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

# Visual Localization Aid for the Blind

Project Student:
Mohammad Y. M. Al-Tamimi

Supervisor:
Dr. Hashem Tamimi

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

2008

# Contents

# Abstract

This is a computer vision based project that aims to help a blind person find her/his location in indoor environment. The blind person is equipped with a mobile phone that consists of a camera, a speaker as well as the stand-alone localization application. The map of the indoor environments is described by a set of images. The localization is performed by extracting a feature from the image of the unknown location and comparing it the features of the environment. The feature which has the nearest distance to that of the novel image is considered and a voice massage is sent the blind person with the location information. Color histogram is proposed as the image feature. The project is developed using C++ under Symbian OS. The simulation was performed in a supermarket and the results show a success rate of 98% and a cycle duration of 2 seconds.

# ملخص المشروع

هذا مشروع مبني باستخدام تقنيات الرّؤية المحوسبة، يهدف لمساعدة الشّخص الضّرير في عمليّة التّعرف على موقعه/ها داخل البيئات المغلقة، عن طريق استخدام جهاز هاتف محمول مزوّد بكاميرا وسمّاعة، إضافة إلى برنامج مستقل لتحديد المواقع. يتمّ وصف خريطة المبنى باستخدام مجموعة من الصّور الممثّلة للمواقع المختلفة، حيث يقوم نظام تحديد المواقع بالتقاط صورة موقع تواجد المستخدم، ويجري عمليّة استخراج للعناصر المميّزة للصّورة الملتقطة، ومقارنتها مع العناصر المميّزة لصور المبنى المعرّفة لإيجاد أقرب صورة معرّفة من ناحية التّفاصيل، وبناء على النّتيجة يتم إبلاغ المستخدم عن معلومات الموقع الحالي بواسطة رسالة صوتية. تمّ بناء النّظام باستخدام المدرّجات التّكرارية للألوان كسمات مميّزة للصّور، وطُورّ برنامج تحديد المواقع باستخدام لغة البرمجة "C++" للعمل تحت بيئة تشغيل "Symbian OS" المخصّص لتشغيل الهواتف المحمولة. ثم أجريت التّجربة على النّظام في أحد المراكز التجارية، وأظهرت النّتائج نسبة تحديد صحيحة للمواقع مقدارها 98%، وزمن تشغيلي يبلغ ثانيتين لكل عمليّة تحديد للموقع الحالي للمستخدم.

# Chapter 1

# Introduction

## 1.1 Overview

The people have unlimited dependency on using their senses, each sense has its functions that help the person to understand and interact with the surrounding environment, because these functions are integrated to provide the full answer to person's questions (where, what, and why), then the loss or malfunction of one or more senses causes environmental ambiguity to the person, sometimes he requires external assistance to do his activities, this assistance in the form of a relative or friend may not be available, so the person has to rely on himself, and finds that his activities are difficult to be done.

One of the common sensual problems is the optical disabilities, when taking a look on the case of blindness, for example: if we apply the previous scenario on a person who lost the ability to see, partially or totally, then we will understand how troublesome his activities become; visiting new area will not be easy, because the details of this place are not known to him, how should he decide to go in some direction if he doesn't know his current location, without a guide, the operation becomes more complex, so to solve this problem, an electronic system is needed to be built for the people in need.

This project is a contribution to the solution of the localization problem, with concentration on the indoor environments, the main task is deciding the location of the user by implementing image processing methods of comparing the current location image's data with the building's predefined images data, and telling the user about his location, the system work is achieved using lightweight mobile device that gathers the data from the images taken, performs the comparison operations with the predefined data that are stored in the device memory, and returns the results to the user, all these operations run on the device only, with no need to communicate to other systems in the surrounding.

## 1.2 Project Objectives

This project is expected to achieve the following objectives when it is completed:

- Localization Inside Indoor Environment: the system takes the image(s) from the environment as input, process it, and output the current location of the user, in high level description of the building, like mentioning the names of rooms and halls.

- User Interaction: to provide an easy system for interaction with disabled user, the input should be minimized, and the output should be available in way that the user understands easily.

- Real-Time Response: the system should provide the answer to the 'Where' question as soon as the input is entered and processed, so the user does not have to wait for a long time before getting the result, as he will decide if he is in the right place or not, which means: the mechanisms applied in the system operation should be capable of generating the output in small period of time.

- No More Hardware Is Required: the system should be installed on a lightweight mobile device that performs all the localization operations from the input of current location's images to the output of the results, with no need to communicate with any hardware system to accomplish the localization of the user.

- Hardware Resources Utilization: the mobile device to be used in the system implementation is limited in the runtime resources, and these resources should be managed properly to maintain the operation of the system perfectly, as the users will not interact with the system using graphical interfaces, which occupy large amounts of memory at runtime, and since the device processor speed is not as high as the ordinary computers, the system should be designed to meet with the capabilities of the devices to be used, with considering the operation time.

## 1.3 Project Hypothesis

The system built in this project should work correctly if these conditions are met:

- The mobile device should have no functionality or power problems that cause the application to end while the user works on.
- The indoor environment should contain distinguishable areas; each area has enough decorations that can make the comparison of its details with the available data in the memory as easy as possible, just like the most people identify their locations by recognizing the details that are related to that location.
- The test of the system will be operated on static environment that does not contain momental changes that affect the output, like hiding the area's details from being photographed, or having details that are not positioned in their original context.
- Whenever the environment details are changed in a way that makes the view of that location different from the former view, the change should be translated into a change in the data stored in the mobile device, which are compared to the input data, to have an (adaptive) system.

## 1.4 Problems to be Solved in this Project

After we have seen the objectives and the summary of the system operations, the opportunity of solving a number of problems arises, when building and running the localization system successfully, these solutions are related to problems such as:

- Hardware Complexity and Costs: until this time, many systems have been constructed to solve the localization problem, there are many ways of implementing these systems that depend on complex connections with devices and circuitry over the covered area (more details are shown in Chapter 2 of this document), while this system is capable to be run on wide variety of mobile devices that are common to be used for wireless telephony, thus having no need to buy more special and additional equipment than the mobile device.

- User Interaction With Mobile Systems: the mobile phones have limited random access memory space to run the applications on, and slower processors than the ones in personal computers, these limitations put anyone working on mobile development on facing the problems that are generated from these limitations. In this project, the dependency on the mobile's capabilities will be in the comparison stage, which is related to calculations rather than drawing and changing the user interface to meet the user inputs, as the user will not enter input data, the required resources that are needed to run the system will be small in size.

- Software Speed: as the image processing operations are ranging from fast to complex, and because of the mobile's processor and memory limitations, a good operation is the one that gives the desired output with small period of time, and with little resources consumed, the operations are described in detail in Chapter 3 in this document.

## 1.5 The VLAB Stages

The VLAB project has passed two stages before reaching this stage, the two previous stages are:

- The Introductory Stage: this stage is the first project, which imposed the idea of using the image processing in indoor localization; to perform the localization operation, it was suggested that the system should capture the current view that the user faces as an image, then apply the captured image retrieval from several known images using 'Content Based Image Retrieval'[16], in this stage the applied method of comparing the images depends on the special characteristics created from the image using the S̈cale Invariant Feature Transformälgorithm, after comparing the images characteristics (or as called İmage Features)̈, the Markov model is applied to keep the probability of selecting a location accurate by knowing the previous location, this stage resulted in good success rate of localization (85% without the Markov model, and 100% using Markov model), however, the time required for performing one localization operation is 74.7 seconds, that shows that the usage of image processing techniques to state the location in the indoor environments is possible, but its performance should be enhanced to make its time smaller, thus providing a capable system to operate in real-time.

- Solving the Speed Problem Stage[1]: this stage was devoted to perform the performance enhancement that was not accomplished in the first stage, this was done by indexing the images features that were used for the locations definition, this provided more flexible way of comparing the images features, as it does not require the system to check all the feature sequentially for the nearest one to the query feature, so the system performance has been notably increased, resulting in operating time of 284 milliseconds for one query, with maintaining the localization accuracy that was achieved in the first stage (which reached 87%), this stage made it possible to implement the system in the real-time environments, that was not implemented until this point, since the experiments being done during these stages were simulations on personal computers.

These previous stages form the background that this stage relies on, as it is the stage of implementing the image processing based localization in real-time environments, this implementation is explained throughout this document.

## 1.6 The Organization of This Document

This document is organized into the following chapters:

- Chapter 1- Introduction: this chapter introduces the purposes of working on VLAB project, mentions the problems that can be solved using this system, the objectives of this stage, the assumptions and hypothesis that guarantee the best performance and results, and the previous stages of this project.

- Chapter 2- Background: this chapter aims to provide a clear view on the related projects and researches, which are initiated to solve the base problem of indoor localization, after reviewing a number of these studies, the technique of finding the similar images using their contents is presented, and ending with general introduction about the platform that operates the runtime mobile device - 'Symbian OS' mobile phones operating system, with mentioning the Symbian OS development basic terms and information.

- Chapter 3- Methodology: in this chapter, the details of the VLAB system are explained, in terms of system general description that is followed by listing the context diagram, the data flow diagram, the localization operations flowcharts, and a look at a supportive system that is built in this project to prepare the information required for the localization application, which is called "The Setup System".

- Chapter 4- Experiments And Results: this is the portion in which the experiments and their details are listed, while providing information about the indoor environment that is the source of testing images, and the data specification of the tests, along with more sections about experiments' information and results discussion.

- Chapter 5- Conclusion: in this chapter, the results are revisited to check what has been achieved with the project objectives, followed by mentioning some work that can be done in the future to enhance the functionality of the VLAB system.

- Appendix - How to Install S60 Developer Tools: this provides a basic level of how to install and configure the development tools for the S60 SDK, that is used in building mobile applications, along with installing the computer vision library that supports VLAB system operations.

# Chapter 2

# Background

## 2.1 Overview

In this chapter, the related works that are aimed to solve the localization problem are presented, each case is explained briefly and the tools used to implement the experiments are listed to provide a comparison in terms of complexity and cost, then the talk will be about the principle used in this project: Content Based Image Retrieval 'CBIR', which enables the system to recognize the location, and the chapter ends with general information about the mobile phones operating system "Symbian OS", which is used in this project.

## 2.2 Previous Works

The localization is a problem that has great impression on the researchers, many studies suggest different ways to handle this issue with available resources and equipment, these studies use different types of equipment, and integrate them into systems used to identify the location inside various environments, that return the answer to the main question in this case (Where am I?).

The first work in this review is a system called "Chatty Environment" [4], that is aimed to provide an independent life for the blinds; as they go to the supermarkets to do their shopping, they find it difficult to get the products inside unknown parts of the store by themselves, so they depend on the employees of the supermarket to give them what they need, alternatively, the Chatty Environment provides two functionalities: the ability to know what products are nearby, and what path can the user go along to get to certain type of products, to achieve these goals, the system used electronic markers to identify certain places, which can be either Radio Frequency Identification 'RFID' tags or Berkeley Motes, each marker tag contains the identification data of itself to be sent when a user comes to the neighborhood of the tag, the user carries a mobile device that communicates with the marker tags and gets the identification data, and state the items associated with the marker tag, the output was in the form of voice messages generated by text-to-speech engine, that can be understood by the user who have no ability to read.

Another system that was designed to perform localization and object recognition for the blinds[11] consists of sensory module attached by cable to a portable computer, the user of the (Assistant System) carries the computer on his back using a backpack, and queries the system using a keyboard, the sensor contains some 3D based systems for environment exploration and stereo images input, as the data are collected, the portable computer processes the input, or the input can be sent to a server via wireless communication, the output is produced by text-to-speech engine, or as Braille portable display if the user was a blind-deaf, the comparison is operated using stereo images, orientation, and 3D model information.

The last system took in consideration the possibility of using the system by doubly-handicapped users, such a case was the interest of [10], who have built a system for guiding the blind-deaf users, the system depends on the communication between a wearable computer and one of the following types of Finger-Braille devices: the vibration motor type, or the solenoid type, these types provides output to the user in the form of Braille text that the user can interpret, the system decides the user position by using a combination of RFID tags fixed on the floor with the observation of cameras that are linked to wireless LAN, the system was designed to be used in barrier-free spaces, but it can be adapted to meet temporary occurrences of barriers existence.

More systems are engaged in providing solutions to the indoor localization problem, like the one described in [9], that uses a laser path finder and 3-axis gyroscope that are mounted on the blind's white cane, as he travels in the building, the position is found in two layered way using Extended Kalman Filter, that calculates the location based on the cane's estimate, followed by the estimate of the person's location himself by having the floor divided into landmarks that are having known positions.

Another system that uses artificial vision[5], that was implemented using stereoscopic vision, by capturing the images with two video cameras, and then sent to digital signal processing systems to be analyzed, while getting the relationship between the images by calculating the sum of related differences, the result is used to construct a disparity map, and depending on the disparity values, the output is sent as vibrations on tactile actuators, this system requires wearing a computer that performs these operations and communicates with the tactile actuators.

A semantic indoor navigation system is applied in [2], that takes into account the geometric information as equal to the semantics of building area and user profile, which are not taken in consideration by other systems, the introduction of user profiles allows the system to get benefit of information concerning user's preferred areas and paths, this system can be applied to be used by most people, like children, who don't know the details of the building they exist in, by querying the desired location using some devices, for example: Personal Digital Assistant 'PDA', the request is delivered to the (Navigation Service) that acts as intermediate between the system and the user, then the (Geometric Path Computation Service) calculates the paths from the current location to the destination, finally the (Semantic Path Selection Service) provides the path to go along by comparing the user data and his profile to existing repositories that contain the users data, the output then is viewed as the user case implies, for example: using images if the user can see the surrounding and does not understand textual instructions.

There is an application named as (Ubibus)[3], that was designed to help the blind and visually impaired in public transportation, as the user arrives to the station, he will get in a certain bus, with the possibility of accessing a PDA or Bluetooth enabled mobile phone, with input of bus number, the user can choose what bus to reserve in advance, and when the reserved bus is available, the user is informed that it is available, enabling the user to issue a stop message using speech recognition system, and get a voice message that the bus has stopped as the bus driver is informed of a person existence who have reserved a ride, thus stopping the bus to get him in, all these activities are done using spatial programming.

For guiding the blinds in outdoor environments using an integrated navigation system, Drishti[8] is initiated, it is a wireless system for guiding the pedestrians, the input is passed to a wearable computer, that gets details from Geographic Information System 'GIS' and Global Positioning System 'GPS', using wireless network, send queries that contain the environmental details (specially: dynamic obstacles), to a central server containing spatial database, which returns the result that is played as voice message, the user can interact with this system all his way, along with using it a supportive system to existing ways of navigational aids.

The last system[22] in this review is a one that uses a (RFID tags) grid to represent the indoor environment, each RFID tag is programmed to provide certain data that uniquely distinguish the current location from all other locations, the RFID tag contains the spatial coordinates along with the description of the surrounding area, with no need to use centralized databases or wireless communication infrastructure, the use of the data stored in the RFID tags is done through RFID reader, which provides the mobile devices (PDA, cellular phones, etc) with this data to produce the output.

As we have seen in the previous review, there are various ways to localize and find the navigation paths to help the blind and healthy people too, but the implementation becomes complex in most of them, in terms of equipment required to build the system infrastructure, to the size and multiplicity of data stores that hold the environmental data, along with the devices that the user interacts with, some of these devices does not provide a comfortable way to use, like carrying portable computers on back, so this system has the advantages of elimination of additional environmental infrastructure, reduction the user input, and dependence on image processing operations that represents the environmental matching that simulates the person perception, to generate the output that tells the user about his location.

In the Section 2.3, an overview about the image processing topic that is used in this project to find the location of the user inside indoor environments using environment details, captured in images.

## 2.3 Content Based Image Retrieval (CBIR)

Content Based Image Retrieval (CBIR), is "the application of computer vision to the image retrieval problem, that is, the problem of searching for digital images in large databases."

The definition above introduces the use of the images content to get the similar images, not the images description that is written due to personal point of view, since the second way is dependent on what is the description that accompanies the image, whether it contains all the objects presented in the image or not, since it is difficult to describe every single image with text, the content of the image becomes the more attractive to be studied to know what details exists, these details can be expressed in terms of expressive representation of the image's data, called (image's features).[16]

### 2.3.1 CBIR Steps

The CBIR involves two steps to produce the result:

- Feature Extraction: in which the expressive representation know as (feature vector) is built using the image's data.

- Matching: this operation involves the comparison of the extracted feature with a predefined database of features, to produce the output in terms of quantitative values, that describes degree of difference of the query image to the current database feature. When performing matching operations, the quantity that expresses the difference is the Mean Squared Error, to find this value, we have to apply the Model 2.1:

$$MSE = \frac{1}{N}\sum_{i=0}^{N}\left(F_i^A - F_i^B\right)^2 \qquad (2.1)$$

The value of MSE then is compared to the other MSE values produced in this way, the less the MSE value implies more similarity between images, with ability to arrange the features as their MSE in ascending order, the most matching features come at the beginning of the list.

The values involved in calculating the MSE for any two features are: the query feature, the second feature, and the size of the feature, the size is described in terms related to the feature nature, an overview on the type of features used in this project is at the end of this chapter.

### 2.3.2 Images Features

The images features are classified on the way they are constructed into two types:

- Global Features: this type includes the features that are constructed from the entire image, that is, to utilize every point in the image to get the characteristics, this type is easy to implement, and the most famous features built in this way are the (Global Color Histograms), or intensity distributions, that consist of elements called (bins) or (buckets) [20], the bin holds the number of images' pixels that have the color value represented by the bin, a gray image with its histogram are shown in the Figure 2.1, the histograms are widely used as features since they are robust to occlusion, but they have a limitation of not differentiating two images in terms of object or regions of interest that are viewed, this can be solved by using Coherence Color Vectors - CCV, as described in [15] to classify the pixel as coherent or not using the neighboring values, to discover if it is a part of larger region of the same range of colors or not.

- Local Features: this type of features is constructed from certain portions of interest, that are holding certain details of objects, instead of depending on the whole image to get the feature vector, the local regions of interest can be represented as standalone features, when matching the query image features with the images stored features in the database, each feature is compared as independent entity, and the image that has the suitable number of matches is considered as the

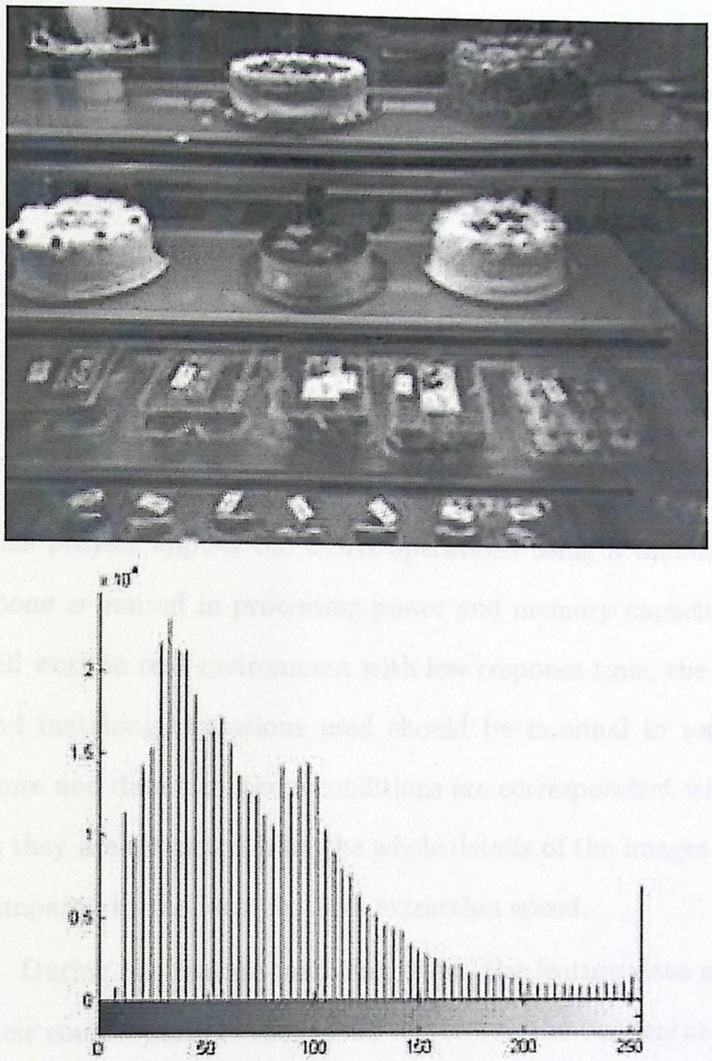

Figure 2.1: A gray scale image with its histogram, which consists of 64 bins.

matching image.

This type has the advantage of being used in partial image matching and object recognition, on the other hand, this type is more complex than the global features, since it requires additional preprocessing to state the regions of interest inside the image. An example of local features extraction algorithm is (Scale Invariant Feature Transform), that has output considered as invariant to geometric transformations, and robust to photometric transformation.

### 2.3.3 Global Color Histogram

This project applies the CBIR operations using a mobile phone, since the phone is limited in processing power and memory capacity, and the system will work in real environment with low response time, the feature extraction and matching operations used should be minimal in terms of calculation times and data size, these conditions are correspondent with global features, as they are extracted from the whole details of the images with higher speed compared to the local features extraction speed.

During the feature matching stage, the feature data are compared with their counterparts of the second feature, so the two features should have the same number of bins, the same color models that represent the source images, in the Mean Squared Error model represented in the Equation 2.1, the number of bins represents the size of the feature, and the values of the bins in the query feature are the data that are subtracted from their correspondents in the second feature, the use of this type of features in the VLAB system operations is mentioned in Chapter 3.

## 2.4 A Look at Symbian OS

"Symbian OS" is an operating system that is a proprietary for a company named "Symbian Ltd."[19], which in turn is shared among a number of electronics manufacturers: Nokia, Ericsson, Sony Ericsson, Panasonic, Siemens AG, and Samsung, it is built to run on a type of processors architecture called Advanced RISC Machine 'ARM', that are beneficial in conserving the energy while operating, causing them to be used in mobile phones, specially for the latest versions of the generations 2.5 and 3.

The Symbian OS was started in 2001 as a descendent to operating environment for the Personal Digital Assistants - PDA, called "Psion EPOC", the first version of Symbian OS was (6.0), it grew rapidly to reach the version 9.5 in 2007, each publication has a number of important features, for example: the version 9.2 (which runs the mobile device used to test this project) provides high security measures, powerful graphics, support for communication protocols [19]. It was available for every developer to write and distribute phone applications with no extra requirements, however, the release of Symbian OS v9.1 in 2005, contained new security regulations that require the developer to sign the new applications using security certificates before distribution, the Symbian company said that it is more protective and preserves the application's content, while others said that the signing of the application each time built violates the rights of the end-user [21], the stages of signing an application are mentioned in the next section.

The Symbian OS is built using C++ programming language, but it is not the only language to develop applications for the Symbian mobile phones, the development for Symbian OS is supported through the usage of special Software Development Kits - SDK, that are compliant with other programming technologies, like Java Micro Edition 'Java ME', and written to facilitate dealing with any model of Symbian OS devices.

There are two main groups for classifying the Symbian devices: S60 and UIQ, more devices may come within other groups, but they are smaller in number, these names point to the devices, or family of devices, that are compatible with respective SDKs (S60 SDK, or UIQ SDK), each model of the Symbian OS phones is capable of running the applications built with its own SDK, that is identified on the websites of the Symbian development resources, namely Forum Nokia[1], after the specification of what SDK to be used for application development for certain device, the download and installation of the SDK and its Integrated Development Environment are provided by the previous website. As the S60 SDK was used to develop this project's software, its installation steps are presented in (Appendix A) after this chapter.

## 2.5 Symbian Application Development

It is noted that the development and submission of the application to be executed on Symbian OS devices is different from the personal computing development, although it uses programming languages like C++, and totally

[1] Forum Nokia Website: *http://www.forum.nokia.com/*

follows the Object Oriented Programming methods, it is not easy for C++ developer to start writing programs for Symbian OS immediately, as there are more concepts should be covered and studied carefully about mobile devices and its respective SDK.

- The properties of the target devices: as Symbian OS operates small devices with little computational resources (specially operational memory) in comparison with personal computers, it is the developer work to maintain high level of resources utilization, in a way that decreases the execution time, to prevent wasting the user time, and maintaining the information integrity and security through granting capabilities of performing certain operations that may affect the state of device operation, such as reading or writing user data and device data, using the mobile phone's peripherals, power management, accessing local and network services, and more capabilities are listed in [12], the capabilities are required to be mentioned when signing an application before delivering it on devices to ensure that the application operation is limited within these capabilities, causing the prevention of device misuse.

- How to use the specially built functionalities for mobile devices: as achieving the needs to manage the device resources efficiently, a built-in class called "Cleanup Stack" to provide a way of handling the space loss that occurs when an object is unloaded from the memory without the de-allocation of its memory, as this problem causes the applications to halt and frequently exit without notification because of memory shortage - or as it is called "Memory Leakage" [6], the developer should be working with the Cleanup Stack when any object is dynamically allocated, this practice require some effort from the developer to cope up with, and the training on using it is essential.

- The implementation of strings in Symbian OS: it does not implement the string types that many developers are regularly using, but introduces new structure of ASCII or Unicode characters called "Descriptors" [17], this type of classes has the property of storing the character string along with a value that indicates the length of the string, thus the use of a terminating character is not needed, and it is possible to manipulate binary data using these descriptors, however, the multiple classes of the descriptors and how each class is implemented, and using them with other functionalities are troublesome for new developers, so they require more study and attention, and using them with care.

- The conventions and idioms of coding in Symbian OS SDKs are necessary to be known as they provide means to understand the classes and methods content, how they work, what may occur during a failure, the marking of user defined data with prefixes and suffixes (as needed) differentiates the dynamic classes from the resources classes, and from the data types, also it provide information about the methods operations in terms of a leave [6][17] (or Symbian exceptions) occurrence expectation, or changing the state of allocation for some object, etc..

The stages of creating a S60 mobile device application and delivering it to real-time operation are as follow:

- After finishing the program code writing, the application will be tested using Windows Emulation Application that comes with the SDK, the application is built using "Windows Simple Code Warrior" configuration to create the executable file for Windows debugging and testing, then the emulator is used to test and debug the built version, note that the emulator is designed to work as the real device, however the execution is much faster than the real-time execution, because it works using the PC resources that are greater than in capacity than the device resources, and it does not support some features that exist in the real-time, such as code-managed audio playing, and capturing using cameras, so such tests should be done using the mobile device itself, the Figure 2.2 shows the S60 SDK emulator.

- When the emulation round is finished, and to create a Symbian real executable file, the program is built again with the release configuration using GCC tool chain (that contain all the required operations to compile and link the components).

- After successfully building the application, the Symbian installation file is created using a utility contained in the SDK, the result is a file with the extension "SIS", that is ready to be signed. When signing the installation file using the services provided at Symbian Signed website[2], with providing the device unique identifier (called: International Mobile Equipment Identity - IMEI), the application is ready to be installed on the mobile device.

[2] *https://www.symbiansigned.com*

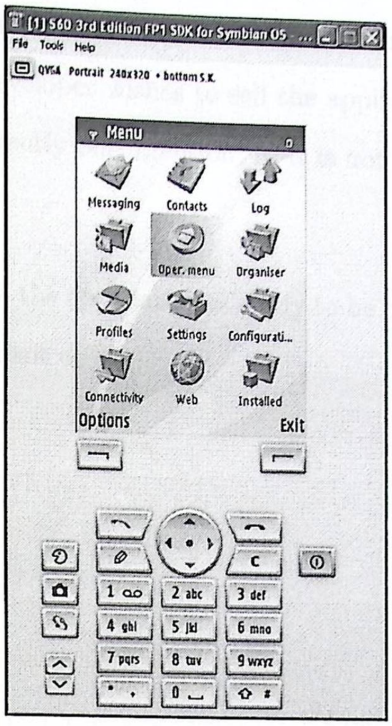


Figure 2.2: The S60 SDK emulator, displaying main menu.

- Using the software that comes with the mobile device, and after establishing a connection the PC from the device, the signed application is installed through small number of steps, and is ready to be run, until this time, the device with the specified IMEI in the signing test is the only allowed device to install the signed application, else the installation will fail because of signature inconsistency.

- When the application tests are finished, and the program is ready for release to the public users, the application is sent with enough documentation to special test houses that check the application for meeting the security regulations and predefined criteria, this stage requires the submission of more information with costs mentioned in [18], this step is required only if the developer wishes to sell the application for public use, which is not currently our intention, so it is not applied in this project.

After finishing these stages, the application is ready to be distributed and can be installed on any compliant device.

# Chapter 3

# Methodology

## 3.1 Overview

In Chapter 2, an overview on the localization projects is introduced, along with simple introduction to the Content Based Image Retrieval, as a technique of finding similar images, these information provide the basis to describe the methodology of this project, what operations will be done to achieve the localization goal, and the sequence of events that occurs to generate the results.

This chapter is divided into two parts, the first part presents the localization system details, and the second part will include the database training system, the further information about the experiments and results will be presented in the Chapter 4.

## 3.2 System General Description

The (Visual Localization Aid for the Blind - VLAB) system aims to provide a way of telling the blind or visually impaired person about his location within indoor environment, applying such a system using common wireless communication and user interaction ways is not possible, for the following reasons:

- The wireless guiding system, known as "Global Positioning System" (GPS), is available outdoor, that means, the indoor systems can not get the building details using the satellites that operates this service.
- The proposed users of the localization system are generally people who suffer from optical disabilities, that prevent them from interaction with the systems as other people, like reading information from screens, or providing complex inputs.

So the orientation is to use other ways of interaction and data collection, to provide the user with the best possible answer, while not ordering him to input data to start the operation of the system, and to provide him with a comfortable set of equipment that is available in the local market.

To achieve this vision, the VLAB project is built as an application installed on a mobile phone, the mobile phone contains a camera that is placed to take images required to be processed, there is a database file that is stored in the mobile phone memory, that contains the features of the images from all the desired parts in the building, in which the system will operate, the database features will be compared with the features extracted from the captured images at runtime, and select the best match, get its location as defined in the application, and give the user a message that tells him where he is.

## 3.3 Context Diagram

Following is the context diagram that represents the VLAB system inside the work environment.

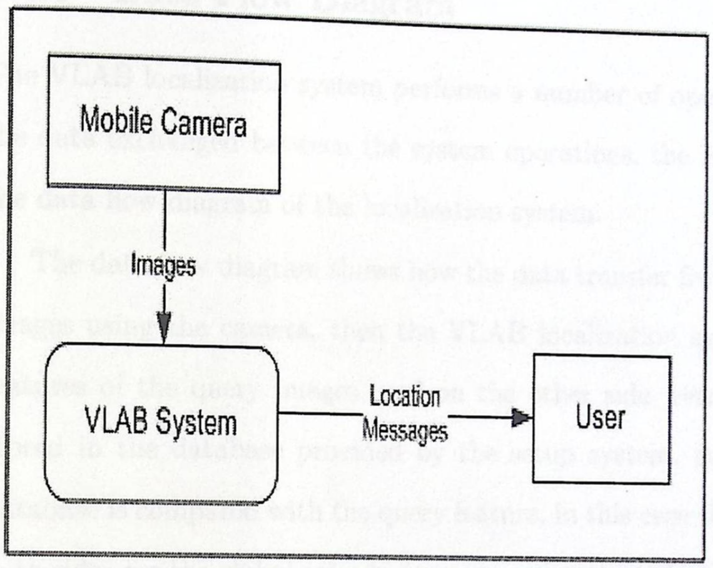


Figure 3.1: The VLAB Context Diagram.

As shown in the Figure 3.1, the mobile phone camera acts as the input side of the system, for capturing and sending images of the current location to the phone's memory, to be used in core processing operations of the CBIR as query images, after extracting their features, the comparison will be performed, and the output is sent to the user as a message, it is clear that there is no need for more circuitry to assist the system operations.

## 3.4 Localization System

This part of the chapter is dedicated for the VLAB localization application that is used by the disabled users, the following sub-sections provide information about the system operations, in terms of data flow diagrams, and operations flowcharts.

### 3.4.1 Data Flow Diagram

The VLAB localization system performs a number of operations that affect the data exchanged between the system operations, the Figure 3.2 presents the data flow diagram of the localization system.

The data flow diagram shows how the data transfer from the input of the images using the camera, then the VLAB localization system extracts the features of the query images, and on the other side, retrieves the features stored in the database provided by the setup system, each feature in the database is compared with the query feature, in this case the used features in both sides are the global color histograms, using the (RGB) color model, that represents the colors in terms of (RED, GREEN, BLUE) components, the comparison between the images results into the generation of Mean Squared Error (Equation 2.1), that is calculated from the query feature, and the current database feature, along with the number of histogram's bins.

The best match is the value that has the least Mean Squared Error between the query feature and its counterparts in the database, the best matching feature in the database is used to get the location in the environment, and select the message that is contained in the messages list, and output it

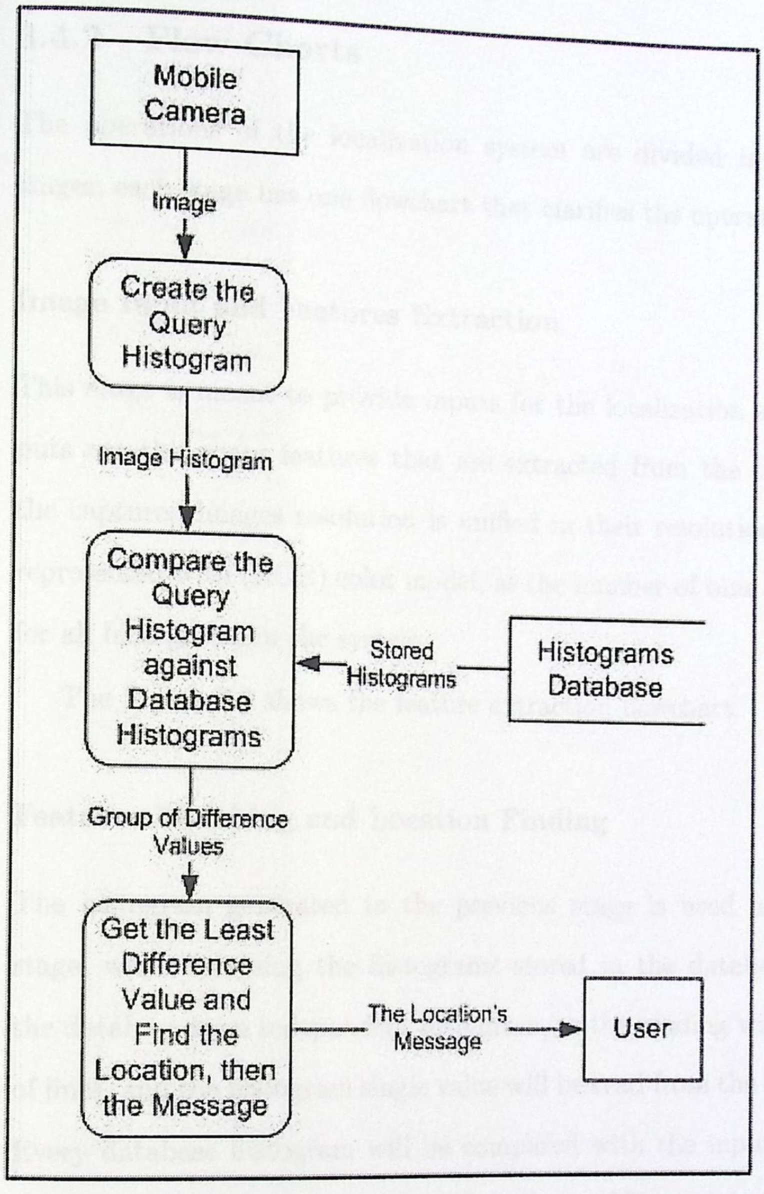


Figure 3.2: The VLAB Data Flow Diagram.

for the user.

### 3.4.2 Flow Charts

The operations of the localization system are divided into the following stages; each stage has one flowchart that clarifies the operation details:

#### Image Input and Features Extraction

This stage is meant to provide inputs for the localization system, these inputs are the query features that are extracted from the captured images, the captured images resolution is unified in their resolutions, and they are represented with (RGB) color model, as the number of bins should be stated for all histograms in the system.

The Figure 3.3 shows the feature extraction flowchart.

#### Features Matching and Location Finding

The histogram generated in the previous stage is used as input for this stage, while accessing the histograms stored in the database, each line of the database is an independent histogram, so the reading will be in the form of lines, and the histogram single value will be read from the histogram's line. Every database histogram will be compared with the input histogram and the result is written in an array of Mean Squared Errors, that has a size equal to the number of features in the database, after finishing MSE calculation, the minimal MSE is found, by getting its index in the MSE array, we can get which image is the nearest match to the query image, Figure 3.4 shows this

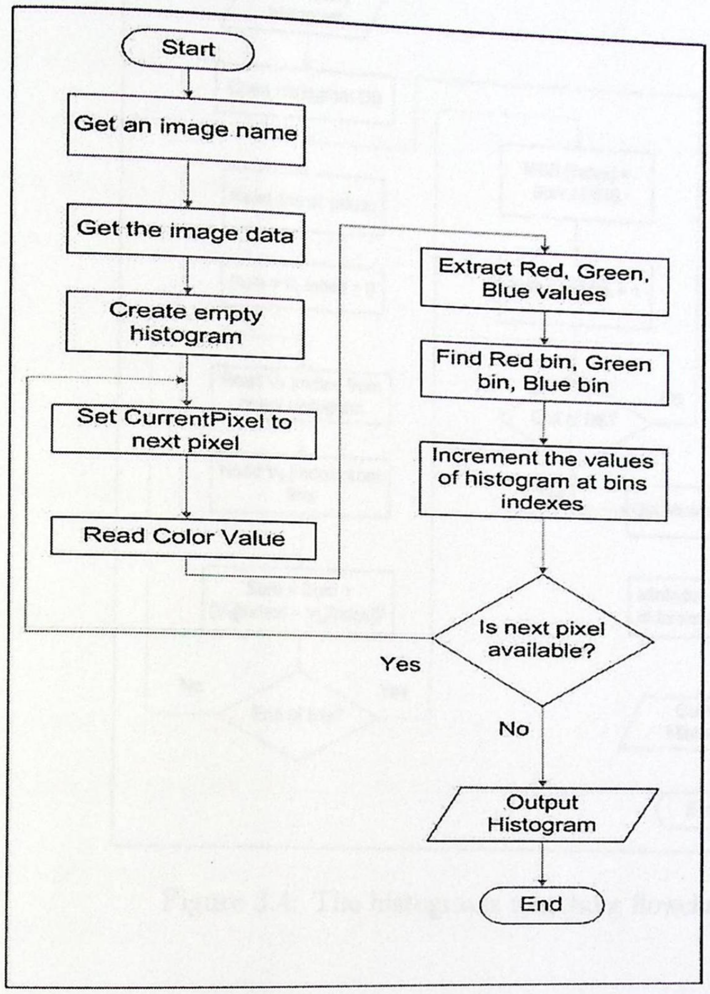


Figure 3.3: The query histogram creation flowchart.

stage's operations.

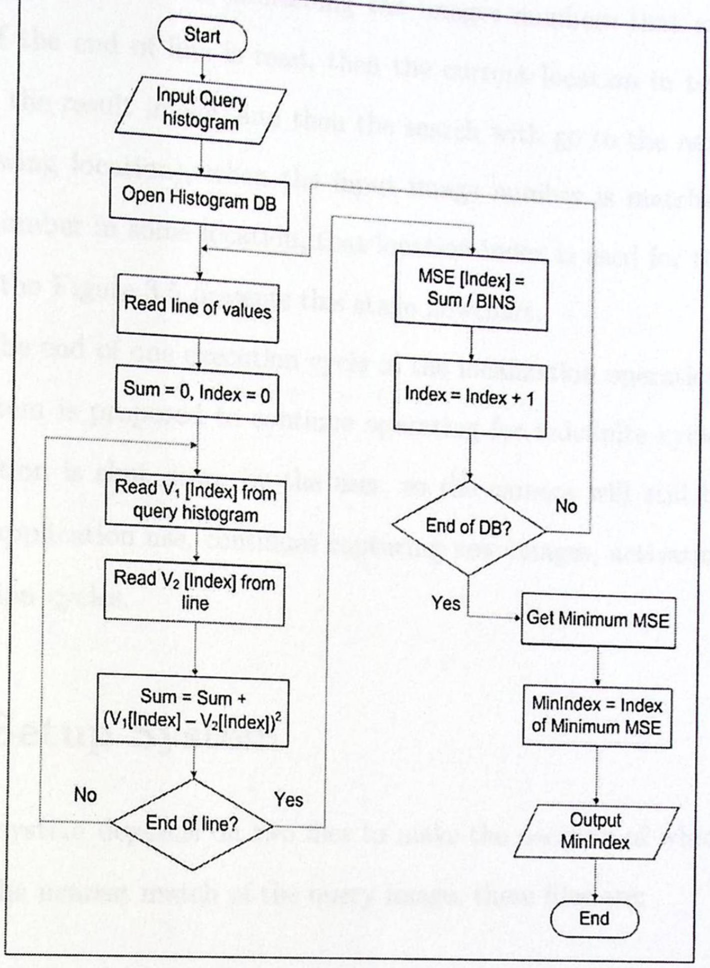


Figure 3.4: The histograms matching flowchart.

### The Output Stage

The final stage in the localization system is to translate the output (the nearest image number) in the terms of location number, and giving the result to the user; the image index that is entered in this stage is used to get the location number by reading line by line from locations images file, each line corresponds to one location, containing the images numbers that are determining it, if the end of line is read, then the current location in test does not contain the result image, and then the search with go to the next line (of the following location), when the input image number is matched with the image number in some location, that location index is used for the message output, the Figure 3.5 presents this stage flowchart.

This signals the end of one execution cycle of the localization operation, however, the system is proposed to continue operating for indefinite cycles until the application is shut down by the user, so the camera will still be reserved for the application use, continues capturing new images, activating more new execution cycles.

## 3.5 The Setup System

The localization system depends on two files to make the decision of which known image is the nearest match of the query image, these files are:

- The locations file: that contains the information about the images that identify the different locations of the environment, each location is identified by a number of images, those images are captured and numbered sequentially, then their numbers are stored in text based file, in linear format: each location has a line, and the corresponding images numbers are written in that line, with white-spaces separating between them, when new location images are entered, a new line will be created for the second location, writing its images numbers, and so on.

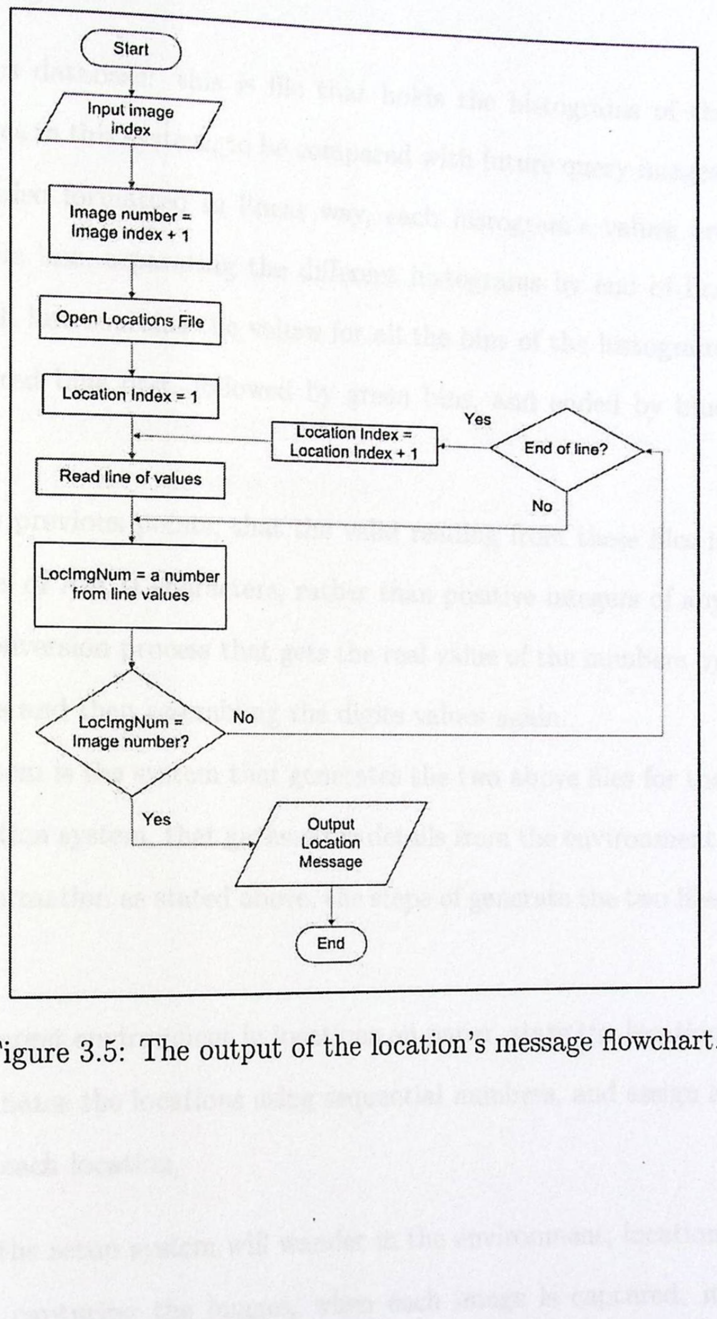


Figure 3.5: The output of the location's message flowchart.

This mechanism helps to read the locations numbers in terms of images numbers in the third stage of the localization system, when finding the needed image in specific line, that line number is used to get the output message.

- The features database: this is file that holds the histograms of the stated images in this system, to be compared with future query images, this file is also formatted in linear way, each histogram's values are stored in one line, separating the different histograms by end of line symbol, each line contains the values for all the bins of the histogram, ordered as red bins first, followed by green bins, and ended by blue bins.

A note on the previous points, that the valid reading from these files is restricted in terms of ASCII characters, rather than positive integers of any size, so there is conversion process that gets the real value of the numbers by dividing the digits and then assembling the digits values again.

The setup system is the system that generates the two above files for the use of the localization system, that gathers the details from the environment, and stores the information as stated above, the steps of generate the two files are listed below:

- Divide the target environment in locations on paper, state the location boundaries, name the locations using sequential numbers, and assign a message for each location.

- The user of the setup system will wander in the environment, location by location, capturing the images, when each image is captured, it will be stored inside the phone's images directory, the user moves these images into a special directory for the setup system application, number the images sequentially, this will enable the setup system to read the image, create its histogram, write it to the database, and write the image number into the location file at the current line.

- When the current location images are ready, the user gives the system a signal of ending the current location information gathering, and may issue a command for another location store to be created, this continues until all locations information are available.
- The setup operation ends with storing the messages in the specified folder of the localization application.

will be stored inside the phone's images directory, the user moves these images into a special directory for the setup system application, number the images sequentially, this will enable the setup system to read the image, create its histogram, write it to the database, and write the image number into the location file at the current line.

- When the current location images are ready, the user gives the system a signal of ending the current location information gathering, and may issue a command for another location store to be created, this continues until all locations information are available.

- The setup operation ends with storing the messages in the specified folder of the localization application.

# Chapter 4

# Experiments And Results

## 4.1 Overview

In Chapter 3 the methodology of the VLAB was presented, while translating these information into working system that performs tasks successfully, a good choice of runtime platform that will carry out the required operations is a crucial consideration, thus the execution of the operations is arranged to be done on a mobile phone (some advanced types are called "Smart phones"), as it is a suitable device that is small, lightweight, and capable of capturing environmental data like images.

This chapter provides basic information about the testing of the VLAB localization system, using one of the latest versions of these "Smart phones", that runs the "Symbian OS" operating system for mobile phones, after introducing the mobile device used for testing, along with other testing specifications, the testing process and results are described.

## 4.2 Testing Environment

The details of the system testing are presented in this section, in terms of hardware, software, and data are as follows:

### 4.2.1 Hardware and Software Specification

The testing process was done on a personal computer in the first stage, and on the mobile phone shown in the Figure (4.1) as the second stage.

Figure 4.1: The Nokia N95-8GB Mobile Phone.

**First Stage Specification**

- Processor: Pentium Core 2 Duo, 2GHz and 2GHz processors.
- Memory Size: 512 MB.

- Operating System: Microsoft Windows XP Service Pack 2.
- Symbian SDK: S60 3rd Edition SDK for Symbian OS, Supports Feature Pack 1, For C++.
- Integrated Development Environment: Carbide.vs v3.0.1 on Microsoft Visual Studio 2005.

This collection of hardware and software were used to ensure that the application functionalities work properly before getting to the localization testing on the phone, and there are no results that are related to this stage.

**Second Stage Specification**

This stage is the real test of the system on the mobile phone, which was chosen to be done on the following phone:

- Mobile Phone Model: Nokia 95-8G.
- Manufacturer: Nokia.
- Class: Nokia N-Series.
- Processor: ARM 11, Dual CPU, 332 MHz.
- SD-RAM Memory: 128 MB, about 90 MB is free.
- Operating System: Symbian OS v9.2.
- Main Camera: 5.0 Mega pixels sensor.[7]

### 4.2.2 Data Specification

The test of the localization system was carried out at Plaza Marketing Center[1], as a public place with rich cues and decorations, it has collections of information to be captured and used in the localization system, so the data obtained from the center concentrate on the images capture of homogenous goods and marking their area as a location, the images of a location were taken twice at different times, once afternoon and the second time at the evening, the details of the images include the same products from different views, the difference in the views resulted because:

- The current available products are changing continuously, as the people who come for shopping buy them, and since the addition of new products to the existing ones, this results in unavailability of some products in some collection of location images.
- The locations images capture process include moving the camera to obtain as much details as possible, by moving left and right, forward and backward, at different times of the day since the illumination changes, all these effects result in a comprehensive collection of images, which details cover the maximum possible aspects of the location cues.

Following are the data specifications that were used in testing the system on the mobile phone:

[1]Plaza Marketing Center - also known as "Plaza Mall", is a branch of the marketing centers owned by the Arab Palestinian Company for Marketing Centers, a subsidiary of the Arab Palestinian Investment Company - APIC, it was opened in Hebron in the 8th of August 2007, and is located in Ein-Sarah street.

- Images:

  Resolution: 320*240.

  Color model: RGB and Grayscale-256.

  Format: Image object in the main memory during application run-time.

- Database of features:

  Number of locations: 16 locations.

  Number of Images: 1307 images.

  Average number of images per location: 81 images per location.

The average number of images does not present that each location has 81 images, since there are various location's size and appearances, that may require more images than this average, or has enough details to be grouped into smaller number of images, the table tab:LocImgs provides the locations with number of images in that location:

The Figures 4.2 and 4.3 present sample images from the locations mentioned above.

## 4.3 Experiments

The following tests were performed to find the most suitable image details and feature length:

- Different feature lengths were applied to the database and the query images, the chosen color model was RGB, and the length of features

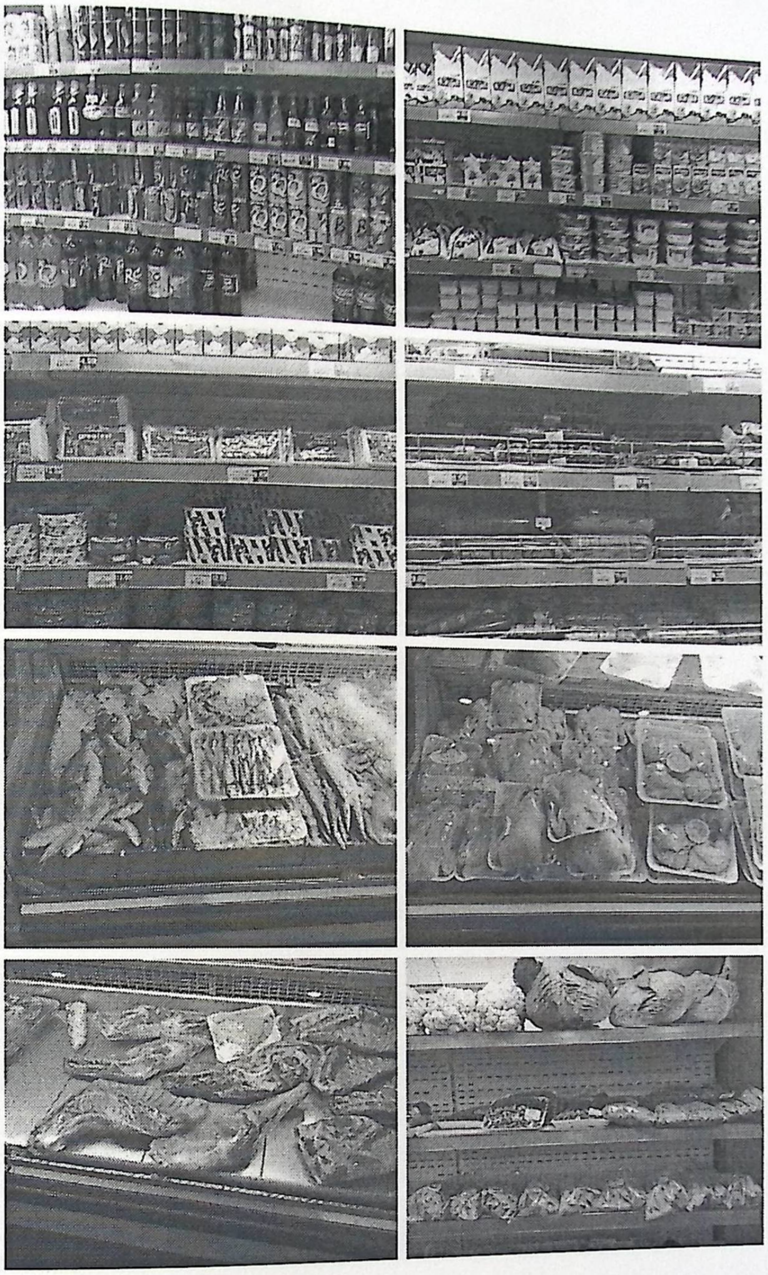

Figure 4.2: Sample images from the defined locations (Part 1/2).

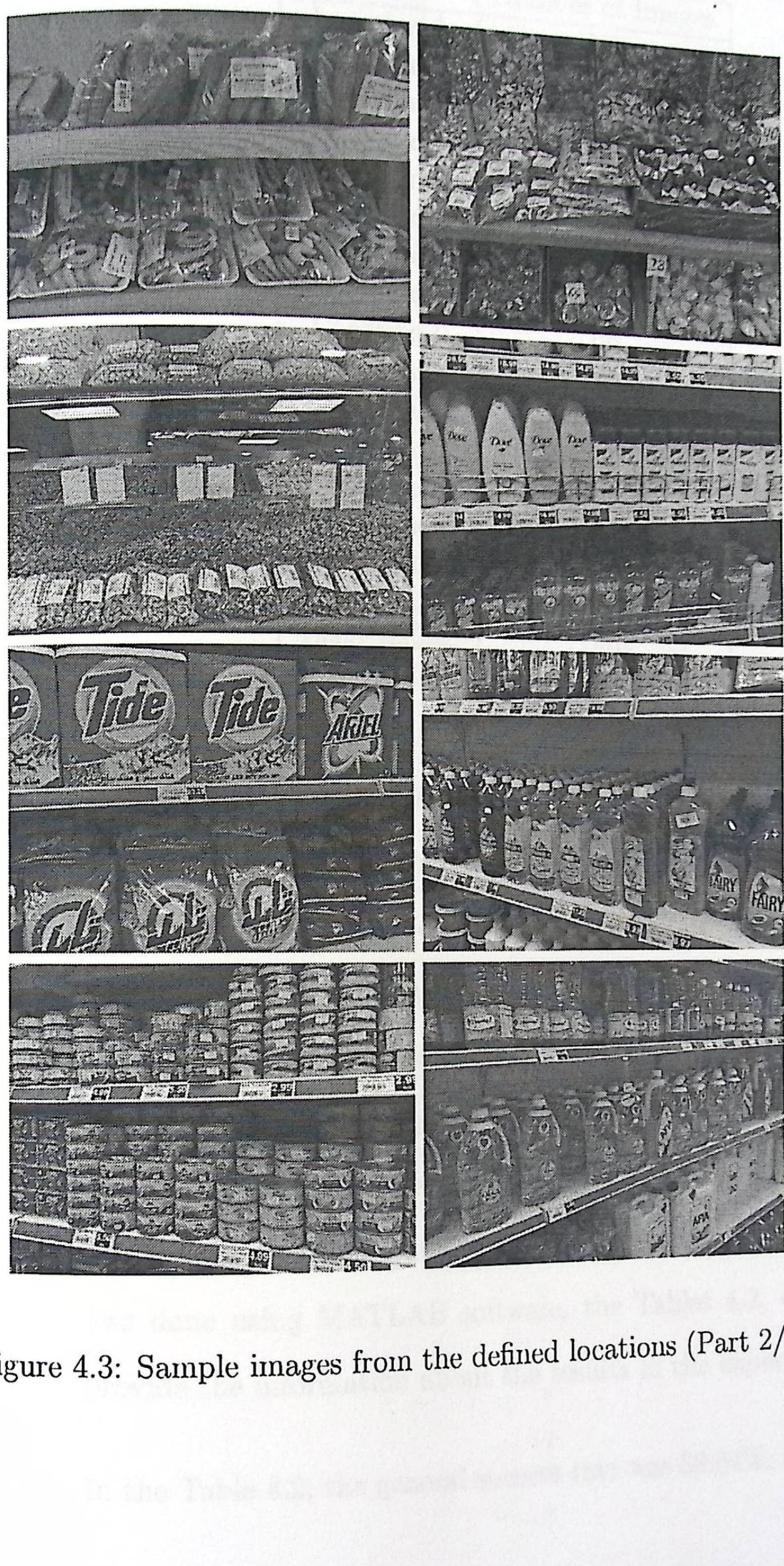


Figure 4.3: Sample images from the defined locations (Part 2/2).

Table 4.1: The locations with the number of respective images

| Location | Number of Images |
|---|---|
| Location 1 | 168 |
| Location 2 | 186 |
| Location 3 | 72 |
| Location 4 | 48 |
| Location 5 | 30 |
| Location 6 | 30 |
| Location 7 | 36 |
| Location 8 | 96 |
| Location 9 | 90 |
| Location 10 | 72 |
| Location 11 | 36 |
| Location 12 | 197 |
| Location 13 | 90 |
| Location 14 | 54 |
| Location 15 | 60 |
| Location 16 | 42 |
| Total Images | 1307 |

were 4, 8, 16, and 32 bins per channel, each test was done five times on random sample of the images using (K-fold cross-validation) with K = 3: all the images will be divided into three groups, each group has one-third of the total images, two of these groups are used as training sets for the database, and the third will be the query set, after performing the test using the query set, the formation changes to until all the groups are passed as query sets, while making the other two groups in each test as database training set, the simulation of this type of testing was done using MATLAB software, the Tables 4.2, 4.3, 4.4 and 4.5 provide the information about the results in the experiments.

In the Table 4.2, the general success rate was 89.64%, due to the small

Table 4.2: Results with feature length = 4 bins/color

| | Query Set 1 | Query Set 2 | Query Set 3 | Success Rate | Time |
|---|---|---|---|---|---|
| Test 1 | 89.68% | 91.03% | 91.51% | 90.74% | 2 seconds |
| Test 2 | 90.14% | 89.89% | 88.07% | 89.37% | 2 seconds |
| Test 3 | 90.37% | 90.34% | 86.93% | 89.21% | 2 seconds |
| Test 4 | 90.37% | 89.66% | 90.37% | 90.13% | 2 seconds |
| Test 5 | 88.99% | 88.05% | 89.22% | 88.75% | 2 seconds |
| | | | Gen. Success Rate | 89.64% | |

number of bins that create the histogram, which include less information about the color distribution in the images than any other larger histograms (with more bins), as the rest of the tables show.

Another note is the time spent to generate the result, as the processes depend on file reading operations, which is fast in Symbian OS, the time required to read the data and calculate the output is fixed to (2 seconds per query), this time is consumed when the query image is matching the required resolution before histogram construction, but if the image was not so, the operational time will increase 2.5 more seconds due to image resizing operation, so it is suggested to capture query images with the specified resolution, in order to prevent resizing them, and lose additional time periods.

In Table 4.3, the result (96.07%) is better than the one shown in Table 4.2, as more bins were used in the histogram (8 bins this time), with the same time period, this result can be extended by further increase in the number of bins, in the following tests.

In Table 4.4, greater general success rate is achieved (98.13%) by using

Table 4.3: Results with feature length = 8 bins/color

| | Query Set 1 | Query Set 2 | Query Set 3 | Success Rate | Time |
|---|---|---|---|---|---|
| Test 1 | 95.87% | 95.63% | 94.95% | 95.48% | 2 seconds |
| Test 2 | 96.79% | 96.55% | 95.87% | 96.40% | 2 seconds |
| Test 3 | 97.71% | 94.02% | 97.02% | 96.25% | 2 seconds |
| Test 4 | 95.41% | 97.70% | 96.10% | 96.40% | 2 seconds |
| Test 5 | 95.18% | 96.09% | 96.10% | 95.79% | 2 seconds |
| | | | Gen. Success Rate | 96.07% | |

Table 4.4: Results with feature length = 16 bins/color

| | Query Set 1 | Query Set 2 | Query Set 3 | Success Rate | Time |
|---|---|---|---|---|---|
| Test 1 | 97.94% | 97.24% | 98.17% | 97.78% | 2 seconds |
| Test 2 | 97.94% | 98.39% | 98.39% | 98.24% | 2 seconds |
| Test 3 | 97.71% | 97.24% | 99.31% | 98.09% | 2 seconds |
| Test 4 | 97.02% | 99.31% | 98.85% | 98.39% | 2 seconds |
| Test 5 | 97.94% | 97.70% | 98.85% | 98.16% | 2 seconds |
| | | | Gen. Success Rate | 98.13% | |

16 bins in the histograms, while maintaining the operational time as small as possible, which performance is considered as suitable for real-time implementation.

Table 4.5: Results with feature length = 32 bins/color

| | Query Set 1 | Query Set 2 | Query Set 3 | Success Rate | Time |
|---|---|---|---|---|---|
| Test 1 | 99.08% | 99.08% | 96.33% | 98.16% | 2 seconds |
| Test 2 | 97.25% | 98.85% | 98.17% | 98.09% | 2 seconds |
| Test 3 | 99.08% | 99.08% | 96.33% | 98.16% | 2 seconds |
| Test 4 | 98.17% | 97.93% | 97.02% | 97.71% | 2 seconds |
| Test 5 | 99.31% | 97.93% | 98.39% | 98.54% | 2 seconds |
| | | | Gen. Success Rate | 98.13% | |

As shown in the tables, the highest average success rate (98.13%) is found when applying the feature length of 16 and 32 bins, as the histogram with length of (32 bins) better express the image feature than the one with length of (16 bins), since it has more details, and does not increase the time required to get the results, so the next test will depend on histograms of length = 32 bins to choose between the RGB or (Gray Scale) representation.

- The second experiment was done using the 32 bins as the feature length, and was compared using the RGB color model with the Grayscale-256 color model, also done five times using the (K-fold cross-validation) with K = 3, the Table 4.6 shows the results of this experiment.

Table 4.6: Results of testing "Gray Scale-256" images using histograms with length = 32 bins

| | Query Set 1 | Query Set 2 | Query Set 3 | Success Rate | Time |
|---|---|---|---|---|---|
| Test 1 | 94.72% | 96.09% | 94.50% | 95.10% | 2 seconds |
| Test 2 | 94.04% | 95.86% | 94.27% | 94.72% | 2 seconds |
| Test 3 | 95.64% | 93.79% | 95.64% | 95.02% | 2 seconds |
| Test 4 | 95.87% | 94.71% | 96.33% | 95.64% | 2 seconds |
| Test 5 | 94.27% | 95.17% | 92.43% | 93.96% | 2 seconds |
| | | | Gen. Success Rate | 94.89% | |

The test results in Table 4.6 show the superiority of the colored images histograms in differentiating the images form the gray-scale images histograms (94.89% for the gray-scale image histogram against 98.13% for the colored ones), since the colored ones are composed of multiple individual histograms (red, green and blue color histograms), that provide more bins (in total, three times of the bins number in the gray-scale histograms), this increase the certainty of the degree of similarity or difference between two images.

## 4.4 Results and Discussion

The experiments results that are represented by the tables in Section 4.3 shows an excellent result of using colored images as sources for query and histograms values (that is better than using Gray-scale images which rate was 94.89%), since it has more information about the images details by reading the images data three times rather than once using gray scale images, this superiority of the colored images is supported by a good number of bins that are contained in the histograms, which is adequate to be implemented on the mobile phone.

The success rate of (98.13%) in Table 4.5, which passes the other histogram sizes (4 bins rate = 89.64%, 8 bins rate = 96.07%, and equal to the 16 bins rate), has been achieved by operations executed on the mobile phone, with average latency near to 2 seconds per query (from capturing the query image until the result is available), thus the user is not going to wait too long before he gets the current location, which makes the localization system efficient to be used in real-time.

# Chapter 5

# Conclusion and Future Work

In this project, the person location inside a building with known details can be identified using image processing techniques, without the need of establishing any connection with sensors and other devices found in the area, with high accuracy (localization rate of 98 %) and little latency (2 seconds per localization operation), that makes this localization system a good choice when having enough information about the appearance of the indoor environments, all these characteristics of the resulting system meet with the objectives listed in Section 1.2.

In the future, the system can be extended to be implemented with storing the history of locations visited ordered by their time, so the results can be stated with less overhead using Markov Localization or Monte Carlo Localization methods.

The quantity of the images gathered to construct the database can be decreased using the HSV color model, which can be stable against illumination changes, thus the system can operate perfectly in the environment at longer periods of time.

The localization system has to be used by a number of blind users, to record what events may occur, and how they adopt themselves to the environment using it, the system then have to use the voice messages as the output method in order to make it more interactive.

One scenario may be achieved by extending the localization system, is the specification of additional mechanism that receives voice commands from the user, the command expresses a location that the user wants to move to, and using a predefined map of the environment the system guides the user along the way until reaching the specified location, this mechanism can be integrated with localization system to provide full details of indoor environments as the user needs.

# Appendix A

# How to Install S60 Developer Tools

In this appendix, I will give the reader a look at the way of preparing the computer for developing (Symbian-Based) applications using the (S60 3rd Edition SDK for Symbian OS, Supports Feature Pack 1, For C++), along with its configuration, and finishing with configuration of Nokia Computer Vision Library that provides vast capabilities of working with images on the mobile phones..

## A.1 Recommended Hardware Requirements

The computer that will be used to develop the S60 applications should have the following characteristics:

- 1.0 GHz 32-bits Processor.
- 512 MB Random Access Memory.

- 16-bits color display.
- 1.0 GB of free Hard Disk space.
- Windows compatible sound card for audio support.
- Keyboard and Mouse.

## A.2 Software Requirements

The following are the software requirements to install and run the S60 SDK correctly:

- Windows XP Professional Edition, Service Pack 2.
- Java Runtime Environment (JRE) 1.4.2 or newer.
- ActiveState ActivePerl 5.6.1: No older or newer versions are required; since some incompatibility issues occur with these versions, the version stated here is the only operational one.

In addition to the previous software, the developer is encouraged to use an Integrated Development Environment (IDE) to facilitate the writing, building, testing, and releasing of the mobile applications, the IDE used here is called (Carbide.vs 3.0.1) that operates under the Microsoft Visual Studio 2005, which is suitable for the developers who have past experience with the development using Microsoft Visual Studio.

## A.3 Installation of the S60 SDK

The installation of the S60 SDK and its related software should be done by an administrator, and arranged in the next steps[14]:

- Make sure that you have installed Microsoft Visual Studio 2005 (specially Visual C++ option) on a hard disk partition that has free space larger than 1.0 GB, the (Microsoft Visual J# Redistributable Package) is also required to be installed in order to run the S60 IDE, the Carbide.vs 3.0.1..

- Install the Java Runtime Environment (JRE), you can download the installer from: *http://java.sun.com*

- Install the ActivePerl 5.6.1, you can download the installer from: *http://downloads.activestate.com/ActivePerl/Windows/5.6/*

- The (S60 3rd Edition SDK, FP1) is ready to be installed, you can download the installer in compressed file from: *http://forum.nokia.com*, then extract the files from the archive and run *setup.exe* file.

- The Setup wizard guides you through the installation process, follow those instructions, and note that the installation path should contain alphanumeric characters only, without any space. Also make sure that you install the SDK in the same Hard Disk Partition that contains the IDE, which in our cause, the Microsoft Visual Studio 2005.

- When the SDK Setup copies the files to the hard drive, it will show the list of installed SDKs on the local disks, if you are having the first

SDK installed right now, check its box to the left to make it the default SDK, if you have more than one SDK in the menu, check the one that you want to use as default, then click (NEXT) button.

- The SDK Setup will check for the existence of the (CSL ARM Tool chain for GCCE) in the installed programs, if it does not exist, the Setup will show a message asking if you want to install the tool chain, click the (YES) button to install it, the tool chain contains some tools to build the applications for the real mobile phones, after this installation completion, the end of the SDK Setup is reached, click (FINISH) to close the wizard. You can access the applications of the SDK by selecting (Start :: All Programs :: S60 Developer Tools :: 3rd Edition FP1 SDK :: 1.0).

- The last installation is the Carbide.vs 3.0.1, download the setup file from the Nokia Forum Site: *http://forum.nokia.com*, and run the installation process.

## A.4 Testing the SDK Configuration

There is small number of steps that are performed to check if the SDK has been successfully installed, these are the SDK testing steps:

- Open the Windows Command Prompt by selecting (Start :: Run), type:

```
cmd
```

And click (OK).

- Type:

```
devices
```

And press (ENTER) key to list the installed SDKs, if there was no SDK marked as default, type:

```
devices -setdefault @S60_3rd_FP1:com.nokia.S60
```

To choose the default SDK, which is in this occasion (S60_3rd_FP1), you can check your SDK name by going to the disk drive that contains the installed SDK, open the installation directory -which I will refer to as [SDK-Directory] in this text-, and getting the name of the directory inside, for example:

```
C:
cd Symbian\9.2
dir
```

The result looks like this:

```
... S60_3rd_FP1 ...
```

So the required SDK in this case is: (S60_3rd_FP1). And the [SDK-Directory] is:

```
C:\Symbian\9.2\S60_3rd_FP1\
```

- Type:

  ```
  epoc
  ```

  And press (ENTER) key to run the S60 emulator, please wait a while it starts. When you run the emulator for the first time, it asks for the current date and time on separate steps, use the emulator buttons to enter the numbers that form the date and time according to your system time settings, after doing so, close the emulator.

- Now we will test the IDE using a simple existing S60 program: Open the Visual Studio 2005, select from the main menu (File :: Import Symbian Project), a dialog appears.

- Select the project file attribute and enter the path of the project's file called (bld.inf) or the one that has the extension (.mmp), set the project type as (Symbian), and click (NEXT).

- Select the SDK that you want to build the application with, and check the options of building the application: the debug mode (WINSCW) is checked by default, and the release mode (GCCE) is left for the developer choice. You can manage the SDKs that are to be enabled and disabled by clicking the (Enable/Disable SDKs...) button. After finishing these steps, click (FINISH) button to perform the importing operation, when it finishes, the project is opened inside a Visual Studio Solution.

- Select the build mode from (Standard toolbar :: Solution Configuration) and build the solution by selecting (Build :: Build Solution).

- If the build mode was (Deb_S60_31_WINSCW) and the build operation was a success, click the (Start Debugging) button to open the emulator and run the program. After the emulator opens, use its buttons to run the program, by clicking the applications button (the blue button that is located to the left of (4) key on the emulator), the main menu is shown, use the five-way navigation key (the upper central key in the form of a circle) to highlight the (Installed) option, by clicking the center of the five-way navigation key, the installed applications are listed on the screen, select the application and press the center key again to run it on the emulator, you can go to the previous screen by pressing the (Right Soft Key) below the screen.
- Else if the build mode was (Rel_S60_31_GCCE) and the build operation was a success, use the (Solution Explorer) window in the Visual Studio, expand the (sis) folder, right click on the file that is named (*Project Name*.pkg) and select (Create sis File) from the context menu that appears, the SIS file is used guide the actual mobile to what files needed to run the program. Reaching this point, you have installed and tested that the SDK is running successfully.

There is some other notes about S60 SDK tools:

- Some users may face a condition when using the emulator with JRE newer than 1.5, in which the emulator diagnostics, utilities, and preferences dialogs can not be loaded, the message that appears to the user is:

```
Ecmt Manager
```

```
Cannot start Ecmt Manager
```

To solve this issue, open the following file in notepad:

```
[SDK-Directory]\Epoc32\tools\ecmt\config\config.properties
```

Add the JRE version you are using into the line 5 (following is the line 5 after adding JRE Version 1.6):

```
epdt.java.version.start=1.4.1,1.4.2,1.5,5.0,1.6,6.0
```

Save the file and close it, run the emulator and the dialogs are now opened.

- Some users may see the following error message when the emulator starts:

```
Application Closed: ncnlist KERN-EXEC 3
```

This error is insignificant and can be ignored.

## A.5 Nokia CV Library Installation

The Nokia Computer Vision Library is a project run by the Nokia Research Center, this project aims to provide extension of the Symbian application's image capabilities, including: Capture Enhancements, Post-Capture Editing, Gaming and User Interfaces.

The next points describe how to configure the SDK to use the Nokia Computer Vision Library along with its libraries[13]:

- Download the library compressed file from the project's home page:
*http://research.nokia.com/research/projects/nokiacv/index.html*
- Unpack the compressed file, go to the extraction directory and open the directory

`..\NokiaCVPublicRelease\Ncvlib\NcvLib\Lib`

Copy the contents of the sub-directories as following:

1. From

   `..\armv5\lib`

   To

   `[SDK-Directory]\Epoc32\release\armv5\lib`

2. From

   `..\GCCE\urel`

   To

   `[SDK-Directory]\Epoc32\release\gcce\urel`

3. From

   `..\winscw\udeb`

   To

   `[SDK-Directory]\Epoc32\release\winscw\udeb`

- When including the header files into a project, copy the contents of:

`..\NokiaCVPublicRelease\Ncvlib\NcvLib\inc`

From the extraction directory and paste them inside a directory within the project, then add the new headers directory to the project's (Additional Include Directories) property by: (Project :: *Project-Name* Properties :: Configuration Properties :: C/C++ :: General :: Additional Include Directories), and add the directory name.

Also include the Computer Vision's static library within the project properties by: (Project :: *Project-Name* Properties :: Configuration Properties :: Linker :: Input :: Additional Dependencies), and add the library file-name (ncvlib.lib) to the values of the property.

- To run the programs that are built using the Computer Vision Library on the actual mobile phone, send the SIS file:

  `..\NokiaCVPublicRelease\Ncvlib\NcvLib\sis\NcvLib.sis`

  To the mobile phone. This is not an executable application, so you will not see it within the (installed) menu in the phone.

# Bibliography

[1] AL Takrouri, M., Awwad, W., *Solving the Speed problem in VLAB "Visual Localization Aid for the Blinds"*, Graduation Project, Palestine Polytechnic University, Hebron, Palestine, 2008.

[2] Anagnostopoulos, C., Tsetsos, V., Kikiras, P., Hadjiefthymiades, S., *OntoNav: A Semantic Indoor Navigation System*, the 1st Workshop on Semantics in Mobile Environments, MDM05, 2005.

[3] Banâtre, M., Couderc, P., Pauty, J., Becus, M., *Ubibus: Ubiquitous Computing to Help Blind People in Public Transport*, 6th International Symposium, MobileHCI 2004, Glasgow, UK, 2004.

[4] Coroama, V., *Experiences from the Design of a Ubiquitous Computing System for the Blind*, Conference on Human Factors in Computing Systems, Montréal, Québec, Canada, 2006.

[5] Costa, G., Gusberti, A., Graffigna, J., Guzzo , M., Nasisi, O., *Mobility and Orientation aid for blind persons using Artificial Vision*, Journal of Physics: Conference Series 90, 2007.

[6] Coulton, P., Edwards, R., *S60 Programming, A Tutorial Guide*, John Wiley & Sons Ltd., 2007.

[7] Forum Nokia, *Nokia N95 8GB Technical Specifications*, 2007, available at: *http://www.forum.nokia.com/devices/N95_8GB*.

[8] Helal, A., Moore, S., Ramachandran, B., *Drishti: An Integrated Navigation System for Visually Impaired and Disabled*, Fifth International Symposium on Wearable Computers (ISWC'01), 2001.

[9] Hesch, J., Roumeliotis, S., *An Indoor Localization Aid for the Visually Impaired*, IEEE, 2007.

[10] Hirose, M., Amemiya, T., *Wearable Finger-Braille Interface for Navigation of Deaf-Blind in Ubiquitous Barrier-Free Space*, The University of Tokyo, Tokyo, Japan, 2003.

[11] Hub, A., Hartter, T., Ertl, T., *Interactive Localization and Recognition of Objects for the Blind*, California State University, Northridge Center on Disabilities' 21st Annual International Technology and Persons with Disabilities Conference, Los Angeles, CA, USA, 2006.

[12] Morris, B., *A Guide to Symbian Signed*, 3rd Edition, Symbian, 2008.

[13] Nokia, *Nokia Computer Vision Library Documentation*, Nokia Research Center, 2007.

[14] Nokia, *S60 3rd Edition SDK for Symbian OS, Supporting Feature Pack 1, for C++ S60 platform: Installation Guide*, Version 1.0, 2006.

[15] Pass, G., Zabih, R., Miller, J., *Comparing Images Using Color Coherence Vectors*, Cornell University, Ithaca, NY, USA, 1996.

[16] Sinnoqrot, F., Younes, I., *Visual Localization Aid for the Blind*, Graduation Project, Palestine Polytechnic University, Hebron, Palestine, 2007.

[17] Stichbury, J., Jacobs, M., *The Accredited Symbian Developer Primer*, John Wiley & Sons Ltd., 2006.

[18] Symbian, *How do I get my Symbian OS application signed? A guide to Symbian Signed*, version 2.3, 2005.

[19] Symbian, *Symbian OS v9.2 Product Sheet*, 2005, available at: *http://www.symbian.com/symbianos/releases/v92/productsheet.html.*

[20] Tamimi, H., *Vision-based Features for Mobile Robot Localization*, University of Tübingen, Tübingen, Germany, 2006.

[21] Wikipedia - the Free Encyclopedia, *Symbian OS*, available at: *http://en.wikipedia.org/wiki/Symbian.*

[22] Willis, S., Helal, S., *RFID Information Grid for Blind Navigation and Wayfinding*, in ISWC05, IEEE Press, 2005.