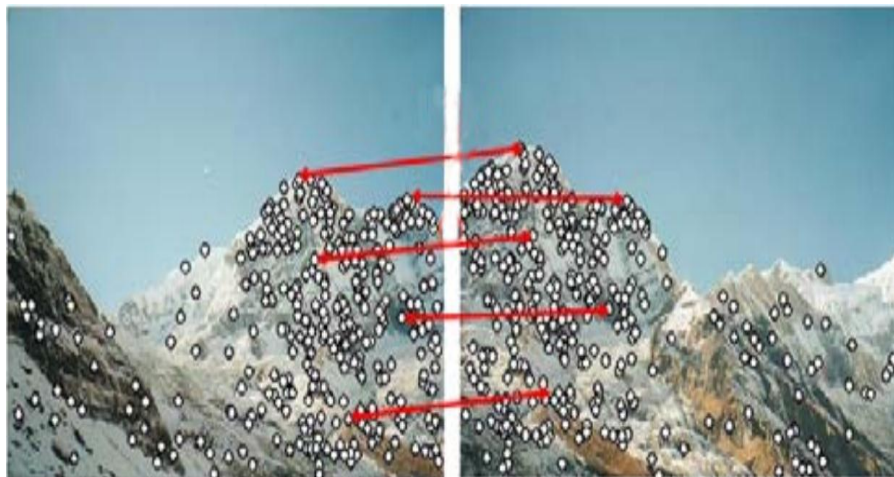


Figure 2.4 Lines indicate common matches between the query image and the database image (R. Szeliski, 2010)

2.6 Descriptors

In this thesis, the Scale-invariant feature transform (SIFT), Speeded Up Robust Feature (SURF) and Features from Accelerated Segment Test (or FAST) descriptors were used and evaluated. Below is a description of these detectors which are also descriptors. These descriptors were considered keeping in mind the domain and application of asset identification.

2.6.1 Scale-Invariant Feature Transform (SIFT)

(Lowe, 1999), introduced the Scale Invariant Feature Transform (SIFT) which transforms image data into scale-invariant coordinates, relative to local features. The SIFT algorithm is a robust method which extracts distinctive features from images invariant to rotation, scale and distortion. In order to identify such invariant keypoints that can be repeatedly found in multiple views of varying scale and rotation, local extreme are detected in Gauss-filtered difference images, (Ruf & Detyniecki, 2009). SIFT recognition might fail in cases with poor illumination conditions (excessive, non-uniform, or poor lighting, shadows), weather, dirt, high occlusion from other objects, and camera wide (perspective) view angle (low contrast), which results in poorly detected features, as researched by Psyllos et al.(2010). The SIFT descriptor has a size of 128.

2.6.2 Speeded Up Robust Feature (SURF)

Speeded Up Robust Features (SURF) uses integral images to compute an approximation of the Hessian matrix.

When rotation invariance is not considered, which results in a scale-invariant version of the descriptor, leads to the descriptor ‘upright SURF’ (U-SURF). In applications, like mobile robot navigation or visual tourist guiding, the camera often only rotates about the vertical axis. The benefit of avoiding the overkill of rotation invariance in such cases is not only increased speed, but also increased discriminative power, as stated in the paper by (Bay et al., 2006).

(Bay et al., 2006), state that the SURF algorithm is a variation of the SIFT algorithm. Its major differences include a Hessian matrix-based measure as an interest point detector and approximated Gaussian second order derivatives using box type convolution filters. Here, the use of integral images enables rapid implementation. The SURF descriptor has a size of 64.

2.6.3 Features From Accelerated Segment Test (FAST)

Features from Accelerated Segment Test (FAST), introduced by Rosten and Drummond, evaluate a small number of individual pixel intensities using decision trees. It computes the fraction of pixels within a neighborhood which have similar intensity to the center pixel. FAST compares pixels only on a circle of fixed radius around the point, (Tuytelaars & Mikolajczyk, 2008).

CHAPTER 3

PROBLEM DESCRIPTION

3.1 Asset Management Problem

Unnecessary data center costs include human errors in every aspect of asset management from keeping track of assets, maintaining inventory and identifying assets which have been moved. Keeping in mind all this and also the large number of assets that a data center can have, a cost effective, error free and efficient method is required.

The problem addressed above is taken care of by creating a mobile phone or iPad application which is used to visually identify an asset with the help of the camera on the device. It identifies the asset in a rack and retrieves real-time relevant information about it. This information is then overlaid and presented on the screen of the mobile device so that the IT personnel can then take the necessary actions.

3.2 Data Center Layout

A data center can occupy one room of a building, one or more floors, or an entire building. Most of the equipment is often in the form of servers mounted in 19 inch rack cabinets, which are usually placed in single rows forming corridors (so-called aisles) between them.

3.3 Proposed Solution

Using an asset management application on the iPad, pictures of a complete asset are taken. Once the picture is taken, keypoints in the image are detected (which includes the step of feature extraction from chapter 2). Once the keypoints are detected, feature vectors for each keypoint are extracted. The set of descriptors extracted for the query image are then used to compare it against the feature descriptors of unique assets that are stored in the database.

A match is then performed on the basis of the kNN (k- Nearest Neighbors). Nearest neighbor matching identifies only the best match and rejects all the others below a given threshold value yielding fewer false matches and the overall precision is high. The distance between the descriptors is the main similarity criterion, (Mikolajczyk & Schmid, 2003).

We perform this kNN match in two directions, i.e. for each point in the query image we find the two best matches in the reference image (in the database), and then we do the same thing for the feature points in the other direction from the reference image to the query image. Thus for each feature point, we have two possible matches. We then use a ratio test to find their distances. If this calculated distance is very low for the best match, and much larger for the second best match, we can safely accept the first match as a good one since it is unambiguously the best choice. If the two good matches are pretty close in distance, then there exists a possibility that the match is erroneous and hence, these matched get rejected. This is done by making sure that the ratio of the distance of the best match over the distance of the second best match is not greater than a given threshold.

Later, after all the matches have been eliminated in the query match set and the reference match set, we extract those matches that are symmetrically similar and agree with each other. This is the symmetry test which imposes that, for a match pair to be accepted, both the matching points have to be the best matching feature of the other.

The matches that finally pass through the ratio test and symmetry test are then returned back to the user with details about the asset. These details include model number, part number, slot in the server rack etc.

This entire process of asset identification is illustrated in Figure 3.2

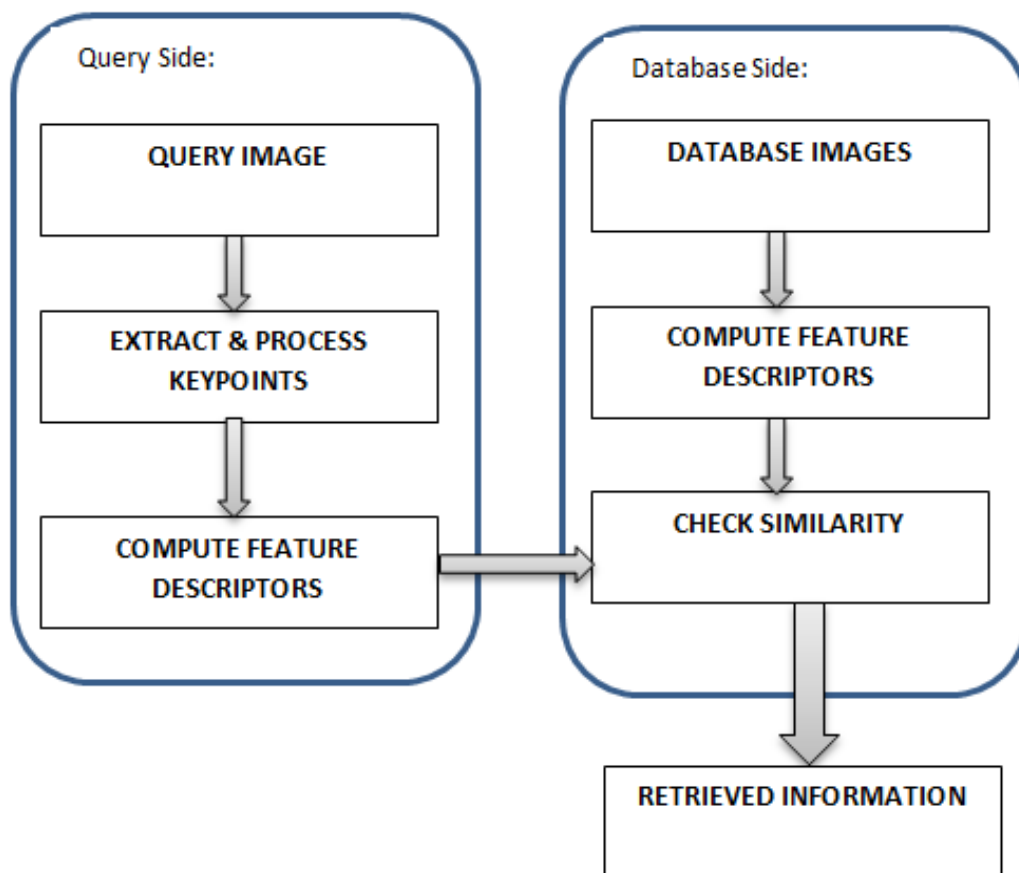


Figure 3.2 Asset Identification Procedure

3.4 System Considerations

Based on the application requirements and taking into consideration the data center conditions, the following goals were identified as essential:

- In the asset identification problem, our primary goal was to identify the best possible match fast, implying that the time taken for the descriptor extraction was to be the least.
- Search speed was a secondary goal.
- The asset identification system has to work within stringent memory and the computational power constraints of the mobile device. The processing on the mobile devices must be fast and economical in terms of power consumption as well.
- The algorithms used for the asset identification should be capable of delivering accurate results in the lowest time possible.
- The retrieval system should be robust to allow a reliable recognition of objects captured under various lighting conditions as data centers are not always well lit.

The main goal for building this application is to keep the total time at its minimum as this application is used on smartphones and tablets. The total time includes the time of extracting keypoints (T_k), computing the descriptors (T_x) and the time of matching the extracted descriptors with the descriptors stored in the database (T_m), and this is computed as shown in equation 3.1.

$$T = T_k + T_x + T_m \dots\dots\dots \text{Equation 3.1}$$

The time T_k depends on the size of the image and the descriptor used. The number of keypoints extracted depends on the type of image and the time T_x is a function of the number of keypoints and the type of descriptor. The final component of time T_m is a function of the matching algorithm and number of keypoints in the query image and the keypoints in each of the images in the database. These dependencies are discussed further in the next chapter.

CHAPTER 4

PERFORMANCE EVALUATION

The performance of the descriptors and complexity tradeoffs are evaluated using the assets database created from a data center at the Florida Atlantic University.

4.1 Dataset

The data for this thesis was collected from the local server room at Florida Atlantic University. The server room was sufficiently lit up and had assets mounted on server racks with 15 distinct assets in all the racks. The unique asset images were annotated and populated in the database with the feature vectors. The challenges include images captured by low resolution mobile phone cameras, server room lighting conditions and the speed of returning the matches. Figures 4.1 and 4.2 show examples of asset images used in this study. Typical asset sizes were 1U and 2U.



Figure 4.1: Image taken at full resolution with a Nikon D80 camera. Example of 1U asset



Figure 4.2: Image taken by an iPad and used as a query image. Example of 1U asset

4.2 Choice Of Descriptor

After evaluating several methods for object recognition, Scale-Invariant Feature Transform (SIFT), Speeded Up Robust Features (SURF), Upright SURF (USURF) and Features from Accelerated Segment Test (FAST), as discussed in Chapter 2, were chosen. Since SURF descriptors are mostly based on intensity differences, they are faster to compute. However, SIFT descriptors are generally considered to be more accurate in finding the right matching feature. SIFT detectors are slow to compute, because of the high dimensionality of their descriptors, but produces highly repeatable features. The SURF interest-point detector provides significant speed up over SIFT. The FAST corner detector is an extremely fast interest-point detector that offers very low repeatability. To study the effect of the various descriptors the below experiment was carried out.

A single asset image, taken with an iPad with resolution of 1280x100, was used as the query and one image of 2821x1053 resolution was saved in the database. The application was then made to run on these images taking one descriptor at a time. Keypoints were found for both the query image and the reference image. Descriptors were extracted for the query image. Descriptors for the reference image were extracted offline and stored in the database.

To evaluate a match, the query image descriptors are compared against the descriptors stored in the database. Figure 4.3 shows the number of keypoints that were extracted by each descriptor for the query image.

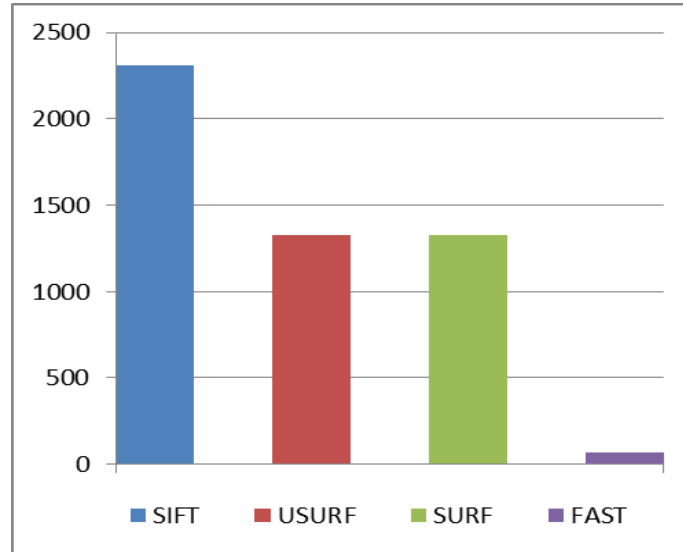


Figure 4.3 Number of keypoints detected for the asset

As seen from the figure 4.3, SIFT extracts the most number of keypoints and thus the features are highly repeatable. SURF and USURF have approximately equal values and FAST extracts the least number of keypoints.

It is seen from the paper by (Schmid, 2000), that if all descriptors lie close to each other, they do not convey much information, implying that the information content is low in that region of the descriptors. On the other hand if the descriptors are spread out, information content is high and matching is likely to succeed. Using this, we focus on reducing the keypoints in a given region.

Table 4.1 Comparison of descriptors where time is reported in milliseconds

Descriptor	No. of keypoints	Keypoint extraction time (Tk)	Descriptor extraction time (Tx)	Matching time (T m)	Matched or not	Total Time (T)
SIFT	Q=1370	271.1098	145.9			
	R=2167	532.2285	287.4561	168.3196	Match	585.3294
USURF	Q=762	24.69606	50.4129			
	R=1467	48.07026	87.93381	172.2474	Match	247.3564
SURF	Q=762	65.55889	45.27735			
	R=1467	114.9554	111.7241	108.724	Match	219.5602
FAST	Q=55	0.961546	4.179278			
	R=113	1.873061	4.873557	7.555001	No match	12.69583

Legend:

Q = Query

R = Reference

As seen from table 4.1, time complexity i.e. total time is high for all the four descriptors. SIFT taking the most time and FAST taking the least time. We also note from Table 4.1 that the matching performance of SURF is good with much less time complexity. FAST on the other hand has a very good time complexity but the accuracy of finding the correct asset match is relatively poor. Thus we selected SURF which will be the descriptor discussed in the rest of the thesis.

From the above table, the following strategies were used to reduce time complexity:

- Changing the resolution of the images in the database to yield fewer descriptors.
- Keypoint Reduction based on the importance of the points.
- Removing overlapping keypoints by further reducing them using a distance based approach.

These approaches are discussed in the sections below.

4.3 Choice For Resolution Of Images In The Database

Time complexity is largely a function of the number of keypoints in the image which in turn depends on the resolution of the images. The bigger the image the more the number of keypoints is extracted.

On plotting the number of keypoints versus the image resolution on a graph, we find that the number of keypoints increases linearly with the increase in the image resolution, as seen in figure 4.4. The dashed line indicates that this was the best resolution for which we were able to get a 100 percent accuracy of identifying the asset.

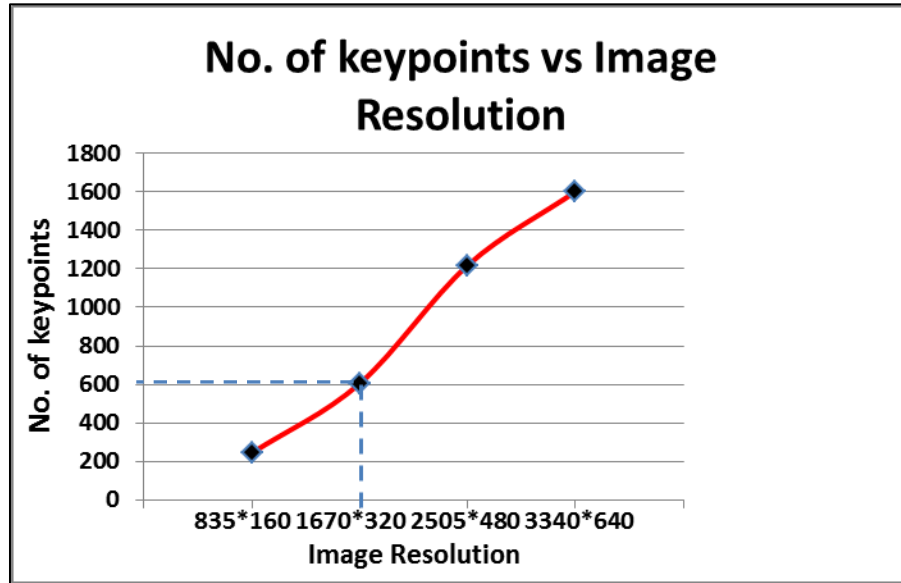


Figure 4.4 Impact of the number of keypoints on the image resolutions

We also see that the keypoint extraction time (T_k) in equation 3.1 is a function of the number of keypoints, seen in Figure 4.5. It increases linearly as the image size increases.

The dashed line in figure 4.5 indicates that this was the best resolution for which we were able to get a 100 percent accuracy of identifying the asset achieving a good time complexity.

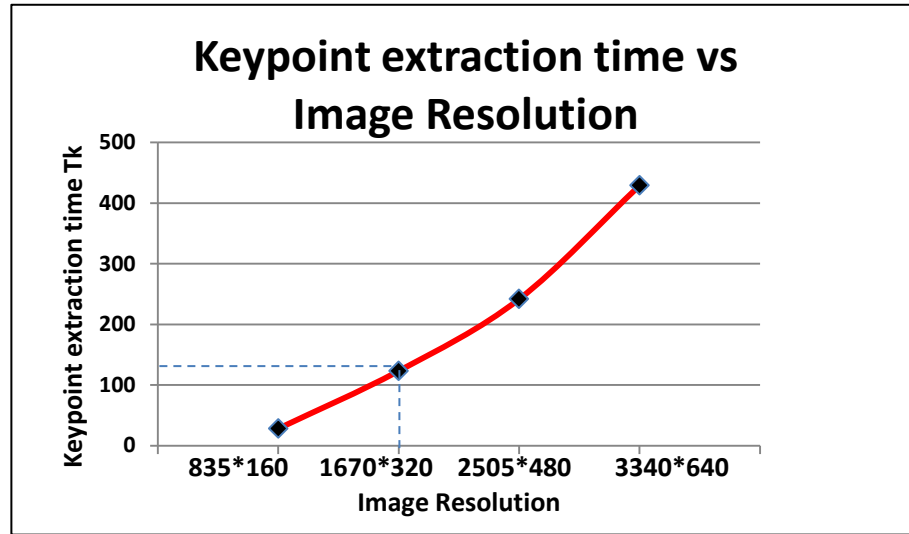


Figure 4.5 Impact of the keypoint extraction time on the image resolutions

Time complexity increases at each stage of the asset identification. Since complexity is largely a function of keypoints and image size, complexity can be reduced by reducing the image resolution.

A lower resolution image will reduce the keypoint extraction time linearly and the number of keypoints detected will also be reduced as image features are lost because of sub-sampling. We have to tradeoff reduced complexity for reduced accuracy and this tradeoff has to be balanced. Moreover, SURF is known to perform very poorly for the low resolutions, as pointed out from the paper by (Ruf, B., & Detyniecki, M., 2009).

To identify the correct resolution for the image in the database, the following experiment was carried out. Images were taken using a high resolution Nikon D60 (10.2 mega pixel) camera.

The images were cropped to include just the asset in consideration. It was saved with four different resolutions i.e. full resolution, three quarter, half and quarter resolutions.

We then ran the program using one query image against all the four images in the database. We found that it was able to find matches of the queried asset with the full resolution asset image (taking more time), the three quarter resolution asset image (in lesser time) and the half resolution asset image (with a still further decrease in time). The quarter resolution image (which resembles the iPad quality resolution) on the other hand took the least time in executing but there was no match identified. At this step we identified that it is best to save the image at half resolution because there is a gain in time and the accuracy of identifying the asset is maintained.

Table 4.2 Comparison of the query image against different resolutions of the same asset in the database

Images	No. of Keypoints	Keypoint Extraction time (Tk)	Descriptor extraction time (Tx)	Matching time (Tm)	Matching accuracy
Query Image [1280*200]	478	54.667227	28.698912		
Database Images:					
Full Resolution 3340*640	1602	428.902017	155.538674	99.62859	Match
Three Quarter Resolution 2505*480	1219	241.517794	101.802712	80.22565	Match
Half Resolution 1670*320	605	123.035711	53.380485	51.65225	Match
Quarter Resolution 835*160	244	27.912033	13.420825	33.4768	No match

Table 4.2 shows the matching performance and total time at different resolutions for reference images in the database. As shown in Table 4.2, computational complexity can be reduced by approximately 50% without affecting the matching performance.

4.4 Keypoint Reduction

The next consideration was to minimize the total time of retrieval by reducing the number of keypoints used to identify a match. Reducing the keypoints also reduces the descriptor extraction time and the matching time.

Table 4.3 shows the matching performance when the number of descriptors in the query image are reduced by selecting only a portion of the keypoints. As shown in Table 4.3, as the number of keypoints is reduced, total time reduces but the matching performance also drops.

Table 4.3 Impact of using reduced keypoints

Method	Total time (T) in ms	Match
SURF_100%keypoints	2415.12	all assets matched
SURF_75%keypoints	2061.16	80% of assets matched
SURF_50%keypoints	1342.05	60% of assets matched
SURF_25%keypoints	680.57	50% of assets matched

4.4.1 Assessing the keypoints based on feature strength

Not all keypoints in an image are equally good in identifying matches. We prioritize the keypoints based on the feature response of the keypoints computed during keypoint extraction in OpenCV.

Feature response or feature strength is a characteristic of the OpenCV keypoint. This value gets populated during the feature extraction phase which means it is dependent on the type of feature detector used. SURF populates the feature responses from two first-order HAAR wavelet filters (1, -1), and are collected on each feature point.

From table 4.3 it is seen that random elimination of keypoints reduces the possibility of finding accurate matches of an asset. For this we came up with the following method:

- 1) Sort the keypoints based on the feature response.
- 2) Keypoints with smaller feature strengths are removed keeping only the more significant set of keypoints.

The results are summarized in Table 4.4. Prioritizing keypoints reduces total time without affecting matching performance. Only 50% of the most significant keypoints can be used in descriptor extraction without affecting matching performance while reducing total time by almost 50%.

Table 4.4 Impact of using reduced keypoints

Method	Total time (T) in ms	Match
SURF_100%keypoints	2415.12	all assets matched
SURF_75%keypoints	2061.16	all assets matched
SURF_50%keypoints	1342.05	all assets matched
SURF_25%keypoints	680.57	80% of assets matched

As we see in table 4.4, if we use only 50% of most significant keypoints, the complexity reduces by almost 50% without affecting the matching performance.

4.4.2 Removing Overlaps Among Keypoints

Once the keypoints are reduced using the technique in section 4.3.1, we still try to identify a possibility for further reduction in time complexity. For this, we make use of the size of the keypoint i.e. its diameter.

Steps done for this reduction:

- 1) Keypoints are sorted in decreasing order of their feature response values and 50% of those strong keypoints are stored in a vector K (from section 4.3.1)

- 2) \forall keypoints in K, compare the keypoints such that:

$$\text{distance}(K_i, K_j) < \text{Threshold value and } i \neq j.$$

- 3) If the distance is lesser than the threshold, eliminate those keypoints. This means that the closely placed keypoints are eliminated. This elimination falls from the conclusion derived from (Schmid, 2000), that the closer the keypoints the less information is retrieved from them.

An example of such an elimination can be seen in the below image, figure 4.6

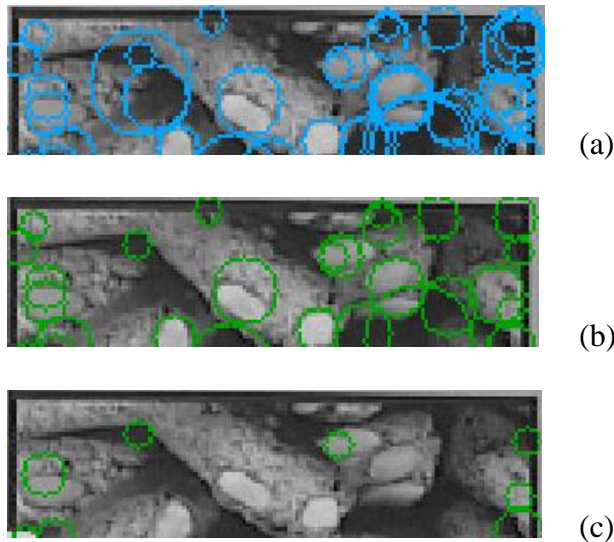


Figure 4.6 Images to show the elimination of the keypoints

Figure 4.6, shows:

- (a) Keypoints before overlapped circle elimination
- (b) Keypoints with the overlapped circle removed at a distance of $0.15 \times \text{size}$
- (c) Keypoints with the overlapped circle removed at a distance of $1 \times \text{size}$

From figure 4.6, it is seen that when the threshold is the size of the keypoint ($1 \times \text{size}$), a lot of keypoints get eliminated. This is not desirable since a lot of information content is lost. Assets which differ in a single digit of the model number will suffer with such a threshold value. Example, Poweredge 1750 and Poweredge 1950.

To identify a good steady threshold value, a series of experiments were carried out with the given domain of assets. Threshold values which were 1 times the size, 0.5 times the size of the keypoint, etc. were selected. The results of these experiments are as seen in the table 4.5 below.

Table 4.5 Results of different threshold values chosen

Method	No. of Keypoints	KP Extraction time (Tk)	Descriptor extraction time (Tx)	Matching time (T)	Percentage of matches
$1 \times \text{size}$	61	56.21	4.7	236.7989	47%
$0.5 \times \text{size}$	108	56.7	7.11	479.2166	60%
$0.25 \times \text{size}$	164	55.03	10.94	613.9084	87%
$0.15 \times \text{size}$	180	57.01	10.67	669.8754	100%

From table 4.5 it can be seen that, as the matching time decreases the accuracy of finding a perfect match also decreases. Using the value of 0.15 as the threshold, we can achieve 100 percent accuracy in identifying all the images.

4.5 Evaluation Of The Proposed Methods

Using resolution reduction and keypoint reduction approaches, we were able to reduce the time complexity significantly and have made it possible to have a responsive iPad asset detection application.



Figure 4.7 Different stages of the keypoint reduction steps

Figure 4.7 above shows:

- (a) Entire query image with all its keypoints
- (b) Query image with 50% of its reduced keypoints
- (c) Query image with the reduced overlapping keypoints
- (d) Matched output with lines indicating the matches from the query image to the image in the database

The final evaluation which compares the original method to the methods described in the thesis is shown in table 4.6. The same application was run, with 1 query image tested against the 15 images in the database. Total time (T) was noted on both the PC and the iPad. We see from the results obtained that we were able to achieve approximately a 40% reduction in time without any effect on the accuracy of identifying the correct asset.

Table 4.6 Comparison of implemented methods (all time in milliseconds)

Method	On PC				On iPad				% Reduction in Time
	Tk	Tx	Tm	Total time	Tk	Tx	Tm	Total time	
SURF	58	60	1597	1715	1745	680	15713	18138	
With resolution reduction	55	30	852	937	1740	360	7810	9910	45% decrease
With feature strength keypoint reduction	55	19	647	721	1735	326	5459	7520	57% decrease
With removal of closely placed keypoints	55	12	491	558	1740	285	3755	5780	67% decrease

4.6 Limitations Of The Proposed Solution

Matching performance is sensitive to the quality of the camera. If the resolution is very poor, the camera has to be moved to as close as possible to the asset to capture the image. The iPad2 resolution is very poor (1280x720 pixel) which gives rise to extremely poor images that these images cannot be used as reference images in the database.

The performance also drops when the lighting conditions of the data center are rather poor. This problem can be overcome using a camera with a good flash.

Assets whose logos are obscured with wires and cannot be noticed even normally by the human eye would be difficult to identify by the image processing application.

Figure 4.7 is an example of such a situation where the asset is almost completely obscured with wires hanging down in front of the asset.



Figure 4.7 Example of an asset obscured by wires

4.6 Evaluation Of Our Method Using The Stanford Dataset

Using the methods described above, we tested the application on a publicly available Stanford's Mobile Visual Search dataset (SMVS) which includes 8 categories of objects like CDs, DVDs, book covers, printed matters, video clips, business cards, museum paintings. The SMVS query data set has the following key characteristics that are lacking in other data sets: rigid objects, widely varying lighting conditions, perspective distortion, foreground and background clutter, realistic ground-truth reference data, and query images from heterogeneous low and high-end camera phones, (Chen et al., 2011). Samples of these images are seen in Figure 4.8.

The query images were taken with query images with different camera phones, including some digital cameras, some of which include Apple (iPhone4), Palm (Pre), Nokia (N95, N97, N900, E63, N5800, N86), Motorola (Droid), Canon (G11). The resolution of the query images varies for each camera phone. Product categories like CDs, DVDs, books were captured indoors under widely varying lighting conditions over several days.

The references are clean versions of images obtained from the product websites.

All reference images are high quality JPEG compressed color images.

The resolution of reference images varies for each category.

SMVS Examples

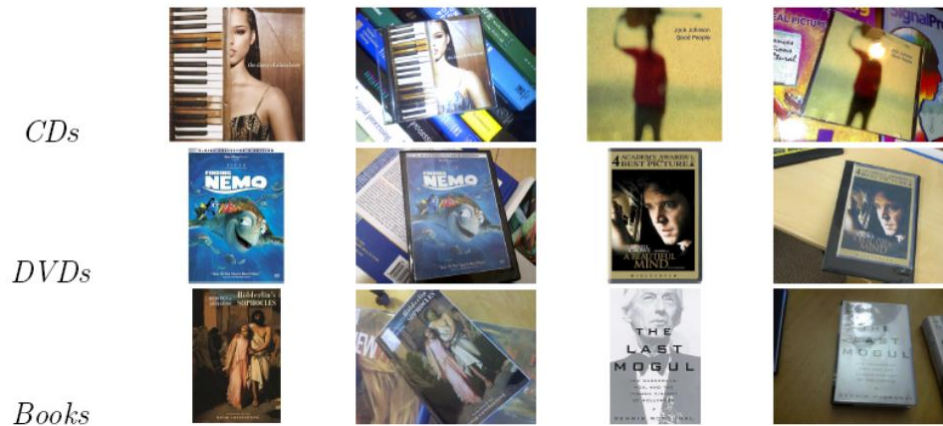


Figure 4.8 Sample images from the Stanford Mobile Visual Dataset

For our experiment we used three categories i.e. books, CD covers and DVD covers. In all three categories there were approximately 300 query images which were compared against 200 reference images.

We ran the application as a batch process for the original SURF implementation and our implementation. The results obtained were noted. It is seen, from Table 4.7 that the average search time for each image shows a significant decrease of approximately 50% for the identification of the books, CD and DVD covers.

Table 4.7 Comparison of time taken for the SMVS dataset

Method	Average Search Time (in ms)	% Time
Book Covers dataset with original SURF	37503	
Book Covers dataset with our SURF	16314	56% decrease
CD Covers dataset with original SURF	15802	
CD Covers dataset with our SURF	8071	48% decrease
DVD Covers dataset with original SURF	24359	
DVD Covers dataset with our SURF	11475	52% decrease

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 Conclusion

In this thesis we were able to provide a computer vision based approach for asset management in data centers, wherein a user can take a photo of an asset with a camera phone and the assets in the image are identified. Leveraging camera phones which are affordable and ubiquitous enables us to build a cost-effective asset identification system. Most of the challenges arise out of the poor resolution of the iPad camera, the low lighting conditions in server rooms, speed and accuracy of providing a good user experience. Common descriptors were evaluated and found that for this particular application SURF performs well both in terms of execution time as well as finding accurate matches. Since most of the mobile devices require a fast execution time, complexity reductions and optimizations in the form of reduced resolution and keypoint reduction is used.

As long as there is a small database, the images can be stored on the mobile device, and the image-retrieval algorithms can run locally. Increasing the size of the database would need to have a server and the retrieval algorithms would have to be done remotely. Database optimizations including better ways of storing the descriptors have to be considered.

Matching performance is limited by the quality of the camera on the iPad (1280x720 pixels) and better performance can be expected on mobile devices with a better camera.

5.2 Other Applications

This asset identification procedure can be extended to suit applications like identification of buildings. We were able to test this same application with a database of the FAU campus buildings. Images were captured with different mobile devices like the iPhone, Canon PowerShot A2200 and NIKON D80 cameras. The database was populated with the feature descriptors. On testing this application, it was able to successfully identify most of the campus buildings. This application of using the campus buildings would prove very useful for visitors and students who are new on campus.

Other domains where this application could be used are Healthcare organizations, manufacturing and retail industries. All these help personnel to quickly find equipment that is often moved around. Using this application we can track misplaced tools which often result in excess inventory and low productivity of specialized labor.

5.3 Future Work

Currently the user of the iPad application has to position the iPad in such a way so as to be able to capture each individual asset within a specified guideline on the iPad. A real time system with the iPad able to scan through an entire rack with the least interaction and identify an asset and overlay details of the assets on the displays is a good add on to this application and would be highly desirable. Further on we would extend it to support augmented reality and be able to run on any platform irrespective of the mobile device and tablet.

BIBLIOGRAPHY

- A.P. Psyllos, C.N. Anagnostopoulos, E. Kayafas. (June 2010). *Vehicle Logo Recognition using a SIFT-based Enhanced Matching Scheme*. IEEE Trans. on Intelligent Transportation Systems, Vol. 11, issue 2, pp. 322-328.
- B. Girod, V. Chandrasekhar, D. Chen, N.-M. Cheung, R. Grzeszczuk, Y. Reznik, G. Takacs, S. Tsai, and R. Vedantham. (July 2011). *Mobile visual search*. IEEE Signal Processing Magazine, Vol. 28, No. 4.
- D. Lowe. (1999). *Object recognition from local scale-invariant features*. In the Proceedings of the Seventh IEEE International Conference on Computer Vision, Vol. 2, pp. 1150–1157.
- D. Chen, S. S. Tsai, C. H. Hsu, K. Kim, J. P. Singh, and B. Girod. (2010a). *Building book inventories using smartphones*. In Proc. ACM Multimedia.
- D. Chen, S. S. Tsai, V. Chandrasekhar, G. Takacs, N. M. Cheung, R. Vedantham, R. Grzeszczuk and B. Girod. (2010b). *Mobile product recognition*. In Proc. ACM Multimedia 2010.
- D. Chen, S. S. Tsai, V. Chandrasekhar, G. Takacs, H. Chen, N. M. Cheung, Reznik Y., R. Vedantham, R. Grzeszczuk, Bach J. and B. Girod. (February 23–25, 2011). *The Stanford Mobile Visual Search Data Set*. In MMSys'11. San Jose, California, USA

- Google's insane number of servers visualized*. (n.d.). Retrieved from <http://gizmodo.com/5517041/googles-insane-number-of-servers-visualized>
- Google goggles*. (2011). Retrieved from <http://www.google.com/mobile/goggles/>.
- H. Bay, T. Tuytelaars, and L. J. V. Gool. (2006). *SURF: Speeded Up Robust Features*. In ECCV, Lecture Notes in Computer Science, vol. 3951. Springer, pp. 404–417.
- Ibach, P., Stantchev, V., Lederer, F., Weiss, A., Herbst, T., and Kunze, T. (2005, December). *WLAN-based asset tracking for warehouse management*. In Proc. IADIS International Conference e-Commerce, 1–8.
- ISO/MPEG. (July 2011). *Compact Descriptors for Visual Search: Call for Proposals*. MPEG output document N12201.
- Kooaba*. (2011). Retrieved from <http://www.kooaba.com/>.
- McCathie, L. and Michael, K. (2005, October). *Is it the end of barcodes in supply chain management?* In Proc. Collaborative Electronic Commerce Technology and Research Conference, LatAm, 1–19.
- Mikolajczyk, K., & Schmid, C. (2003). *A performance evaluation of local descriptors*. In International conference on computer vision & pattern recognition.
- Nokia point and find*. (2012). Retrieved from <http://pointandfind.nokia.com/>.
- N. Quoc and W. Choi. (September 2009). *A framework for recognition books on bookshelves*. In Proc. International Conference on Intelligent Computing (ICIC'09), pages 386–395, Ulsan, Korea.
- Ouertani, M. Z., Parlikad, A. K., & Mcfarlane, D. (2008). *Towards an approach to select an asset information management strategy*. International Journal of Computer Science and Applications, 5(3b), 25-44. Technomathematics Research Foundation

- Patil, A., Munson, J., Wood, D., and Cole, A. (2008). *Bluebot: Asset tracking via robotic location crawling*. Computer Communications 31(6), 1067–1077.
- R. Szeliski. *Computer Vision: Algorithms and Applications*. Springer, 2010.
- Ruf, B., & Detyniecki, M. (2009). *Identifying paintings in museum galleries using camera mobile phones*. Paper presented at Proceedings of the Singaporean French Ipal Symposium - sinfra'09, Singapore.
- Schmid, C., Mohr, R., and Bauckhage, C. (2000). *Evaluation of interest point detectors*. International Journal of Computer Vision, 37(2):151–172.
- Snaptell*. (2007-2009). Retrieved from <http://www.snaptell.com/>.
- Travis Burkhard, AJ Minich, Christopher Li. (2011). *Vehicle Logo Recognition and Classification: Feature Descriptors vs. Shape Descriptors*. EE368 FINAL PROJECT Stanford University.
- Tuytelaars, T., & Mikolajczyk, K. (2008). *Local invariant feature detectors: A survey*. In Foundations and trends. Vol. 3, No. 3, pp. 177-280.