

**AUGMENTED REALITY BASED INDOOR POSITIONING
NAVIGATION TOOL**

By

Muhammad Fadzly Bin Abdul Malek

16416

DISSERTATION

Submitted to the Electrical & Electronics Engineering Department

in Partial Fulfilment of the Requirements

for the Degree

Bachelor of Engineering (Hons)

(Electrical & Electronics Engineering)

JANUARY 2016

Universiti Teknologi PETRONAS

Bandar Seri Iskandar

31750 Tronoh

Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

**AUGMENTED REALITY BASED INDOOR POSITIONING
NAVIGATION TOOL**

By

Muhammad Fadzly Bin Abdul Malek

16416

DISSERTATION

Submitted to the Electrical & Electronics Engineering Department

in Partial Fulfilment of the Requirements

for the Degree

Bachelor of Engineering (Hons)

(Electrical & Electronics Engineering)

JANUARY 2016

Approved by,

(Mr.Patrick Sebastian)

Universiti Teknologi PETRONAS

Perak Darul Ridzuan

JANUARY 2016

CERTIFICATE 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.

(MUHAMMAD FADZLY BIN ABDUL MALEK)

ABSTRACT

Nowadays, indoor navigation gained people's attention. Lots of techniques and technologies have been used in order to develop the indoor navigation. Indoor navigation is far away behind the outdoor navigation. For outdoor navigation, we have GPS to guide and give direction to the desired place. Unfortunately, it is restricted for the outdoor purpose only. Thus, the main objective of this project is to develop an interactive indoor navigation system and augmented reality is being use to superimposed the directional signage. In this project small computer which is Raspberry Pi has been used as a computing device. Probably in the future, all smartphones will have augmented reality based indoor navigation tools because it already equipped with many sensors such as an accelerometer, gyro, and compass which will improve the accuracy of positioning. Basically, the project has been tested at Universiti Teknologi PETRONAS's Information Resource Centre (IRC), and it has shown its flexibility in working as an indoor positioning tool to navigate to 5 different locations with multiple levels.

ACKNOWLEDGEMENTS

I would like to take this opportunity to gratitude to all persons that helped me a lot throughout completing this project. First of all, I would like to thank to my supervisor, Mr. Patrick Sebastian for his support and guidance in making this project successful. He has contributed a lot in this project in term giving idea and suggestions in order to make the outcome of this project outstanding.

I also would like to thanks to my friends including post graduates student from Electrical & Electronics department for guiding me on how to use Raspberry Pi board as well as installation of ARToolKit software.

Not forgotten, my family members, thank you for supporting me in my academic.

TABLE OF CONTENTS

CERTIFICATION OF APPROVAL.....	ii
CERTIFICATE OF ORIGINALITY.....	iii
ABSTRACT	iv
ACKNOWLEDGEMENTS	v
 CHAPTER 1: INTRODUCTION	
1.1 Background of Study	4
1.2 Problem Statement	5
1.3 Objectives and Scope of Study	5-7
 CHAPTER 2: LITERATURE REVIEW	
2.1 Augmented Reality	8
2.2 Basic Principle of ARToolKit.....	9
2.3 Concept and related work.....	9-11
2.4 Critical Analysis.....	11
 CHAPTER 3: WORK SCHEDULE	
3.1 Project Key Milestones and Gantt Chart.....	12-14
 CHAPTER 4: METHODOLOGY	
4.1 Research Methodology.....	15
4.2 Project Activities.....	16
 CHAPTER 5: RESULT AND DISCUSSION	
5.1 Tools and Hardware Connections.....	17-18
5.2 Start-up Program.....	18
5.3 Recognize Multiple Patterns.....	19-22
5.4 Route Planner Algorithm.....	23-25
5.5 Camera Parameter.....	26-27
5.6 Creating and training new marker.....	28
 CHAPTER 6: OVERALL OPERATION OF COMPLETE SYSTEM	
6.1 Flow Chart of Overall Operation.....	29-30
6.2 Flow Chart of Audio Module.....	31
 CHAPTER 7: CONCLUSION.....	32
 REFERENCES	33
APPENDICES	34-56

LIST OF FIGURES

Figure 1: Indoor navigation using AR [1]	4
Figure 2: Example of OpenGL programming	7
Figure 3: virtual image of lamp and chairs	8
Figure 4: Principle of ARToolKit [9]	9
Figure 5: Overview of project proposed in [2]	10
Figure 6: Virtual 3D map of Sunway University ground floor over a physical floor plan in [10]	11
Figure 7: Tools required	17
Figure 8: Selection of the desired location by user	18
Figure 9: Marker pattern 2 showing to the right direction.....	20
Figure 10: Marker pattern 3 showing to the left direction	21
Figure 11: Multiple markers recognition	21
Figure 12: Camera parameter setting	26

LIST OF TABLES

Table 1 : Comparison of various positioning technologies [3]	5
Table 2: Marker patterns declaration.	20
Table 3: Legend	23
Table 4: Route planner	24
Table 5: ARToolKit Camera Setting	27

LIST OF ABBREVIATIONS

AR	: Augmented Reality
IRC	: Information Resource Centre
GPS	: Global Positioning System
WLAN:	Wireless Local Area Network
RFID	: Radio Frequency Identification RFID
UTP	: Universiti Teknologi PETRONAS

CHAPTER 1

INTRODUCTION

1.1 Background of Study

Recently, Augmented Reality (AR) is getting popular and captured people's attention. Thus, a lot of research and development had been carried out in order to bring this technology to a new level. Basically, the aim of this project is to develop an AR Based Indoor Positioning Navigation Tool on raspberry pi which is an open source electronic board. At the same time, ARToolkit software is used to create a virtual image on the user's real view where the image will be displayed on the detected marker. A different pattern of the marker will create a different virtual image. So this concept will be used to show direction or path to the user as shown in Figure 1 below.



Figure 1: Indoor navigation using AR [1]

1.2 Problem Statement

A navigation tool such as Global Positioning System (GPS) is widely used for navigation purposes. However, it is restricted to outdoor navigation due to the satellite signals being restrained by the structure of the building [2, 3]. The available solutions to this problem are by using wireless technology such as GSM, Bluetooth, infrared, Wireless Local Area Network (WLAN) and Radio Frequency Identification (RFID). However, based on the comparison shown in Table 1 [3], it shows that these positioning technologies are not effective to be implemented for indoor navigation.

	GPS	GSM	WLAN	Bluetooth	Infrared	RFID
Accuracy	No signal	Low	Low	Low	High	High
Signal error rate	No signal	Low	Mid	Low	Lowest	Lowest
Range	Wide range	Wide area	Micro area	Micro area	Pico area	Pico area

Table 1 : Comparison of various positioning technologies [3].

Using wireless technology as indoor localisation has limitation on area or signal coverage. As a consequence, indoor navigation is still relying on the conventional and non-interactive way of navigation such as signboard and map around the building. However, the conventional indoor navigation does not guide the user directly like GPS.

1.3 Objective and Scope of Study

The objectives of this project are:

- To develop an interactive way of indoor navigation using a raspberry pi.
- To use the idea of AR in indoor navigation tools.

Scope of study:

1.3.1 Raspberry pi :

The Raspberry Pi is a low cost , a small sized computer which enables people to explore more about computing as well as learn programmings such as Scratch, Python, and C language [4]. Not just that, it also capable of doing things like a normal desktop computer can do. Raspberry pi has come out with several boards such as Raspberry pi (RPI) Model A+, RPI Model B, RPI Model B+ and the latest one is RPI 2 Model B. Basically, these boards can support mainly Linux based operating system such as Raspbian, NOOBS, Ubuntu Mate and Snappy Ubuntu Core as well as Windows 10 IOT core.

1.3.2 Augmented Reality ToolKit (ARToolKit):

ARToolKit is an open source software use to develop AR application. It uses C and C++ language. The function of this software is to create a virtual graphical image in the real world. The features of this software are single camera orientation tracking, tracking simple black squares, ability to use any square marker patterns, easy camera calibration code [5] and multiple languages supported. Besides that, this software is multi-platform where it can run on Linux, Windows, iOS, and android.

1.3.3 OpenGL and GLUT:

Glut is actually the utility toolkit for OpenGL and it makes the learning and exploring OpenGL programming become easier. It provides a portable API so that the OpenGL program can work across all PC and workstation OS platforms [6]. Meanwhile, the OpenGL is a multi-platform application programming interface (API) used for 2D or 3D graphic renderings such as a cube, polygon, cone and much more. So in this project, OpenGL will be used to draw an arrow as a direction indicator to the user. The figure below shows an example of Object generated by using OpenGL programming.

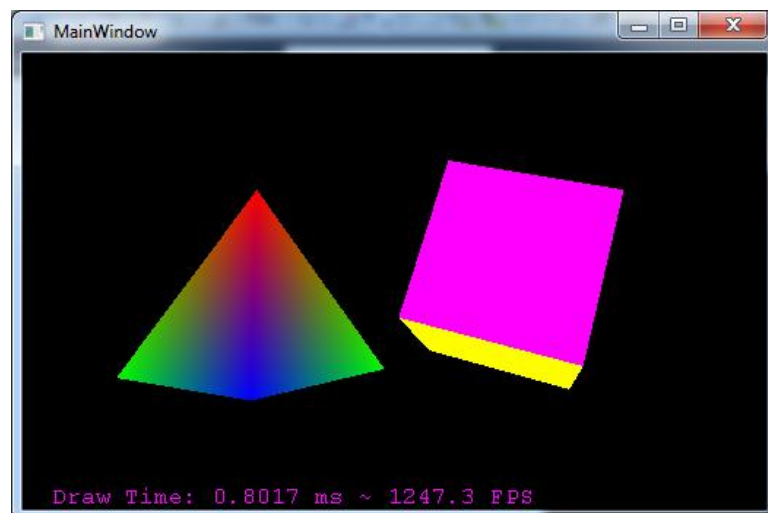


Figure 2: Example of OpenGL programming

CHAPTER 2

LITERATURE REVIEW

2.1 Augmented Reality:

Nowadays, technology is developing rapidly and a lot of new technology has been introduced especially for navigation. Although the navigation technology is advancing but it only restricted to outdoors only. With the increasing of the complexity of building internal design, indoor navigation has become a hot topic than outdoors. A study in [7] shows that the advancement of smartphones technology in combination with AR as an emerging technology has capabilities of creating new indoor navigation tool in the future.

The most common software used to build AR application is ARToolKit. It was developed by Dr.Hirokazu and supported by many institutions such as the University of Washington Human Interface Technology Laboratory (HIT Lab), the University of Canterbury, (HIT Lab Newzealand) and Seattle's ARToolworks, Inc [8]. The Figure 1 below is an example of augmented reality. The table and telephone are real meanwhile, the table lamp and chairs are virtual images generated by the computer.



Figure 3: virtual image of lamp and chairs

2.2 Basic Principle of ARToolKit:

This software works based on the shape of the black squares as tracking markers. Firstly, the camera will capture video and sent it to the computer. At the same time, the software will search for square shapes from the video frame. Once the square shape is found, the software will calculate the distance between the camera and the black square. After that, the computer graphic model is drawn on the top of the video of the real world from the determined position. Basically, this is how ARToolKit works and all steps above are summarized in the figure below.

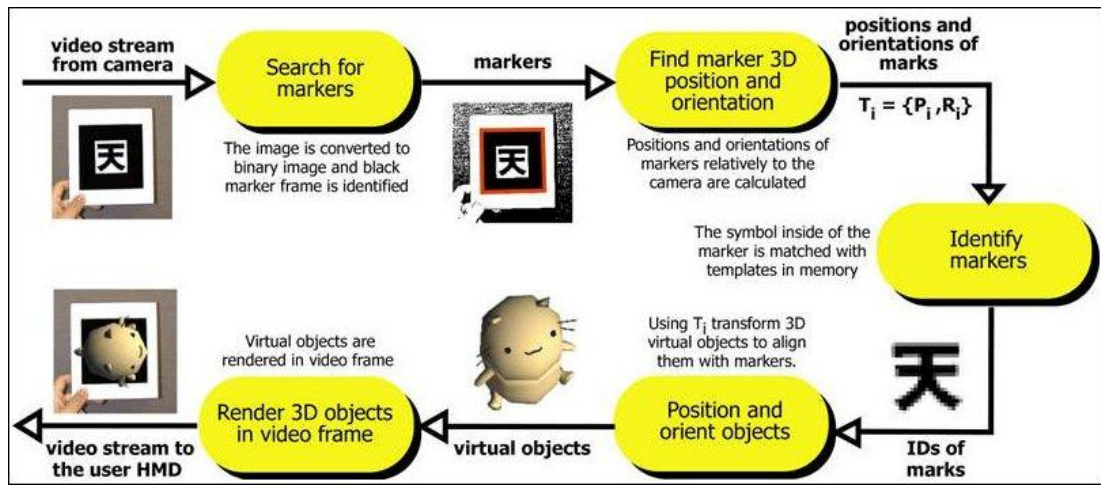


Figure 4: Principle of ARToolKit [9]

2.3 Concept and Related Work:

Recently, there are many projects have been done related to vision-based indoor navigation and the use of augmented reality. For instance, authors in [2] implement indoor navigation on a laptop using ARToolKit software with USB camera installed to face the front view. Besides that, the author also includes audio module along with the direction displayed by the markers to guide the user to reach the desired destination. The webcam will continuously capture the live video frame and sent it to the laptop with ARToolKit installed. Then the ARToolKit's marker recognition module will look for a marker from the frame. If the marker is detected then it will send the marker ID to the route planner module. After that, the module will give direction to go to the user's desired location based on the location input by the user at the beginning of the program. Figure 4 shows the overview of the indoor localisation project proposed in [2].

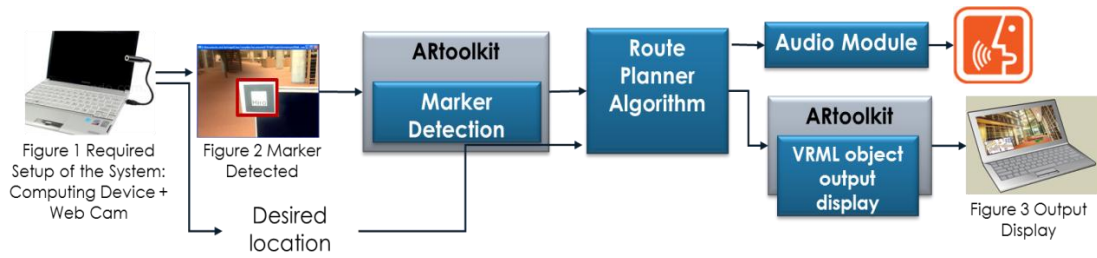


Figure 5: Overview of project proposed in [2]

Another related project is also had been carried out by Sunway University which is in [10]. In this research paper, the author utilized sensors in the smartphone such as gyro, compass, accelerometer and camera to enable the navigation system for the indoor environment. Thus, an android application called SunMap+ has been developed to showcase the possibilities of indoor navigation using AR around Sunway University campus. This software will guide the user to the intended destination using AR concept and it can also model a 3D map of Sunway University campus by referencing a floor plan of the campus.

Furthermore, image recognition technique is used in the project in order to identify the current location of the user. The software that provides image recognition and process capabilities for SunMap+ application is Vuforia SDK developed by Qualcomm. The development platform for this software is based on Unity 3D while the previous project in [2] use ARToolkit as the development platform. For this project, the author decided to use ARToolKit because of its versatility and coverage in installing to most major platforms and devices. Basically, that is the software requirement for this project.

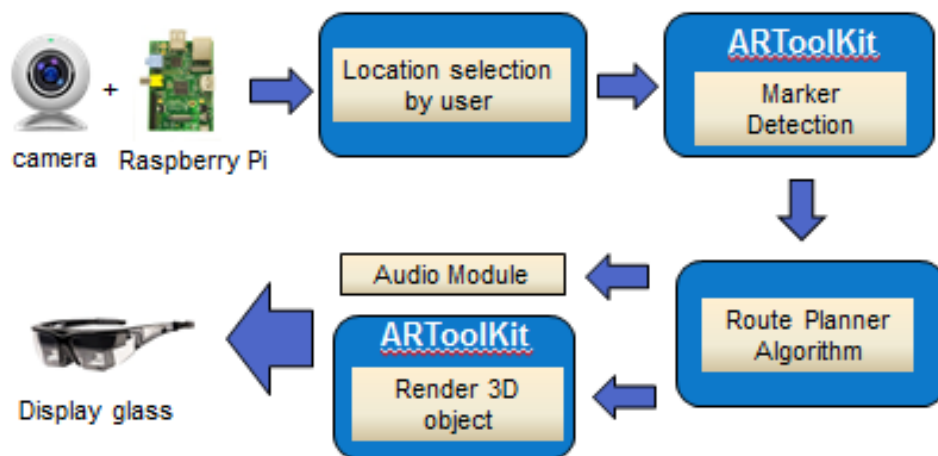
Meanwhile, the hardware required accelerometer, compass and camera in order to ensure the SunMap+ app can function effectively. The camera will capture live video while the compass provides directional orientation. Meanwhile, the accelerometer will track the user's movement by recording the number of steps through PDR developed in the project.



Figure 6: Virtual 3D map of Sunway University ground floor over a physical floor plan in [10].

2.4 Critical Analysis:

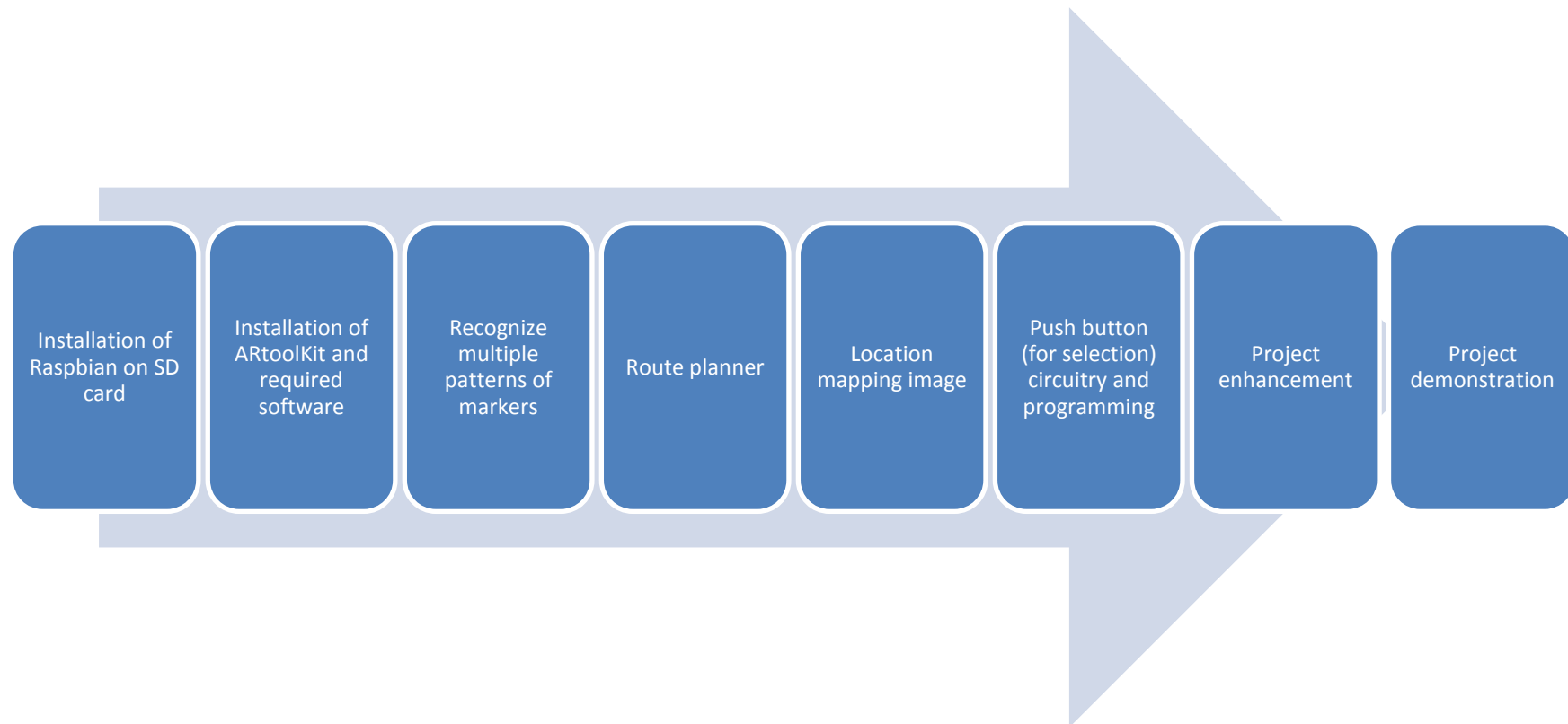
The difference between the project in [2] and [10] are the computing device that has been used either laptop or smartphone and the AR software development. The project carried out by Universiti Teknologi Petronas was using ARToolKit as the AR software development while the project carried out by Sunway University was using Unity 3D. So for this project, ARToolkit will be used because of its compatibility with Linux operating system and it is open source software. This project is almost similar to the project carried out in [2] but the differences are it is implemented on a small computer which is Raspberry Pi and it is using Linux operating system. The overview of this project is shown in the process flow below.



CHAPTER 3

WORK SCHEDULE

3.1 Project Key Milestones and Gantt Chart:



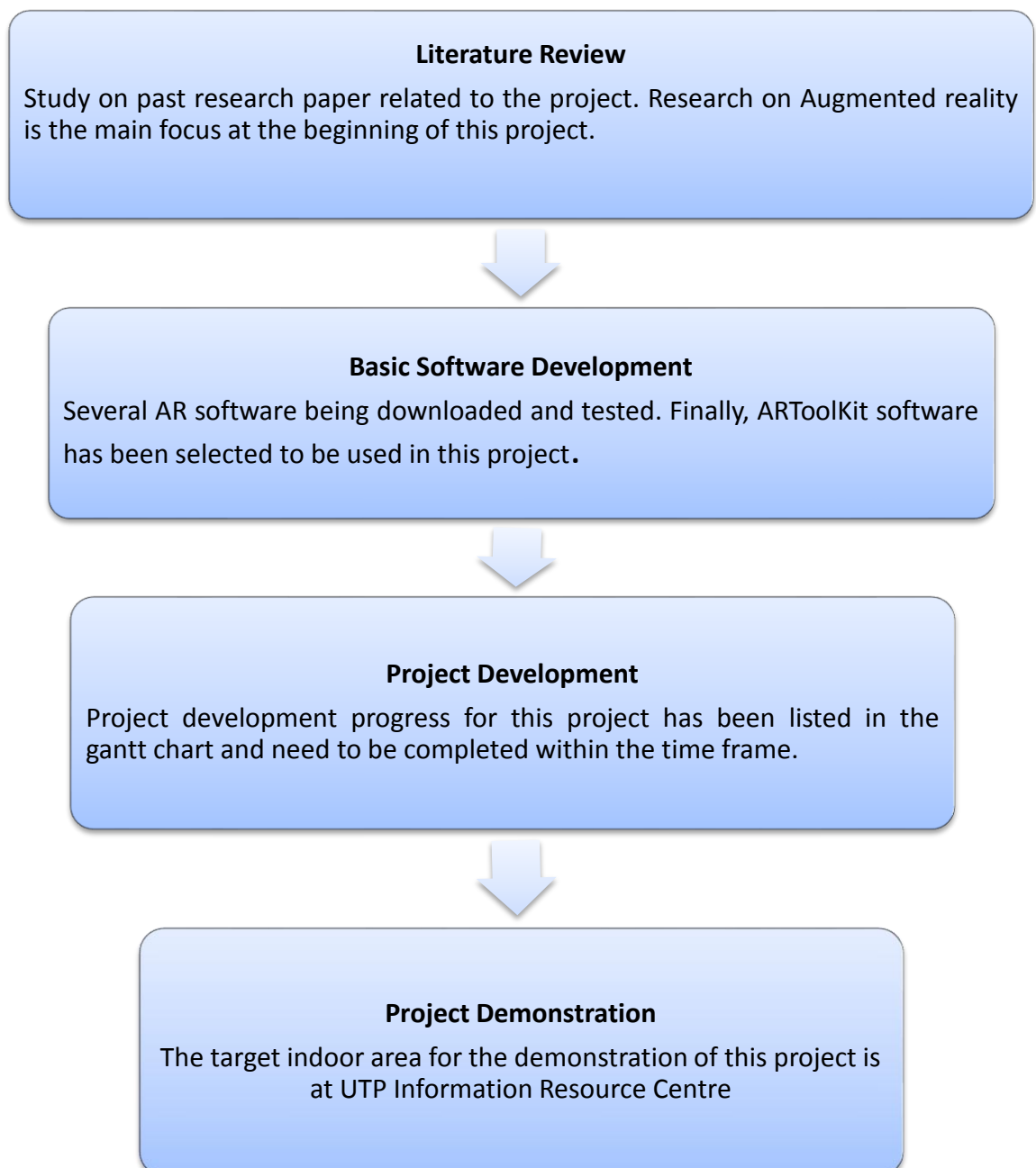
Project Gantt Chart for FYP 1															
No	Detail/Work	Week													
		1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	Selection of project topic														
2	Installation of Raspbian on SD card														
3	Experience and testing Raspberry Pi board and its accessories														
4	Installation of ARtoolKit and required software														
5	Submission of Extended Proposal														
6	Recognize multiple patterns of markers														
7	Route planner														
8	Proposal Defense (Presentation)														
9	Submission of Interim Draft Report														
10	Submission of Interim Report														

Project Gantt Chart for FYP 2															
No	Detail/Work	Week													
		1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	Location mapping image														
2	Submission of Progress Report														
3	Push button (for selection) circuitry and programming														
4	Project enhancement														
5	Pre-SEDEX														
6	Submission of Draft Final Report														
7	Submission of Dissertation (soft bound)														
8	Submission of Technical Paper														
9	Viva														
10	Submission of Project Dissertation (Hard Bound)														

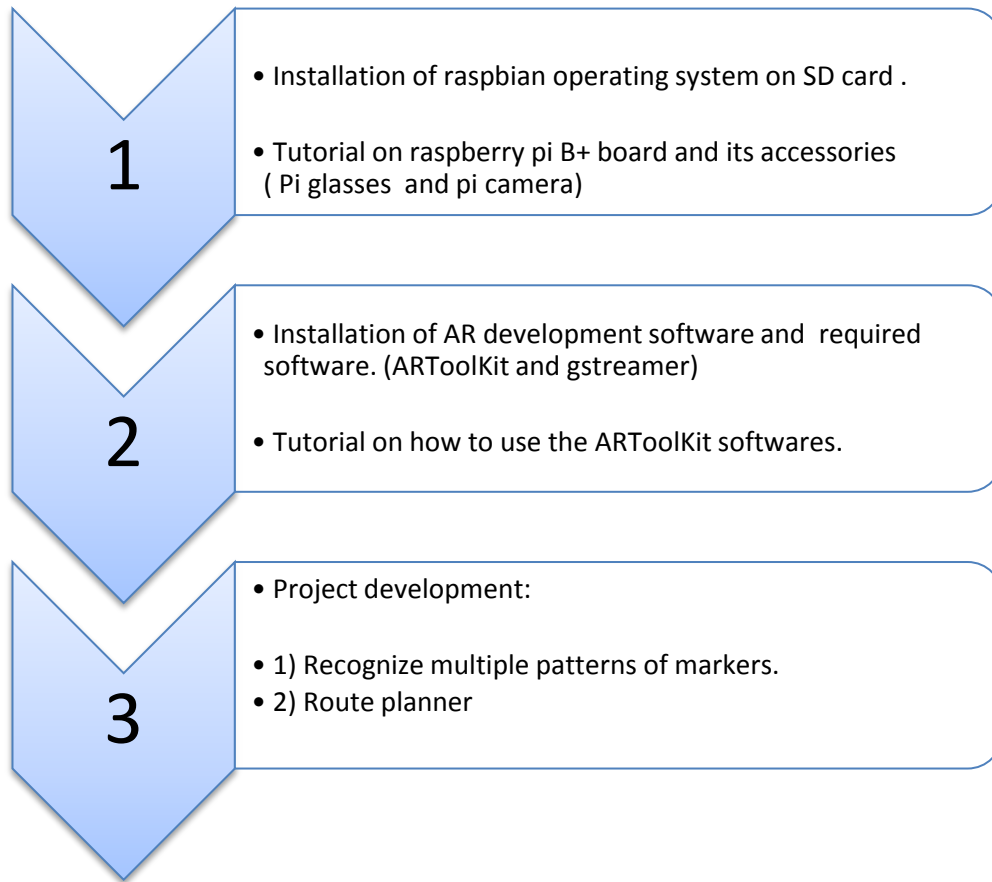
CHAPTER 4

METHODOLOGY

4.1 Research Methodology:



4.2 Project Activities:



There are 3 main activities had been completed in FYP 1 as shown in the chart above. The first step is to install the Raspbian operating system and before you able to install it on SD card you have to install SD card formatter and Windows32 Disk Imager software on your computer. Besides that, you also need a Raspbian image file that can be downloaded from Raspberry Pi official website. Then, use Windows32 Disk Imager software to write the image file on the SD card.

After that, we can proceed with ARToolKit software installation. First of all, we have to download the installer from the official website. In this project, ARToolKit version 2.72.1 is used. As usual before installing this software, you have to install some dependencies such as freeglut3, OpenGL, and GStreamer. These dependencies are essential to be installed before installing ARToolKit especially GStreamer otherwise you will face some problem later such as the software cannot open the USB cam. The detail instruction about the installation process is attached in the appendix.

CHAPTER 5

RESULT AND DISCUSSION

5.1 Tools and Hardware Connections:

The figure below basically shows the required tools for this project and its connection as well as explaining the function of the respective tools.

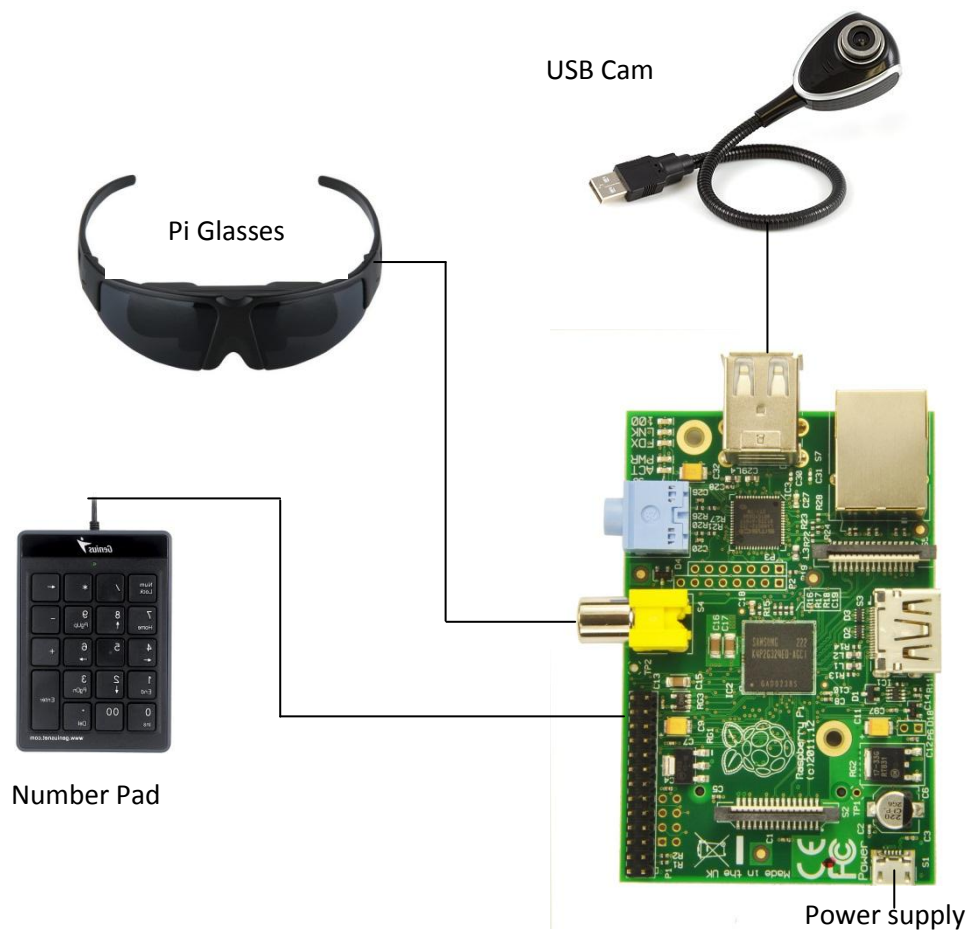


Figure 7: Tools required

The tools required for this project are one PI Glasses which will be connected to RCA video out port, several push buttons that connected to GPIO pins and a USB camera. The USB cam will continuously capture live view frame and the user can see the live video captured by the camera through the Pi Glasses. Meanwhile, the push button will be used by the user at the beginning of the program to select the desired destination. For further reference, the layout of the Raspberry Pi board was attached in the appendix section.

5.2 Start-up Program:




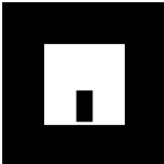




```
pi@raspberrypi ~/installation-scripts-master/ARToolKit/bin $ ./loadMultiple
Please select your destination (based on the genre of books):
1. Circulation
2. History & politics
3. music & art
4. science & engineering
5. lifestyle & religion
>>>>1
```

Figure 8: Selection of the desired location by user

First of all, in order to run this program you have to go to the bin directory using the terminal as shown in the figure above and type `./loadMultiple`. At the beginning of the program, the user has to choose his or her desired destination. There are five locations all together which are Circulation, History & Religion, Music & Art, Science & Engineering and lastly Lifestyle & Religion. After the selection has been made by the user, then the camera will start capturing live video and will search for a marker. Not just that, the user can continue navigating by pressing “C” button and if the user wants to exit this program just press “E” button.

5.3 Recognize Multiple Patterns:

Currently, nine markers with different patterns are used to represent each destination as shown in the table below.

Marker pattern	Destination
<u>Pattern 1</u> 	Circulation
<u>Pattern 2</u> 	History & Politics
<u>Pattern 3</u> 	Music & Art
<u>Pattern 4</u> 	Checkpoint 1
<u>Pattern 5</u> 	Checkpoint 2
<u>Pattern 6</u> 	Checkpoint 3




<u>Pattern 7</u> 	Checkpoint 4
<u>Pattern 8</u> 	Science & Engineering
<u>Pattern 9</u> 	History & Religion

Table 2: Marker patterns declaration.

Not just that, the program is also able to distinguish between markers by giving indications in green colour or red colour. The green colour indicates that the software recognizes the marker and it has its information in the marker database. Meanwhile, Red colour indicates that the software does not recognize the marker and has no information about the marker in the database.



Figure 9: Marker pattern 2 showing to the right direction



Figure 10: Marker pattern 3 showing to the left direction

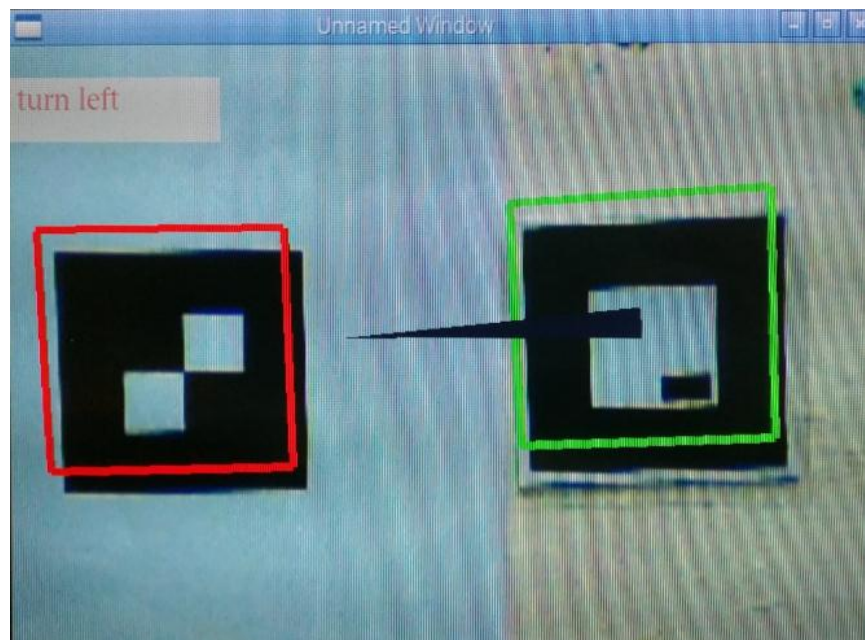


Figure 11: Multiple markers recognition

Basically, each marker has its own unique binary information which differentiates it with other markers. Actually, the pattern of each marker is what makes the black square shape unique between each other. The binary information of the marker is save using “patt.MARKERNAME” format and it is stored in the marker

database. Firstly, when the camera detects a marker, the image will be converted into a binary image. After that, the ARToolKit software will identify the marker's pattern. If the pattern matches with the templates in the marker database, then the software will render a virtual image that appears overlaid on the tracking marker.

5.4 Route Planner Algorithm:

The table below shows the route planner for navigating around IRC based on five locations which are Circulation, History & Religion, Music & Art, Science & Engineering and lastly Lifestyle & Religion whereby the route planner will generate a route to the targeted location decided by the user. There are two important variables here which are L and D, L stands for location and continuously updated when the camera detect different markers along the way while D is destination or targeted location.

Code	Indication
1	Circulation
2	History & Politics
3	Music & Art
4	Checkpoint 1
5	Checkpoint 2
7	Checkpoint 3
8	Science & Engineering
9	Lifestyle & Religion
	Exactly at the particular point
	Turn Back/ Make U-turn
	Go forward
	Turn right
	Turn left
	Go upstairs
	Null

Table 3: Legend


















































































$\begin{matrix} D \\ L \end{matrix}$	1	Arrow	2	Arrow	3	Arrow	4	Arrow	5	Arrow	6	Arrow	7	Arrow	8	Arrow	9	Arrow
1	11		12		13		14		15		16		17		18		19	
2	21		22		23		23		25		26		27		28		29	
3	31		32		33		34		35		36		37		38		39	
4	41		42		43		44		45		46		47		48		49	
5	51		52		53		54		55		56		57		58		59	
6	61		62		63		64		65		66		67		68		69	
7	71		72		73		74		75		76		77		78		79	
8	81		82		83		84		85		86		87		88		89	
9	91		92		93		94		95		96		97		98		99	

Table 4: Route planner

The floor plans of IRC and markers location were attached in Appendix section. Basically, the router planner works based on the markers location. For example, let say the user currently at circulation and he or she wants to go to Music & Art section. Thus, the route that the user should take to reach the section is $1 \longrightarrow 4 \longrightarrow 3$. So based on the table 4, now the user's destination is Music & Art (3), while the current location is at circulation (1). From location 1 to location 4, the marker that was assigned as circulation should show forward direction. After that, from location 4 to location 3, the marker that was assigned as checkpoint 1 should show right arrow. Next, when the user sees the next marker that was assigned as Music & Art, the current location will become 3 and the destination is also 3. So when the location value is equal to the destination value it indicates that the user already reaches the destination.

If the user accidentally goes to the wrong direction for example the user goes straight instead of turn right from location 4 to location 3 whereby the user should turn right to reach the destination 3. If this case happens, the other markers in the building will show the correct way to the user. For example, after reaches location 4 the user accidentally goes to location 5. Now the current location of the user is 5 and the destination is 3. Thus, based on the table 4 the marker that was assigned as location 5 will tells the user to go back or make a U-turn. Basically, this is how the route planner will correct back the direction or way to the reach a particular location.

Not just that, the route planner also works for multiple level whereby the user can navigates from ground level to first level of the building. In fact, the same procedures and processes are applied if the user wants to go to other locations in the building.

5.5 Camera Parameter:

In the ArtoolKit program, there is a function to setup the camera setting. At the earlier stage of the program, there was a significant lagging during video streaming. However, this problem has been solved after changing the parameter inside the function as shown in the figure below. There is a significant lagging because the camera is actually in zooming mode and this problem can be solved by resetting the camera back to normal mode.

```
/* load in the object data - trained markers and
if( (object=read_ObjData(model_name, &objectnum)
printf("Objectfile num = %d\n", objectnum);

/* open the graphics window */
argInit( &cparam, 1.0, 0, 0, 0, 0 );
}

/* cleanup function called when program exits */
static void cleanup(void)
{
    arVideoCapStop();
    arVideoClose();
    argCleanup();
}
```

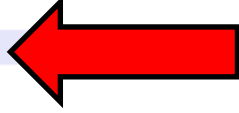


Figure 12: Camera parameter setting

As you can see from the figure above, the arrow shows where the setting located which is under “argInit()” function. Basically, it handles 5 variables all together: “**argInit (ARParam *cparam, double zoom, int fullFlag, int xwin, int ywin, int hmd flag)**”. Then, change the double zoom value to 1.0. As a result, the window size of the camera will be reduced and your camera streaming will be much faster. As a conclusion, the smaller the window size the faster the video streaming and vice versa. The reason why the video streaming becomes slower as the window size increases are due to the limitation of Raspberry Pi processor because more pixels need to be processed. Thus, the processing speed becomes slower.

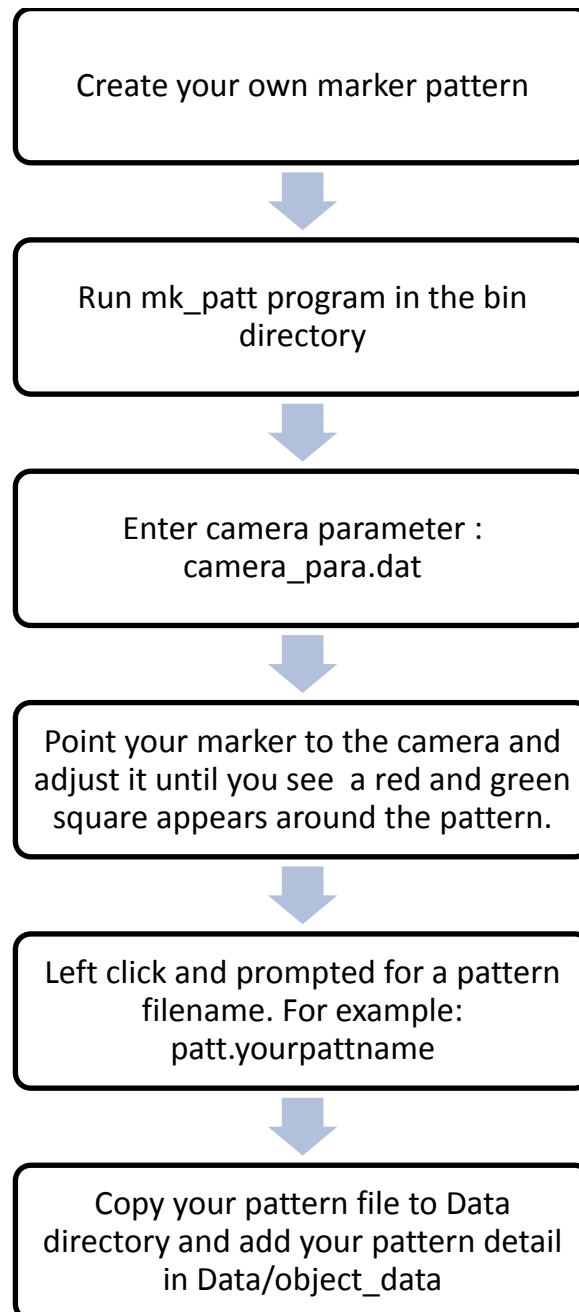
The result of the camera settings versus the processing speed in terms of frame per second is shown in Table 5.

Camera Setting	Frame per second (Fps)
argInit (ARParam *cparam, 2.0, 0, 0, 0, 0)	0.843
argInit (ARParam *cparam, 1.0 , 0, 0, 0, 0)	2.41
argInit (ARParam *cparam, 0.5, 0, 0, 0, 0)	5.56
argInit (ARParam *cparam, 0.1, 0, 0, 0, 0)	25.55

Table 5: ARToolKit Camera Setting

5.6 Creating and Training New Marker:

By default, the ArtoolKit software only provides 5 patterns of markers all together. However, we can add and train our own marker by following these steps.

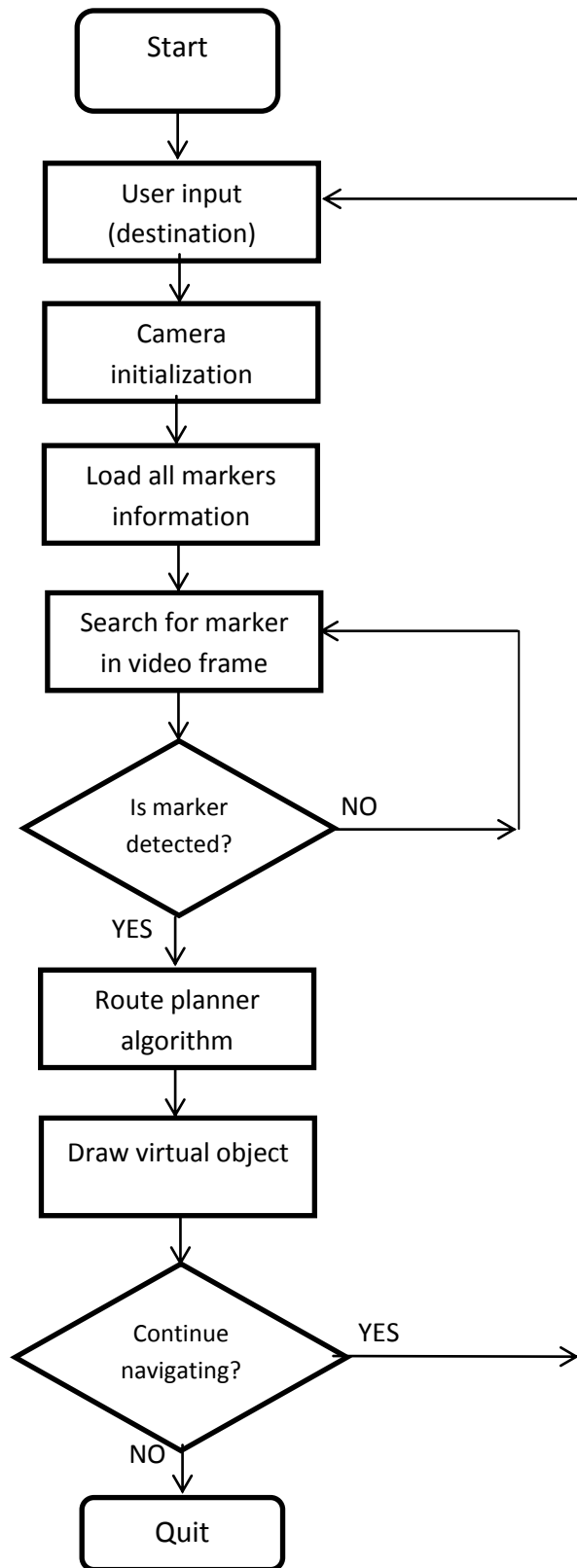


CHAPTER 6

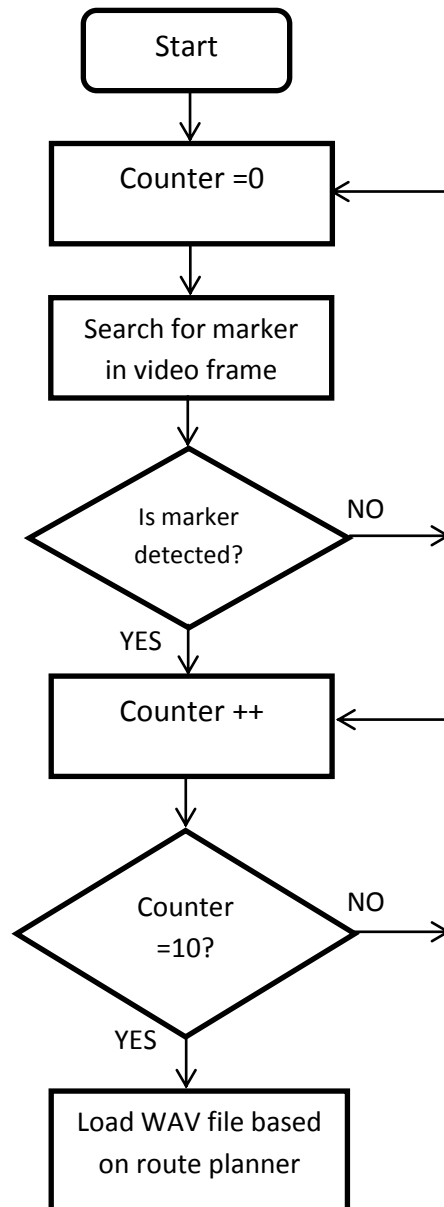
OVERALL OPERATION OF COMPLETE SYSTEM

6.1 Flow Chart of Overall Operation

The flowchart below summarized the overall operation of the system excluded the audio module. Firstly, the user has to select the destination listed in the program. After that, the software will initialize the camera and load markers information. Now, the camera starts capturing video and at the same times the ARToolKit software will search for markers in the video frame. If a marker was detected, the software will draw virtual object overlaid the marker based on the route planner algorithm. Once the user reached the destination, he or she can continue navigating by pressing “C” button and the process repeats again until the user decided to stop navigating by pressing “E” button.



6.2 Flow Chart of Audio Module



The audio module comes in different packages and not included in ARToolKit software. When the camera detects a marker the counter will keep increasing and if the camera does not detect any markers, the counter value will reset back to zero. Once the counter value reaches 10, the program will play the audio file based on the route planner.

CHAPTER 7

CONCLUSION

7.1 Conclusion

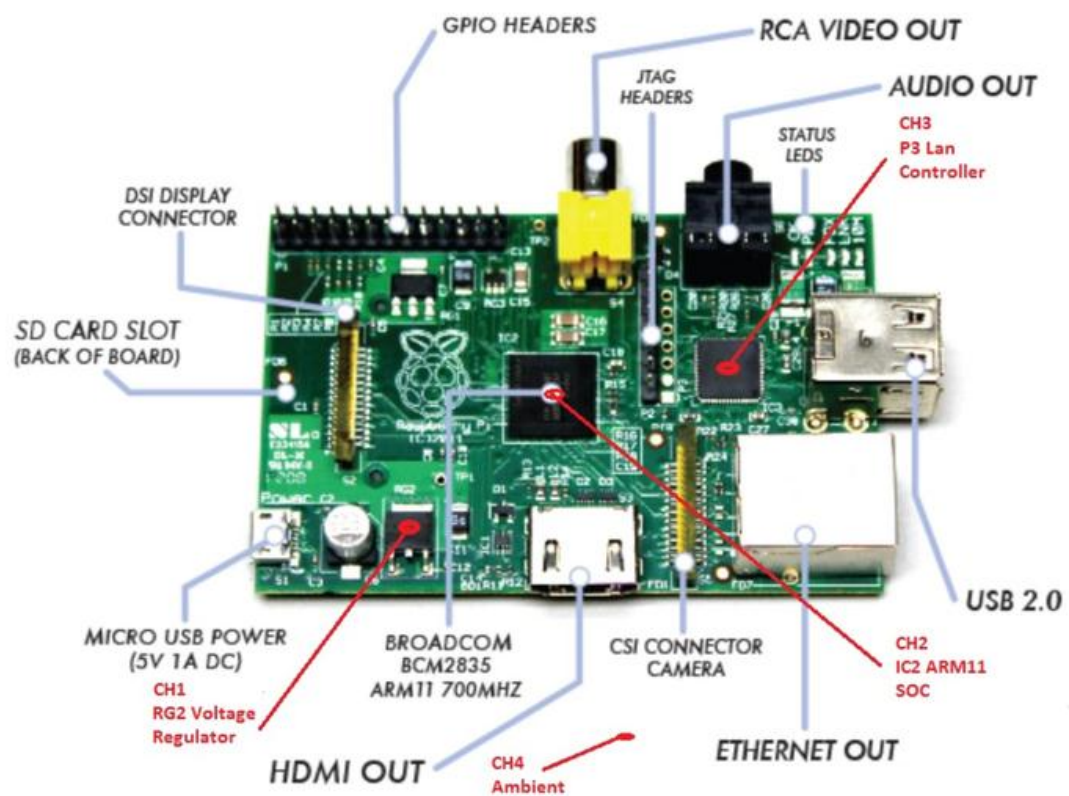
Augmented Reality is the next generation technology. Lots of researches have been carried out to enhance this technology since it incorporates the visual technology enhancement especially in designing the indoor positioning technology which is also another growing technology. Basically, this project has been tested at IRC, and it has shown its flexibility in working as an indoor positioning tool to navigate to 5 different locations with multiple levels. In addition, the route planner is more dynamic whereby the user can continue navigating from current location to the next destination. However, it still has rooms for improvement in terms of system's flexibility and the intelligence of the route planner in handling more complicated indoor layout such as a big shopping mall.

REFERENCES

1. Jongbae, K. and J. Heesung, *Vision-based location positioning using augmented reality for indoor navigation*. Consumer Electronics, IEEE Transactions on, 2008. **54**(3): p. 954-962.
2. Low Chee, H., P. Sebastian, and M. Driberg. *Augmented reality based indoor positioning navigation tool*. in *Open Systems (ICOS), 2011 IEEE Conference on*. 2011.
3. Li-Der CHOU , C.-Y.C., *A Hierarchical Architecture for Indoor Positioning Services*.
4. *What is a raspberry pi*. Available from: <https://www.raspberrypi.org/help/what-is-a-raspberry-pi/>.
5. *ARToolKit Documentation*. Available from: <http://www.hitl.washington.edu/artoolkit/documentation/userintro.htm>.
6. *GLUT - The OpenGL Utility Toolkit*. 2015; Available from: <https://www.opengl.org/resources/libraries/glut/>.
7. Al Delail, B., et al. *Indoor localization and navigation using smartphones augmented reality and inertial tracking*. in *Electronics, Circuits, and Systems (ICECS), 2013 IEEE 20th International Conference on*. 2013.
8. Greg Kipper, J.R., *Augmented Reality: An Emerging Technologies Guide to AR*. 2013: Syngress.
9. *How does ARToolKit work?* ; Available from: <http://www.hitl.washington.edu/artoolkit/documentation/userarwork.htm>.
10. Low, C.G. and Y.L. Lee. *SunMap+: An intelligent location-based virtual indoor navigation system using augmented reality*. in *Frontiers of Communications, Networks and Applications (ICFCNA 2014 - Malaysia), International Conference on*. 2014.

APPENDICES

- Raspberry Pi Model B Board Layout:



- **ARtoolkit Installation**

1. Install the dependencies using terminal:

```
sudo apt-get install freeglut3-dev libgstreamer0.10-dev libgstreamer-  
plugins-base0.10-dev libxi-dev libxmu-headers libxmu-dev libjpeg62-dev  
libglib2.0-dev libgtk2.0-dev
```

2. Download the ARToolKit software from :

<http://sourceforge.net/projects/artoolkit/files/artoolkit/2.72.1/ARToolKit-2.72.1.tgz/download>

3. Go to the software directory in the terminal and run :

- **./Configure** : choose option 5
- **Make**

4. Copy the include files to /usr/local/include :

```
sudo cp -R ./include/AR /usr/local/include/
```

5. Copy libraries to /usr/local/lib :

```
sudo cp ./lib/*.a /usr/local/lib/
```

6. Add the library location path to the LD_LIBRARY_PATH environment variable :

```
export LD_LIBRARY_PATH=/usr/local/lib
```

7. Before we run any of the example programs in “bin” folder of ARToolKit we MUST export an environment variable with the configuration of our capturing device :

```
export ARTOOLKIT_CONFIG="v4l2src device=/dev/video0 use-fixed-  
fps=false ! ffmpegcolorspace ! capsfilter caps=video/x-raw-rgb,bpp=24 !  
identity name=artoolkit ! fakesink"
```


- **Raspberry Pi specification**

	Raspberry Pi 2	Model B+	Model A+
SoC/CPU	BCM2836 Quadcore 900 MHz ARMv7	BCM2835 700 MHz ARMv6k	BCM2835 700 MHz ARMv6k
GPU	Broadcom VideoCore IV @ 250 MHz	Broadcom VideoCore IV @ 250 MHz	Broadcom VideoCore IV @ 250 MHz
RAM	1GB	512 MB	256 MB
Storage	MicroSD	MicroSD	MicroSD
USB	4	4	1
Ethernet	1	1	0
Video output	HDMI/Composite via RCA jack	HDMI/Composite via RCA jack	HDMI/Composite via RCA jack
Audio output	3.5 mm jack	3.5 mm jack	3.5 mm jack
GPIO	40	40	40
Price	\$35	\$35	\$20

- **Sample of working system (Hardware)**



USB Camera

Display Glass

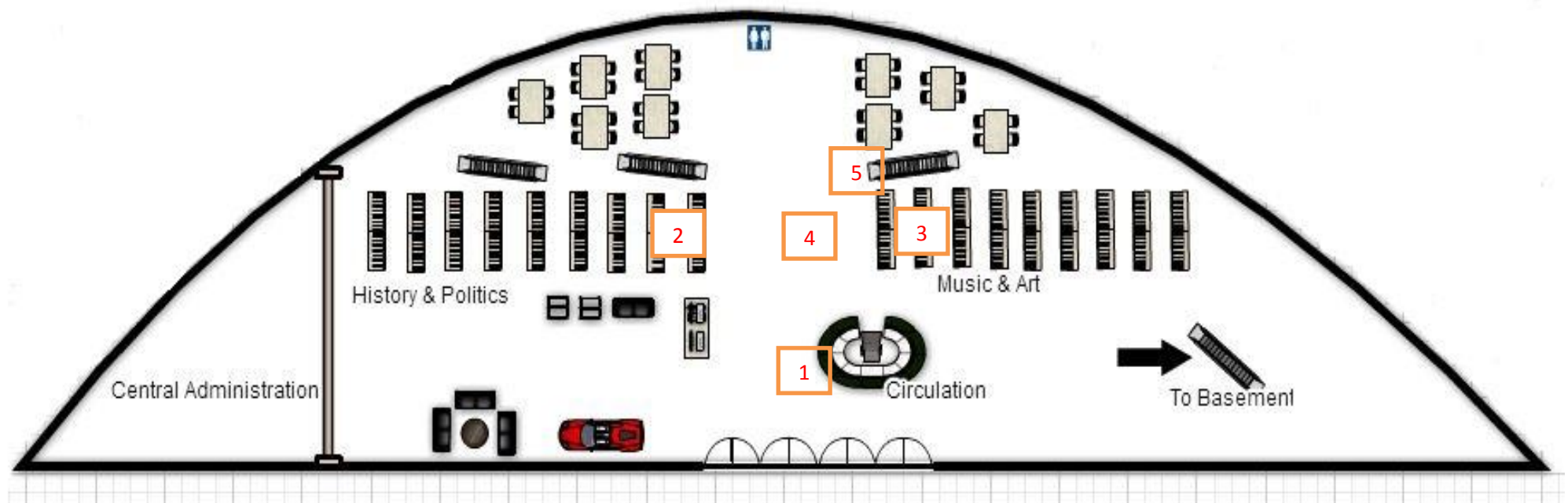


Mouse + Right click +
Left click

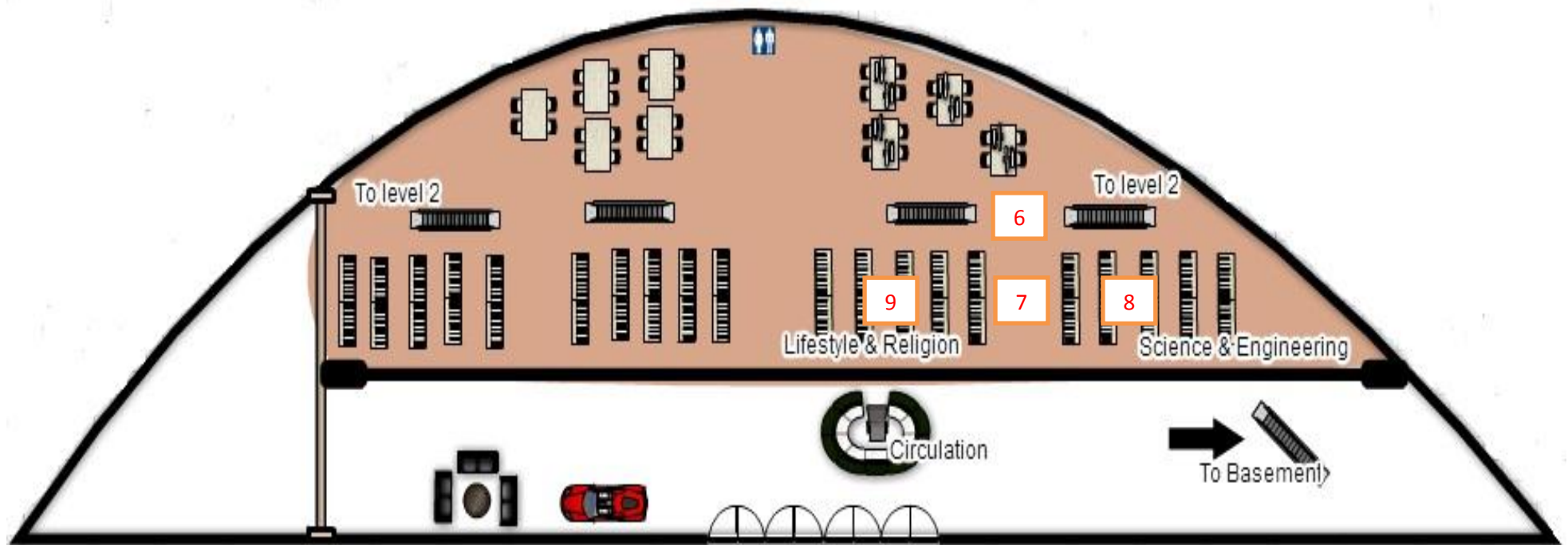
Number Pad

- **Information Resource Centre Floor Planner and location of markers**

Ground Floor



First Floor



- **Raspberry Pi Coding**

```
#ifdef _WIN32
# include <windows.h>
#endif

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
// openAL header
#include <AL/alut.h>
#include <AL/al.h>
#include <AL/alc.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <unistd.h>

#ifdef __APPLE__
# include <GL/glut.h>
#else
# include <GLUT/glut.h>
#endif

#include <AR/gsub.h>
#include <AR/param.h>
#include <AR/ar.h>
#include <AR/video.h>
#include "draw_object.h"
#include "object.h"
#include "info.h"
#define COLLIDE_DIST 30000.0

/*string*/
unsigned int width,height;
```

```

unsigned char *data;
GLuint texture;
char    string[256],string1[256],string2[256];
char    v;
/* Object Data */
char    *filename = "Data/audio";
char    *model_name = "Data/object_data"; // MARKER YG GUNA
ObjectData_T *object;
int    objectnum;
int    new, N;
int    xsize, ysize;
int    thresh = 100;
int    count = 0;
int D,L,P,x;
int c;
/* set up the video format globals */

#ifdef _WIN32
char    *vconf = "Data\\WDM_camera_flipV.xml";
#else
char    *vconf = "";
#endif
char    *cparam_name  = "Data/camera_para.dat";
ARParam  cparam;

static void  init(void);
static void  cleanup(void);
static void  keyEvent( unsigned char key, int x, int y);
static void  mainLoop(void);
static int draw( ObjectData_T *object, int objectnum );
static int draw_object( int obj_id, double gl_para[16] );
static int draw_obj(int obj_id);
GLuint LoadTexture( const char *filename);

```

```

//load wav function
int playsound( char *filename);

//===== AUDIO
=====

//
//load wav prog 2
playsound( char *filename)
{
char command[256];
int status;

sprintf(command,"aplay -c 1 -q -t wav %s",filename);
status=system(command);
return status;
}
////////// main
int main(int argc, char **argv)
{
    if(argc<2){
        exit(1);}
playsound( "Data/voice_navi/start.wav" );
////////// select destination//////////
fprintf(stdout, " Please select your destination (based on genre of books:\n");
    fprintf(stdout, " 1. Circulation\n"); //marker 1
    fprintf(stdout, " 2. History & politics \n"); // marker 2
    fprintf(stdout, " 3. music & art \n");// marker 3
    fprintf(stdout, " 8. science & engineering\n"); // marker 8
    fprintf(stdout, " 9. lifestyle & religion\n");// marker 9
    fscanf(stdin,"%d",&D);
//////////

```

```

if ( D !=0 )
{
//initialize applications
glutInit(&argc, argv);
init();

arVideoCapStart();

//start the main event loop
argMainLoop( NULL, keyEvent, mainLoop );
return 0;
}
else
{ fprintf(stdout, " Tq\n");
}

}

static void keyEvent( unsigned char key, int x, int y)
{
/* quit if the ESC key is pressed */ //ESC=0X1b
if( key == '-' ) {
printf("*** %f (frame/sec)\n", (double)count/arUtilTimer());
playsound( "Data/voice_navi/exit.wav" );
cleanup();
exit(0);
}

if( key == '+' )
{
int i;

draw_obj(object[i].id);
fprintf(stdout, " Please select your destination (based on genre of books:\n");
fprintf(stdout, " 1. Circulation\n"); //marker 1
fprintf(stdout, " 2. History & politics \n"); // marker 2
fprintf(stdout, " 3. music & art \n");// marker 3

```

```

        fprintf(stdout, " 8. science & engineering\n"); // marker 8
        fprintf(stdout, " 9. lifestyle & religion\n");// marker 9
        fscanf(stdin,"%d",&D);
    }
}

/* main loop */
static void mainLoop(void)
{
    ARUInt8      *dataPtr;
    ARMarkerInfo *marker_info;
    int          marker_num;
    int          i,j,k;

    /* grab a video frame */
    if( (dataPtr = (ARUInt8 *)arVideoGetImage()) == NULL ) {
        arUtilSleep(2);
        return;
    }

    if( count == 0 ) arUtilTimerReset();
    count++;

    /*draw the video*/
    argDrawMode2D();
    argDispImage( dataPtr, 0,0 );

    glColor3f( 1.0, 0.0, 0.0 );
    glLineWidth(6.0);

    /* detect the markers in the video frame */
    if(arDetectMarker(dataPtr, thresh,
        &marker_info, &marker_num) < 0 ) {
        cleanup();
    }
}

```



```

        exit(0);
    }

    for( i = 0; i < marker_num; i++ ) {
        argDrawSquare(marker_info[i].vertex,0,0);
    }

    arVideoCapNext();

    /* check for known patterns */
    for( i = 0; i < objectnum; i++ ) {
        k = -1;
        for( j = 0; j < marker_num; j++ ) {
            if( object[i].id == marker_info[j].id ) {

                /* you've found a pattern */
                //printf("Found pattern: %d ",patt_id);
                glColor3f( 0.0, 1.0, 0.0 );
                argDrawSquare(marker_info[j].vertex,0,0);

                if( k == -1 ) k = j;
                else /* make sure you have the best pattern (highest
confidence factor) */
                    if( marker_info[k].cf < marker_info[j].cf )
                        k = j;
            }
        }
        if( k == -1 ) {
            object[i].visible = 0;
            continue;
        }
    }

```

```

        /* calculate the transform for each marker */
        if( object[i].visible == 0 ) {
            arGetTransMat(&marker_info[k],
                object[i].marker_center, object[i].marker_width,
                object[i].trans);c=0;
        }
        else {
            arGetTransMatCont(&marker_info[k], object[i].trans,
                object[i].marker_center, object[i].marker_width,
                object[i].trans);
        }
        object[i].visible = 1;
    }

    /* draw the AR graphics */

    draw( object, objectnum );

    /*swap the graphics buffers*/
    argSwapBuffers();

}

static void init( void )
{
    ARParam wparam;

    /* open the video path */
    if( arVideoOpen( vconf ) < 0 ) exit(0);
    /* find the size of the window */
    if( arVideoInqSize(&xsize, &ysize) < 0 ) exit(0);
    printf("Image size (x,y) = (%d,%d)\n", xsize, ysize);

```

```

/* set the initial camera parameters */
if( arParamLoad(cparam_name, 1, &wparam) < 0 ) {
    printf("Camera parameter load error !!\n");
    exit(0);
}
arParamChangeSize( &wparam, xsize, ysize, &cparam );
arInitCparam( &cparam );
printf("*** Camera Parameter ***\n");
arParamDisp( &cparam );

/* load in the object data - trained markers and associated bitmap files
*/
if( (object=read_ObjData(model_name, &objectnum)) == NULL ) exit(0);
printf("Objectfile num = %d\n", objectnum);

/* open the graphics window */
argInit( &cparam, 0.5, 0, 0, 0, 0 );//lajukn
}

/* cleanup function called when program exits */
static void cleanup(void)
{
    arVideoCapStop();
    arVideoClose();
    argCleanup();
}

```

```

/* draw the the AR objects */
static int draw( ObjectData_T *object, int objectnum )
{
    int    i;
    double gl_para[16];

    glClearDepth( 1.0 );
    glClear(GL_DEPTH_BUFFER_BIT);
    glEnable(GL_DEPTH_TEST);
    glDepthFunc(GL_LEQUAL);
    glEnable(GL_LIGHTING);

    /* calculate the viewing parameters - gl_para */
    for( i = 0; i < objectnum; i++ ) {
        if( object[i].visible == 0 ) continue;
        argConvGlpara(object[i].trans, gl_para);
        draw_object( object[i].id, gl_para);
    }

    glDisable( GL_LIGHTING );
    glDisable( GL_DEPTH_TEST );

    return(0);
}

/* draw the user object */
static int draw_object( int obj_id, double gl_para[16])
{
    GLfloat mat_ambient[]           = {0.0, 0.0, 1.0, 1.0};
    GLfloat mat_ambient_collide[]   = {1.0, 0.0, 0.0, 1.0};
    GLfloat mat_flash[]             = {0.0, 0.0, 1.0, 1.0};

```

```
GLfloat mat_flash_collide[] = {1.0, 0.0, 0.0, 1.0};
```

```
GLfloat mat_flash_shiny[] = {50.0};
```

```
GLfloat light_position[] = {100.0,-200.0,200.0,0.0};
```

```
GLfloat ambi[] = {0.1, 0.1, 0.1, 0.1};
```

```
GLfloat lightZeroColor[] = {0.9, 0.9, 0.9, 0.1};
```

```
argDrawMode3D();
```

```
argDraw3dCamera( 0, 0 );
```

```
glMatrixMode(GL_MODELVIEW);
```

```
glLoadMatrixd( gl_para );
```

```
/* set the material */
```

```
glEnable(GL_LIGHTING);
```

```
glEnable(GL_LIGHT0);
```

```
glLightfv(GL_LIGHT0, GL_POSITION, light_position);
```

```
glLightfv(GL_LIGHT0, GL_AMBIENT, ambi);
```

```
glLightfv(GL_LIGHT0, GL_DIFFUSE, lightZeroColor);
```

```
glMaterialfv(GL_FRONT, GL_SHININESS, mat_flash_shiny);
```

```
//////// LOCATION DECLARATION//////////
```

```
//HIRO
```

```
if ( obj_id == 0)
```

```
{ L= 1;}
```

```
//VERTICAL LINE
```

```
if ( obj_id == 1 )
```

```
{ L=2;}
```

```
//HORIZONTAL LINE
```

```
if ( obj_id == 2 )
```

```
{ L=3;}
```

```
//KANJI
```

```
if ( obj_id == 3 )
```

```
{ L=4;}
```

```

if ( obj_id == 4 ) //PATT CALIB
{ L=5;}
if ( obj_id == 5 ) //patt m1
{ L=6;}
if ( obj_id == 6 ) //patt hurufa
{ L=7;}
if ( obj_id == 7 ) //patt B
{ L=8;}
if ( obj_id == 8 ) //patt L
{ L=9;}

////////////////////////////////////

//sampai destination//

if (L==1 && D==1 || L==2 && D==2 || L==3 && D==3 || L==8 &&
D==8 || L==9 && D==9 )
{

    //fprintf(stdout,"c=%d\n",c);

    c++;

    glMaterialfv(GL_FRONT, GL_SPECULAR, mat_flash);
    glMaterialfv(GL_FRONT, GL_AMBIENT, mat_ambient);

    /* draw a cube */

    //glTranslatef( 0.0, 0.0, 30.0 );

    //glutSolidCube(60);

    //glTranslatef(-1.0, -0.20, 0.0);

    glRotatef( 90.0, 1.0, 0.0, 0.0 );

    glColor4f( 0.0, 0.0, 1.0, 1.0 );

    glutSolidCone(40.0, 80.0, 20, 24);

    sprintf(string1," You've Reach Your Destination,Do you wish to continue?
(Y/N) ");

    print_string1( string1 );

    sprintf(string2,"\nnext destination?\n\n1. Circulation\n\n2. CH\n\n3.
IRC\n\n4.FINANCE");

```

```

print_destination( string2 );

        if(c==10)
            {playsound( "Data/voice_navi/reach.wav" );}

    }

//uturn/goback
if (L==5 && D==1 || L==5 && D==2 || L==5 && D==3 || L==4 &&
D==1)
{
    c++;
    glMaterialfv(GL_FRONT, GL_SPECULAR, mat_flash);
    glMaterialfv(GL_FRONT, GL_AMBIENT, mat_ambient);
    /* draw a cube */
    //glTranslatef( 0.0, 0.0, 30.0 );
    //glutSolidCube(60);
    if(L==5)
        {glRotatef( 90.0, 0.0, 0.0, 1.0 );}
    else if (L==4)
        {glRotatef( 90.0, 1.0, 0.0, 0.0 );}
    glColor4f( 0.0, 0.0, 1.0, 1.0 );
    glutSolidCone(40.0, 150.0, 20, 24);
    sprintf(string, " Go Back ");
    print_string( string );
    if(c==10)
        {playsound( "Data/voice_navi/Uturn.wav" );}
}

```

```

//on ur right ( for check point only)
    if ( L==4 && D==3 || L==7 && D==9)
    {
c++;
glMaterialfv(GL_FRONT, GL_SPECULAR, mat_flash_collide);
glMaterialfv(GL_FRONT, GL_AMBIENT, mat_ambient_collide);

        /* draw a cube */
        //glTranslatef( 0.0, 0.0, 30.0 );
        //glutSolidSphere(30,12,6);
        //glColor4f(1.0, 1.0, 1.0,1.0 );
        glRotatef( 90.0, 0.0, 1.0, 0.0 );
        //glColor4f( 1.0, 0.0, 0.0, 1.0 ); //merah
        glutSolidCone(40.0, 150.0, 20, 24);
        sprintf(string," on your Right ");
        print_string( string );
        if(c==10)
            {playsound( "Data/voice_navi/on_your.wav" );}
    }
// turn right
    if (L==2 && D==1 || L==2 && D==3 || L==2 && D==8 || L==2 &&
D==9 || L==6 && D==8 || L==6 && D==9 || L==8 && D==9)
    {
c++;
glMaterialfv(GL_FRONT, GL_SPECULAR, mat_flash_collide);
glMaterialfv(GL_FRONT, GL_AMBIENT, mat_ambient_collide);

        /* draw a cube */
        //glTranslatef( 0.0, 0.0, 30.0 );
        //glutSolidSphere(30,12,6);
        //glColor4f(1.0, 1.0, 1.0,1.0 );
        if (L==6)
            {glRotatef( 90.0, 1.0, 0.0, 0.0 );}
        else

```



```

        {glRotatef( 90.0, 0.0, 1.0, 0.0 );}
//glColor4f( 1.0, 0.0, 0.0, 1.0 ); //merah
glutSolidCone(40.0, 150.0, 20, 24);
        sprintf(string, " Turn Right ");
        print_string( string );
        if(c==10)
            {playsound( "Data/voice_navi/turn_right.wav" );}
    }
//turn left
    if (L==3 && D==1 || L==3 && D==2|| L==3 && D==8|| L==3 &&
D==9|| L==9 && D==8)
    {
        c++;
        glMaterialfv(GL_FRONT, GL_SPECULAR, mat_flash_collide);
        glMaterialfv(GL_FRONT, GL_AMBIENT, mat_ambient_collide);

        /* draw a cube */
        //glTranslatef( 0.0, 0.0, 30.0 );
        //glutSolidSphere(30,12,6);
        //glColor4f(1.0, 1.0, 1.0,1.0 );
        glRotatef( 90.0, 0.0, -1.0, 0.0 );
        //glColor4f( 1.0, 0.0, 0.0, 1.0 ); //merah
        glutSolidCone(40.0, 150.0, 20, 24);

        sprintf(string, " Turn Left ");
        print_string( string );
        if(c==10)
            {playsound( "Data/voice_navi/turn_left.wav" );}
    }

```

```

////////////////////////////////////
//on ur left
if ( L==4 && D==2|| L==7 && D==8)
{
    c++;
    glMaterialfv(GL_FRONT, GL_SPECULAR, mat_flash_collide);
    glMaterialfv(GL_FRONT, GL_AMBIENT, mat_ambient_collide);

    /* draw a cube */
    //glTranslatef( 0.0, 0.0, 30.0 );
    //glutSolidSphere(30,12,6);
    //glColor4f(1.0, 1.0, 1.0,1.0 );
    glRotatef( 90.0, 0.0, -1.0, 0.0 );
    //glColor4f( 1.0, 0.0, 0.0, 1.0 ); //merah
    glutSolidCone(40.0, 150.0, 20, 24);

    sprintf(string, " on your Left ");
    print_string( string );
    if(c==10)
    {playsound( "Data/voice_navi/on_your_left.wav" );}
}

//go down stairs
if ( L>=6 && D<=3 )
{
    c++;
    glMaterialfv(GL_FRONT, GL_SPECULAR, mat_flash_collide);
    glMaterialfv(GL_FRONT, GL_AMBIENT, mat_ambient_collide);
}

```

```

/* draw a cube */

//glTranslatef( 0.0, 0.0, 30.0 );
//glutSolidSphere(30,12,6);
//glColor4f(1.0, 1.0, 1.0,1.0 );
if(L==6){
    glRotatef( 90.0, 0.0, -1.0, 0.0 );}
else if (L>6){
    glRotatef( 90.0, 1.0, 0.0, 0.0 );}
//glColor4f( 1.0, 0.0, 0.0, 1.0 ); //merah
glutSolidCone(40.0, 150.0, 20, 24);

    sprintf(string, " GO DOWN ");
    print_string( string );
    if(c==10)
        {playsound( "Data/voice_navi/downstairs.wav" );}
}
//go foward
if ( L==1 && (D==2 || D==3|| D==8|| D==9))
{
    c++;
    glMaterialfv(GL_FRONT, GL_SPECULAR, mat_flash_collide);
    glMaterialfv(GL_FRONT, GL_AMBIENT, mat_ambient_collide);

/* draw a cube */

//glTranslatef( 0.0, 0.0, 30.0 );
//glutSolidSphere(30,12,6);
//glColor4f(1.0, 1.0, 1.0,1.0 );
    glRotatef( 90.0, -1.0, 0.0, 0.0 );
//glColor4f( 1.0, 0.0, 0.0, 1.0 ); //merah
glutSolidCone(40.0, 150.0, 20, 24);

```

```

        sprintf(string," go forward ");
        print_string( string );
        if(c==10)
            {playsound( "Data/voice_navi/forward.wav" );}
    }

//go UPSTAIRS
if ( (L==4 || L==5 )&& ( D==8|| D==9))
{
    c++;
    glMaterialfv(GL_FRONT, GL_SPECULAR, mat_flash_collide);
    glMaterialfv(GL_FRONT, GL_AMBIENT, mat_ambient_collide);

    /* draw a cube */
    //glTranslatef( 0.0, 0.0, 30.0 );
    //glutSolidSphere(30,12,6);
    //glColor4f(1.0, 1.0, 1.0,1.0 );
    glRotatef( 90.0, 0.0, 0.0, 1.0 );
    //glColor4f( 1.0, 0.0, 0.0, 1.0 ); //merah
    glutSolidCone(40.0, 150.0, 20, 24);

    sprintf(string," go UP");
    print_string( string );
    if(c==10)
        {playsound( "Data/voice_navi/upstairs.wav" );}
}

```

```

////////////////////////////////////
    argDrawMode2D();
    return 0;
}
static int draw_obj( int obj_id )
{
    argDrawMode3D();
    argDraw3dCamera( 0, 0 );
    glMatrixMode(GL_MODELVIEW);
    glClearDepth( 1.0 );
    glClear(GL_DEPTH_BUFFER_BIT);
    glEnable(GL_DEPTH_TEST);
    glDepthFunc(GL_LEQUAL);
    //load the camera transformation matrix
    glMatrixMode(GL_MODELVIEW);

}

```