SIRIUS: Cost Effective Smart Video Surveillance System with Object Recognition Feature

by

Chong Hui Qing

14781

Dissertation submitted in partial fulfilment of

the requirements for the

Bachelor of Engineering (Hons)

(Electrical and Electronic Engineering)

JANUARY 2015

Universiti Teknologi PETRONAS

Bandar Seri Iskandar

31750 Tronoh

Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

SIRIUS: Cost Effective Smart Video Surveillance System with Object Recognition Feature

by

Chong Hui Qing

14781

A project dissertation submitted to the

Electrical & Electronics Engineering Programme

Universiti Teknologi PETRONAS

in partial fulfilment of the requirement for the

BACHELOR OF ENGINEERING (Hons)

(ELECTRICAL & ELECTRONIC ENGINEERING)

Approved by,

_____

(Dr. Mohd Zuki B Yusoff)

UNIVERSITI TEKNOLOGI PETRONAS

TRONOH, PERAK

January 2015

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

_____

CHONG HUI QING

ABSTRACT

Sirius is a Linux based real time smart video surveillance system implemented mainly using software programs. Yocto Project is used as the base operating system because of its less memory footprint and lower storage requirement compared to other Linux distributions. Sirius uses the concept of Machine to Machine (M2M) and Internet of Things (IoT) since two machines, server and client are connected to each other via Ethernet. Sirius is done to improve the effectiveness and efficiency of current surveillance systems which are costly and lack of features. Instead of turning on the camera all the time, Passive Infrared (PIR) sensor is used to trigger video capturing. Message is sent from client to server using Message Queue Telemetry Transport (MQTT) protocol to inform that someone or something has intruded the observing region. Real time video captured is displayed at the server so that object recognition can be performed using Speeded-Up Robust Features (SURF) algorithm found in Open Source Computer Vision (OpenCV) libraries. Object recognition feature in Sirius helps to remove human eyes in identifying object being captured. The identity of object captured is displayed in table format after analyzing each video. Sirius Project will be delivered as a running system made up of two machines, with accurate results of object recognition. Waterfall Model, which involves several phases is used as the methodology to complete Sirius Project. Open source tools are fully utilized including image building for the operating system. Sirius Project is fully related to IoT especially in the part of machines communication. The accuracy of the system is always higher than 85%

ACKNOWLEDGEMENT

I, Chong Hui Qing, matrix ID 14781, from the department of Electrical and Electronic Engineering would like to express my greatest appreciation to those who gave me the opportunity to complete my Final Year Project (FYP). I would like to thank Universiti Teknologi PETRONAS (UTP) for giving me an opportunity to use what I have learnt in the last few years on a real industrial application.

First and foremost, I would like to acknowledge with much gratitude the continuous guidance and patience from my FYP supervisor, Dr. Mohd Zuki B Yusoff. His constant motivation and encouragement were vital for the completion of this project. Furthermore, I am given the opportunity to expose to new knowledge through this project.

I have to further thank my lecturers who provided me guidance and support to go ahead with my project. They provided not only technical knowledge, but also encouragements and suggestions while I am facing difficulties in completing the project. Last but not least, I would like to give my special thanks to my family and course-mates whose support and love enabled me to complete the project.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS AND NOMENCLATURES

API	Application Programming Interface
CCTV	Closed-Circuit Television
FYP	Final Year Project
GPIO	General Purpose Input/Output
GPU	Graphical Processing Unit
HD	High Definition
IoT	Internet of Things
LTS	Long Term Support
M2M	Machine to Machine
MQTT	Message Queue Telemetry Transport
OE	Open Embedded
OpenCV	Open Source Computer Vision
OS	Operating System
PCA	Principal Component Analysis
PIR	Passive Infrared
RTSP	Real Time Streaming Protocol
SCP	Secure Copy Protocol
SIFT	Scale-Invariant Feature Transform
SQDIFF	Squared Difference
SSH	Secure Shell
SURF	Speeded-Up Robust Feature
USB	Universal Serial Bus
UTP	Universiti Teknologi PETRONAS
YP	Yocto Project

CHAPTER 1

INTRODUCTION

1.1 Background

Sirius is a binary star system consisting two bright stars located next to each other. This Linux-based project is known as “Sirius” because it consists of two machines, the server and client, connecting to each other via Ethernet for messaging and data transferring. The main purpose of Sirius Project is to improve the effectiveness and efficiency of the security system at workplace. Passive Infrared (PIR) sensor is used to trigger video capturing only when there is change in infrared radiation under the observing region. The client performs video capturing while the server performs object recognition using Speeded-Up Robust Feature (SURF) algorithm, which is one of the algorithms under Open Source Computer Vision (OpenCV) library. Real time streaming is performed between two machines using and messages are sent from the client to server using Message Queue Telemetry Transport (MQTT) Protocol.

Yocto Project (YP) is used as the base operating system (OS) of Sirius. YP helps to reduce the memory footprint and the storage requirement of the system. In ensuing chapter, an OS built by using YP is loosely called “Yocto OS” for brevity. Sirius is a project related to Internet of Things (IoT), which is now one of the trends in the world of embedded system. IoT is defined as “the interconnection of embedded devices with the help of Internet”. Hence, instead of being a standalone system, Sirius involves connection and communication between machines as well as real time streaming. This connection is not limited to only two machines, but can be unlimited according to the user’s requirement. However, in order to satisfy the feasibility of project, Sirius only requires the connection between two machines. After completing

this project, student is expected to have more knowledge on YP, Machine to Machine (M2M), OpenCV, and MQTT. Figure 1.1 shows the general idea of Sirius Project.

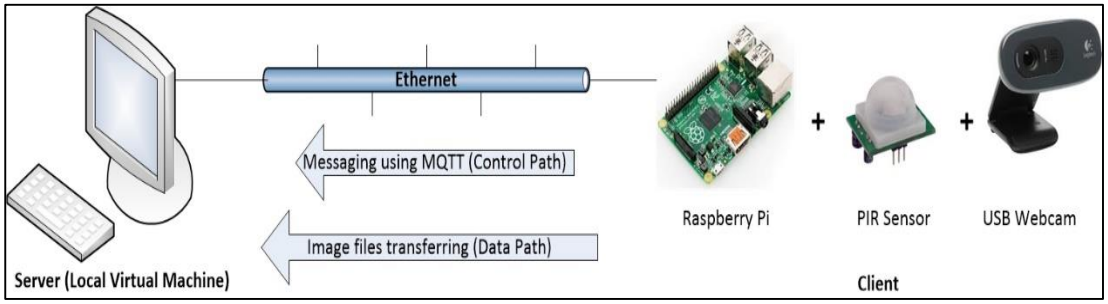


FIGURE 1.1. Sirius Project

1.2 Problem Statement

Nowadays, several kinds of embedded system using different operating systems are developed. The memory footprint and storage requirement of the Linux distributions are large. Larger capacity storage is needed to store the system image and the extra data generated in the system. Furthermore, a large memory system is needed to ensure the system is functioning fast and smoothly. Hence, the initial cost of the system is increased.

The existing CCTVs have high power consumption since the cameras are turned on all the time to capture a region regardless of any change of event has happened. Recently, video filtering feature is added to reduce the storage size. However, such design is sub-optimal because video camera is still need to be on at all time and processing power is consumed to filter off meaningless video frame.

Furthermore, object recognition feature is not included in existing CCTV systems. Users need to watch all the videos to determine what had happened at a particular time. The involvement of human in detecting object for long duration often causes fatigue. Such fatigue typically causes users to skip video frame by fast-forwarding and important event may be skipped.

1.3 Objectives And Scope Of Study

1.3.1 Objectives

Sirius Project is mainly used in industry its objectives are:

i. To capture video of a region to ensure the safety of industry.
ii. To improve the effectiveness and efficiency of the security in industry by reducing the power consumption.
iii. To remove human eyes from object recognition with the help of machine vision object recognition at the server end.
iv. To assist police in searching suspects in different areas.
v. To reduce initial cost of system by reducing memory footprint and storage requirement using open source development tools.

1.3.2 Scope of Study

This project focuses on an embedded system being run in Yocto OS. This study covers the background, function and the advantages of YP. The work flow of YP needs to be understood in order to apply it correctly based on the requirement of Sirius.

Furthermore, the scope of Sirius project includes the concept of M2M. Since Sirius Project consists of a server and a client, communication between two machines is required. MQTT protocol helps to connect two machines in terms of message transferring. Besides, files transferring and real time streaming are done in network data path using several protocols.

In addition, Sirius project focuses on video capturing and processing. All these processes are done using OpenCV Application Programming Interface (API). OpenCV consists of image processing algorithms such as object recognition algorithm. Different types of algorithms are learned and compared so that the best method is used.

1.3.3 Relevancy of Project

The research of Sirius Project is relevant to current inefficient security system available in the market. Several tools and functions need to be integrated not only to reduce the initial cost, but also increase the efficiency of the security systems. Hence, the objectives of Sirius Project are relevant to the current challenges faced by the society especially in terms of security.

1.3.4 Feasibility of the Project within the Scope and Time Frame

This project is feasible within the scope of study because all the information can be found in open source websites. Tutorials can be easily found in Internet websites. Although several researches need to be done, however, students are not required to create the tools and protocols from scratch. By only integrating all the software packages, libraries and functions available in Internet, Sirius Project can be done as an individual project of a student within 28 weeks.

CHAPTER 2

LITERATURE REVIEW AND/OR THEORY

2.1 Yocto Project

Yocto Project (YP) is an open source tool kit consisting of recipes to create a custom Linux distribution for different hardware architectures [1]. “Poky”, which is the core of YP, consists of tools, Open Embedded (OE) packages and metadata to build an image to boot up a target device. The “Bitbake” tool found in Poky helps to download, patch, install and build an image [2]. Packages found in projects other than Poky and OE can be installed using YP. Figure 2.1 shows the work flow of YP [3].

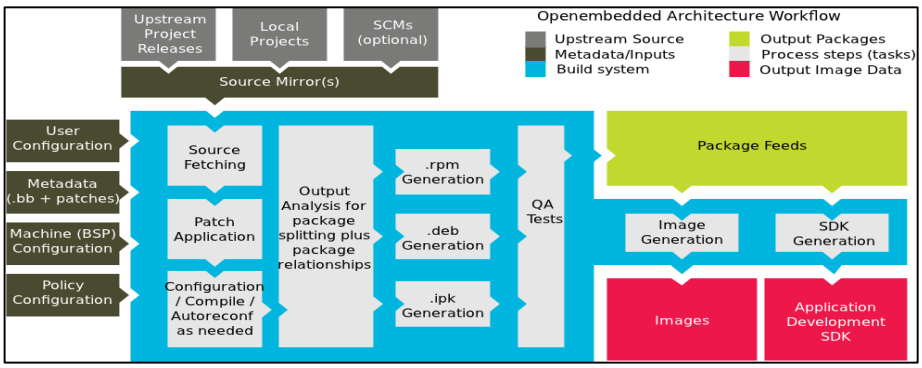


FIGURE 2.1. Yocto Project Work Flow

2.2 Messaging Queue Telemetry Transport

Messaging Queue Telemetry Transport (MQTT) is a protocol that consists of a publish-subscribe system to transfer messages from a server to a client [4], [5]. MQTT protocol allows several parties to publish and subscribe on a same topic for communication. A server named “broker” manages the connections between publishers and subscribers and transfers messages to the subscribers according to their

subscribed topics. Without subscribing to the same topic, message will not be received by the subscriber. Figure 2.2 shows the MQTT protocol model [5].

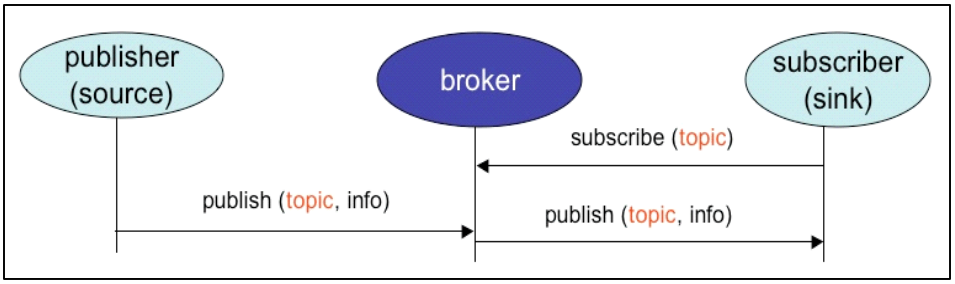


FIGURE 2.2. MQTT Protocol Model

2.3 Real Time Intelligent Video Surveillance System

Recently, the cameras are equipped with several features such as automatic focus, face detection, smile detection, etc. All these features play an important role especially in monitoring the security of a place. Algorithms in computer vision help to achieve this purpose. However, the scenarios should be executed in real time or with a small latency [6]. This can be done with either using Secure Copy Protocol (SCP) to transfer computer files between two machines in a fast speed or Real Time Streaming Protocol (RSTP).

SCP is a network protocol which helps the hosts on a network to transfer files between each other. The data transferring is done by using Secure Shell (SSH) and some mechanisms for authentication. Hence, permissions are needed so that the clients can download the files from the server. The syntax of SCP command is shown in Table 2.1 [7].

TABLE 2.1. SCP Command

Direction of Transmission	Command
Copying file to host	scp Source_file user@hostIP:Target_directory
Copying file from host	scp user@hostIP:directory/Target_file Target_folder

Streaming media technology such as RTSP is included in the system. The raw data being captured by the camera is encoded into signal and being sent to the network by the server. The client can connect to the network and receive the signal. Then, the signal will be decoded back into a video. Figures 2.3 and 2.4 show the structure of a video surveillance system [8].

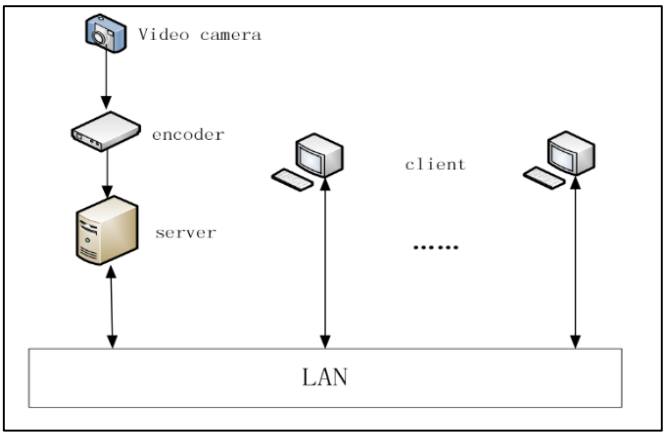


FIGURE 2.3. Real Time Video Surveillance System

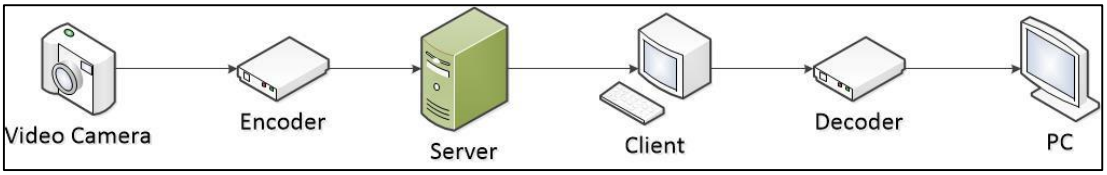


FIGURE 2.4. Process of Encoding and Decoding in Video Surveillance System

2.4 Speeded Up Robust Features (SURF) in Open Source Computer Vision (OpenCV)

Computer Vision helps to program a computer to process images and videos. Open Source Computer Vision (OpenCV) is launched in year 1999. OpenCV library consists of image and video processing algorithms. The functions and algorithms are able to capture images and videos, access pixel values, analyze images, edit images and so on [9].

Recently, researches on real time smart video surveillance system which includes object tracking and detection features without human intervention are done. Although background subtraction has been widely used in several video surveillance systems, object recognition cannot be done using background subtraction algorithm [10]. The purpose of object recognition is to verify the presence and identity of an object in an image. An algorithm named Speeded-Up Robust Features (SURF), which uses Hessian matrix approximation extracts key points, from both template and captured images and describes them as feature vectors [11], [12], [13]. The feature vectors of template and captured images are learned and compared with each other to determine if the object is being captured and recognized [11]. Figure 2.5 shows the pipeline of SURF algorithm [13].

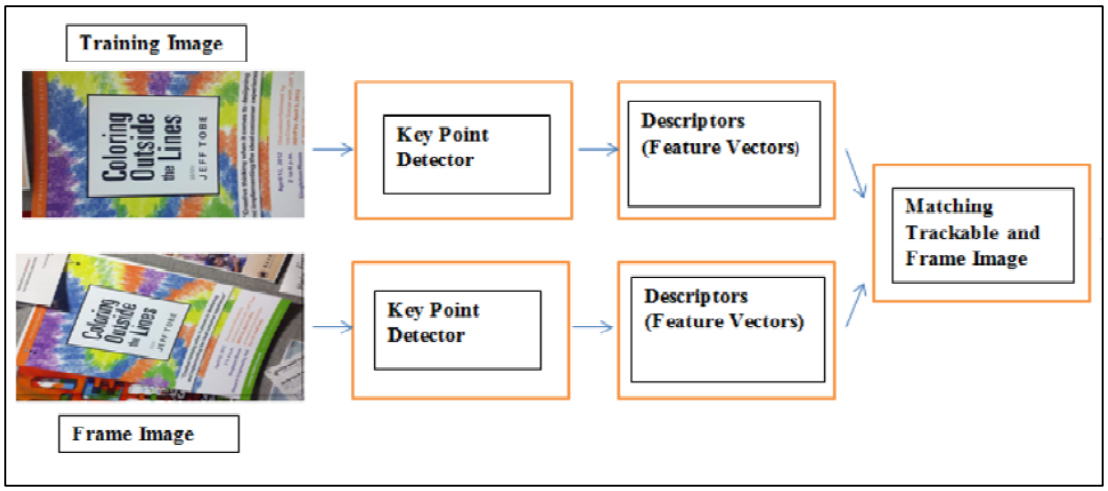


FIGURE 2.5. SURF Algorithm Pipeline

Several researches are done to compare different algorithms for object recognition. The comparison is done on several factors such as speed as well as variance in scale, rotation and so on. However, SURF is proven to be the best algorithm among all. Table 2.2 shows the comparison between Scale-Invariant Feature Transform (SIFT), Principal Component Analysis (PCA)-SIFT and SURF [12], [14].

TABLE 2.2. Comparison between SIFT, PCA-SIFT and SURF

Method	Time	Scale	Rotation	Blur	Illumination	Affine
SIFT	Common	Best	Best	Best	Common	Good
PCA-SIFT	Good	Common	Good	Common	Good	Good
SURF	Best	Good	Common	Good	Best	Good

2.5 Relevancy and Recentness of Literature

The literature is relevant to the project since similar algorithms, theories and studies are done by other researches. Furthermore, the literature referred are done within these six years. Hence, the literatures are useful and reliable for Sirius Project.

CHAPTER 3

METHODOLOGY/PROJECT WORK

3.1 Methodology

Sirius is done mainly using software implementation. Only a small part involves hardware implementation. Hence, Waterfall Model which is a sequential design process mainly used in software development is used to complete this project. The term “Waterfall” indicates that a phase in the development process cannot start if the previous phase is not completed. Besides, this method does not allow developers to make changes in the previous phase [15]. The advantage of Waterfall Model is that it is easy to be understood. The phases are clear defined. Hence, developers can easily arrange their tasks so that everything can be done smoothly [16]. Figure 3.1 shows the phases of Waterfall Model.

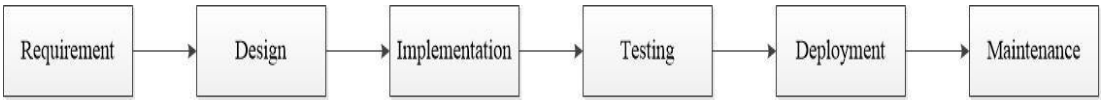


FIGURE 3.1. Waterfall Model

3.2 Key Milestone

Sirius is a project consists of the process of planning, designing, implementing, testing and improving. Similar to Waterfall Model, this project has a few key milestones need to be achieved to complete the whole project. The milestones are:

3.2.1 Identifying Problem Statement and Goals

Before starting the project, it is important to identify the problem statement and why this project is needed to be implemented. Researches about the current systems and user demand are required to be done. Besides, comparison should be made between the new system and the current system available in the market. The strengths and weaknesses of the systems have to be evaluated. Furthermore, goals of this project are needed to be set clearly. It helps developers to plan the schedule and the design of the system so that this project is feasible within the scope and time frame.

3.2.2 Software and Hardware Research

Sirius Project involves several software developments such as building Yocto image, OpenCV and MQTT protocols. Hence, researches are needed before any implementation is started. Students need to research about Yocto Project, how to build a Yocto OS and what extra software packages are needed. Besides, OpenCV and MQTT protocols as well as their applications need to be understood. Furthermore, several methods about object recognition are compared so that the most suitable method is chosen. On the other hand, datasheets and schematic diagrams of hardware components are studied so that all the requirements are met and the connections are correct.

3.2.3 System Design

System design is important before any implementation is made. The high level design of the project should be done. All the conditions are considered and should be included in the design. For example, how long should the video being recorded once the camera is being triggered, what is the suitable frame per second, what is the threshold value needed for object recognition, what criteria should the system consider during the process of object recognition, etc. The flow of system is planned well and flow chart is drawn to ease the implementation. Furthermore, Gantt chart is prepared to estimate the time needed for each phase. It ensures everything is done within the time given.

3.2.4 Hardware Connection

Sirius Project involves both software and hardware implementation. Before starting software development, the hardware components have to be connected correctly. Schematics and datasheets are being referred while connecting the components. The functionality of components are tested before software implementation is started. This helps to narrow down the scope of debugging if error is found during the implementation phase.

3.2.5 Software Implementation

At client side, C++ language source codes are written to trigger video capturing depending on the input of PIR sensor and publish message to the server. Real time streaming is done between two machines. On the other hand, Python script is written at server to subscribe MQTT topic and read the message. Then, object recognition is performed. A table containing the identity and location of object is displayed at the end of each video analytic. Necessary testing are done so that the result obtained is accurate.

3.2.6 Benchmarking/Validation and Maintenance

After the Sirius Project has done, it is tested to ensure that it is user friendly and the accuracy is high. There are some factors considered during the process of testing such as sensitivity of sensor, resolution of image, the speed of communication between both machines as well as the speed of image transferring. Furthermore, a validation algorithm is written to calculate the accuracy of the object recognition algorithm. Theoretical results are generated and it is compared with the experimental results. Maintenance is needed when there is any fault or bugs found in the system. If the accuracy obtained from the benchmarking process is low, the system needs to be modified. Changes made include using different threshold values or combining different object recognition methods. If the resolution of video is bad, a better USB camera should be used.

3.3 System Flow Chart

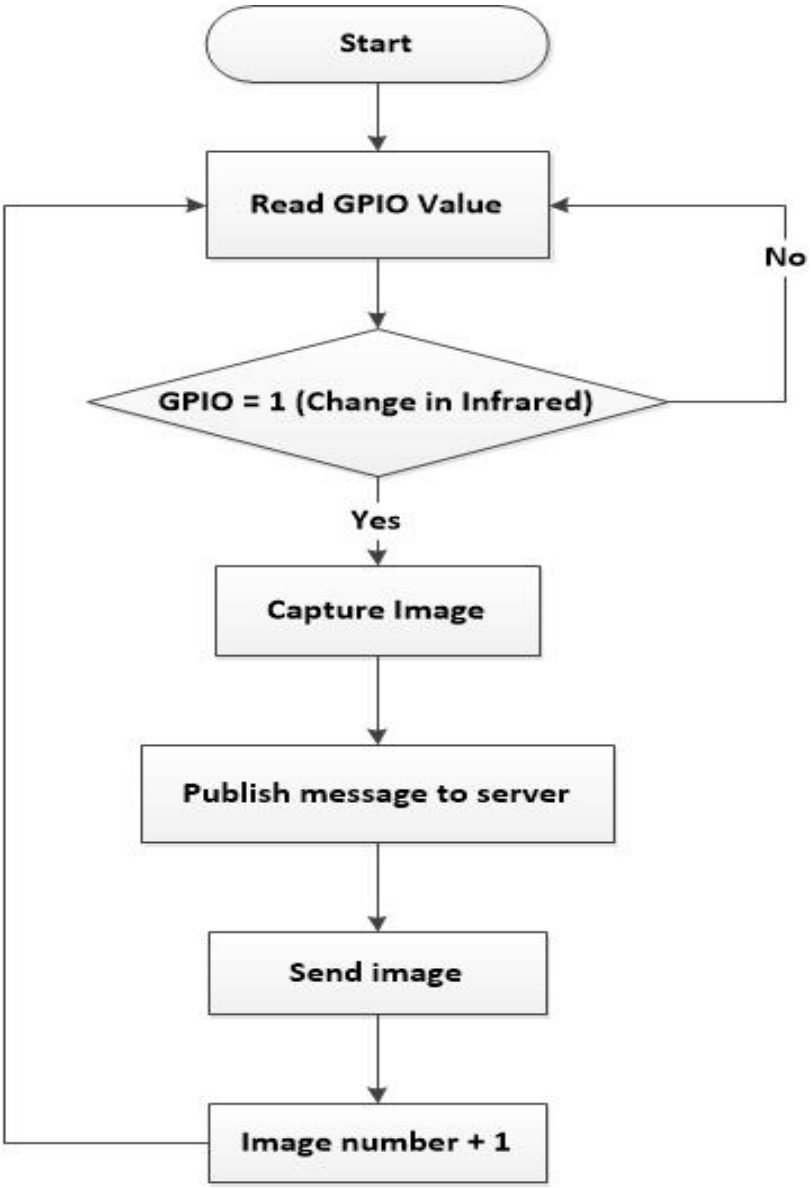


FIGURE 3.2. Client System Flow Chart

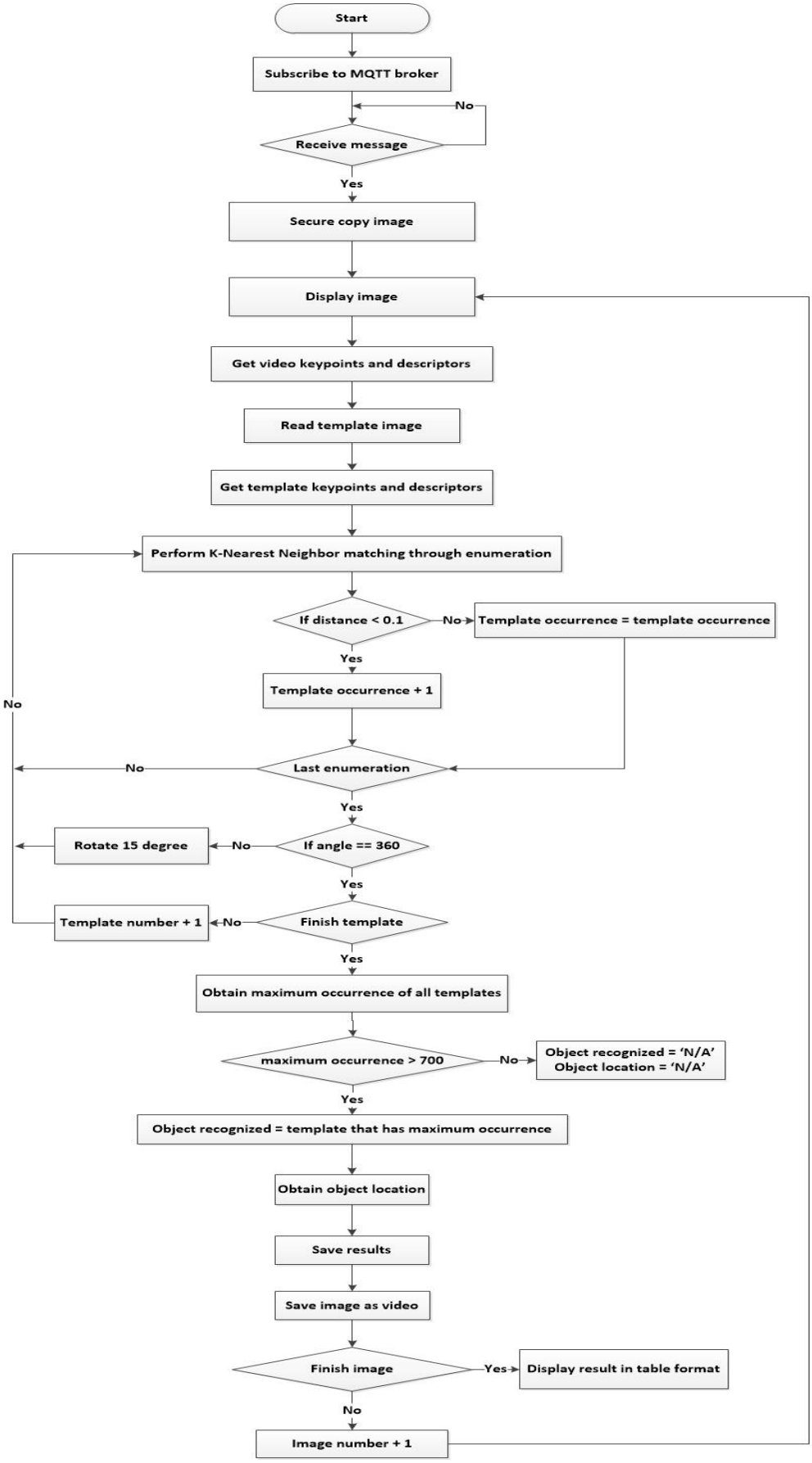


FIGURE 3.3. Server System Flow Chart

3.4 Tools and Equipment Required

There are several components and equipment are used for Sirius Project. The tools used and the functions are shown below.

- **Local Virtual Machine (Server) -** The operation system of the server is Linux and Ubuntu 12.04 LTS is used in this development machine. This machine is used to stream the video captured and perform object recognition.

FIGURE 3.4. Ubuntu 12.04 LTS

- **Raspberry Pi with Yocto Image (Client)** - The client is operated by Yocto Project with software packages needed installed. The client consists of Raspberry Pi platform, PIR sensor and USB camera.

FIGURE 3.5. Raspberry Pi B+

- **Passive Infrared (PIR) Sensor** – Used to detect changes in infrared radiation so that the camera will be triggered when the output of sensor shows there is object detected in a certain distance.

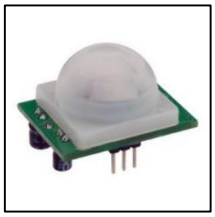


FIGURE 3.6. PIR Sensor

- **Logitech HD USB Camera** – Used to capture video in high definition mode.

FIGURE 3.7. Logitech USB Webcam

- **Ethernet Switch** - Used to connect both the server and the client via Ethernet. IP address is set for both machines so that they are able to be connected.

FIGURE 3.8. Ethernet Switch

- **Ethernet Cables -** Used to connect both server and client machines to Ethernet switch.

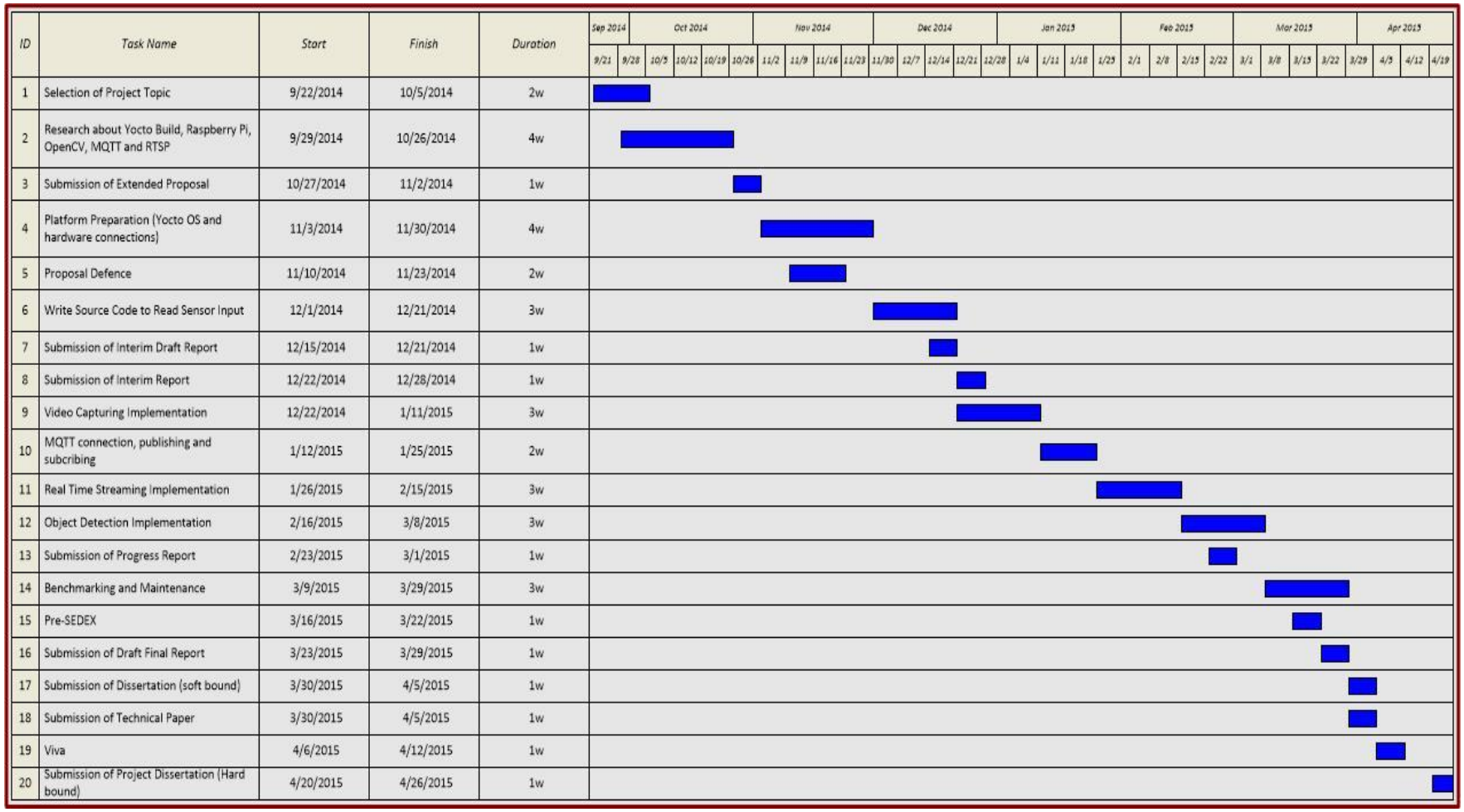
FIGURE 3.9. Ethernet Cables

- **USB Ethernet Adaptor** - Used to connect server to Ethernet switch.

FIGURE 3.10. USB Ethernet Adaptor

3.5 Gantt Chart

Figure 3.11 illustrates the Gantt chart for the completion of Sirius Project.



ID	Task Name	Start	Finish	Duration
1	Selection of Project Topic	9/22/2014	10/5/2014	2w
2	Research about Yocto Build, Raspberry Pi, OpenCV, MQTT and RTSP	9/29/2014	10/26/2014	4w
3	Submission of Extended Proposal	10/27/2014	11/2/2014	1w
4	Platform Preparation (Yocto OS and hardware connections)	11/3/2014	11/30/2014	4w
5	Proposal Defence	11/10/2014	11/23/2014	2w
6	Write Source Code to Read Sensor Input	12/1/2014	12/21/2014	3w
7	Submission of Interim Draft Report	12/15/2014	12/21/2014	1w
8	Submission of Interim Report	12/22/2014	12/28/2014	1w
9	Video Capturing Implementation	12/22/2014	1/11/2015	3w
10	MQTT connection, publishing and subcribing	1/12/2015	1/25/2015	2w
11	Real Time Streaming Implementation	1/26/2015	2/15/2015	3w
12	Object Detection Implementation	2/16/2015	3/8/2015	3w
13	Submission of Progress Report	2/23/2015	3/1/2015	1w
14	Benchmarking and Maintenance	3/9/2015	3/29/2015	3w
15	Pre-SEDEX	3/16/2015	3/22/2015	1w
16	Submission of Draft Final Report	3/23/2015	3/29/2015	1w
17	Submission of Dissertation (soft bound)	3/30/2015	4/5/2015	1w
18	Submission of Technical Paper	3/30/2015	4/5/2015	1w
19	Viva	4/6/2015	4/12/2015	1w
20	Submission of Project Dissertation (Hard bound)	4/20/2015	4/26/2015	1w

FIGURE 3.11. Gantt Chart

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Results

Sirius Project mainly focuses on integrating sensors, machines and communication systems. It is a project related to Internet of Things and it consists of a few main parts, which are:

- Build Yocto image
- Prepare template images
- Obtain input from PIR sensor
- Trigger USB camera for image capturing
- Publish message to server and transfer image
- Object recognition
- Save video and display result

4.1.1 Build Yocto Image

Sirius Project is started with building Yocto image. Yocto Project is a reference operating system which ease developers to build an operating system. Compared to other Linux distros, Yocto OS requires smaller storage size and it has lesser memory footprint. There are a few types of images that are compatible to Raspberry Pi platform. The images include rpi-hwup-image, rpi-basic-image, rpi-test-image, core-image-minimal, core-image-sato, core-image-sato-sdk, etc. In this project, core-image-sato is used because it is able to show pop-out window containing captured images on the screen. The extra software packages such as OpenCV, Gstreamer, ffmpeg, and Mosquitto are installed successfully in the image. Hence, all the libraries can be used.

Similar to Windows or Linux OS, Yocto OS consists of terminal, keyboard settings, appearance configuration, shutdown button, etc. Figure 4.1 shows the desktop of Yocto OS.

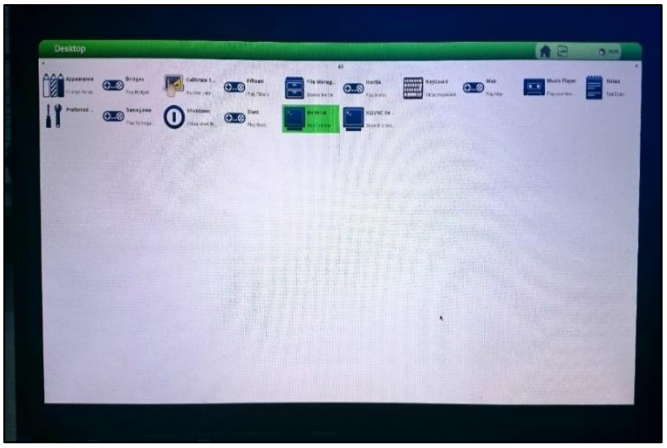


FIGURE 4.1. Desktop of Yocto OS

A comparison is made between Yocto OS and Raspbian Image to determine their storage requirement and memory footprint. It is proven that Yocto OS requires smaller storage requirement compared to Raspbian Image. Furthermore, after booting up the image, the memory used is determined. The result shows that Yocto OS uses lesser memory when the image capturing process is running. Figures 4.2 to 4.5 show the storage requirement and memory used for both Yocto OS and Raspbian Image while Table 4.1 shows the comparison between both images.

```
Filesystem      Size  Used Avail Use% Mounted on
/dev/root       614M  488M   94M  84% /
devtmpfs        215M  4.0K  215M   1% /dev
tmpfs            40K     0   40K   0% /mnt/.psplash
tmpfs           219M  208K  219M   1% /run
tmpfs           219M   64K  219M   1% /var/volatile
/dev/mmcblk0p1   20M   13M  7.6M  63% /media/mmcblk0p1
```

FIGURE 4.2. Disk Usage of Yocto OS

```
Filesystem      Size  Used Avail Use% Mounted on
rootfs          5.6G  4.9G  420M  93% /
/dev/root       5.6G  4.9G  420M  93% /
devtmpfs        183M     0  183M   0% /dev
tmpfs            38M  312K   38M   1% /run
tmpfs           5.0M     0  5.0M   0% /run/lock
tmpfs            75M     0   75M   0% /run/shm
/dev/mmcblk0p5   60M  9.4M   50M  16% /boot
/dev/mmcblk0p3   27M  397K   25M   2% /media/SETTINGS
```

FIGURE 4.3. Disk Usage of Raspbian Image

```
MemTotal:         447876 kB
MemFree:          351024 kB
Buffers:            6608 kB
Cached:            51676 kB
SwapCached:            0 kB
Active:            36104 kB
Inactive:          44904 kB
Active(anon):      22808 kB
Inactive(anon):      912 kB
Active(file):      13296 kB
Inactive(file):    43992 kB
Unevictable:           0 kB
Mlocked:               0 kB
SwapTotal:             0 kB
SwapFree:              0 kB
Dirty:                 8 kB
Writeback:             0 kB
AnonPages:         22732 kB
Mapped:            16044 kB
Shmem:              1000 kB
Slab:               7420 kB
SReclaimable:       3212 kB
SUnreclaim:         4208 kB
KernelStack:         528 kB
PageTables:         1080 kB
NFS_Unstable:          0 kB
Bounce:                0 kB
WritebackTmp:          0 kB
CommitLimit:      223936 kB
Committed_AS:      74288 kB
VmallocTotal:     565248 kB
VmallocUsed:        3860 kB
VmallocChunk:     340524 kB
```

FIGURE 4.4. Memory Usage of Yocto OS

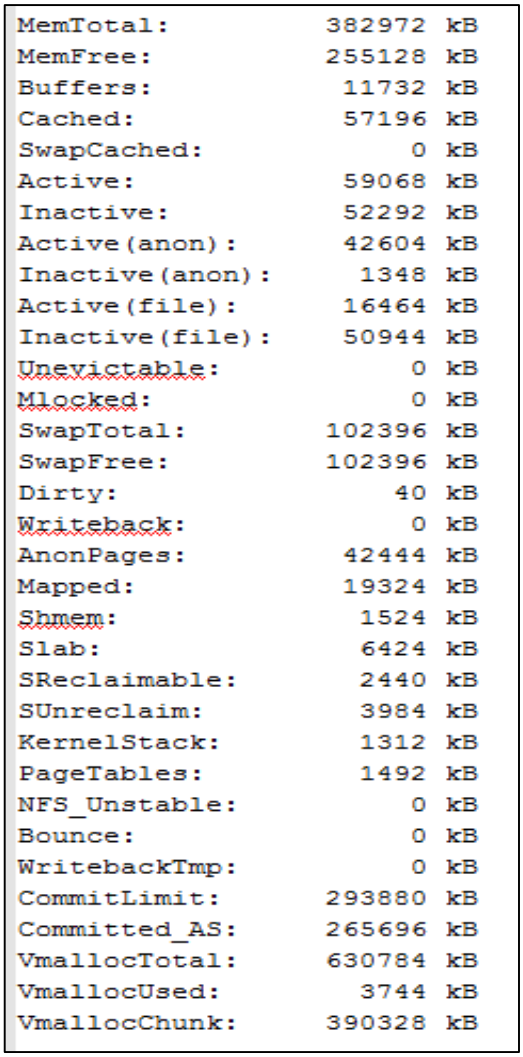


FIGURE 4.5. Memory Usage of Raspbian Image

TABLE 4.1. Comparison between Yocto OS and Raspbian Image

Parameter	**Yocto OS**	**Raspbian Image**
Disk Usage	501 MB	9.81GB
Memory Usage	$\frac{97MB}{448MB} \times 100\% = 21.65\%$	$\frac{124MB}{373MB} \times 100\% = 33.24\%$

4.1.2 Prepare Template Images

Before object recognition can be carried out, the machine needs to learn the objects which are needed to be recognized. The number of template images will affect the processing speed of object recognition. The larger number of template images, the longer the time needed to complete the process. In this project, three template images are used. Figures 4.6 to 4.8 show the template images.

FIGURE 4.6. Aeon Card Template Image

FIGURE 4.7. KimGary Card Template Image

FIGURE 4.8. Watson Card Template Image

4.1.3 Obtain Input from PIR Sensor

PIR sensor and USB camera are connected successfully to Raspberry Pi platform. The voltage source of PIR sensor is connected to 3.3V while the output is connected to the General Purpose Input Output (GPIO) Pin 26 so that the USB camera can start capturing image depending on the input from the pin. The sensitivity knob on the PIR sensor is adjusted so that it is sensitive enough to detect infrared changes. On the other hand, the delay is reduced to its minimum so that the GPIO value can be changed almost instantly. Once the PIR Sensor detects any change in infrared radiation, the GPIO value will become HIGH for around 3 seconds. Then the value will become LOW again. If there is changes in infrared radiation within the 3 seconds, the GPIO value will keep HIGH until there is no changes detected. Figure 4.9 shows the hardware connections of the client of Sirius.

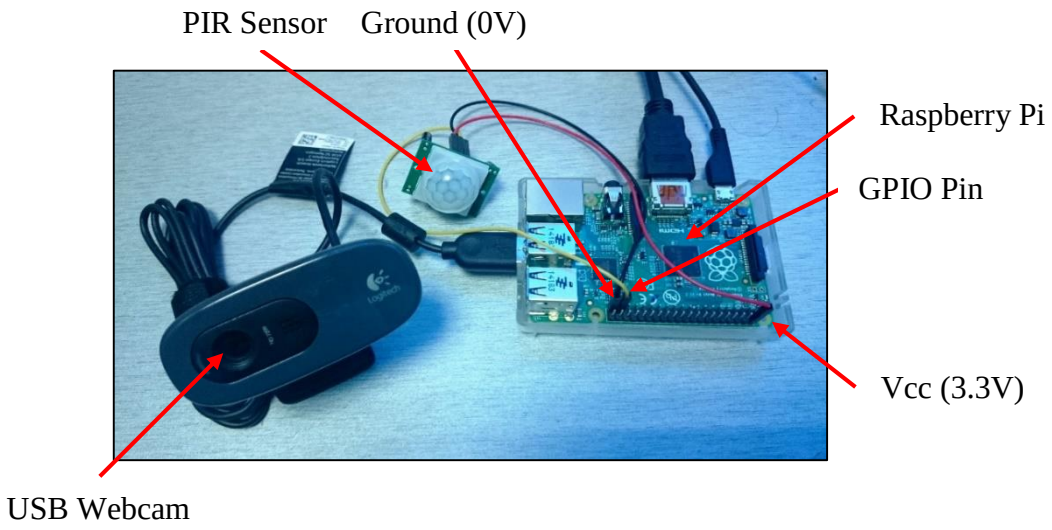


FIGURE 4.9. Hardware Connections

When the machine is booted up, the GPIO pin that is used is not initialized yet. Hence, several commands are needed to initialize the GPIO pin and indicate the direction whether it is an input or an output. In this case, GPIO pin 26 is used. The commands are:

```
// Export GPIO pin 26
echo 26 > /sys/class/gpio/export

// Set the output of PIR sensor as the input of Raspberry Pi
echo in > /sys/class/gpio/gpio26/direction
```

FIGURE 4.10. Commands to Configure GPIO Pin

4.1.4 Trigger USB Camera for Image Capturing

Source codes is written in C++ language to read the value of GPIO pin so that the camera can be triggered depending on the input of sensor. When there is change in infrared, PIR sensor will output a logic "1" to the GPIO pin, while logic "0" indicates there is no event happening. Besides, image can be captured successfully using OpenCV libraries. The image is captured and saved in jpg format. Besides, since the GPIO value becomes low after three seconds if there is no change in infrared radiation, which causes the USB camera to turn off. Hence, the algorithm is written to ensure that the camera will continue capturing images for 20 seconds. If there is change in infrared within the 20 seconds, the capture time will be increased. Figure 4.11 shows one of the images captured by the USB camera.

FIGURE 4.11. Image Frame Captured

4.1.5 Publish Message to Server and Transfer Image

The protocol used to publish and receive message between two machines is MQTT. A message is sent to the broker under the topic “huiqing/sirius”. The purpose is to inform the server that someone or something has intruded the observation region. On server side, it is set to be always connected to the broker and subscribed to the topic name. By doing this, the server will be able to receive the message sent by the client. Even if the server is not connected, the message will still be able to be received once it is connected to the broker.

On the other hand, while the camera is capturing images, the image is sent to the server one by one using SCP protocol. With a 640 × 480 pixels image, the sending time is less than 1 second. Once the image is sent to the server, the server will open it and start the object recognition process. Since no password is set for the Yocto OS, SCP command will not prompt user to key in the password.

4.1.6 Object Recognition

Apart from running in Yocto OS, the value of Sirius Project is that object recognition feature is included in the system. This helps the users to determine what is happening during a particular time without watching the entire video. Object captured will be recognized if the template object has been learnt by the machine. In Sirius Project, SURF method is used to perform object recognition. After receiving messages and images from the client, the images will be opened one by one for further processing. Key points and descriptors of both image source and template images are extracted. Then, the similarities between image source and template images are determined using K-nearest neighbor method. The number of key points matched will be recorded. These steps will be repeated for all the templates. If the maximum number of key points among all templates is higher than the threshold value, it can be concluded that the object is captured. Figures 4.12 and 4.13 show the image when the object is recognized successfully.

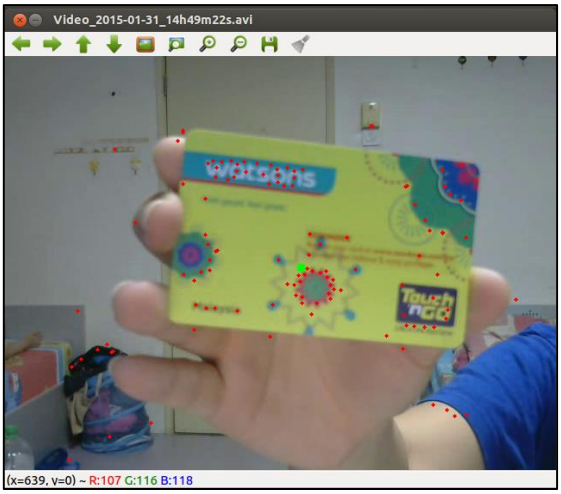


FIGURE 4.12. Successful Object Recognition 1

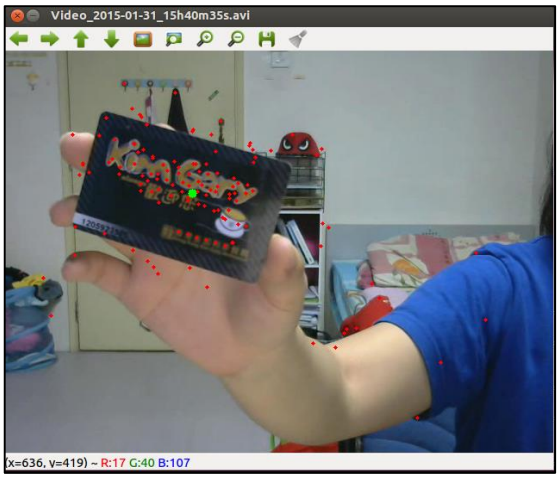


FIGURE 4.13. Successful Object Recognition 2

The red color points represents the common key points of both image source and template image. Threshold value is set to be 700 by trial and error and if the number of red color points exceeds the threshold value, it can be concluded that the object in template image is being captured in the video. However, SURF method has its own weaknesses. SURF method does not work well for rotated images. If the object captured is rotated with a large angle, the object will not be recognized. Hence, each template image is being rotated from 0° to 360° so that rotated object can be recognized. However, this takes longer time for the process to be done. Figure 4.14 shows the

successful recognition for a rotated object while Figure 4.115 shows the successful recognition for an upside down object.

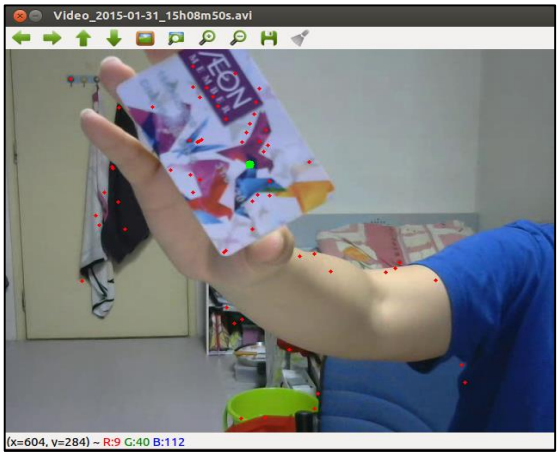


FIGURE 4.14. Successful Recognition for Rotated Object

FIGURE 4.15. Successful Recognition for Upside Down Object

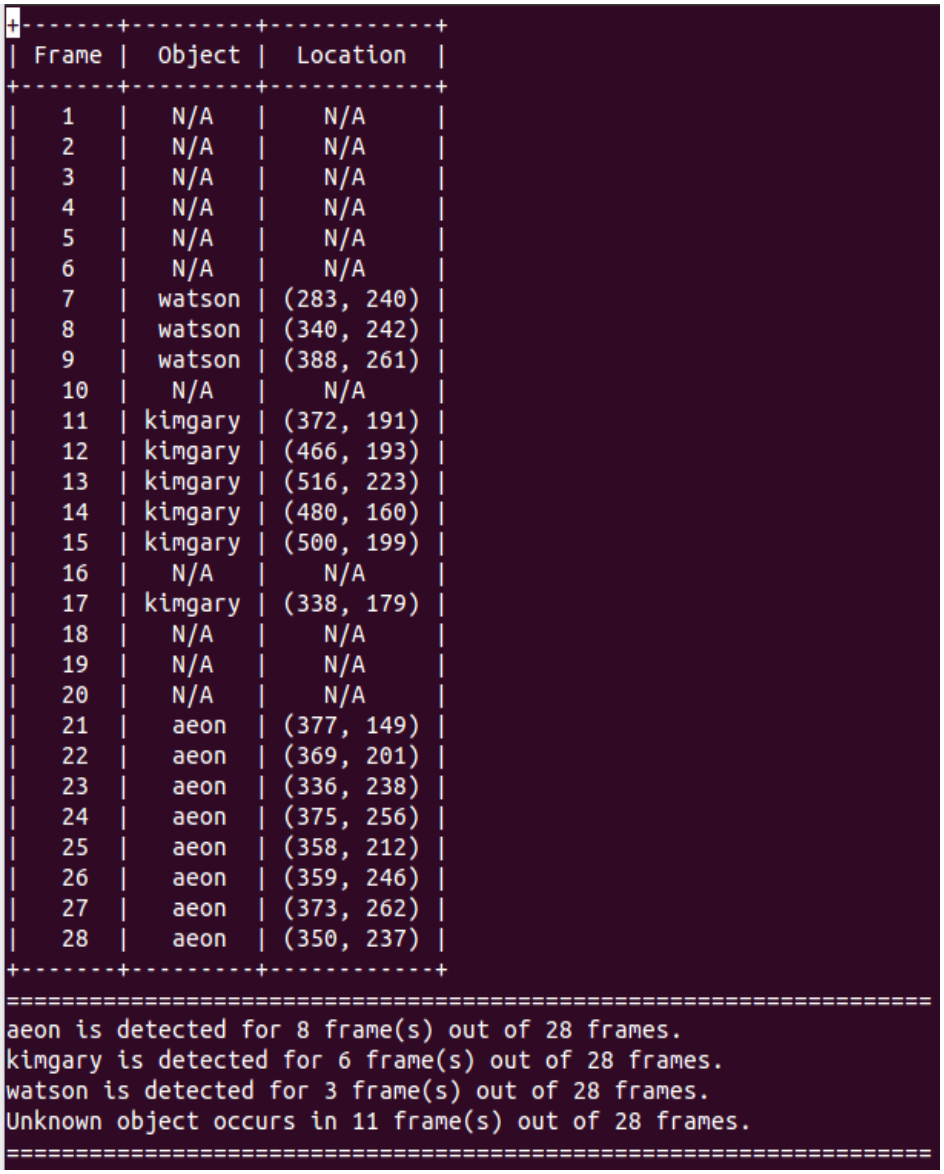
Apart from object recognition, the algorithm is able to determine the location of the object being detected. This process is done by scanning through all the pixels of the frame in order to determine if there is similar key point. If similar key point is found, the coordinate will be recorded. Once the scanning process is done, the mean of both x and y coordinates will be calculated and the result will be the object location. However, the coordinate found is not necessary the center of the object. It is because the key points are located at some interesting points such as corner or colorful spots.

Hence, the coordinate found depends on the location of key points. The following figures show the coordinate found for the object. The green spot represents the coordinate calculated while red spot represents the similar key point. The location of green spot is located at the center of red spots. Hence, if the red spots only locate at certain part of the object, for example the bottom part of the object, the green spot will be shown at the bottom part as well. This scenario is shown in Figure 4.12.

4.1.7 Save Video and Display Result

Images are sent one by one instead of sending the whole video from the client to the server. Although there are latency compared to real time streaming, but it is still acceptable. Hence, after analyzing all the images, the images will be saved as a whole video using date and time as the video name. This is to ensure that different videos will not overlap with each other. Furthermore, if the user wants to trace something from the video, they can easily decide which video to be opened. Video saving can be achieved by using the function "VideoWriter" available in OpenCV library.

After the analytic, the result of object recognition is saved as a table in a text file. The contents include frame number, object recognized, location of object as well as a summary showing the number of frames an object is recognized. Figure 4.16 shows an example of the result of object recognition.



Frame	Object	Location
1	N/A	N/A
2	N/A	N/A
3	N/A	N/A
4	N/A	N/A
5	N/A	N/A
6	N/A	N/A
7	watson	(283, 240)
8	watson	(340, 242)
9	watson	(388, 261)
10	N/A	N/A
11	kimgary	(372, 191)
12	kimgary	(466, 193)
13	kimgary	(516, 223)
14	kimgary	(480, 160)
15	kimgary	(500, 199)
16	N/A	N/A
17	kimgary	(338, 179)
18	N/A	N/A
19	N/A	N/A
20	N/A	N/A
21	aeon	(377, 149)
22	aeon	(369, 201)
23	aeon	(336, 238)
24	aeon	(375, 256)
25	aeon	(358, 212)
26	aeon	(359, 246)
27	aeon	(373, 262)
28	aeon	(350, 237)

aeon is detected for 8 frame(s) out of 28 frames.
kimgary is detected for 6 frame(s) out of 28 frames.
watson is detected for 3 frame(s) out of 28 frames.
Unknown object occurs in 11 frame(s) out of 28 frames.

FIGURE 4.16. Result of Object Recognition

Based on Figure 4.16, “N/A” indicates “Not Available”, which means object is not detected for that particular frame. This scenario is caused by the following reasons:

- **No object is captured in the video.**
- **Number of key points matched is less than the threshold value.**
- **Object is captured but its template image is not learnt by the machine.**

4.2 Discussions

In order to complete Sirius Project which gives a high accuracy in object recognition, several experiments are done. Firstly, a few template matching methods such as SURF and Squared Difference (SQDIFF) template matching are used in order to compare the accuracy of object recognition. It is found that SURF method produces results with higher accuracy compared to SQDIFF method. Experiment is also done by combining both SURF and SQDIFF methods. However, the accuracy becomes worse. Besides, background subtraction is tried to combine with SURF to increase the accuracy. It is planned to extract the moving object from the static video before analyzing the moving object. But similar to SQDIFF, the desired accuracy is not met. Hence, it is decided to use only SURF method for object recognition.

However, there are some other factors which affect the accuracy of object recognition using SURF method. All these factors become the limitation of Sirius Project. However, the accuracy of Sirius Project is still high. It is because the factors do not affect the accuracy as much as the factor of rotation. Firstly, the scale of the object will slightly affect the accuracy. When the object is captured from far, it appears very small in the image. Hence, number of key points will decrease and they scatter to the background images. Besides, since the frame rate is high, many frames will be captured and if the object is moving fast, it causes the image to be blur. Therefore, the object cannot be recognized. Another factor is affined image. If the object captured is significantly distorted, it is difficult for it to be recognized. Figures 4.17 and 4.18 show the failed object detection cases due to different factors. It can be seen that there are only a few red color points in the image and the number of red color points does not exceed the threshold value set.

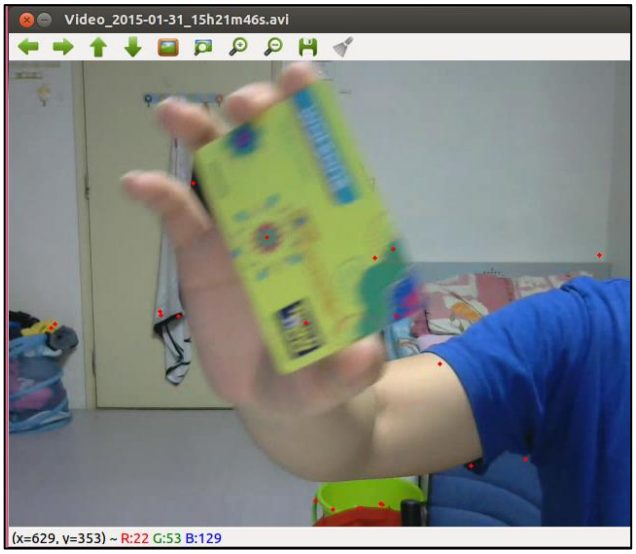


FIGURE 4.17. Blur Image Recognition 1

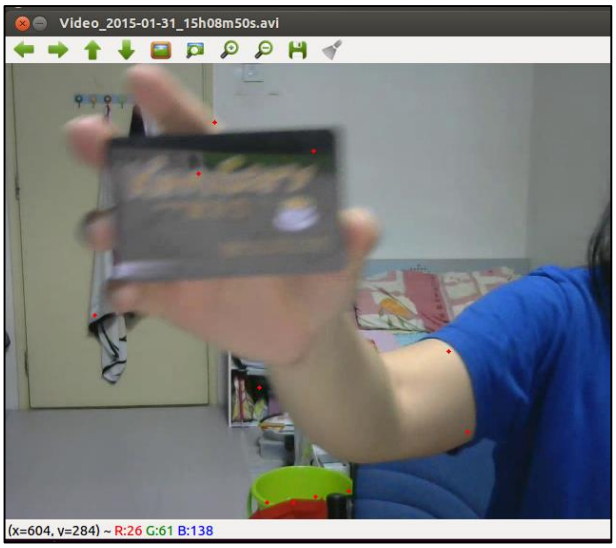


FIGURE 4.18. Blur Image Recognition 2

Apart from that, SURF algorithm is dependent on the complexity of background. Since the key points are extracted according to the colorful spots, corners, etc, a complex background might cause the result to be inaccurate. Figure 4.19 shows an inaccurate location detection due to the complex background. It can be seen that the green dot is totally out of the object. Even though the identity of object is recognized correctly, the location might give some error.

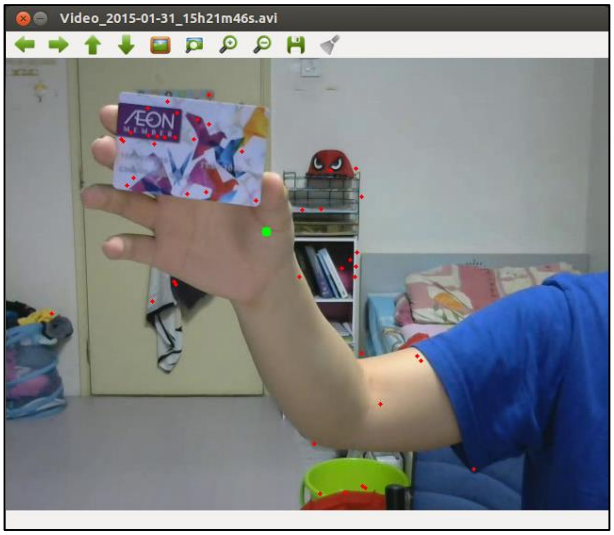


FIGURE 4.19. Wrong Location Detected

To make the system more user friendly, the source codes are included in the init-scripts so that the source code will start running as soon as the machine is booted up. The source code includes the process of setting up IP address and initializing GPIO pin. Hence, at the client side, monitor is not needed once the source code is completely done. Besides, multiple videos can be opened during the process of object recognition. There are cases when the new images are captured while the previous video is still in the progress of analytic. Therefore, in order to reduce the delay time, multiple windows will pop out so that two or more videos can be analyzed at the same time. However, the increase of number of windows will increase the processing time because the processor is too busy to manage all the tasks. On the other hand, in terms of time spent, building a Yocto OS which is compatible to Raspberry Pi needs around three hours depending on the speed of internet connection. On the other hand, 5 minutes are needed to complete the object recognition process for one video which contains of 50 frames. However, the processing time will increase if the number of template image increases.

Furthermore, in order to determine the accuracy of the object recognition algorithm, a validation process is done. Different template images are inserted into a static background video randomly. The resolution, size, rotation angle and the location of the template images are also adjusted randomly. While inserting the template

images frame by frame, an expected result of the objects' identity is generated. The modified video is then being fed into the object recognition algorithm to get the real result. Both results are compared to calculate the accuracy of the system. Figure 4.20 shows the flow chart of the validation process while Figures 4.21 and 4.22 illustrate the video frames which the template image has already being inserted randomly.

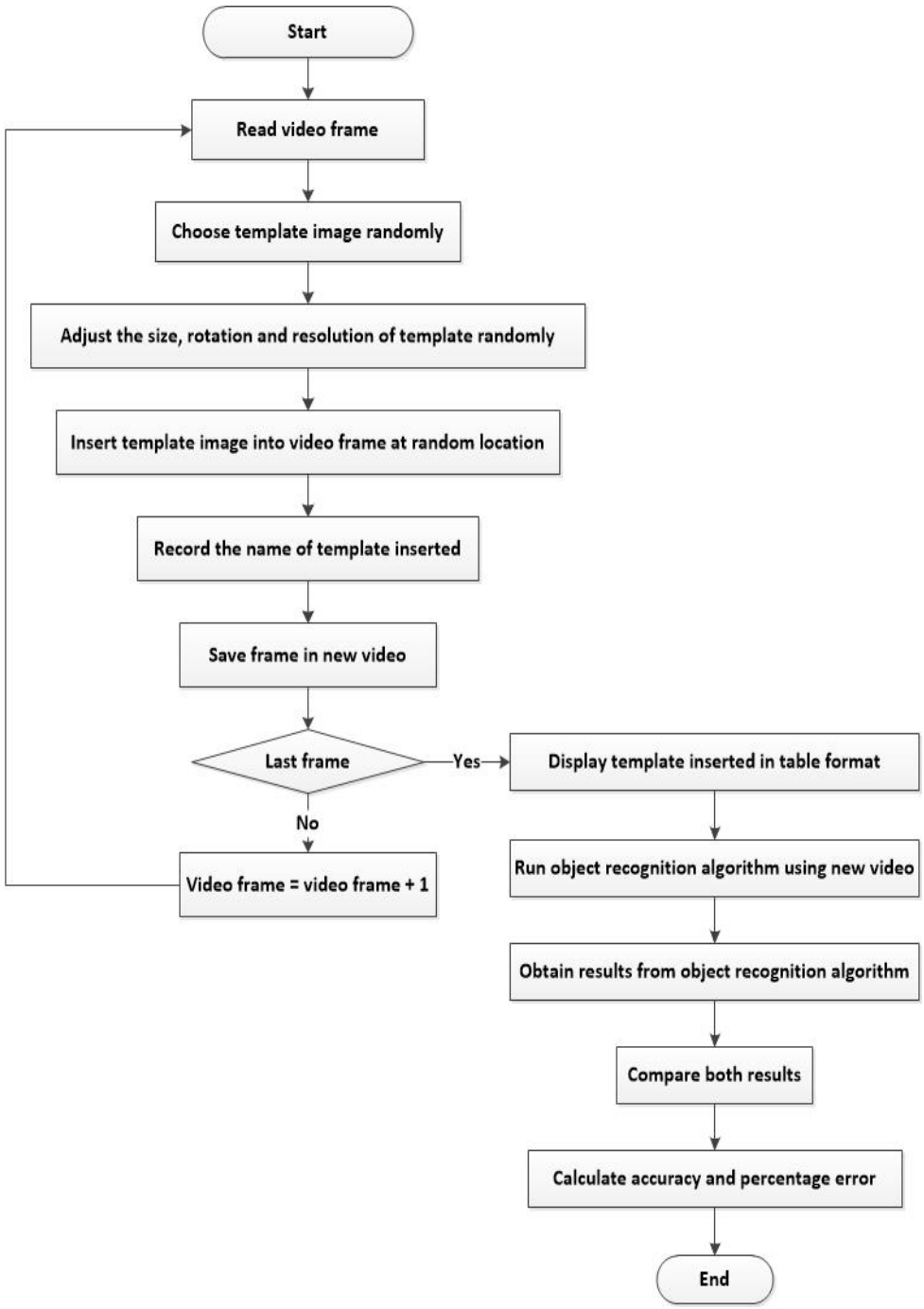


FIGURE 4.20. Object Recognition Validation Flow Chart

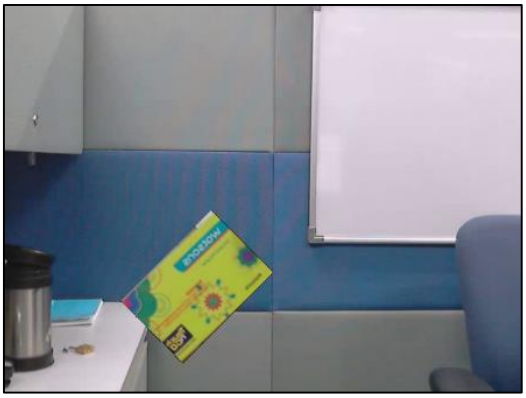


FIGURE 4.21. Video Frame with Template Image Inserted

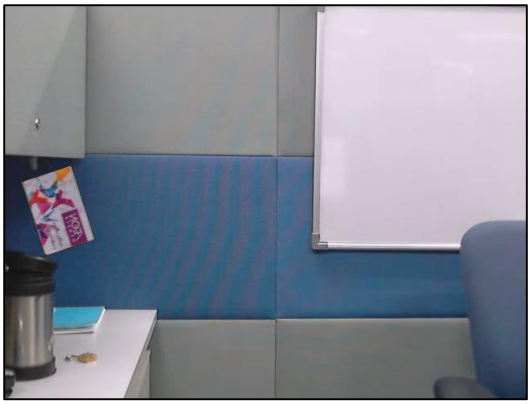


FIGURE 4.22. Video Frame 2 with Template Image Inserted

Based on the comparison between expected and real result, it is found that there are four scenarios that will occur during the process of object recognition. The percentage of four scenarios are analyzed and the accuracy of object recognition algorithm is calculated using a video consisting 242 frames. It is found that the accuracy for each run varies due to the random properties of template image. However, the accuracy is always above 85%. In this case, the accuracy is 88.43% while the percentage error is 11.57%. The summary of accuracy is shown in Table 4.2.

TABLE 4.2. Summary of Accuracy

Scenario	Description	Percentage
Correct Recognition	- No object is captured and results show "N/A". - Object is captured and the exact object is recognized.	$\frac{214}{242} \times 100\% = \mathbf{88.43}\%$
Wrong Recognition	- No object is captured but results show that something is recognized.	$\frac{2}{242} \times 100\% = \mathbf{0.83}\%$
Missed Recognition	- Object is captured but results show "N/A".	$\frac{0}{242} \times 100\% = \mathbf{0}\%$
Extra Recognition	- Object is captured but results show another object is being recognized.	$\frac{26}{242} \times 100\% = \mathbf{10.74}\%$

4.3 Project Deliverables

Sirius Project is delivered as a system controlled by software programs. Although hardware components are needed, Sirius Project is still depending on the source code so that it can be run successfully. Sirius Project needs to be completed by a specific deadline, which is the period of 28 weeks. Therefore, there are still limitations in the system. However, Sirius Project can be improved in the future and delivered in phases or releases. Different features can be added into the system to enhance the function of the project.

After Sirius Project is done, a presentation is conducted in order to demonstrate the system functionality to the project supervisor as well as internal and external examiners. The usage, strengths and weaknesses of the system are explained to the audiences. Besides, all the files and source code are well version controlled by “git” tool.

CHAPTER 5

CONCLUSION AND RECOMMENDATION

5.1 Conclusion

Sirius Project involves both software and hardware implementations. Two machines are connected to each other for the purpose of messaging and data transferring. This connection relates to the concept of M2M communication. Sirius Project is able to improve the effectiveness and efficiency of current video surveillance system available in the market by reducing the initial cost and including the feature of object recognition. Human eyes are removed from object recognition since it is performed automatically by the system. The system helps in assisting police to find the suspects all around the world. Sirius Project will be delivered as a full-fledged functioning system controlled by software programs written in both C++ language and Python scripts. At the end of this project, student is expected to have well understanding about Yocto Project, Raspberry Pi, M2M, MQTT, OpenCV, SURF algorithm, etc. All hardware connections are done completely. It is proven that the storage and memory requirement of Yocto OS are smaller than the Raspbian Image. This helps to reduce the initial cost of the system. Besides, sensor inputs can be read successfully and the image capturing process is performed successfully. The validation results show that the accuracy of object recognition algorithm is always higher than 85%. All the objectives are achieved.

5.2 Recommendations

Sirius Project can be further enhanced in the future by:

- **Improving the accuracy of object recognition**

  Based on the result of validation process, it is found that the accuracy of object detection algorithm is always higher than 85%. Although the accuracy is high, it should be improved to a better system. Since there are several methods in template matching, feature matching and object recognition, the suitable methods can be tried to combine so that a better result can be yield. The percentage of Wrong Detection, Missed Detection and Extra Detection should be reduced. After future expansion, Sirius Project should be able to handle the case of low resolution and affined object. Besides, it should be able to differentiate objects even though they are similar in appearance such as shape and color.

- **Improving the processing speed of object recognition**

  The processing speed of object recognition algorithm takes some time. Multi-thread or Graphics Processing Unit (GPU) are some of the techniques to speed up the whole process. This helps to reduce the delay time so that actions can be taken immediately.

- **Adding user alarm/alert feature**

  Similar to the current alarm system, if the alarm rings for a certain time, a message will be sent to the user or even the police. Hence, this type of feature should be included in Sirius so that if certain objects or features are recognized, the user and security guard will be informed so that actions can be taken immediately.

- **Adding voice recording feature**

  Voice recording feature helps in determining what is happening in the video. If the video captured is silent, users can only see the movement and actions of the object or person being captured. Everything is only a prediction. If voice is recorded together with the video, users can hear the sound and if someone is talking, users will be able to understand the contents of the dialogue. Hence, it helps in investigating the scenario more easily.

REFERENCES

[1] LINUX FOUNDATION. (2013). Yocto Project. [Online]. Available: https://www.yoctoproject.org/about

[2] Rudolf Streif. (2011, Aug.). Yocto Project – Big in Embedded Linux. [Online]. Available: http://eecatalog.com/embeddedlinux/2011/08/23/yocto-project-big-in-embedded-linux/

[3] Linux Foundation. (2014). Yocto Project Quick Start. [Online]. Available: https://www.yoctoproject.org/docs/current/yocto-project-qs/yocto-project-qs.html

[4] U. Hunkeler, T. Hong Linh, and A. Stanford-Clark, "MQTT-S — A publish/subscribe protocol for Wireless Sensor Networks," in *Communication Systems Software and Middleware and Workshops, 2008. COMSWARE 2008. 3rd International Conference on*, 2008, pp. 791-798.

[5] G. Nalin, "Orchestration of smart objects with MQTT for the Internet of Things," master degree dissertation, Dept. Information Engineering, University of Padua, Padoba, Italy, 2014.

[6] K. Pulli, A. Baksheev, K. Kornyakov, and V. Eruhimov. (2012, June). Real-Time Computer Vision with OpenCV. *Communications of the ACM*. [Online]. 50 (6), pp. 61-69. Available: http://dl.acm.org/citation.cfm?id=2184337&picked=formats&CFID=630781323&CFTOKEN=61987775

[7] Secure copy. (n.d.) In Wikipedia. Retrieved February 10, 2015, from http://en.wikipedia.org/wiki/Secure_copy

[8] C. Dian, J. Chun-hua, H. Zong-bo, and J. Wei, "The Design and Implementation of Video Surveillance System Based on H.264, SIP, RTP/RTCP and RTSP," in *Computational Intelligence and Design (ISCID), 2013 Sixth International Symposium on*, 2013, pp. 39-43.

[9] I. Culjak, D. Abram, T. Pribanic, H. Dzapo, and M. Cifrek, "A brief introduction to OpenCV," in MIPRO, 2012 Proceedings of the 35th International Convention, 2012, pp. 1725-1730.

[10] Y. Caiyan, Z. Xiaoshi, Z. Yanling, L. Guangqi, and L. Na, "Review of intelligent video surveillance technology research," in *Electronic and Mechanical Engineering and Information Technology (EMEIT), 2011 International Conference on*, 2011, pp. 230-233.

[11] I. Masmoudi, M. El'arbi, and C. B. Amar, "Vocabulary Tree schema based on SURF descriptor for real time object detection and recognition in video," in *Hybrid Intelligent Systems (HIS), 2013 13th International Conference on*, 2013, pp. 134-139.

[12] P. M. Panchal, S. R. Panchal and S. K. Shah. (2013, Apr.). A Comparison of SIFT and SURF. *International Journal of Innovative Research in Computer and Communication Engineering*. [Online]. 1 (2), pp. 323-327. Available: http://www.ijircce.com/upload/2013/april/21_V1204057_A%20Comparison_H.pdf

[13] A. Ganesh, and K. Vudata. (2012). SCAVENGAR HUNT. [Online]. Available: https://www.ece.cmu.edu/~ee551/projects/S12/Final_Report_Group7.pdf

[14] L. Juan, and O. Gwun. (2009). A Comparison of SIFT, PCA-SIFT and SURF. *International Journal of Image Processing (IJIP)*. [Online]. 3 (4), pp. 143-152. Available: http://www.cscjournals.org/library/manuscriptinfo.php?mc=IJIP-51

[15] Abhinab Choudhury. (2011, Dec.). Waterfall Model. [Online]. Available: http://www.sdlc.ws/waterfall-model/

[16] Tutorialspoint. (2014). SDLC Waterfall Model. [Online]. Available: http://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm