Palestine Polytechnic University
Collage of Administrative Sciences and Informatics
Department of Information Technology

Solving the Speed problem in VLAB
"Visual Localization Aid for the Blinds"

Project team:

Malek Hashem AL Takrouri

Waseem Yousef Awwad

Supervisor:

Dr. Hashem Tamimi

A final project submitted in partial fulfillment of the requirements for the degree of B.Sc. in Information Technology

2008

Dedication

To all the honest people in these days, to the martyrs and the prisoners...

To my dear parents, my brothers, my family and my friends...

And to all who are not cited and whose names may be inadvertently not mentioned...

malek

To my beloved parents whose heart beat with every notion flashed in our mind and every single letter we wrote down

To all those who love Palestine and those who died for it or still waiting and all friends.

To the child, Huda Ghalia who lost all her family members for Palestine

waseem

Acknowledgment

We are very thankful to the supervisor Dr. Hashem Tamimi for his time, efforts and for his continuous help.

We will never forget our university, information technology department and our intelligent teachers.

Many thanks to all our friends and teachers for the times and efforts they spend in this project.

Malek and Waseem

TABLE OF CONTENTS

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF EQUTAIONS

LIST OF TABLES

Abstract

VLAB is a system designed to help the sight impaired people to determine the position within indoor environment using image processing. This system uses a mobile camera that captures images from the environment, analyze it and give a voice message for the blind person. The system has three phases: simulation, solving the speed problem and deploying the system onto a mobile device. The first phase was a simulation for the system to test if the system can compete with the other non-image processing system, the results for this phase was accurate but there was the speed problem, the system gives a result in 73 seconds, that means the system can not work in the real-time unless the speed is increased. The second phase, our project, aims to solve the speed problem without affecting the accuracy of the system, the speed problem arises from the sequential comparison between the features in the query image and the database; the proposed solution is using a KD-Tree to index the database. This has decreased the time of comparisons to less than 0.082 second per query and the accuracy was maintained.

CHAPTER ONE

INTRODUCTION

1.1 Overview

1.2 Project objectives

1.3 Preview of the system

1.4 Project hypothesis

1.5 Project problem

1.6 The testing environment

1.7 The project phases

1.8 The next chapters

1.9 Summary

1.1 Overview

One of the human responsibilities towards each others is to provide help to those who need it. A major sector of these people, are those who lost their seeing ability. Blind people are facing a lot of difficulties while moving, whether inside or outside their homes. Because of that, this project aimed to provide the blind people with aid, in order to live a normal life, and help them find their position easily.

There are many artificial sensors alternatives in the latest technology that could cover the absence of the sight of the human. None of them is as perfect as the human eyes but they can be used to give the blind person some clues about this location. For example, RFID, laser sensors, wireless waves etc. only recently cameras were introduced as a new alternative to these sensors. Cameras need high computation power, but the rapid development of a high speed processors and hardware; brought to light a new direction to work on solving this problem, which is using real-time image processing. A new system that uses image processing is preferred to provide visual localization aid for those who lost their sight because of them many advantages of the cameras over the other sensors.

This project is to propose a system that is intended for indoor environment, for example: "universities, hospitals, museums, houses and schools". The project is intended to work as follows: First an image is captured from the surrounding environment by the camera mounted on a mobile device. An image retrieval system is used to determine the closest location that contains this image. This image retrieval system is stored on the mobile device. In addition, information about the map of the environment is stored on the mobile device. The result of the image retrieval system is used to indicate the position of the blind person. Finally, a suitable voice message is played for the user.

1.2 Project objectives

The system is supposed to accomplish the following objectives:

1. *Availability*: The system must be available all the time whenever the blind user needs help.
2. *Reliability and robustness*: We will try to minimize the number of negative voice messages by avoiding the software failure and guide the user with instructions through voice messages.
3. Efficiency: We aim to maximize the efficiency of the system, by studying and optimizing the localization rate and the speed of localization process.
4. *Update and upgrade:* The system stored data which represent the map of the environment could be updated easily in order to adapt to any significant changes needed in the environment.
5. *Size of the system environment*: The system will be tested in a indoor environment of one floor in a building.
6. *Type of interaction:* The system will interact with the user through voice messages, since that blind user can not benefit from Graphical User Interface (GUI).
7. *Ease of use*: The equipments that are necessary to make this project works effectively should be easy to use and understand, regardless of the users experience and knowledge. No special equipment is needed which will minimize the cost of the project.
8. *Infra-structure:* The system does not need any kind of infra-structure, not even special markers.

1.3 Preview of the system

As mentioned in the overview, the project has two phases: in the initial phase, the system must learn the environment so a set of image are captured from the different locations and a database from the corresponding features is constructed. The database is stored on a mobile device. This phase is created before the blind person uses the system. In the second phase the blind person can use the system. Whenever he/she needs help about his/her position, the mobile device is used as follows: the mobile device captures an image and the image is converted to a set of features. The features are compared with the features in the database and the result is played to the blind person as a voice message. For the first time the system is loaded on a mobile device, after that the camera starts to capture pictures from the surrounding area, for each picture; the system converts it to a number of features, each feature is compared with the system features which are stored previously, the closest match for this feature; resembles the highest location possibility for it, then the next query feature is treated the same until comparing all the query features is finished. After that, the system counts the occurrences for each location, the highest number of occurrences will give the most possible location. These steps are repeated for each query image. After a number of images; the system returns a voice message about the location for the user. Figure 1.1 shows the context diagram of the proposed system.

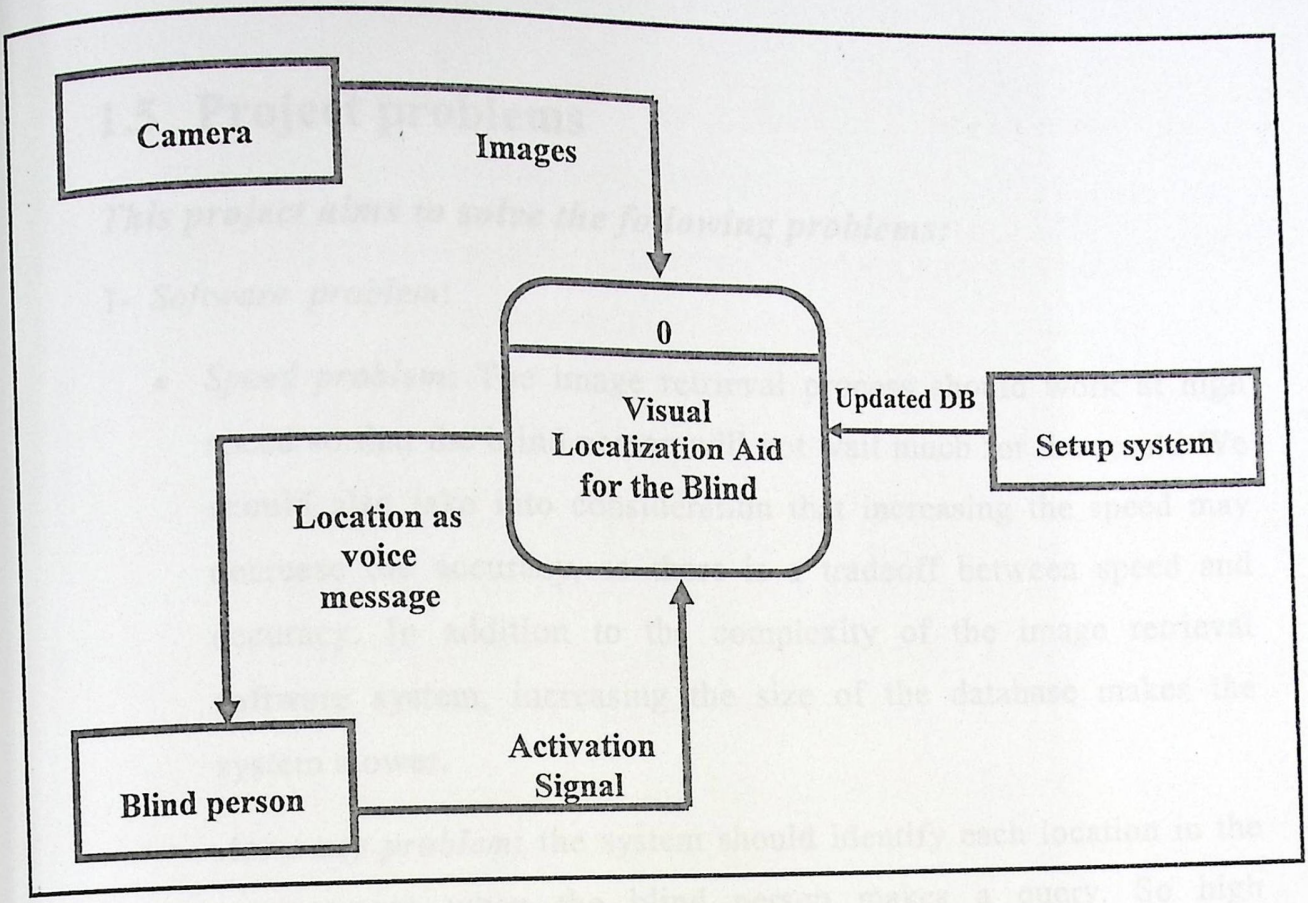


Figure 1.1: context diagram for the system.

1.4 Project hypothesis:

- Any significant update occur on the building must be followed by an update on the system database that represents the environment. For example, moving the seminar room to another location must be followed by an update on the system data that indicates its new location.
- There must be enough natural decorations to make every location different from others, in order to help the system to differentiate among these locations. For example, using only doors will confuse the system and the results will be misleading.
- All the hardware parts supposed to be functioning without any problem or lack of power.

1.5 Project problems

This project aims to solve the following problems:

1- *Software problem*:

- *Speed problem*: The image retrieval process should work at high speed so that the blind person will not wait much for the result. We should also take into consideration that increasing the speed may decrease the accuracy, so there is a tradeoff between speed and accuracy. In addition to the complexity of the image retrieval software system, increasing the size of the database makes the system slower.
- *Accuracy problem*: the system should identify each location in the environment when the blind person makes a query. So high localization rate is required. Nevertheless, precision is not highly required because human do not need to know their location precisely in terms of centimeters. In order to have accurate results, we should select an excellent feature extraction algorithm that can differentiate between the images.

2- *The location problem*:

- The accuracy of this system depends heavily on the rich environment. This means, it will be difficult for the system to decide the location when the image does not provide enough information (such as a wall, a door or an empty space).

3- *Hardware problem*:

- In order to maintain a low price; there are no sensors used other than the camera, no infrastructure is needed in the environment. Also no GPS sensors can help us because the GPS signals can not reach

indoor. This will make the work challenging because we rely only on the image to decide the location;

- During the development of the project, we will use a standard personal computer to simulate the whole system, but when the system is successful we will deploy it on a mobile device equipped with a camera. The mobile device should have the following general specifications:

  1) Speed: the speed of the processor is heavily affecting the performance of the system. Current mobile devices support a processing power up to 600MHz.

  2) Memory: enough memory to contain the data about the whole environment. Current mobiles devices reach 2GB of flash memory.

  3) Camera: the mobile device must have a camera for taking pictures for the environment. Current mobiles devices reach a resolution of 2 mega pixel.

  4) Weight: the device must be light so the user can easily carry it and move around.

  5) Ease of use: the device must not be complicated, but easy to use by the user.

  6) The project hardware requirements are based on the results that we reach, so it will be discussed in the results in Chapter 5.

4- Human computer interaction problem:

Since the graphical user interface is not suitable for this project; we will rely on voice messages only. These voice messages should be both brief and clear. The blind person may get annoyed if the voice message is

repeated many times. This means we must study the blind person attitude toward the system.

1.6 The testing environment

To simulate a real case of indoor environment, this project will be tested at Friends of Fawzi Kawash Information Technology Center of Excellence (FFKITCE)[1].

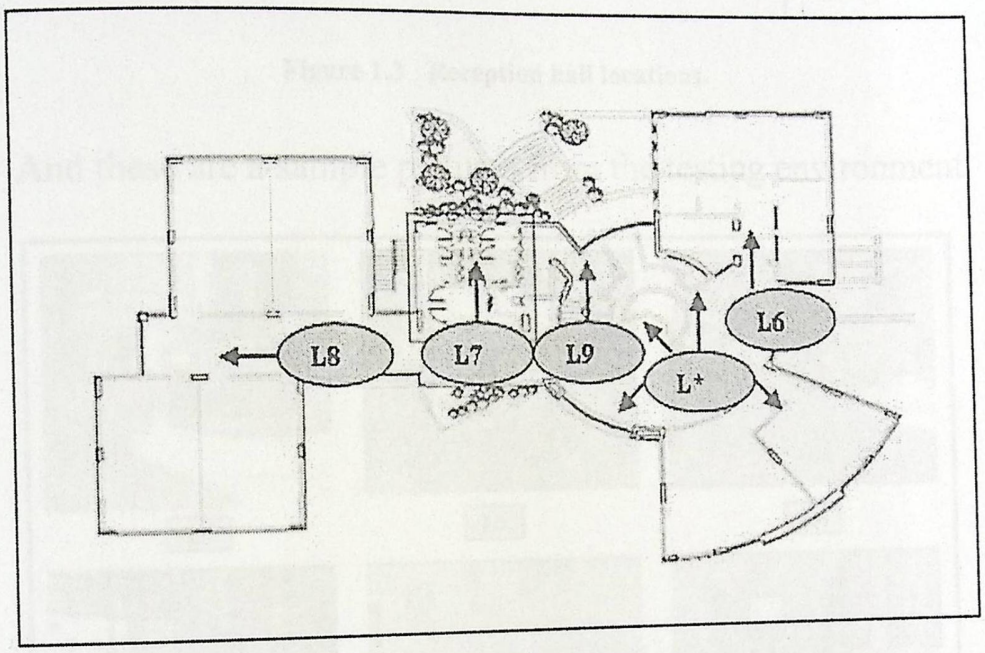


Figure 1.2: The First floor locations at FFKITCE.

Images from the first floor of The FFKITCE will be taken for experiments. Figure 1.2 and Figure 1.3 show the map of the first floor where the environment where divided into 10 important locations. Figure 1.4 shows a sample image from each location.

[1] FFKITCE is one of few purpose-built IT development-training centers in Palestine; it was established at Palestine Polytechnic University-Hebron in 2005. It aims to encourage and support innovative IT research and projects that have significant potential value, and to provide exemplary dissemination, training and support in IT for the University, the region and the country. The IT Business Incubation Service at the Center will enable people with good ideas in IT to find a place to work and develop their ideas.

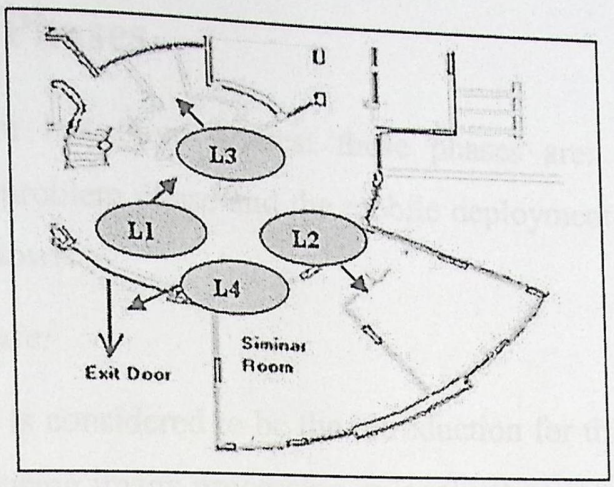


Figure 1.3 : Reception hall locations.

And these are a sample pictures from the testing environment.

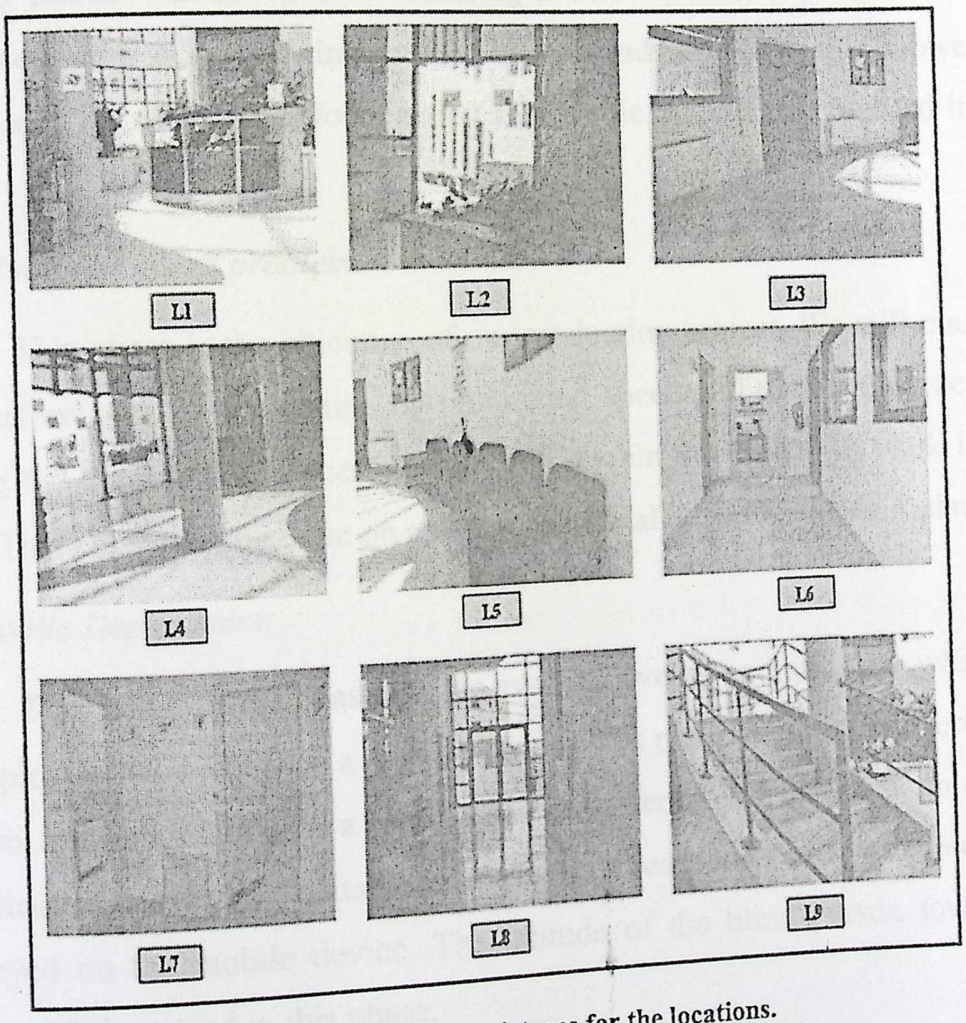


Figure 1.4 : Sample pictures for the locations.

1.7 Project Phases

This system has three phases: these phases are: simulation phase, solving the speed problem phase and the mobile deployment phase, each phase is described as follows:

1- Simulation phase:

This phase is considered to be the introduction for this system. The idea is to see whether using image processing in localization is possible and helpful or not. This phase was done during a graduation project prepared by Fatinah Sinnoqrot and Iman Younes. Good localization results were reached. (85% without markov model[1] and 100% using markov model) [1] but the average time needed for one query image was 74.7 seconds. These results showed that using image processing in localization is possible but can not be used in real-time.

2- Solving the speed problem:

This phase is the objective of our graduation project. We will maximize the performance of the system by optimizing speed and maintaining accuracy during the comparison process, which will enable the system to work in real-time. This phase will be done on a normal personal computer as a simulation.

3- Mobile Deployment:

This is the future phase for the system development process, which will be deploying the system on a mobile device. This phase includes converting the system from simulation to a mobile device system. When the system reaches this phase, it is certainly suitable to work in the real time and it just needs to be deployed on the mobile device. The attitude of the blind person toward the system will be tested in this phase.

[1] Markov model: A probabilistic approach to track the localization process. It is commonly used for robot localization.

1.8 The outline of this report

Chapter 2 will be a background and will view for the reader to introduce some important terms that are important in understanding the system, in addition to the related studies. Chapter 3 will discuss some related works in the indoor environment localization problem, and will compare between our system and these studies. Chapter 4 will talk about the methodology for developing this system and solving the speed problem that faced the system in its first phase. Chapter 5 will view the experiments and results then will discuss the results by comparing them with results found in the previous phase.

1.9 Summary

This chapter threw the light on the main motivation for this system which is helping the visually impaired people. Then, we moved to talk about the objectives which the system is supposed to accomplish. After that, the chapter mentioned the system hypothesis and explained the main problems of the system, which were divided into software, location and hardware problems. Following that, a map for the testing environment was provided with a sample from the database for the locations. Finally, a preview for the next chapters was clarified.

CHAPTER TWO

THEORETICAL BACKGROUND

2.1 Introduction

2.2 Motivation

2.3 Image retrieval

2.4 The features

2.5 Introduction to SIFT

2.6 The Indexing method

2.7 Data flow diagram

2.8 Flow chart

2.9 Summary

2.1. Introduction

This chapter will introduce new terms for the reader, because of their importance in understanding this system, namely: image retrieval system, the features, SIFT and the indexing method that is used to organize the database of features. These terms will help the reader in understanding the coming chapters.

2.2. Motivations for using image processing

First, we want to explain the motivation for using image processing for localization. What directed us toward using image processing for indoor localization was the huge information that we can obtain from the camera. It is known that image processing applications require a huge number of computational operations but nowadays there is an increase of computer power that technology achieves which makes many image processing applications possible. On the other hand, failure of other systems to work within indoor environment also affected our choice to use image processing for localization. These systems fail either because the information that it delivers is very low or because of the complexity a cost of infra-structure needed to use them.

There are other alternatives than using a camera but they all have disadvantages. We will talk about four alternatives: GPS, RFID, Bluetooth and Wireless LAN.

The global positioning system (GPS[1]) is a satellite-based navigation system made up of a network of 24 medium earth orbit satellites that transmit precise microwave signals. The system enables a GPS receiver to determine its location, speed and direction. GPS system is normally unavailable. Its signals are provided by satellites. Because of that, using GPS signals can not reach indoor. Other systems like Bluetooth and RFID also have other problems:

[1] See wikipedia.org, available at http://en.wikipedia.org/wiki/GPS

RFID[1] is an automatic identification method, relying on storing and remotely retrieving data using devices called RFID tags. Most RFID tags contain at least two parts. One is an integrated circuit for storing and processing information, modulating and demodulating an (RF) signal and can also be used for other specialized functions. The second is an antenna for receiving and transmitting the signal.

Bluetooth[2] provides a way to connect and exchange information between devices such as mobile phones, laptops, PCs, printers, digital cameras, and video game consoles over a secure, globally unlicensed short-range radio frequency.

RFID and Bluetooth require a complex and a costly infra-structure when installing the RFID tags or Bluetooth access points. In addition RFID and Bluetooth transmit signals in a short range only. Other than that, using radio waves in RFID may affect the human health.

Wireless LAN[3] (Wireless Local Area Network), utilizes modulation technology based on radio waves to enable communication between devices in a limited area, also known as the basic service set. This gives users the mobility to move around within a broad coverage area and still be connected to the network.) Wireless LAN has its own limitations such as being subject to interference.

All these reasons supported toward using image processing to build this system. These four alternatives are used by different projects. These projects are discussed in details in Chapter 3.

---

[1] See wikipedia.org, available at http://en.wikipedia.org/wiki/RFID

[2] See wikipedia.org, available at http://en.wikipedia.org/wiki/ Bluetooth

[3] See wikipedia.org, available at http://en.wikipedia.org/wiki/ Wireless_LAN

2.3. Image retrieval

An image retrieval system is a system used to search for an image from a huge database using Query by Example (QBE) [2].

This query is based on methods that can be divided into:

2.3.1. Name-based image retrieval :

Most traditional and common methods of image retrieval utilize some method of adding metadata, such as: captioning, keywords, or descriptions to the images so that retrieval can be performed over the annotation words.

2.3.2. Content-based image retrieval (CBIR)

CBIR is also known as query by image content (QBIC) and content-based visual information retrieval (CBVIR). Figure 2.1 shows the steps in CBIR systems [3].

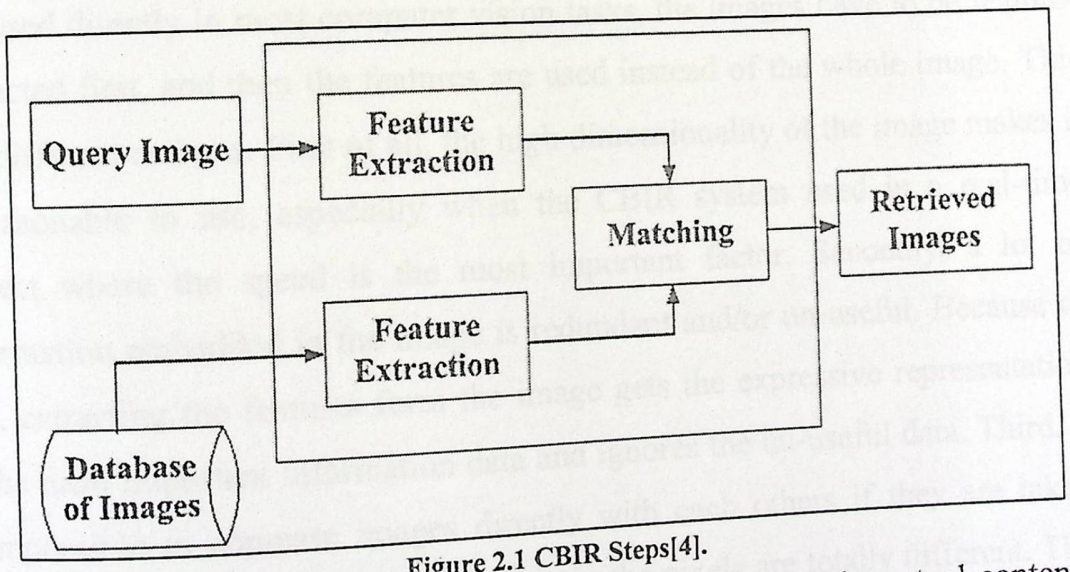


Figure 2.1 CBIR Steps[4].

"Content-based" means that the system will analyze the actual contents of the images in order to compare them with each other. The term 'content' might refer to colors, shapes, textures, or any other information that can be derived from the image itself; these contents will be called later "the features of the image".

CBIR involves two main steps:

i. ***Feature extraction***: In this step, the CBIR system converts the images into a set of values; these values can distinguish an image from other images. The feature extraction step is done for the query image on system runtime and for a set of images in the system training phase, in order to create a database for the system to search for the query image within.

ii. ***Matching***: This step in the CBIR system involves comparing the feature of the query image, with the features in the system database. The result will be the image with the highest matches with the query image. There is a lot of comparison methods used. The one that we used is called "The mean squared error" MSE. It is explained later in this chapter.

2.4. The features :

The Features are the core of the CBIR systems. The raw image data is not used directly in most computer vision tasks, the images have to be feature-extracted first, and then the features are used instead of the whole image. This has different reasons. First of all, the high dimensionality of the image makes it unreasonable to use, especially when the CBIR system used in a real-time project where the speed is the most important factor. Secondly, a lot of information embedded in the image is redundant and/or un-useful. Because of that, extracting the features form the image gets the expressive representation of the most important information data and ignores the un-useful data. Third, it is impossible to compare images directly with each others if they are taken from different locations because the images the pixels are totally different. The solution is to extract the features and the resulting representation is called the feature vector. This shall reduce the time needed in image comparison, and will lead to a higher speed and performance and will increase the accuracy for the system. [4]

2.4.1. *Invariant Features*

For these feature to be usefully used, they must remain unchanged (i.e. they should be invariant) if the image content is subjected to a transformation that changes its current situation. Transformations can be both geometric (for example: rotation or scaling) or can be photometric (like a change in lighting conditions).

Global vs. Local features:

There are two types of features [4]: global features and local features, here is a discussion of each of them:

2.3.1. *Global features:*

The first category of features is called global features. If the features are extracted from the visual content of the entire image, then these features are called global features. This means that all the regions of the image contribute in this feature, and any loss or change on the image, geometric or photometric, will affect these features. Global features have been used successfully for image retrieval. The easiest and the most famous example is the global color histogram[1].

The main advantage of global features is their speed but their problem is that the resulting description cannot differentiate between different image parts. In other words, if a new object appeared in the image, it will be different from the original image, and the system will have difficulties to match them. Therefore, the global features are usually not suitable for tasks like partial image matching and object recognition or retrieval in cluttered or complex scenes.

2.3.2. *Local features:*

In contrast to global features, local features are extracted from regions of interest or objects in the image. This means that any transformation on the

[1] It is a vector that shows the number of occurrences for each color in the image.

image will not affect the entire image. Even though, introducing new objects in the image will affect only some of the image features, not all the features. This gives the features the attribute that is called "robust to occlusion". Being robust to occlusion means if there is an image with some features, and some of these features are hidden or missing, the other features can differentiate this image from any other image. Having this attribute will make using local features in partial image matching successful and effective, where global features fail.

The main problem of local features is the need of high preprocessing power to perform a number of computational processes to determine the regions of interest. The rapid development in technology nowadays guaranteed the availability of powerful hardware with high speed and performance [4].

2.5. Introduction to SIFT

Scale-invariant feature transform (or SIFT) is an algorithm in computer vision to detect and describe local features in images. The algorithm was invented by David Lowe in the year 2004[5].

SIFT transforms the image data into scale-invariant coordinates relative to local features. An important aspect of this approach is that it generates large numbers of features. A typical image of size 500x500 pixels will give rise to about 2000 features, although this number depends on the image content.

The local features are based on the appearance of the object at particular interest points. These features are invariant to image scale and rotation and robust to changes in illumination. They are also robust to noise, occlusion and minor changes in viewpoint.

3.5.1. SIFT Stages

SIFT uses local features and call them keypoints. Following are the major stages of computation used to generate the set of image features:

1. Scale-space extrema detection:

2. Keypoint localization.

3. Orientation assignment.

4. Keypoint descriptor.

These stages are explained in [5] and we mention them here just for completion.

3.5.2. *Features comparison*

There are a number of comparison methods that can be used in comparing features with each others; one of these methods is the mean squared error (MSE) (see Section 3.5.2).

$$MSE = \sum_{i=0}^{l} \left(v_1[i] - v_2[i]\right)^2$$

Equation 2.1 Mean Squared Error

Where: MSE is Mean Squared Error,

l = vector length.

v_1 = query feature vector

v_2= compared feature vector

MSE measures the average of the square of the "error", the error is the amount by which the query feature differs from the compared feature in the system database [6].

3.5.3. *Robust comparison*

To maintain a robust comparison between the features, the system first compares the query with the whole database and finds the closest feature, and then it finds the second closest feature. The choice of taking this comparison as positive one is based on the following: assume the MSE (see Section 3.5.2) of the first comparison is mse_1, and the MSE for the second comparison is mse_2,

the positive match is when $mse_1/mse_2 < 0.4$. Otherwise the comparison is rejected.

Finally, the similarity between two images is based on the number of positive comparisons.

2.6. The Indexing method

The main cause for the speed problem in the previous phase (simulation phase) came from the comparison process between the query features and the features stored in the system database. The previous system database was created by storing features sequentially, and the comparison process had to compare all features in the database. This is called linear order. In the new phase, the system database is built like a tree, in order to decrease the time of comparison process, and to decrease the number of features from the database that will be compared.

The following is a comparison between the two methods:

3.6.1. Linear order

Linear order [7] is a simple way to organize the database, also known as sequential order, which is suitable for searching a set of data for a particular value.

For a large number of features this is a huge time consuming process. Because the comparison must be made with all the features in the system database; and must be repeated for every query feature. Suppose that the system data has N features and the query image has M features. We need M*N comparisons which will not enable the system to work the real time. For example; if a system has 140,000 features, and the query image has 2000 feature, then the number of comparison operations because 2,000 * 100,000 = 2 * 10^8 operations.

3.6.2. *KD-Tree*

A KD-Tree (short for k-dimensional tree) is a method for organizing points in a k-dimensional space [8]. Each point resembles a feature. KD-Tree, a binary tree data structure, is a useful method for several applications, such as searches involving a multidimensional search key. The following figure shows a sample tree of two dimensional data elements (k=2).

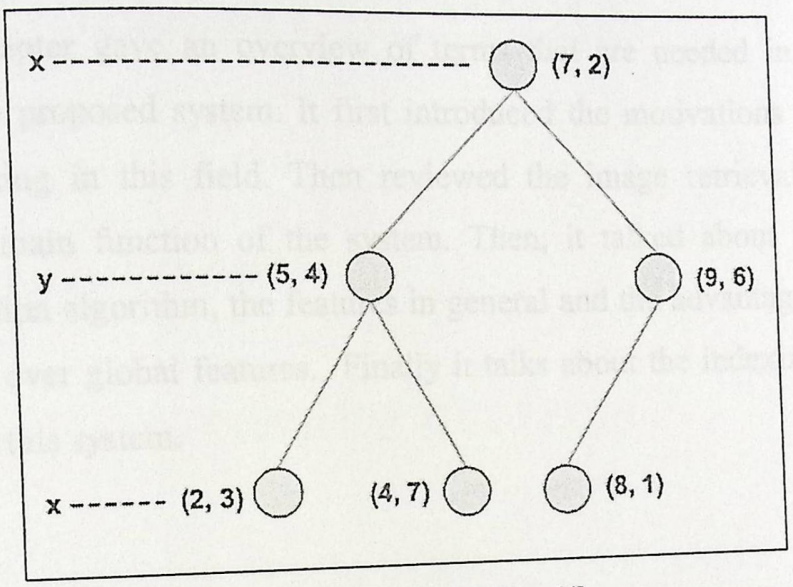


Figure 2-2 a sample of 2D-Tree [4].

The KD-Tree is used for searching when the data vector is multidimensional. The Levels of the tree are split every time on a different dimension depending on mathematical consideration. It has the advantage that's easy to build and has a simple algorithm for closest points and ranged search.

Using KD-Tree is faster than using sequential search. Using KD-Tree excludes some features that have low possibility to be close to the query feature. Because of that; there is no need to compare them and waste time. Comparing the query image with the system data using KD-Tree needs M*(logN) operations.

For example the number of operations for a system with 100,000 feature as system data and an image with 2,000 feature, will be 2000*log100000 = $1*10^4$. Compared with sequential comparison, it is $2 * 10^4$ times faster.

Since KD-Tree is much faster than the linear order method (sequential search) we will use it in our project. In each node in the KD-Tree we have a feature vector. It is important to say that, we are not searching for an exact feature in the KD-Tree, but we are look for the closest feature based on MSE (*see Equation 2.1*).

2.9 Summary

This chapter gave an overview of terms that are needed in order to understand the proposed system. It first introduced the motivations for using image processing in this field. Then reviewed the image retrieval process, which is the main function of the system. Then, it talked about SIFT, the feature extraction algorithm, the features in general and the advantage of using local features over global features. Finally it talks about the indexing method that is used in this system.

CHAPTER THREE

RELATED WORK

3.1 Introduction

The previous chapter explained some terms that are important in understanding this system. After reading Chapter 2, you should not have any problem in understanding the studies and the projects that will be viewed in this chapter, and the way the system works that will be explained later on.

This chapter will review some projects and research ideas which were initiated to deal with indoor localization problem. We categorized them into two categories: image processing-based projects and non-image processing-based projects.

3.2 Image processing-based projects

Using image processing in localization projects is a new approach in solving this problem; because using image processing requires high speed and powerful hardware. These requirements have been developed and enhanced; so using image processing is now not only possible, but also preferred, because no specific infrastructure is needed and the cost is very small. There are many researches in the literature that are based on image processing. Here are some of them:

1. **Localization using a mobile**: One of these projects was made by Nishkam Ravi et al [9]. Their system is for general purposes and not for blind localization, they used a mobile camera warn by the person, but after capturing an image, it is sent to a web server by GPRS[1] to extract its features and determine its location, then send the answer back to the user. The following figure shows the components of the system.

1 General Packet Radio Service (GPRS) is a Mobile Data Service available to users of Global System for Mobile Communications (GSM) and IS-136 mobile phones. It provides data rates from 56 up to 114 Kbps.

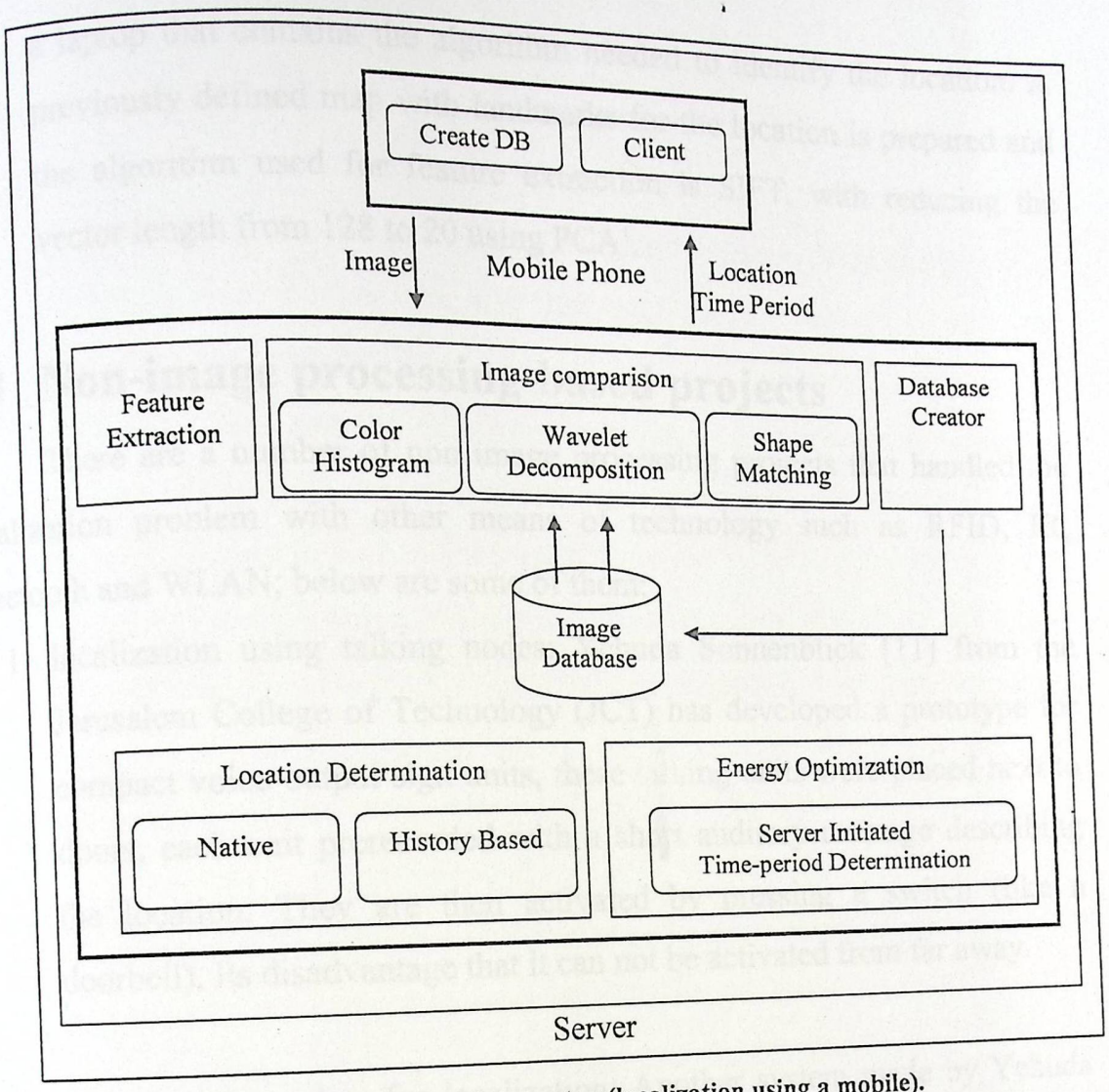


Figure 3-1 the componet of the system(localization using a mobile).

Unlike our idea of using SIFT, they use composite feature extraction algorithm such as histograms, wavelets, shape matching. The result was 90% accurate when tested on a floor of the Computer Science Department building, which includes 16 rooms, staircase, a bridge and the corridors.

The challenges that faced their system were varying lightning conditions, and the presence of moving objects in the image, which this system overcame by using a suitable feature extraction tool.

2. A seeing-wheelchair: Another project that investigated computer vision for robotic localization was made by Punarjay Chakrarty [10]; it was a motorized wheelchair with a camera mounted on the top and connected on a laptop that shows the results. The wheelchair can move from one place to another, while the camera is taking pictures and passing them to

a laptop that contains the algorithm needed to identify the location. A previously defined map with landmarks for the location is prepared and the algorithm used for feature extraction is SIFT, with reducing the vector length from 128 to 20 using PCA[1].

3.3 Non-image processing-based projects

There are a number of non image processing projects that handled the localization problem with other means of technology such as RFID, IR, Bluetooth and WLAN; below are some of them:

1. localization using talking nodes: Yehuda Sonnenblick [11] from the Jerusalem College of Technology (JCT) has developed a prototype for compact voice-output sign units, these talking units were placed next to doors, each unit prerecorded with a short auditory message describing the location. They are then activated by pressing a switch (like a doorbell). Its disadvantage that it can not be activated from far away.

2. Using access points for localization: Another system made by Yehuda Sonnenblick [10]; was a system consists of static transmitter units mounted on the ceilings along the building's corridors and a portable receiving unit for each user. The various building locations are coded into digital values, which are written into the corresponding transmitter units. These codes are continuously transmitted from the ceiling using infrared beams. The portable receiver unit senses the beam closest to it and decodes the transmitted signal into a building location value. The receiving unit also contains a speaker module, which announces the user's location to him.

3. Using tags indoors: RFID is also used in the field of localization. Myungsik Kim [12] has design a localization system using the RFID but

[1] Principle component analysis: is a technique used to reduce multidimensional data sets to lower dimensions for analysis.

is used by robots; a robot can easily identify objects through the ID code automatically transmitted form transponders that are attached to each object without additional sensors or complex operation progress. The main difficulty involved is that the spatial location information of an object for executing a task with the target object can not be acquired through the current RFID.

The location sensing RFID system using a dual directional antenna is designed. The dual direction antenna is an antenna set composed of two identical antennas perpendicularly positioned to each other having a 90 degree phase difference. It finds the direction of arrival of signals by the ratio of the signal strength between two antennas. The location of the transponder is triangulated using the directions sensed at different positions.

4. A radar system: Shashank Tadakamadla has made a paper[13] on localization using the *Received Signal Strength Indicator* (RSSI). RSSI is used to determine the quality of the communication from one node to another. By tagging objects with a special node and deploy a number of nodes at fixed position in the room, the received signal strength indicator can be used to determine the position of tagged object. This system operates by recording and processing signal strength information at multiple base stations positioned to provide information in the area of interest. It combines Euclidean distance technique with signal strength matrix obtained during offline measurement to determine the location of user.

3.4 Discussion

Both projects that use image processing; are sharing very common properties with our system. The first project uses a mobile phone as a camera and captures and image, then send it to a server where the process of localization is done, then the result is returned to the phone. In our system a mobile device is also used, and the result will be given by the same device. One difference between this system and ours is that processing is done on the mobile device and not on a middle hardware like a server, another significant difference is the algorithms used in image comparison that affects the accuracy of the system, that is this system uses three types of comparison: color histogram, wavelet decomposition and shape matching, where our system uses SIFT algorithm to generate robust and invariant features, and the mean square error is used for comparison.

The other system is made for robots not for the blinds, but it used image processing to solve the localization problem, the processing is done on a laptop not in a mobile device, and the algorithm used for feature extraction is the same: SIFT, but with an enhancement on the feature length.

The non-image processing system supported the idea by showing that they need a special and complicated infra-structure. The first and the second projects were by Yehuda Sonnenblick, the first one used a number of voice output units at every location, and this unit is activated by pressing (as a door bell). The other project mounted a number of transmitters in every location, and gave the user a hand held device worked as a receiver, the transmitters sent the signals that convey their location; and the receiver interpreted the signal and found the location.

The third non-image processing project used RFID to enable robots in finding their localization. The robot can identify the location by receiving the signals from the surrounding objects, but there is a problem when receiving multiple signals in the same location. A similar problem faced our system; that was when an image of a far location is taking by the cam, the solution for this problem is using markov model, where each location has a possibility depending on the current location.

The last system is similar to radar, there are a number of nodes spread in the locations, while a robot is moving it receives signals, by measuring the signal strength and depending on past data like the speed; the location can be determined.

3.5 Summary

This chapter has viewed some of the projects that planned to aid the visual impaired to localize their position within indoor environment using different types of technology, and from the mentioned above projects it is clear that using image processing can perform an accurate and fast localization with an advantage of the low cost and not needing a special infrastructure.

The following chapter will give a background for some terms which are related to the system environment. The chapter is supposed to help the reader to understand the system.

CHAPTER FOUR

METHODOLOGY

3.1 Introduction

3.2 The On-line system

3.3 The Setup System

3.4 Pseudo code

3.5 Summary

4.1 Introduction

This chapter goes through the details that are required to realize the system. As mentioned in Chapter 1, we have two phases in the VLAB system. First, the setup phase: where the images are collected and the features are extracted and stored with the locations in a setup database in addition to the voice messages. The blind person is not involved at this phase. Second, the on-line phase: where the blind person uses the system. In order to implement the proposed system we have proposed to make the project in two separated systems: the setup system and the on-line system. We will present the dataflow diagram and the flowcharts for each system.

The major problem that faced the VLAB project in its first testing was the speed. The speed problem came from the indexing method that used in the database. Using a sequential file as database to store the features made the localization process very slow and inefficient. This prevented the system from working in the real-time. Because of that an indexing method is suggested in order to increase the efficiency and the speed of the localization process.

4.2 The Setup System

The main objective of the Setup system is to make a map from the environment by collecting enough indoor images that represents the whole map. The features are extracted. Then the labels for the corresponding locations are stored with suitable voice messages to each location. It is obvious that this is not done by a blind person. Also when there are changes in the environment, the Setup system must be activated to update the map. When the database is ready, it is uploaded to the mobile database.

4.2.1 Data flow diagram of the Setup system

In this section we will clarify the dataflow between the elements of the system and its transitions from one operation to another, and will include

the context diagram, and level1 diagram. Figure 4.1 shows context diagram for the Setup system.

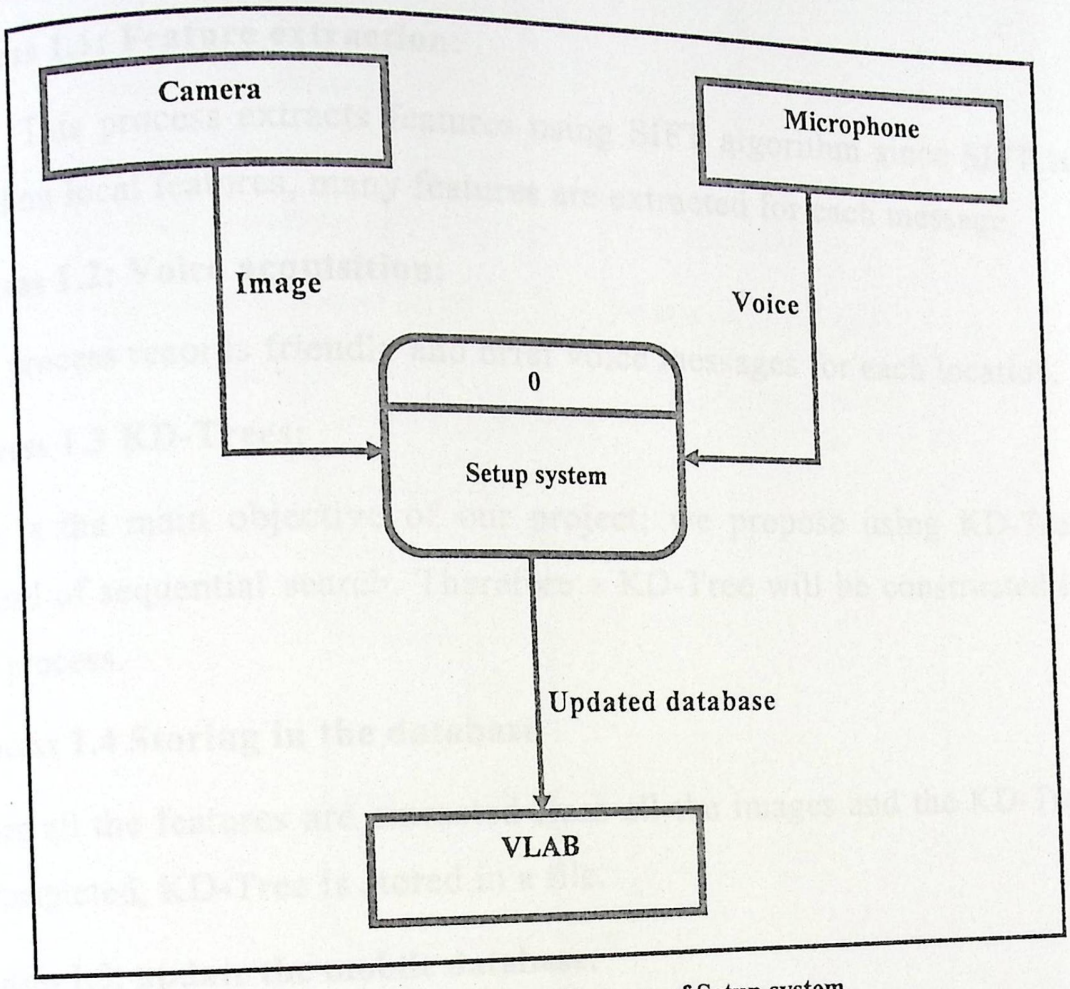


Figure 4.1: The Context diagram of Setup system.

This system is a preparatory system to the On-line system so that the work of construction of the database that will be used in the second part of the system and will also work to update the database and then send the new system comparison, which will carry out research and comparison of the images with query.

4.2.2 Level 1 for Setup system

Figure 4.2 shows level one of data flow diagram for Setup system. This system interacts with several entities. The system captures images using the camera from the indoor environment then the features are extracted. After that, voice messages are recoded through the microphone, and then each set of images that belong to a certain location are combined with the suitable voice message. The whole data is stored in the setup database.

The following is a brief description of each process in the dataflow diagram of Figure 4.2.

Process 1.1: Feature extraction:

This process extracts features using SIFT algorithm since SIFT is based on local features, many features are extracted for each message.

Process 1.2: Voice acquisition:

This process records friendly and brief voice messages for each location.

Process 1.3 KD-Trees:

This is the main objective of our project; we propose using KD-Tree instead of sequential search. Therefore a KD-Tree will be constructed in this process.

Process 1.4 Storing in the database

When all the features are extracted from all the images and the KD-Tree is completed, KD-Tree is stored in a file.

Process 1.5: update the mobile database:

This process transfers the updated database form the Setup system to the On-line system.

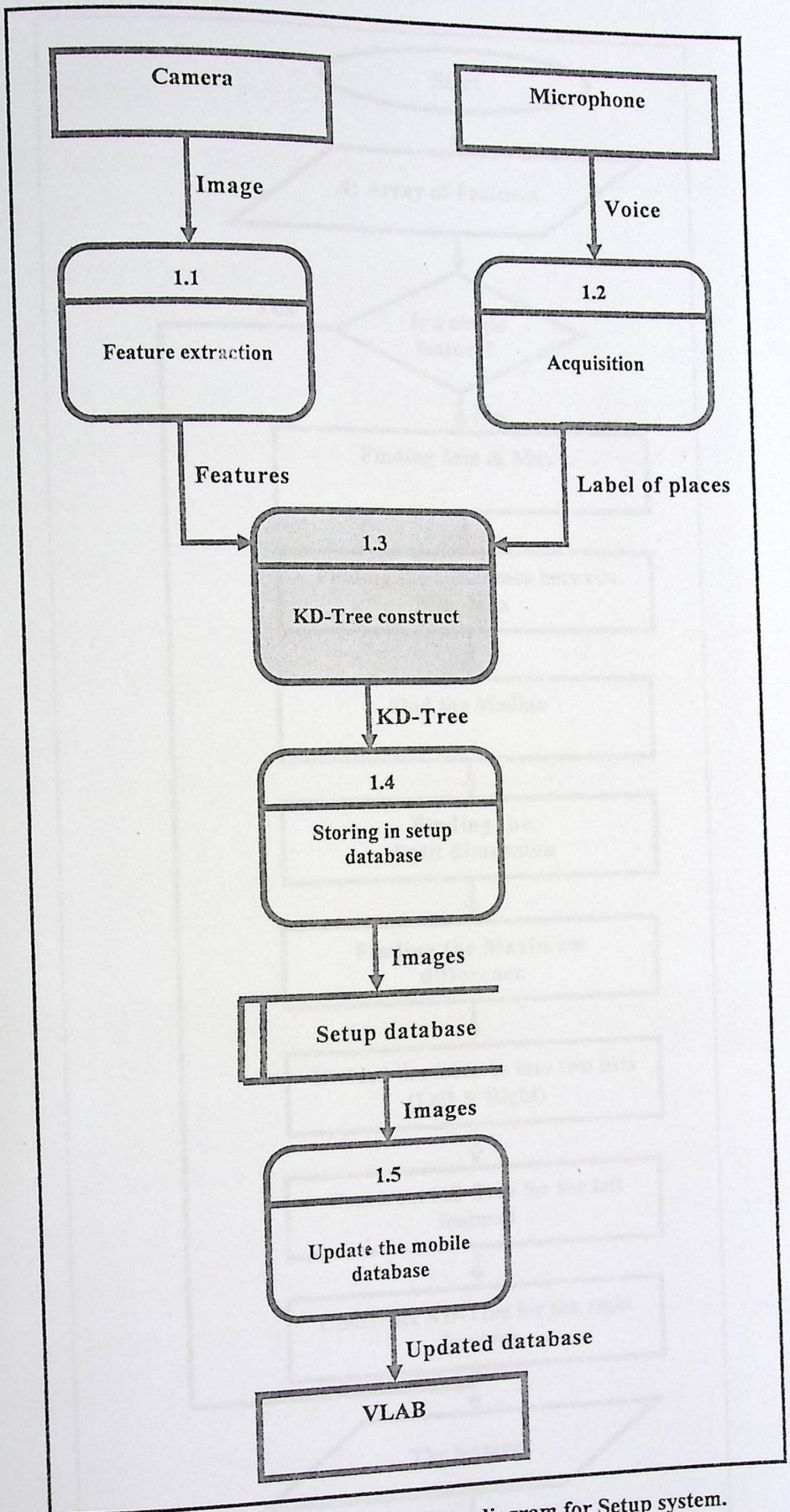


Figure 4.2: Level one of data flow diagram for Setup system. The process 1.3 is the main scope for this project.

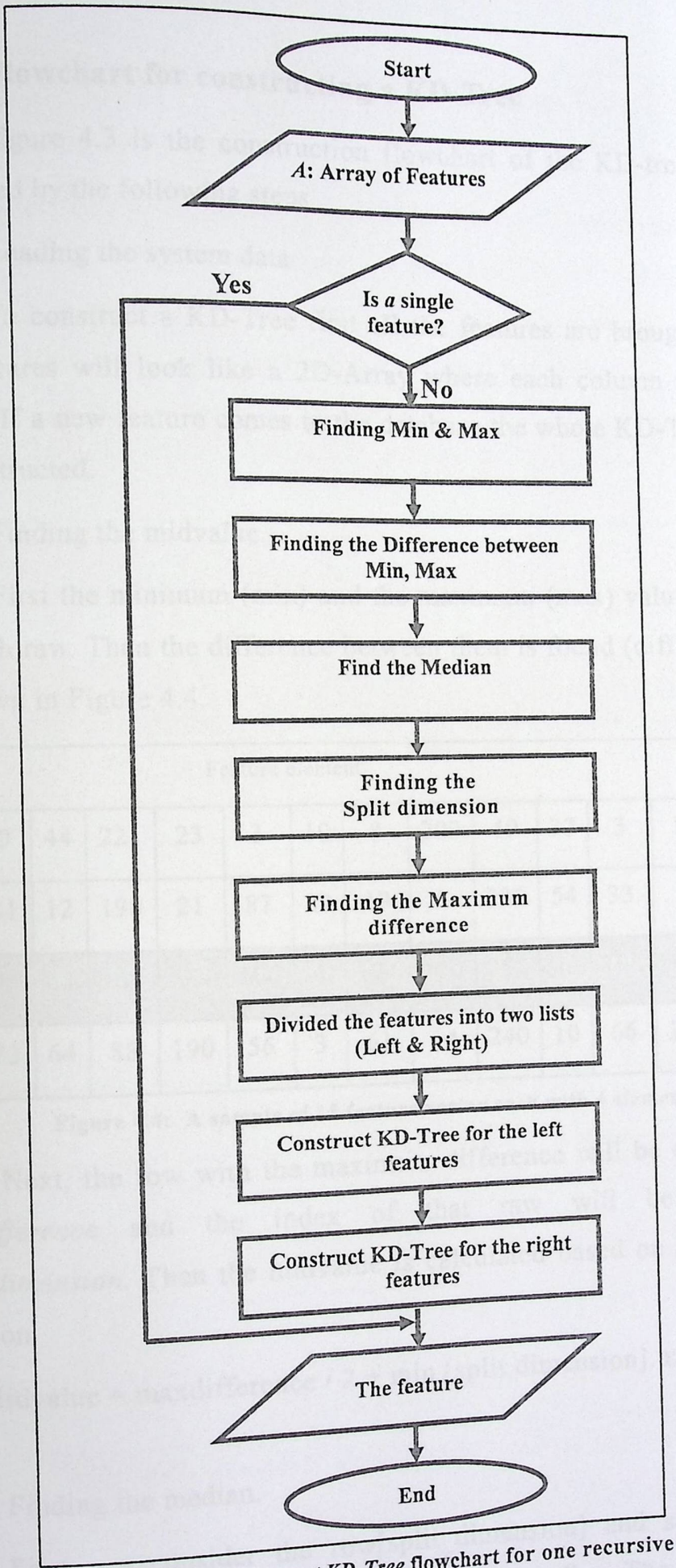


Figure 4.3: The *Construct KD-Tree* flowchart for one recursive step.

4.2.3. Flowchart for constructing a KD-Tree

Figure 4.3 is the construction flowchart of the KD-tree, which is explained by the following steps

4.4.1 Loading the system data

To construct a KD-Tree first all the features are brought together, the features will look like a 2D-Array where each column is a feature vector. If a new feature comes to the database the whole KD-Tree must be re-constructed.

4.4.2 Finding the midvalue.

First the minimum (min) and the maximum (max) values are found for each raw. Then the difference between them is found (diff = max-min) as shown in Figure 4.4.

Feature element															Min	Max	diff
7	66	1	0	44	221	23	3	10	8	200	49	37	3	17	0	221	221
1	3	56	31	12	198	21	87	22	19	8	230	54	33	1	1	230	229
2	66	0	45	44	168	53	100	4	174	150	255	6	198	66	0	255	255
8	34	22	33	64	88	190	56	3	61	64	240	10	66	123	3	240	237

Figure 4.4: A sample of 15 feature vector each with 4 elements.

Next, the row with the maximum difference will be considered as *maxdifference* and the index of that raw will be called the *split_dimension*. Then the midvalue is calculated based on the following equation

Midvalue = maxdifference / 2 + min [split dimension]. **Equation 4.1**

4.4.3 Finding the median.

First, we consider the row[split_dimension] and search for the closest value to the *midvalue,* we call this the median. The feature vector

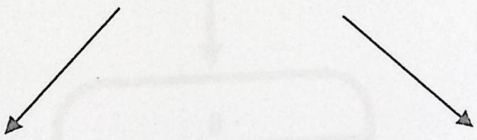
that contains the median value becomes the root of the KD-Tree.

4.4.4 Splitting the system data.

After finding the median, the whole 2D-array will be split into two new 2D-arrays, *left_array* and *right_array* as follows: For each feature vector (V), we compare *V[split_dimension]* with the median. If the result of this comparison is greater than the median, the feature is added to the *right_array,* else it is added to the *left_array.* As shown in Figure 4.5.

Feature element															Min	Max	diff
7	66	1	0	44	221	23	3	10	8	200	49	37	3	17	0	221	221
1	3	56	31	12	198	21	87	22	19	8	230	54	33	1	1	230	229
2	66	0	45	44	168	53	100	4	174	150	255	6	198	66	0	255	255
8	34	22	33	64	88	190	56	3	61	64	240	10	66	123	3	240	237

root	3	87	100	56



left_array							
7	66	1	0	44	23	10	37	17
1	3	56	31	12	21	22	54	1
2	66	0	45	44	53	4	6	66
8	34	22	33	64	190	3	10	123

right_array				
221	8	200	49	3
198	19	8	230	33
168	174	150	255	198
88	61	64	240	66

Figure 4.5: Splitting the array into left_array and right_array.

4.4.5 Recursion

The whole process is recursively repeated for left_array and right_array, until each array has only one feature.

4.3 The On-line system

As mentioned before, this system includes two sub-systems. This sub-system is the second part of the project. It is the core system which will be used by the blind person directly. The user will get the system installed on a mobile device and will start using the system.

4.3.1 Context diagram for the On-line system

Figure 4.6 shows the context diagram for the On-line system .The system interacts with two external entities that are the mobile camera, which

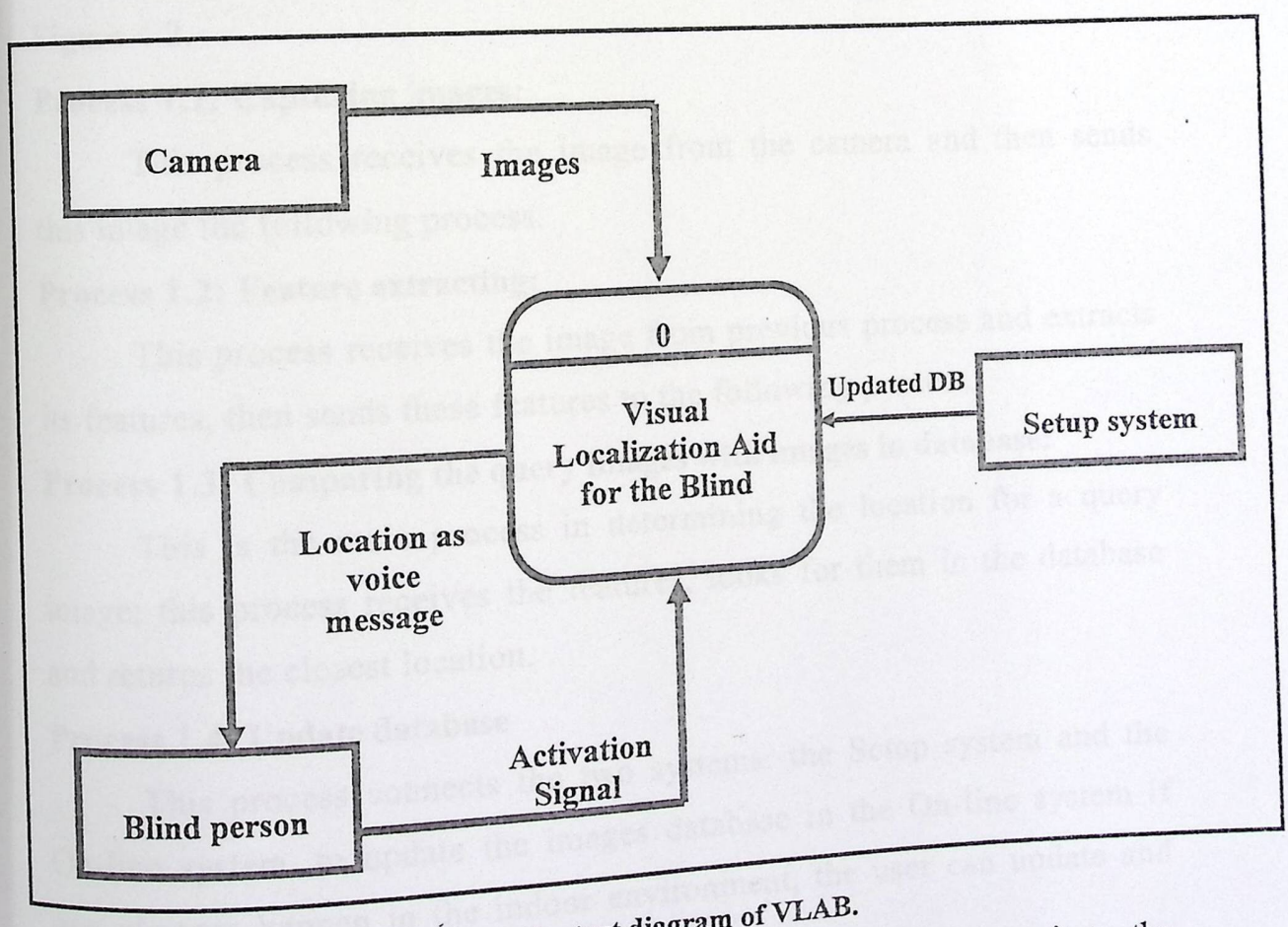


Figure 4.4: Context diagram of VLAB.

will capture images from the surrounding environments and pass it to the system. Then the image will be analyzed by the system, the second external

entity is the blind person who will deal with our system. The user will receive a voice message telling him about the location.

4.3.2 Level 1 for Setup system

Figure 4.7 shows the data flow-level one diagram, which contains the main processes, databases, and the flow of data between them. But our graduation project's objective is Process (1.3).

- The system works as follows:

The mobile camera is supposed to capture images from the surrounding environment and send the images to the system. The system in return will look for the image in the database and return the closest location as a result for this query image. Then, a suitable voice message will be chosen, and played to the user. The following is a brief description of each process in the dataflow diagram that is shown in Figure 4.2.

Process 1.1: Capturing images:

This process receives the image from the camera and then sends this image the following process.

Process 1.2: Feature extracting:

This process receives the image from previous process and extracts its features, then sends these features to the following process.

Process 1.3: Comparing the query images with images in database:

This is the main process in determining the location for a query image; this process receives the features, looks for them in the database and returns the closest location.

Process 1.4: Update database

This process connects the two systems: the Setup system and the On-line system, to update the images database in the On-line system if any changes happen in the indoor environment, the user can update and enlarge the image database.

Process 1.5: Markov model

This process receives the closest location and compares it with the previous location to see if there is it possible to move from the past location to this location or the system has missed the real location. Then returns the result if this image should be taken or ignored.

Process 1.6: Select appropriate voice message to blind person

This process receives the new location from the previous process and selects the suitable voice message to be sent to the user in order to help the blind to determine his/her location.

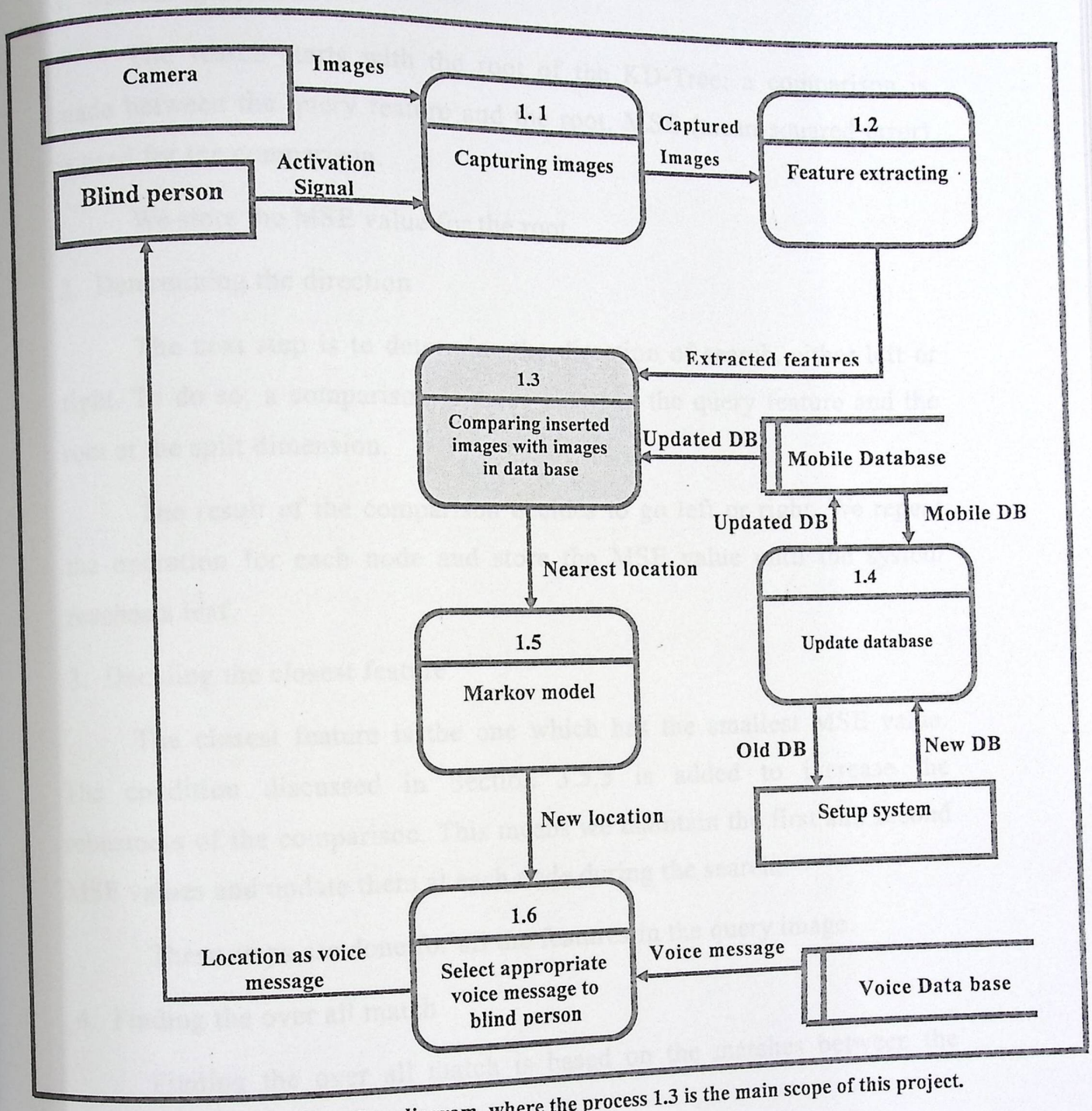


Figure 4.7: Level 1 of dataflow diagram, where the process 1.3 is the main scope of this project.

4.3.3 Flowchart for Searching a KD-Tree

- Figure 4.8 shows part(1) of the flowcharts for Searching in KD-Tree. The search and compare images made through set of steps as follows:

For a new image with a set a features, we apply the following steps for each feature:

1. Searching the root

The search starts with the root of the KD-Tree; a comparison is made between the query feature and the root. MSE (mean squared error) is used for the comparison.

We store the MSE value for the root.

2. Determining the direction

The next step is to determine the direction of search, either left or right. To do so; a comparison is made between the query feature and the root at the split dimension.

The result of the comparison decides to go left or right. We repeat the operation for each node and store the MSE value until the system reaches a leaf.

3. Deciding the closest feature

The closest feature is the one which has the smallest MSE value. The condition discussed in Section 3.5.3 is added to increase the robustness of the comparison. This means we maintain the first and second MSE values and update them at each node during the search.

These steps are done for all the features in the query image.

4. Finding the over all match

Finding the over all match is based on the matches between the features in the image and the KD-Tree.

5. Searching the root

The search starts with the root of the KD-Tree; a comparison is made between the query feature and the root. MSE (mean squared error) is used for the comparison.

We store the MSE value for the root.

6. Determining the direction

The next step is to determine the direction of search, either left or right. To do so; a comparison is made between the query feature and the root at the split dimension.

The result of the comparison decides to go left or right. We repeat the operation for each node and store the MSE value until the system reaches a leaf.

7. Deciding the closest feature

The closest feature is the one which has the smallest MSE value. The condition discussed in Section 3.5.3 is added to increase the robustness of the comparison. This means we maintain the first and second MSE values and update them at each node during the search.

These steps are done for all the features in the query image.

- Figure 4.7 shows part(2) of Flow charts for Searching in KD-Tree through set of steps as follows:

8. Finding the over all match

Finding the over all match is based on the matches between the features in the image and the KD-Tree.

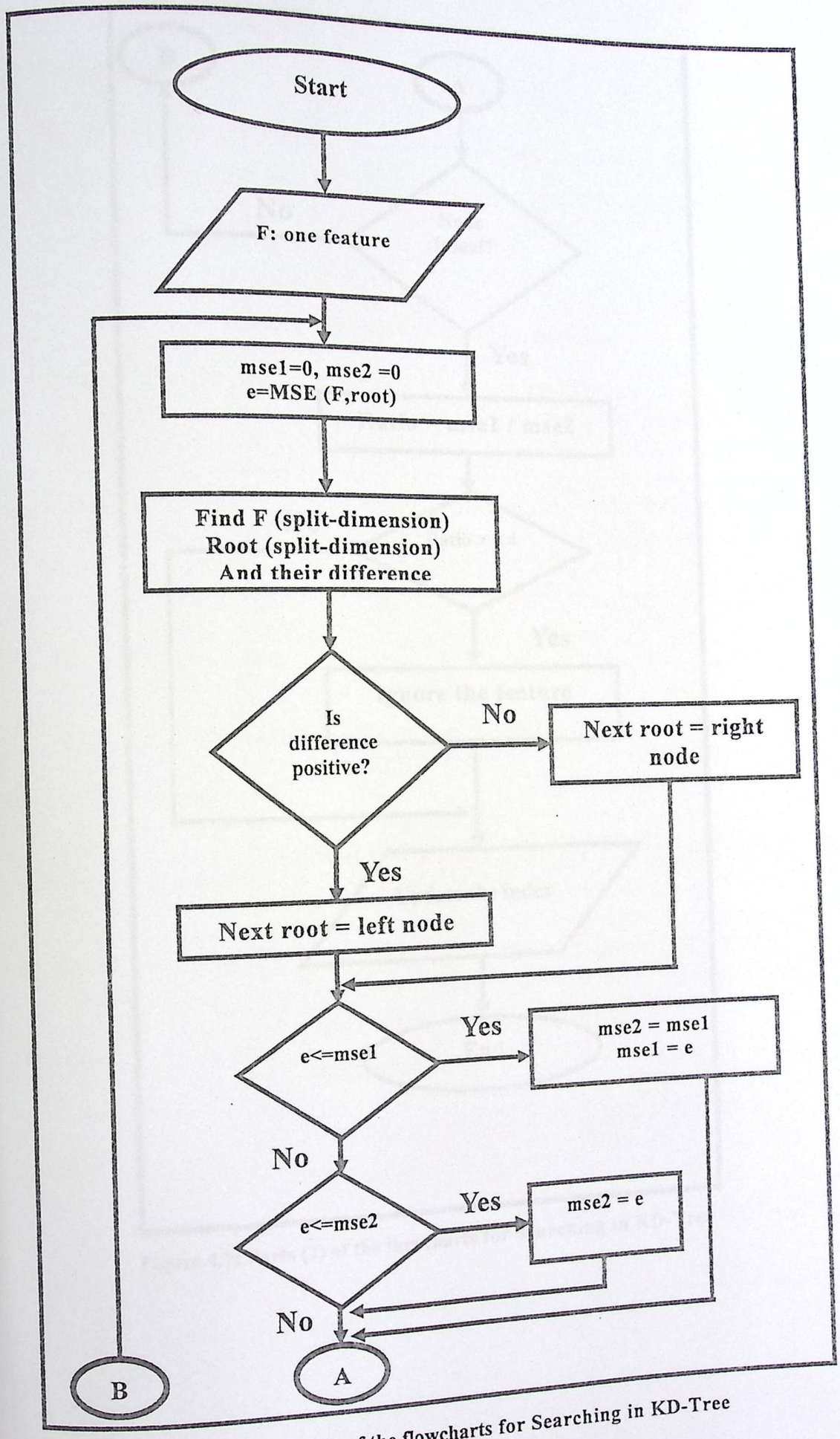


Figure 4.8: parts (1) of the flowcharts for Searching in KD-Tree

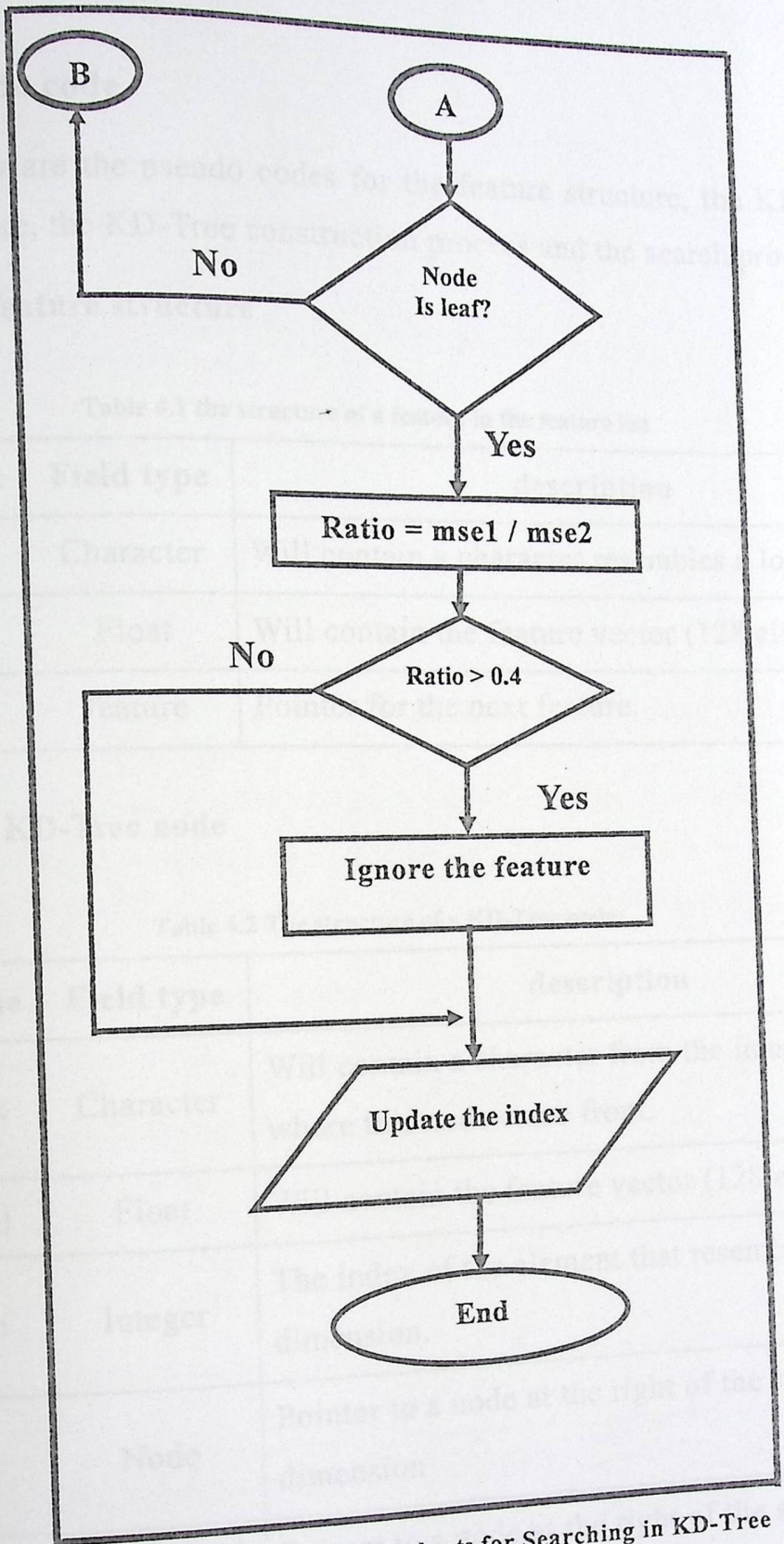


Figure 4.7: Parts (2) of the flowcharts for Searching in KD-Tree

4.5. Pseudo code

Below are the pseudo codes for the feature structure, the KD-Tree node structure, the KD-Tree construction process and the search process.

4.5.1 The feature structure

Table 4.1 the structure of a feature in the feature list

Field name	Field type	description
imgIndex	Character	Will contain a character resembles a location.
data[128]	Float	Will contain the feature vector (128 element)
next	feature	Pointer for the next feature.

4.5.2 The KD-Tree node

Table 4.2 The structure of a KD-Tree node

Field name	Field type	description
imgIndex	Character	Will contain a character from the image name where this node came from.
data[128]	Float	Will contain the feature vector (128 element)
splitDim	Integer	The index of the element that resembles the split dimension.
Left	Node	Pointer to a node at the right of the split dimension
Right	Node	Pointer to a node at the right of the split dimension

4.5.3 The construction pseudo code:

Table 4.3: The construcion process for the KD-Tree.

Input	list of features
Output	KD-Tree root node
Pre	Count (feature list) > 1
Post	None
Code	1. Find the split dimension. 2. Find the median. 3. Splitting the list of features into two new lists of features (left and right). 4. Left = recursively construct KD-Tree from list "left". 5. Right = recursively construct KD-Tree from list "right". 6. KD-Tree = <imgIndex, splitDim, data[128], left, right>

4.5.4 The searching pseudo code:

Table 4.4: The searching process for the KD-Tree.

Input	KD-Tree, query features
Output	Closest location
Pre	None
post	None
code	1. Find MSE for the root and the first feature. 2. Choose the next node to be the root (left or right) 3. Find the MSE for the new root 4. Repeat step 2 and 3 until reaching a leaf node. 5. If the smallest two MSE values close to each other ignore the feature. 6. Repeat these steps for all the features.

4.6 Summary

This chapter has introduced the KD-Tree as a theory, and showed the algorithm for building and searching for a feature in a KD-Tree.

The steps done for creating a KD-Tree are: initializing system data, finding the midvalue, median, and splitting data .This is repeated until the whole system data being organized.

The steps for searching a feature are: comparing the root feature, continue either left or right, compare the new feature, repeat this step till the tree gets to an end. This process is done for all the query features.

CHAPTER FIVE

RESULTS AND CONCLUSION

5.1 Introduction

5.2 Simulation environment

5.3 Experiments

5.4 Results and discussion

5.5 Conclusion and future work

5.1. Introduction

In this chapter we will describe the simulation environment, in order to make the experiments and results clear for the reader. Then we will talk about the experiments that were made to measure the system speed and accuracy, and the results that we got from these experiments. After that there we will be a discussion about the experiments and the results that we collected from these experiments. Finally, a conclusion based on this discussion will be viewed for "Solving the speed problem" phase of the VLAB System.

5.2. Simulation environment

In this section we will describe the environment where the simulation for the "Solving the speed problem" phase was done. There is general specification that consider the hardware and the development environment, and special specification about the data of the system. Both are explained below.

5.2.1. Hardware and software Specifications:

This simulation was done on a personal computer. The following are the important specifications of the hardware and the software that can affect the simulation process:

- Processor Speed: 3000MHz
- Memory Size: 512MB
- Operating System: Microsoft Windows XP sp2
- Programming Environment: Microsoft Visual studio .NET (2003)
  - Microsoft Visual C++ (2003)

5.2.2. Data Specifications:

- **Images:**

Resolution: 352 * 288

Color model: Grayscale 256

Format: pgm (Portable Grayscale Map)

- **Features:**

Extraction algorithm: SIFT (Scale Invariant Feature Transformation)

Feature length: 128 values

- **Database of features:**

Locations in the environment: 10 locations,

Images used: 150.

Average number of images per location: 15.

The number of images per location is not the same for all the locations because there were certain locations that needed more than 15 images to be found, and there were locations that are highly distinctive from the other locations that needed fewer images. Also, the number of features in the database varies depending on the details of the images. The time for feature extraction depends also on the image details as in Figure 5.1.

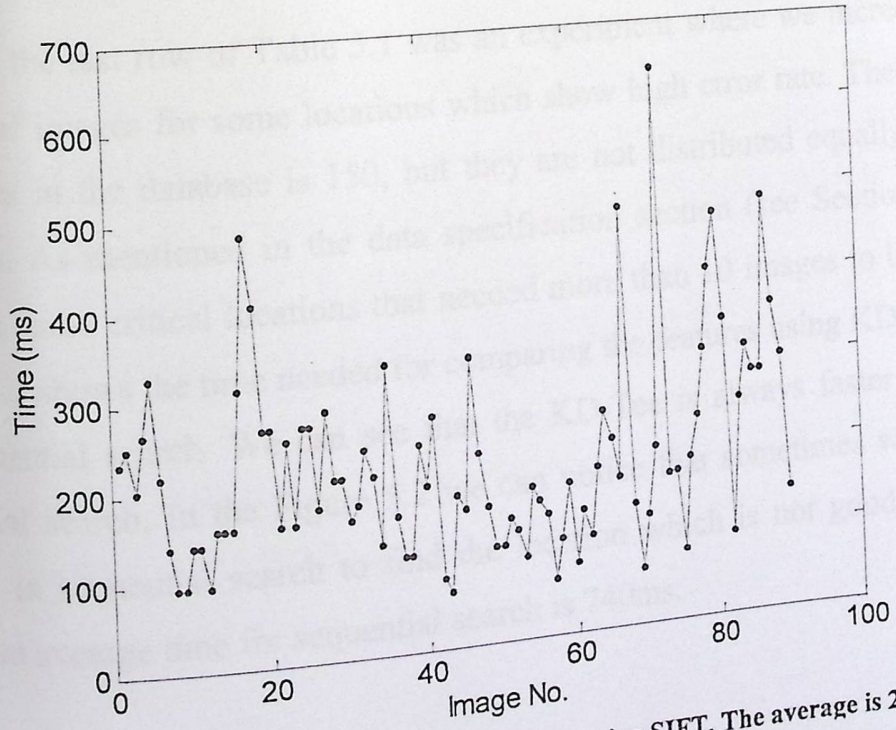


Figure 5.1: The time for feature extraction using SIFT. The average is 200ms.

- **The test images:**

The number of query images that were used in the testing was 90 query images, distributed among the locations.

5.3. Experiments

A number of experiments were made to test if the KD-Tree indexing method will decrease the needed time for localization. The main concentration was on the size of database, which will affect the size of the KD-Tree and will affect the localization time and rate. Three experiments were done using a database that contains 50,100, and 150 image. Table 5.1 shows the summary of these experiments, where we can see that the localization rate becomes better as the number of images in the database increases but also the searching time increases.

Table 5.1: Results using different databases.

No. of images in the database	Localization Rate	Searching time using KD-Tree(Average)
50	71%	1.5 ms
100	82%	22 ms
150	87%	82 ms

In the last row of Table 5.1 was an experiment where we increased the number of images for some locations which show high error rate. The number of images in the database is 150, but they are not distributed equally for the locations. As mentioned in the data specification section (see Section 5.2.2.) there are some critical locations that needed more than 10 images to be found. Figure 5.2 shows the time needed for comparing the features using KD-tree and the sequential search. We can see that the KD-Tee is always faster than the sequential search. In the Figure 5.2 we can notice that sometimes we need 2 seconds in sequential search to find the location which is not good for real-time. The average time for sequential search is 740ms.

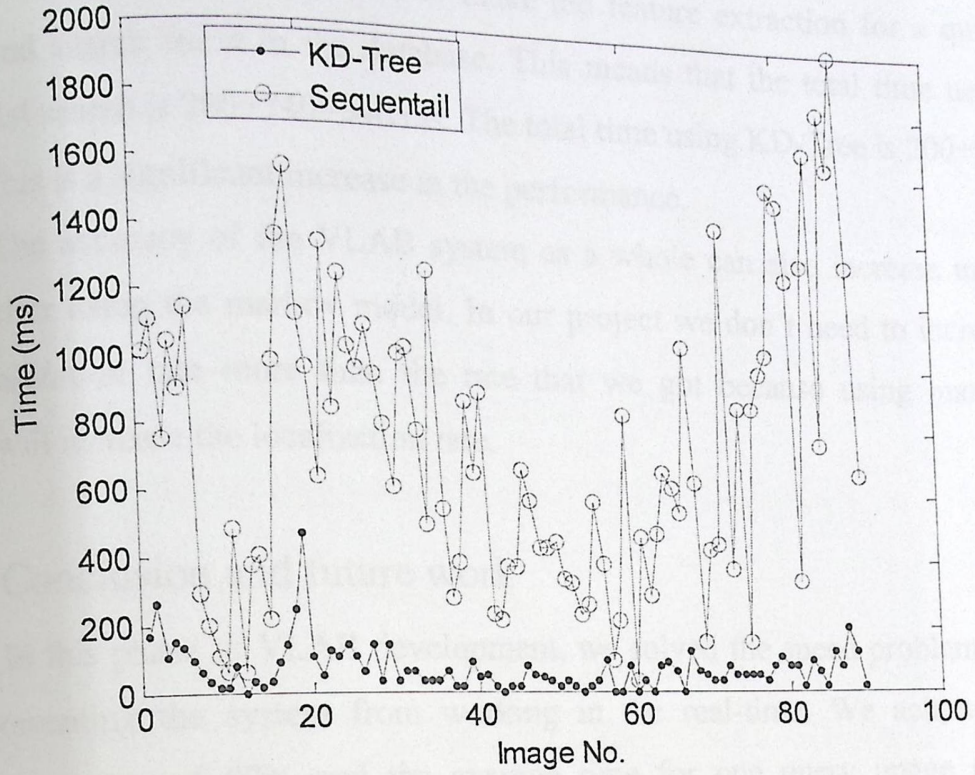


Figure 5.2: The time to compare the features in the KD-Tree and the sequential search for a database of 150 images.

5.4. Results and discussion

This section will discuss the result and compare them with the results found in the previous using phase, which depended on a sequential database to look for the features.

Using the KD-Tree as an indexing method instead of using a sequential database showed a high increase in performance and localization speed without affecting the localization rate. The results that were found in the previous phase of the VLAB were 74 second on average, and the localization rate was 85%.[1]

In our project; we have repeated the experiments using sequential search with a smaller database and found the following:

- The average feature extraction time using SIFT is 200 ms.
- The localization rate is 87%.
- The average required time for the comparison using KD-Tree is 82 ms.
- The average required time for the comparison sequentially is 740 ms.

In the localization we have to make the feature extraction for a query image and search for it in the database. This means that the total time using sequential search is 200+740=940 ms. The total time using KD-Tree is 200+84 =284. This is a significant increase in the performance.

The accuracy of the VLAB system as a whole can also increase up to 100% after using the markov model. In our project we don't need to increase the localization rate more than the rate that we got because using markov model will increase the localization rate.

5.5. Conclusion and future work

In this phase of VLAB development, we solved the speed problem that was preventing the system from working in the real-time. We achieved a localization rate of 87% and the average time for one query image to be localized was 282 ms. we are proud to say that the localization time is optimized without decreasing the localization rate. We recommend the following notes to be done in the third phase in order to deploy the system and make it ready to be used:

- The KD-Tree must be stored in away that every time the system is initialized, it will be loaded, not created.
- The two phases of VLAB: the simulation and the speed problem phase must be merged together and deployed on a mobile device.

REFERENCES

1 Visual Localization Aid for the Blind, "F. sinokrot and I. Younes", graduation project, Palestine polytechnic University, 2007.

2 Wikipedia , Free on-line encyclopedia, "Image retrieval", available at: http://en.wikipedia.org/wiki/Image_retrieval

3 Wikipedia , Free on-line encyclopedia, "CBIR", available at: http://en.wikipedia.org/wiki/CBIR

4 A. Halawani, A. Teynoe, L. Setia, G. Burnner, H. Burkhardt," Fundementals and Applications of Image Retrieval: An Overview", Datenbank-Spektrum, Heft 18, August 2006.

5 D. Lowe, "Distinctive Image Retrieval from Scale-Invariant Keypoints", International Journal of Computer Vision, 2004.

6 Wikipedia , Free on-line encyclopedia, "MSE", available at: http://en.wikipedia.org/wiki/Mean_square_error

7 Wikipedia , Free on-line encyclopedia, "Linear Search", available at: http://en.wikipedia.org/wiki/Linear_search

8 Wikipedia , Free on-line encyclopedia, "KD-Tree", available at: http://en.wikipedia.org/wiki/Kd-tree

9 N. Ravi, P. Shankar, A. Frankel, A. Elgammal and L. Iftode, "Indoor Localization Using Camera Phones", Department of Computer Science, Rutgers University, Piscataway,

10 P. Chakravarty, "Vision-based Indoor Localization of a Motorized Wheelchair", Australia, 2005.

11 Y. Sonnenblick, "An Indoor Navigation System for Blind Individuals", Jerusalem College of Technology, the 13th Annual

Conference on Technology and Persons with Disabilities, Los Angeles, California, USA, March 17-23 1998.

12 M. Kim, "RFID-based Indoor Localization Systems for Mobile Robot Applications", school of information science.

13 Shashank Tadakamadla, "Indoor Local Positioning System for ZigBee, Based On RSSI", Mid Sweden University, the Department of Information Technology and Media (ITM), 2006.