INDOOR-POSITIONING FOR WAREHOUSE MOBILE ROBOTS USING COMPUTER VISION

BY

NG ZHUN YEE

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE

Faculty of Information and Communication Technology
(Kampar Campus)

JANUARY 2021

UNIVERSITI TUNKU ABDUL RAHMAN

REPORT STATUS DECLARATION FORM

Title: INDOOR-POSITIONING FOR WAREHOUSE MOBILE ROBOTS USING COMPUTER VISION

Academic Session: January 2021

I NG ZHUN YEE

(CAPITAL LETTER)

declare that I allow this Final Year Project Report to be kept in Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.

Verified by,

(Author's signature)

(Supervisor's signature)

Address:
22, Laluan Pengkalan Indah 6
Taman Pengkalan Intan
31650 Ipoh, Perak

Liew Soung Yue

Supervisor's name

Date: 16-APR-2021

Date: 16/4/2021

INDOOR-POSITIONING FOR WAREHOUSE MOBILE ROBOTS USING COMPUTER VISION

BY

NG ZHUN YEE

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE

Faculty of Information and Communication Technology

(Kampar Campus)

JANUARY 2021

DECLARATION OF ORIGINALITY

I declare that this report entitled "**INDOOR-POSITIONING FOR WAREHOUSE MOBILE ROBOTS USING COMPUTER VISION**" is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : ______________________________

Name : ______NG ZHUN YEE______

Date : _________16-APR-2021________

ACKNOWLEDGEMENTS

I would like to express my sincere thanks and appreciation to my supervisor, Ts. Dr. Liew Soung Yue who has given me the chance to take part in this meaningful yet challenging project. Another million thanks also given to Mr Wong Chee Siang for his ideas and critics. I would also take this opportunity to say a thank you to my Academic Advisor, Dr Ng Hui Fuang for the guidance and supports all along the course.

Not to forget about my families and friends who helped me out by giving loves and supports throughout the course. Finally, I would like to say thank you to all of you that supported me.

ABSTRACT

This project is a system which is built for the warehouse. Due to the rapid growth of the e-commerce sector, the operation of the supply chain from upstream to downstream have to be more efficient and effective. In this project, the problem will be specified on warehouses/ sorting centres. The usage of automated robot machines can be easily seen in the logistics sector or supply chain processes. Every processes in a warehouse have to be effective, efficient and also in a safety condition. There is a need for the localization of the mobile robots in the indoor environment, i.e. warehouse. The indoor mobile robots are currently needed to perform various tasks. The localization feature of an indoor mobile robot is important to estimate the position and orientation of the mobile robot in the environment. Localization is required due to it is difficult for the mobile robots to perform their tasks without accurately knowing the exact position in the environment. This project can be considered as both research based and development based project. The purpose of this project is to demonstrate the usage of ceiling mounted cameras for localization of mobile robots in indoor environment. With that said, this project was developed to imitate the real world scenarios having numbers of high speed mobile robots in a huge environment. On the other hand, this project was built to determine whether the usage of overhead cameras is better than mounting the cameras on the robot itself, and to show that this project is able to use low costing equipment to perform localization of mobile robots in an indoor environment. This project proved that overhead cameras can be used to detect and track the mobile robots, which has a low costing as compared to install a camera on the robot itself. For this proposed system, a testbed based experiment will be conducted to prove that this concept works, in a much smaller scale compared to warehouse size. 2 overhead cameras with 90˚ degrees pointing down to the ground were used to capture the videos, with no overlapping of area among the cameras. To uniquely identify each of the toy car, different QR code was placed on top of the toy car for the cameras to detect. A QR code decoder was used to decode all the QR codes detected in the testbed area. All the toy cars were moved in random direction. This system can detect all the QR codes, as well as track them all along whenever detected. When the QR codes were detected, the localized position of the toy cars will be calculated and saved for later use. At the same time, the orientation of the toy car was also calculated by obtaining the coordinates of the positioning detection markers i.e. the 3 square boxes on corners of QR code.

Multiple QR codes could be present in each frame, and this proposed system could handle it. Different experiment has been conducted to get the ideal settings for the best performance of this system. This system has proven to achieve the main goal of this project, which was the localization of mobile robots in indoor environment.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

Symbol	Meaning
°	Degree (Angle)

LIST OF ABBREVIATIONS

AGV	Automated Guided Vehicles
AMR	Autonomous Mobile Robots
FOV	Field of View
ICP	Iterative Closest Point
IoT	Internet of Things
LED	Light-emitting diode
QR	Quick Response
RANSAC	RANdom Sample Consensus
ROI	Region Of Interest
SLAM	Simultaneous Localization and Mapping
SURF	Speedup Up Robust Features

CHAPTER 1 INTRODUCTION

1.1 Problem Statement

Currently, due to the rapid growth of the e-commerce sector, the operation of the supply chain from upstream to downstream have to be more efficient and effective. In this project, the problem will be specified on warehouses/ sorting centres. The function of the warehouse is to act as a storage space for inventory items while sorting centres having conveyor to sort the items. Either warehouses or sorting centres, they both need the help of mobile robots. The usage of automated robot machines can be easily seen in the logistics sector or supply chain processes. Every processes in a warehouse have to be effective, efficient and also in a safety condition. Therefore, every single element either minor or major have the probability to affect the overall performance of every area of a warehouse such as the storage area, shipping area and the manufacturing area. The automation among the mobile robots is important due to automation is the major part to be considered to ensure the logistics industry able to achieve industry 4.0.

Due to the importance stated above, there is a need for the localization of the mobile robots in the indoor environment, i.e. warehouse. The indoor mobile robots are currently needed to perform various tasks. The localization feature of an indoor mobile robot is important to estimate the position and orientation of the mobile robot in the environment. Prior knowledge such as map of the environment or initial position is important too in helping the localization process. Localization is required due to it is difficult for the mobile robots to perform their tasks without accurately knowing the exact position in the environment. Disregard the routing decision algorithm for the mobile robots, the problem of localization of high speed moving mobile robots in an indoor environment has to be addressed.

Due to the automation features of mobile robots in the market, the usage of this technology can help in reducing the human work, thus human can work on other higher value activities rather than moving the objects around. So, the localization of mobile robots in indoor environment is important. When combining this localization feature with the routing decision algorithm, this framework can be used in either warehouses or sorting centres for the mobile robots to move items around. This framework can know the exact location and orientation of the mobile robots in the environment, and can provide the robots with decision on what tasks to do next. However, the routing decision algorithm will not be discussed in this proposal, but the problem of localization of mobile robots in indoor environment will be the main problem to be solved in this project.

1.2 Background and Motivation

In light of the accelerated shift towards e-commerce in recent years, the supply chain members, especially warehouses and distribution centres are playing a huge role in the industry. With the ease of accessing e-commerce website for online shopping through mobile phones nowadays, the revenue growth of e-commerce are increasing rapidly. Despite the recent Covid-19 pandemic, Amazon doubled its net profit compared to last year (Faulkner, 2020). People went for online shopping and the supply chain members have to get their work done faster, from transforming raw materials into finished products, to distributing them to end-user. Together with the applications of machine learning in e-commerce, it will learn the shopping behaviour of customers and will recommend products that might caught their attention, thus promoting the sales of e-commerce. Amazon has also proven that recommendation of products in their website works. As such, warehouses and sorting centres are now having high demand due to the rise of e-commerce. However, lacking of man power is one of the problem faced in the warehouses in the past. Now, the usage of robots are very common in the industry, and also solved the problem of limitations of man power.

Industry 4.0 is the phase in the Industrial Revolution that emphasize on interconnectivity, automation, machine learning, and real-time data in the industry sector (What is Industry 4.0-the Industrial Internet of Things (IIoT)?, 2020). The third Industrial Revolution started in late 1950s, where computers were introduced. The Industry 4.0 optimizes the computerization of Industry 3.0, where computers can communicate with each other and make decisions. When logistic and warehouses meet Industry 4.0, it makes a big changes to the supply chain. With applications of Industry 4.0 including IoT, connected supply chain, autonomous vehicles, and also robots, the e-commerce companies are preparing for a future where smart machines can improve their business (Marr, 2020). From manual sorting and shipment, to automated sorting and deliveries, we are living in the world that emphasizes on Industry 4.0, specifically automation.

The use of robots are quite common in many industries nowadays. For example, DF Automation & Robotics tech-based company has deployed delivery robot for hospitals to support healthcare front-liners by delivering food or supplements to patients with Covid-19 (How an Autonomous Mobile Robot (AMR) can help to fight COVID-19? - DF Automation & Robotics, 2020). According to research study of LogisticsIQ, due to the increasing demand of faster order delivery in the e-commerce sector, the AGV/AMR will contribute more than a quarter of the overall revenue. The AGV market has doubled in last 3 years according to the research. The AGV/AMR are assets in the supply chain and plays a role in optimizing the warehouse operations (Warehouse Automation Market, 2020). The AGV/AMR are used widely for internal transportation tasks, so that employees can focus on higher-value activities, instead of using man power to move materials around. Today, AGVs are challenged by AMRs with better technologies. An AGV has minimal intelligence where it is following the fixed routes and simply stops in its tracks if obstacles are on the tracks, due to AGV is unable to navigate around them. On the other hand, an AMR uses data from sensors to detect its surroundings and choose the best route to the target. Most importantly, it is completely automated (AGV vs. AMR – What's the Difference? | Mobile Industrial Robots, 2020). The main motivation and why localization is important is that the coordinates of the mobile robots in real time is important for determining the further route decision. With the localization information obtained, the data will be transfer to the backend server for further route decision.

1.3 Objectives

The title of this project is specific enough, such that the aim of this project is to perform localization of warehouse mobile robots using overhead cameras. The ultimate objective is to perform localization of mobile robots, but the specifications of the mobile robots and cameras have to take into consideration as well.

This project aims to outperform the current development that has the issue in localizing the mobile robots. This project also aims to get a lower average localization error comparing to method proposed by others. As the title said, this project will be focusing on the localization of the mobile robots in an indoor environment using multiple ceiling cameras. This project covers the area of localizing the mobile robots, but the routing decision or how efficient is the localization information send to server will not be covered in this project.

After the development of the system, a testbed based experiment will be carry out to test the performance and accuracy of the system. The experiment will be carry out in a small scale, where few cameras and mobile robots will be used in the workspace. This project is aimed to mimic the real world scenario.

1.4 Proposed Approach

A testbed based experiment has been conducted. 2 cameras were mounted on top of the testbed area, where the cameras were 90˚ pointing down to the ground. Each of the camera was capturing half of the testbed area, and no overlapping of captured area among the cameras. This project was both development and research based. This project was built to detect, track, and get the localize coordinates of the toy car. At the same time, this project was also built and experimented to prove the concept that multiple cameras on the ceiling were usable to do localization of mobile robots in warehouse. QR codes were proposed to do the localization of the toy cars in testbed area. One QR code was put on top of the toy car as shown in Figure 1.4.1, same goes to other toy cars. Then, the toy cars were controlled to move in random direction. Let's assume the testbed area were split into top part and bottom part, we called it Area1 and Area2 as shown in Figure 1.4.2. Some of the toy cars were stay inside Area1, some of it stay inside Area2, and some of it will cross between the both areas. When toy cars were in the testbed area, the proposed system can detect and track the location of the QR code, which turns out can get the location of the toy car. When the QR code was detected, the proposed system can get the location and calculate the orientation of the QR codes precisely. When QR codes were detected, all the information will be collected immediately.

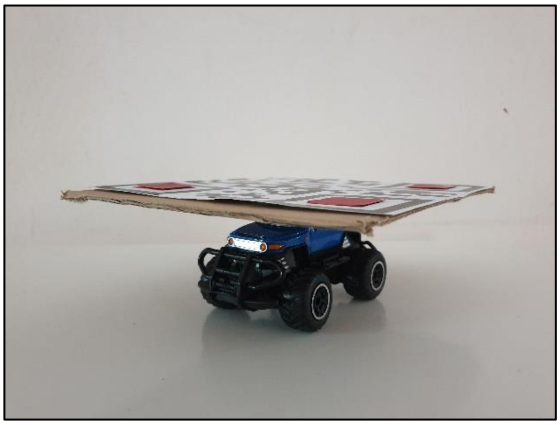

Figure 1.4.1: Toy car with QR code

Figure 1.4.2: Area 1 and area 2

Figure 1.4.3 shows the general system flowchart of the proposed system. The proposed system will detect any QR code present in the testbed area, and it will keep on tracking the QR code from time to time. When the QR code can be detected, the system will obtain the coordinates, orientation, and timestamp. At the same time, there will be a window showing all the coordinates of the detected QR code in dots as shown in Figure 1.4.4. All these process will be done at the same time, meaning that this system will be running in real time. For a more detailed system design, it will be discussed in Chapter 3.

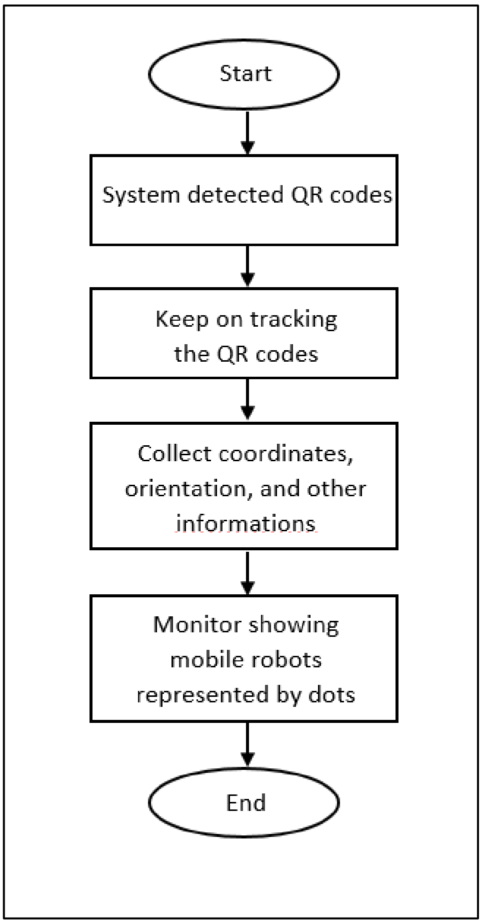

Figure 1.4.3: General design flowchart

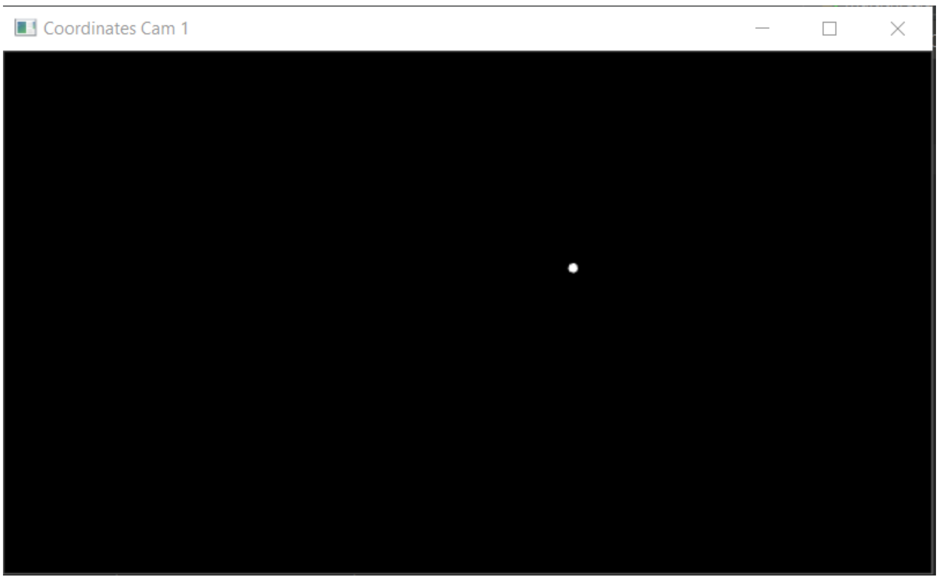

Figure 1.4.4: Dots showing coordinates

1.5 Contributions

The experiment and analysis confirmed that the ability of using multiple cameras were able to perform the detection, tracking, as well as localization. Firstly, it was proved that cheap cameras were able to capture the frame and perform detection. Next, when the QR code was detected and tracked, it will be able to do localization precisely in this project. Thirdly, it was proved that multiple cameras can be used on the ceiling and obtained a satisfactory result instead of installing on every robots, thus reducing the costs. Lastly, the orientation of the robots can be precisely calculated by obtaining the coordinates of the positioning detection markers i.e. the 3 square boxes on corners of QR code.

1.6 Report Organization

The details of this project are shown in the following chapters. In Chapter 2, related backgrounds and techniques were reviewed. Chapter 3 explains the detailed system design of this project. Then, chapter 4 explains the specification of this project. Furthermore, chapter 5 includes the implementation and testing of the project. Lastly, chapter 6 is the conclusion part of this project.

CHAPTER 2 LITERATURE REVIEW

2.1 Ceiling Landmarks and input images from RGB-D camera methods in determining the indoor position of mobile robots

Techniques like ceiling landmarks and images from RGB-D camera were proposed for mobile robot localization in the indoor environment. These methods are like a global method and local method, respectively. For the ceiling landmarks method, specifically designed markers are attached on the ceiling with known locations, with the intentions to calculate the location and orientation of the robot. On the other hand, the RGB-D camera is mounted on the robot and will take both the colour and depth images of the environment. Then the 3D point clouds and features extracted through SURF between consecutive image frames are computed to determine the robot motion. Both techniques can calculate the robot's position by using the real-time detected input images, and does not need any prior information or features acquisition from anything in the environment. The similarity between these two techniques is both methods uses camera as the primary sensor, with the RGB-D camera method having extra depth sensor to detect the distance between the scene and the camera.

For the first method, localization using ceiling landmarks, camera facing upwards are mounted on robots to capture the images of landmarks. The specifically designed landmarks are like the one show in Figure 2.1.1. It is a black square containing combination of few white circles. This pattern is unique for the robots to identify the location. Like Figure 2.1.1, only 3 solid white circles out of 4 will be assigned in one black square. The position of the 3 circles can used to rectify image captured and identify the direction of the robot pointing at. For other circles in the pattern, it is used to uniquely indicate the locations of the markers in the indoor environment.

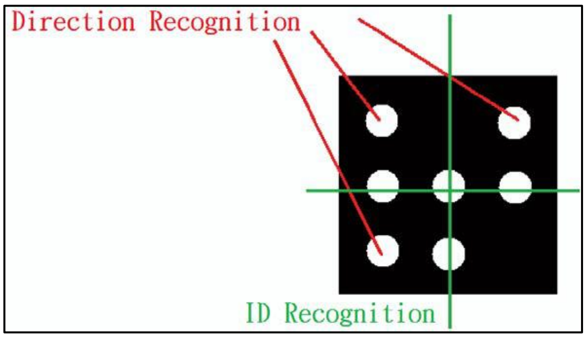

Figure 2.1.1 Example of marker pattern (Designed ceiling landmark)

The detection of marker of the ceiling landmarks method will undergo a number of steps. The input images will first converted to grayscale images since the inputs are colour images. Then the markers are extracted using Sobel edge detection algorithm. The image will then enhanced by applying connected component labelling to further remove the noise. After this process, the large components can be identified as the candidate marker patterns. The image is then segmented to obtain the ROI and used Hough transform to detect the edges of the marker patterns. Since the specialize marker patterns are in square, there will be four lines and four intersection points found in each pattern. If the angle between lines are not 0˚ or 90˚, then the components are discarded as it is not a square.

After obtained the estimation of the corner location, the authors employ the Chen's method to put the detected corner points to a sub-pixel precision locations according to the gradient info (Huang, Tsai and Lin, 2012). The centre point coordinates of the marker can be calculated after the precise coordinates of the corners are obtained. The amount of rotation thus can be found with reference to the pattern of the four white circles position. The real distance of the marker is then calculated and thus the position coordinates can be obtained, if all the coordinates of the markers are known. In the experiment using this method, the images are captured in real scenes and obtained a good result when the robot is allowed to move in a U-shape path.

In the second method using RGB-D camera, the camera is mounted to capture both the colour and depth images of the surroundings. The authors use the Microsoft's RGB-D camera system Kinect. The resolution of the colour images is 640×480 with 32 bits per pixel, while depth images having 320×240 with 16 bits per pixels. Due to the differences in resolution and FOV of the colour and depth cameras, calibration has to be done so that both the images can be aligned. 3D coordinates of each pixel can be calculated where set of 3D point cloud can be formed. The motion of the cameras can be obtained by comparing the two 3D point clouds which are obtained from different viewpoints. In the paper, the authors proposed the ICP algorithm to calculate the rotation and translation of the motion of the robot.

To increase the accuracy for computing the rotation and translation of the motion, the authors use SURF to obtain the feature points of the colour images, as well as RANSAC to filter out the outliers. Then the 3D coordinates of the feature points are used in ICP algorithm for registration. In the experiment, a mobile robot is used to move in a circular path using this method. When threshold of 100 is used, the estimated path are more likely formed a circle compared to others.

The strength of both the methods proposed in the paper is that both methods do not need any prior feature acquisition from anything in the environment, and also use the real-time images to do the computation for position of robots. The advantage of the colour information feature extraction in second method is that it can reduce the unwanted points and resulting in faster processing.

The weakness of the second method is the precision of the depth sensor is low. The authors proposed to use laser rangers instead of the depth sensor to increase the precision. Another weakness of the second method is that the image coordinates of both the depth camera and colour camera do not overlap, thus the proposed solution is calibration step is done.

2.2 Multiple ceiling cameras for mobile robot localization

Effective vision based system using several overhead cameras to determine the position and orientation of mobile robots has been proposed (Visvanathan et al., 2015). The system is able to detect and localize multiple mobile robots simultaneously in a real time scenario. All the images from the overhead cameras are calibrated in order to eliminate the fish eye effect, then further calibrated to remove angle distortion error. A marker will be assigned on top of the mobile robots for the cameras to detect the unique markers. Then a pattern matching algorithm will be used to determine the markers' position, and can eventually track the robots' position. The images obtained from all the cameras are calibrated and then merged together to obtain the localization in the global coordinate frame.

A testbed is used to experiment the system. Four Axis M1034-W network cameras are mounted directly on top of the mobile robots workspace. The camera has maximum resolution of 1280 × 800 pixels at frame rate of 30fps. The field of view of the camera of the horizontal view is 80˚, while 51˚ for vertical view. The mobile robots workspace is 3m × 6m, the cameras are mounted in the way that there will be overlapping views of the workspace. But however, the highest pattern matching score will be taken as the final position of the robot in the event that a mobile robot appears in several views of cameras.

Due to the lens aberration of the camera, the images will be having the fish eye effect. This problem is solved by applying image distortion correction using a calibration grid template, which consists of several polka dots having equal distances in both x and y direction as shown in Figure 2.2.1. The image having fish eye effect is corrected and calibration info is produced. Then the image is further calibrated to remove the camera angle distortion. Figure 2.2.2 shows the sample image having fish eye effect on the left and the corrected image on the right.

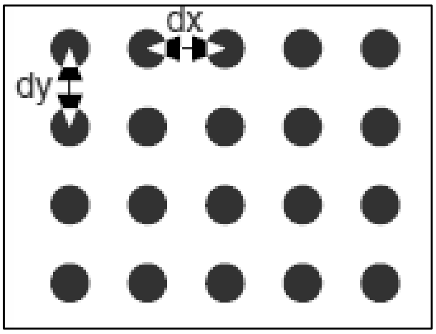

Figure 2.2.1 Calibration grid template

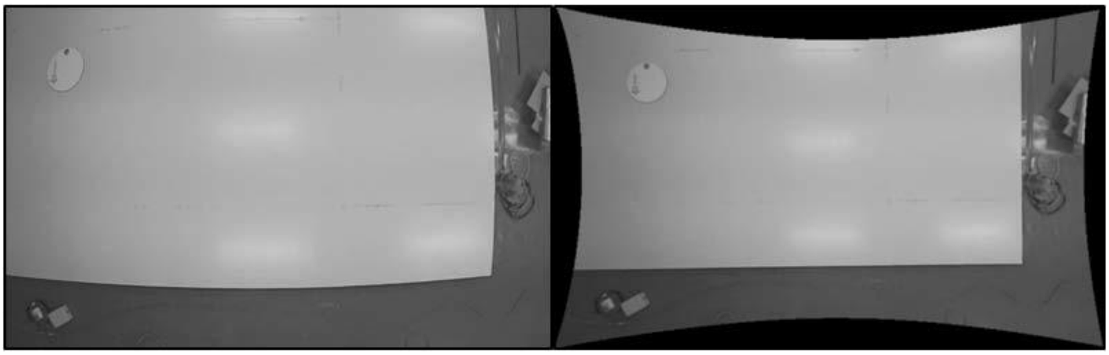

Figure 2.2.2 Distorted image (left) and corrected image (right)

The pattern matching algorithm is trained with some pattern templates designed with different shapes. The algorithm will search for the patterns and will return the pixel coordinate and rotation angle of the images captured in order to perform localization. The pixel data is converted to obtain the real position in the real environment. This algorithm can detect multiple patterns in a single image. The patterns designed are then placed on top of the mobile robots for this algorithm to detect the pattern and return the actual position and orientation of the robot. There will be a problem occur where camera perspective error will arise due to the height of the mobile robot as shown in Figure 2.2.3. The error in position estimation will increases as the height of the mobile robot increases. Referring to line from the camera penetrating the centre of the robot in Figure 2.2.3, when the height of the robot increases, the perspective error will increases too. Since the height of the robot and the camera are known, trigonometry functions is used to calculate the real position of the mobile robot.

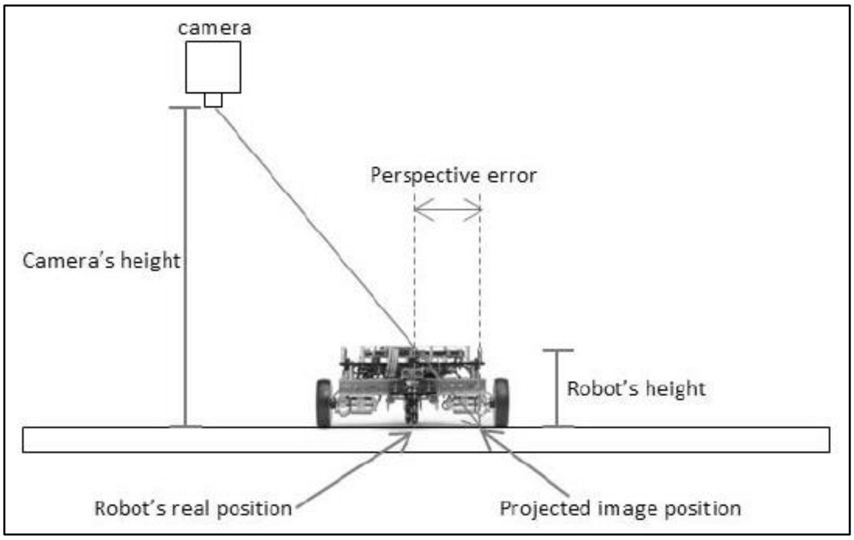

Figure 2.2.3 Pose estimation error

Several experiments were done according to the paper. Several markers are placed on the testbed without the robot in one of their experiment, meaning that the height of the object is 0cm from the view of camera. The results of this experiment are 0.7cm, 0.3cm, and 0.6˚ for average errors of position-X, average errors of position-Y, and average errors of orientation, respectively. This experiment shows that this system can provide an accurate approximation of the ground truth pose. Another experiment is done with the same settings in the first experiment, the difference is the height of the object is 13.5cm. The results of this experiment are 3.5cm for average errors of position-X and 1.2cm of average errors of position-Y. The third experiment is conducted with corrected perspective error mentioned earlier and getting 0.6cm and 0.3cm average errors for both the axes, respectively. The average errors of orientation still remained the same across three experiments, which is 0.6˚, meaning that orientation is not affected by the height of the object.

Another few experiments also conducted in this paper. To demonstrate the ability of the proposed system, the results can show that the system performed well in benchmarking other pose estimation method, the SLAM techniques. The SLAM techniques consists of Gmapping, Hector-SLAM, and Graph-SLAM. The pose estimation using the SLAM techniques are calculated by using the ground truth positions data as the reference. The mean errors of Hector-SLAM and Gmapping are consider low, having mean error of 9.7cm and 9.1cm, respectively. The SLAM techniques are chosen because the performance is better than the Windows-based SLAM algorithm, where the results show that performance of the system can achieve the real time SLAM operation (Kamarudin, K. et al., 2014).

One of the strength of the overhead camera method proposed in this paper is that the system is able to track and localize several mobile robots simultaneously. The proposed system is cost effective if compared to work done by Johnson, where multiple robots are used. In the paper by Johnson, they proposed to use LEDs as unique marker for their mobile robots (Johnson et al., 2012). The method of mounting cameras on the ceiling is cost effective compared to mounting cameras on robot itself. The authors of this paper also compared their work with the work done by Sabattini, where cameras are mounted on the robot. Gum-stix computer and Linksys IPCamera are mounted on one robot (Sabattini et al., 2010). When compared to the method in this paper, this method would be using less cost.

The weakness of this method is the images will appeared distorted (fish eye effect) due to the lens aberration of the camera used. This problem is fixed by using a calibration grid template to perform image distortion correction. The template consists of polka dots having equal distance in both x and y direction as shown in Figure 2.2.1. After calibration process, corrected images are used in further step.

2.3 Improved mobile robots localization based on ceiling landmarks

Artificial landmarks based localization is proposed in this paper. An infrared sensor is used to read the landmarks on the ceiling. However, there exist several estimation error, thus in this paper, the authors built a sensor model to reduce the usage of landmarks and also created a global coordinate system (Lan, Wang and Chen, 2016). The paper used StarGazer sensor and its landmarks. The landmarks are not affected by the lightning problem, and also provides lot of unique IDs. This landmark consists of 2 parts, one for calculate the angle and position, and other act as unique IDs as shown in Figure 2.3.1. However, there will be problems with this localization sensor as stated by the authors, including uncertainty problem due to increases in distance, discontinuity problem, and also the noise problem (Oh, Kim and Lee, 2014).

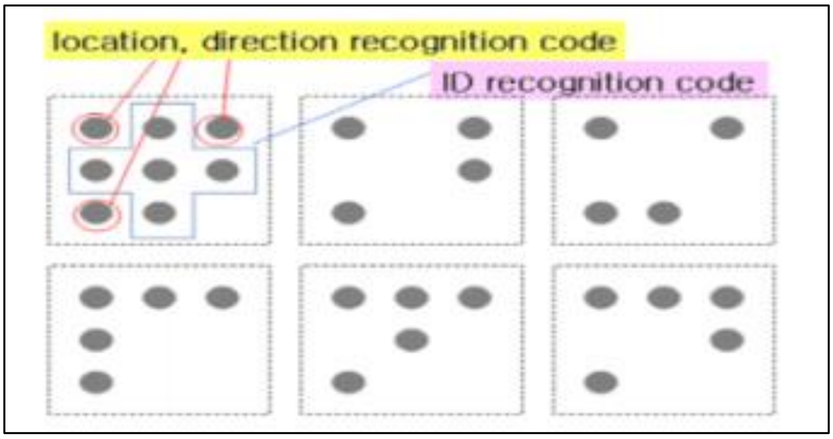

Figure 2.3.1 Landmarks with 2 parts of info provided

The landmarks are made of reflective coated film. First, the paper proposed to get the coverage area of the sensor, the sensor with circle coverage area can detect the landmark when the landmark is inside the area. Different landmark positioning system has been evaluated and analysed (López Fernández, Watkins, Pérez Losada and Díaz-Cacho Medina, 2013). However, there was not error model built by them. In this paper, the author built the error model based on the experiment data. Next, the authors also proposed the method of landmark placement. The paper taking concern of dead zones which will affect the localization of mobile robots. The paper proposed to use the least amount of landmarks and avoiding having dead zones. Then, the paper proposed to create the global coordinate system. With reference of each landmark, a relative position to a reference landmark has been calculated to build the system. In their experiment, they achieved average estimation error of 8.56 cm. The strength of this proposed method is that it uses only one sensor and successfully improve the localization system, comparing to others that uses multiple sensors.

2.4 QR code based indoor mobile robots localization and navigation

QR codes are use as landmarks for global position references is proposed in this paper. An industrial camera is used to read the QR codes at the ceilings. A laser range finder (LRF) is also used to build a map in the environment to achieve a collision-free navigation. Then, Dijkstra algorithm and Dynamic Window Approach (DWA) are used for path planning. The vision system which mounted the camera and LRF on a robot to get information from the QR codes. First, camera calibration is done, then QR code encoding and decoding is done. The information that will obtained is decoded and orientation information, and used as references for the absolute localization (Zhang, Zhang, Yang and Chen, 2015).

The QR codes are generated by a software and placed on the ceiling like Figure 2.4.1. With calculated camera's field of view, there will be at least one QR code will be detected, thus can estimate the location of the robot. Basically, when one of the QR has been detected, it will becoming the reference QR code, thus can easily determine the position of other QR codes.

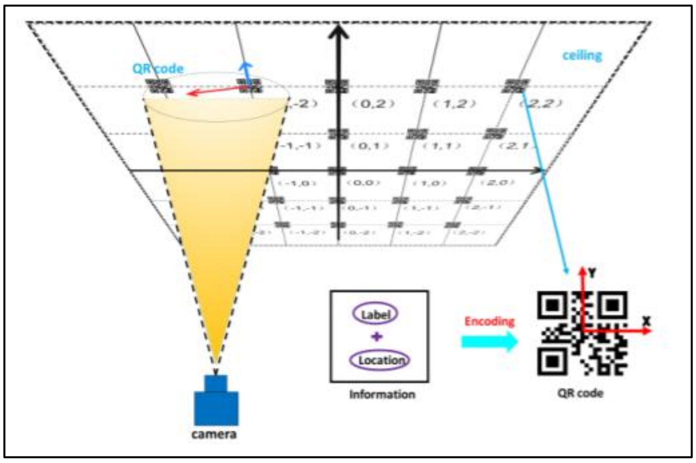

Figure 2.4.1 QR code and placement of QR codes

In the experiment, the environment has ceiling height of 2.45m, QR codes of 0.12m × 0.12m, and 1.4m of distance between 2 QR codes. The experiment obtains localization error of 6.02cm in both the X-Y axis, and is better than the method proposed in other paper (Kroumov and Okuyama, 2012). For the path planning, Dijkstra algorithm and DWA are used for global path planning and local path planning, respectively. The laser range finder is used to detect the obstacles in the environment.

In the experiment, the method proposed is effective and reliable, such that the robot can move from starting point to the ending point. The strength of this method is that the method uses QR codes which are very easy to create and cheap. But for the weaknesses of this method, the QR codes cannot be recognize in a dark environment, and also calibration has to be done.

2.5 A low cost vision sensor for localization of mobile robots

A camera is mounted on the ceiling, and transceivers is connected to enable communication with multiple devices via 6 different data pipes (Lee et al., 2015). Colour codes based landmarks are placed on top of the mobile robots, and after detecting the landmarks, real world coordinates can be obtained. The camera used is the Pixy camera that is able to find the unique colour codes and send back the data to the microcontroller. This vision sensor is also able to provide the angle value to the mobile robot. In the experiment, 6 different colour codes are used and placed on top of the mobile robots. When successfully detected a colour code, it will gives information like id, coordinates, as well as the angle. In the paper, the test environment has area of 2.9m × 1.6m, and distance between the camera and the floor is 1.8m. The proposed method achieved average localization errors from 1.42cm to 5.93cm, and also able to achieve up to 50 frames per second of performance. The strength of this proposed method is that this method is cost effective as is uses low cost cameras and also colour codes that are easy to generate. The weakness of this proposed method is that the camera may not be able to detect any mobile robots due to the strong lightning, in other words is affected by the lightning condition of the environment.

2.6 Real Time Based Tracking and Path Planning of Robot in Indoor Environment

In this paper, the authors developed a method of navigation and exploration of path by the mobile robots where obstacles were placed during experiment. The obstacles may be static or dynamic and the robots will find their ways to travel from starting point to the targeted point (Surgade et al., 2017). Sensors are placed on robots to obtain information of their surroundings. A ceiling camera was used to track the robots. During tracking, the information from the camera and sensors will be sent for further processing in real time, thus enabling the robots to avoid the static and dynamic obstacles.

In the experiment, what they concerned were the orientation and position of the robot, as well as the position of the obstacles, which are equally important information that need to be updated from time to time. Camera is used to capture live images and sent them to process, to detect and track the position. At the same time, calculations of position and path will also be done.

The system starts with capturing the RGB images and undergoes further pre-processing before calculations. Then the position of the robots and the obstacles will be identified. Then here is the computation comes in. The system will detects whether the obstacles is static or dynamic, and will compute the path for the robot to travel. In the algorithm, the path to avoid the static and dynamic path are not the same, as shown in Figure 2.6.1 and Figure 2.6.2. In their experiment, they achieved the maximum error of 4.16% and error of targeted destination of ± 20 pixels.

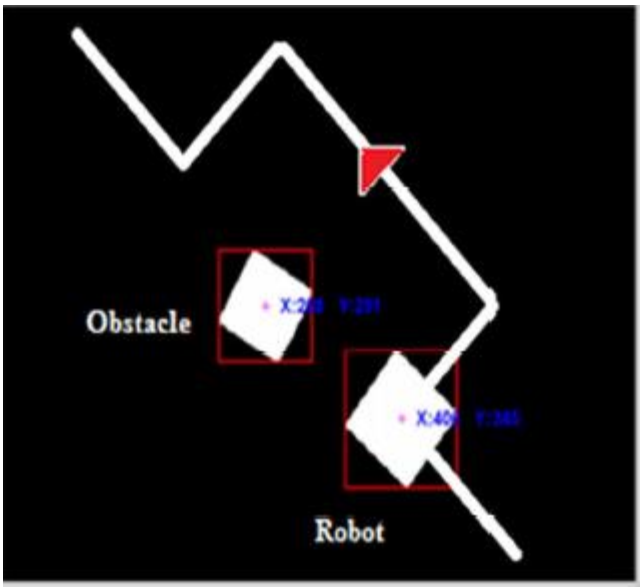

Figure 2.6.1: Path to avoid static obstacles

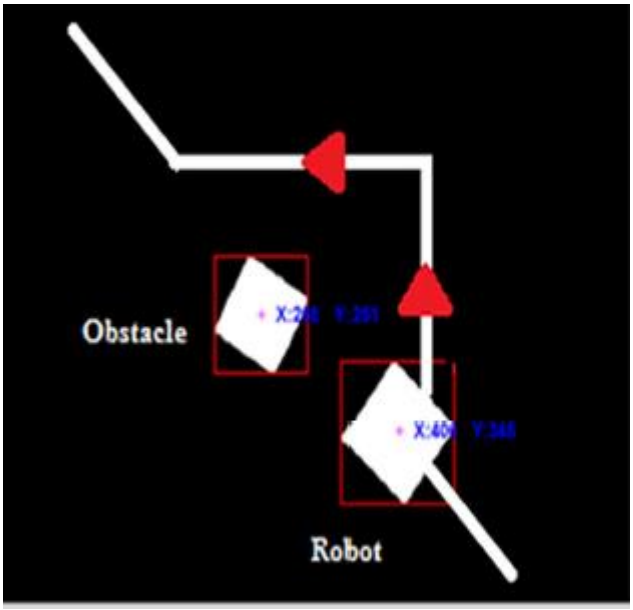

Figure 2.6.2: Path to avoid dynamic obstacles

One of the strength of this proposed method is that it can detect whether the obstacles is static or dynamic. The system will then identified it and further determine the path for the robot to avoid the obstacles. If the obstacle is static, the robot will follow the path that is parallel to the edge of the obstacle like Figure 2.6.1. On the other hand, if the obstacle is dynamic, the path will have the shape of the bounding box of the detected obstacle as shown in Figure 2.6.2. The weakness of this method is that this system assumed that the obstacles are moving only in the horizontal axis, thus the authors proposed to add in more parameters so that the system can detect the obstacles in every direction.

2.7 Analysis of Decoding for QR code in Logistic

This paper is the result of analysis for QR code readability on a package of a product (Tarjan et al.). The results are obtained by scanning QR codes on different Android smartphones. They experimented on different criteria of the QR code which includes the sizes and contents. Based on their results, the readability of the QR codes are influenced by the module size, and also the printed materials of the QR code. As shown in Figure 2.7.1, the QR code are made up of binary modules, which made up of black and white squares.

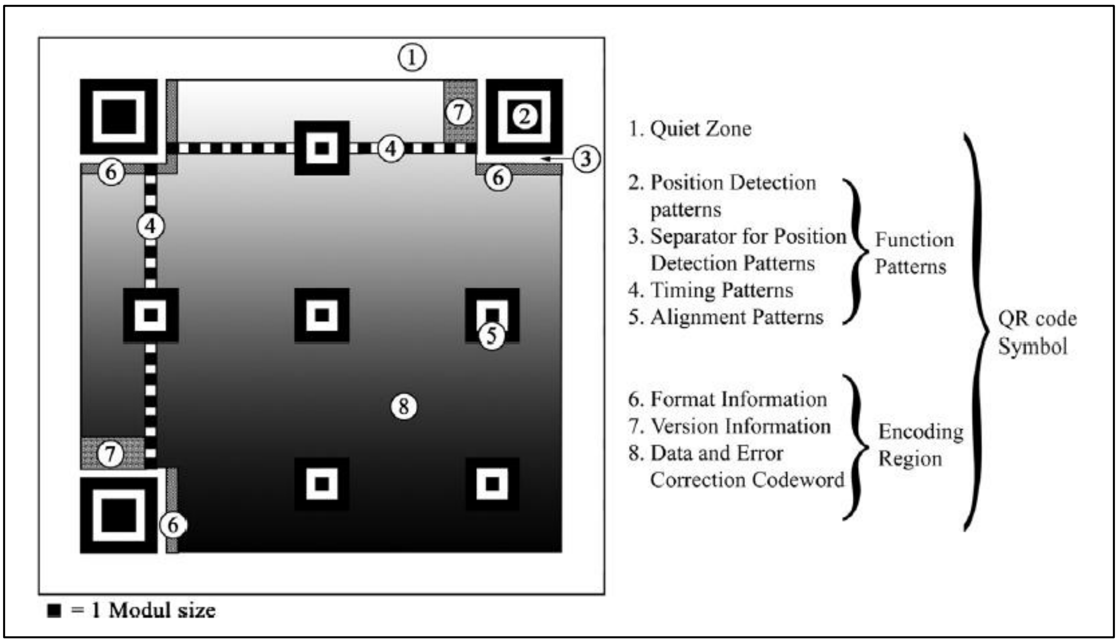

Figure 2.7.1: Different parts of a QR code

In their experiments, different types of printing materials were used to print the QR code. Different types of smartphones with 5MP and 8MP were used as well. The size of the module of the QR code were also experimented. The only constant variable in the experiment is that the placement of QR code, which is on the curved packaging surfaces. The results showed that the module size of less than 4 × 4 (px) has a poor readability, which means that the bigger sizes of module have better readability on the QR code. At the same time, the higher megapixels of the camera would make the readability ease. As shown in Figure 2.7.2, smartphone with higher megapixels and bigger module size would take the lesser time for decoding the QR codes.

Code ID	Module size (px)	Paper types 1		2		3		4		5		6		7		8	
		Camera resolution (mega pixels)															
		5	8	5	8	5	8	5	8	5	8	5	8	5	8	5	8
100.L.Lar	7 × 7	4.7	4.3	5.3	4.2	4.7	3.7	4.6	3.7	4.8	3.7	4.6	3.7	4.8	3.6	4.8	3.7
100.L.Med	5 × 5	4.7	4.0	6.0	3.6	5.9	3.7	5.0	3.6	4.6	3.6	5.3	3.6	5.0	3.7	5.3	3.7
100.M.Lar	7 × 7	4.4	3.8	4.9	3.6	4.6	3.6	4.6	3.6	4.8	3.7	4.5	3.6	4.9	3.6	4.9	3.8
100.M.Med	4 × 4	4.8	4.2	5.8	3.5	6.5	3.6	5.4	3.6	5.7	3.6	5.5	3.6	5.5	3.6	5.8	3.6
100.Q.Lar	6 × 6	4.4	4.3	4.9	3.8	4.9	3.8	4.6	3.7	4.4	3.7	4.6	3.8	4.8	3.6	4.4	3.7
100.Q.Med	4 × 4	4.4	4.5	6.9	3.8	7.8	3.8	5.3	3.7	6.2	3.8	6.3	3.7	6.1	3.7	6.9	4.1
100.H.Lar	5 × 5	4.5	3.8	5.8	3.5	5.6	3.6	4.8	3.6	5.2	3.7	5.2	3.8	4.7	3.7	4.7	3.7
200.L.Lar	5 × 5	4.4	4.4	5.2	3.5	4.9	3.9	4.6	3.6	5.0	3.6	4.7	3.7	4.7	3.8	4.8	3.6
200.M.Lar	5 × 5	5.4	3.8	4.9	6.2	5.4	4.0	4.6	3.9	4.8	3.8	4.7	3.7	4.7	3.7	5.1	3.7
200.Q.Lar	4 × 4	4.7	4.4	6.6	3.7	5.6	3.8	5.4	3.6	5.2	3.7	5.7	3.6	5.5	4.0	7.3	3.6
200.H.Lar	4 × 4	4.8	4.6	9.4	3.6	5.5	4.8	5.8	3.9	7.5	4.0	11.3	3.8	4.7	3.7	7.0	4.0
300.L.Lar	5 × 5	4.9	4.4	6.3	3.7	5.7	3.7	5.3	3.7	6.0	3.8	5.5	3.6	5.3	3.8	5.1	3.7
300.M.Lar	4 × 4	4.8	4.7	7.0	3.9	5.7	4.3	5.5	4.2	6.2	3.9	6.3	3.9	5.7	3.8	6.4	3.8

Figure 2.7.2: Time needed (s) to decode the QR codes

CHAPTER 3 SYSTEM DESIGN

3.1 System Design/ Overview

This system was developed using Python programming language. It was easy to use, and have plenty of modules to be used. This project was using toy cars to mimic the mobile robots in the warehouse. In this system, it was proposed to use cameras mounted on the ceiling to detect and track the toy cars. But to uniquely identify each of the toy car, a QR code was placed on top of the toy car for the cameras to detect.

Now we will talk about the overview of this system. First, the system can be operated to use cameras to capture the videos or processing the recorded videos. When the cameras were opened, it will continuously capture the videos frame by frame and started to detect the QR codes. Note that this process will be continued until the user terminates the program. For the detecting part, a QR code decoding library 'Pyzbar' was used to detect and decode the QR code. After successfully detected QR code(s), the system continued to track the QR code. At the same time, the coordinates of the QR code was obtained by calculating the centroid point of the bounding box of the detected QR code. The orientation of the toy car was also calculated by using the 3 positioning detector markers of the QR code at the same time. After obtaining the required information, it will be saved and printed out in the console. This information is important as it is needed for the path planning algorithm. This project was only covered the ability to get the localize position of the mobile robots, but not the path planning algorithm. The process from starting the camera until obtaining the required information were in a while loop, meaning that it will be processed frame by frame. Note that the system can be terminated at any time whenever the user wants. Figure 3.1.1 shows the detailed system design.

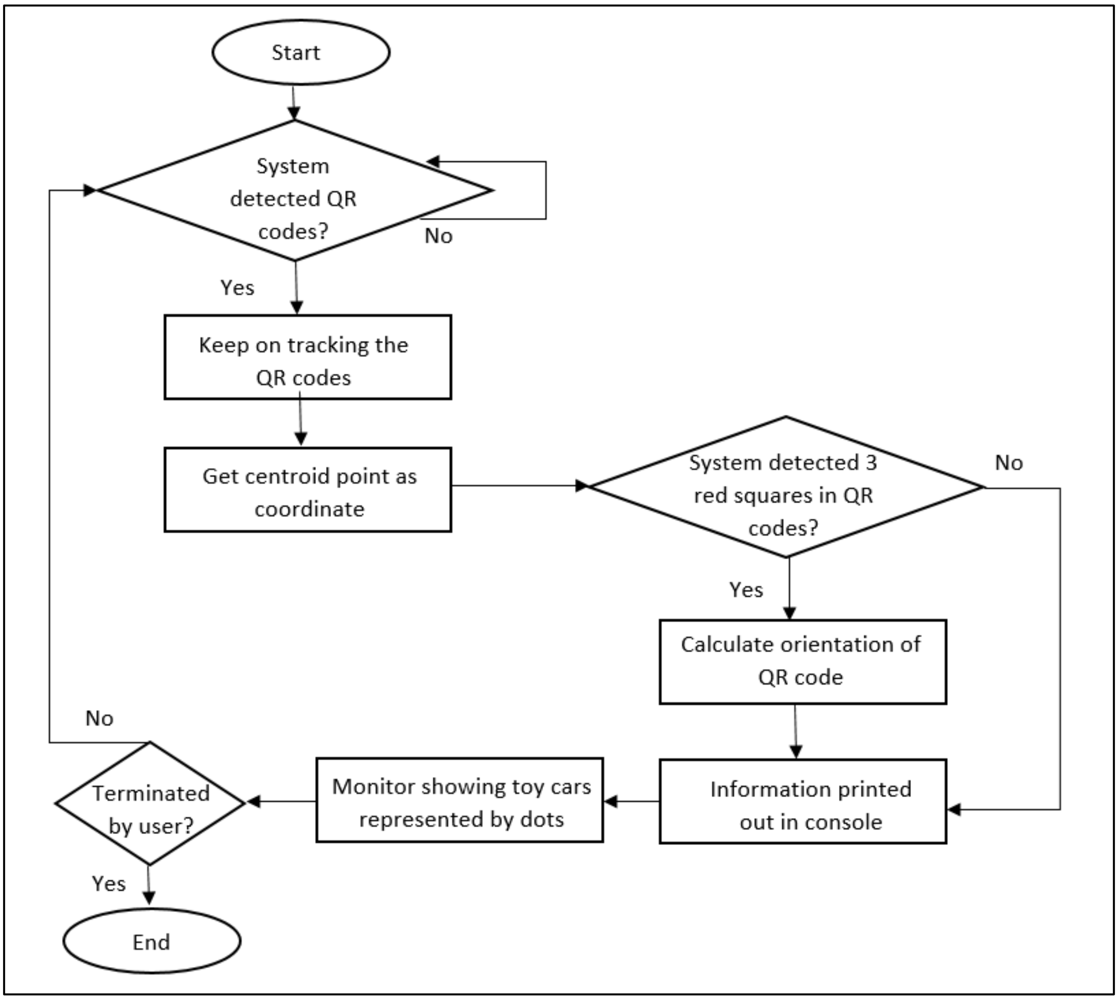

Figure 3.1.1: Detailed system design

The system will start by detecting the QR codes in each frame. If it's detected, it will continue to track the QR codes and get the centroid point as coordinates. If no, the system will keep on searching for QR codes to decode. Then the system will check whether there are 3 red coloured boxes to be detected inside the QR code, the system will calculate the orientation of the QR code if detected. Regardless of detected a not, the system will then print out the information like the coordinates, timestamp, and the orientation of QR codes in the console. At the same time, a monitor will show the coordinates of QR codes represented by dots. Throughout the process, if the user does not terminate the program, it will continue to search for QR code, will terminate if user choose to.

When viewing this at a bigger scope, it could be seen that this module of tracking can be integrate with the path planning to form a complete project, thus the information obtained from this module has to be accurate.

3.2 Detailed System Process

3.2.1 Multithreading

The video processing is a resources intensive task. With that said, applications that requires to be processed in real-time seems not possible. The input/output (I/O) operations like processing video stream was the major bottleneck that makes the applications unable to perform in real time. Therefore, this application was implemented in multithreaded way such that the computational loads has been split between multiple threads. As shown in Figure 3.2.1.1, a class called VIdeoStreamWidget is defined. Inside it, it has init method, where this method will be called whenever the class is initialized. In the init method, it will first called the VideoCapture method by cv2 to open the camera or video file. The parameter ‘src’ is the source of the camera.

```
class VideoStreamWidget(object):
    i = 0

    def __init__(self, src):
        self.capture = cv2.VideoCapture(src)
        # Start the thread to read frames from the video stream
        self.thread = Thread(target=self.update, args=())
        self.thread.daemon = True
        self.thread.start()

    def update(self):
        # Read the next frame from the stream in a different thread
        while True:
            if self.capture.isOpened():
                (self.status, self.frame) = self.capture.read()
            time.sleep(.000001)
```

Figure 3.2.1.1: VideoStreamWidget class

Then, the method will instantiate the Thread class to create thread for reading frames from the video source. In the next line, the daemon value of the thread instance is set to True. Daemon threads are threads that executing in the background, and does not block the main thread from exiting and then continues to run in background. The start() method will be called to start the thread.

Next, the update function is used to read the frame from the video source stream in another thread. A while loop is used to continue the process of reading frames until the user terminates the programs. Then, it will check whether the video source is opened in the 'if self.capture.isOpened()' line. When the condition is met, It will keep on reading from the stream. The time.sleep() line will let the input stream to stop for some time before reading next frame. Note that the argument passed into the sleep function is in seconds. In the example, it was set to let the code to read every frame in a 0.01 milliseconds interval.

3.2.2 Importing Libraries

Before starting implement the code, let's import the needed libraries. The threading library is used to perform threading as explained in last sub-section. The cv2 is the open-cv library that mainly used for computer vision, which has many modules that can be implemented easily. The time library is used to get the timestamp of certain codes at given time. Next, the numpy library is a powerful yet lightweight library that have large collection of functions to deal with array. This library is fast in processing arrays and matrices. Then the datetime library is used to print out the timestamp in date and also time. The math and itertools libraries have some mathematical functions that ease the calculation of orientation of the toy car. Lastly, which is the 'pyzbar' library that used to detect and decode the QR codes.

Note that the libraries are not available in default. The following commands are needed to import the libraries:

pip install opencv-python

pip install pyzbar

```python
from threading import Thread
import cv2, time
import numpy as np
import datetime
import math, itertools
from pyzbar.pyzbar import decode
```

Figure 3.2.2.1: Imported libraries

3.2.3 Detecting the QR codes

From this sub-section onwards until Section 3.2.5, these process will be executed in the function called ‘show_frame’, which is located inside VideoStreamWidget class in Section 3.2.1.

```python
def show_frame(self, framename):
```

Figure 3.2.3.1: show_frame function

```python
def show_frame(self, framename):
    coordinates = np.zeros((self.frame.shape[0], self.frame.shape[1], 1), np.uint8)

    if decode(self.frame):
        for barcode in decode(self.frame):
            carname = barcode.data.decode('utf-8')
            qrroi = barcode.rect
            cv2.putText(self.frame, carname, (qrroi[0], qrroi[1]), cv2.FONT_HERSHEY_COMPLEX, 0.9, (255,0,0), 2)
            cropped = cv2.rectangle(self.frame, (qrroi[0], qrroi[1]), (qrroi[0] + qrroi[2], qrroi[1] + qrroi[3]),
                                    (0, 255, 0), 3)
            ts = datetime.datetime.now()
            global idxx
            print(idxx, " ", carname, " ", ts, '\t\t\t', framename)
            idxx+=1
```

Figure 3.2.3.2: Detection of QR code

In the first line, it creates a black window for visualizing detected toy car in dots later as shown in Figure 3.2.3.3.

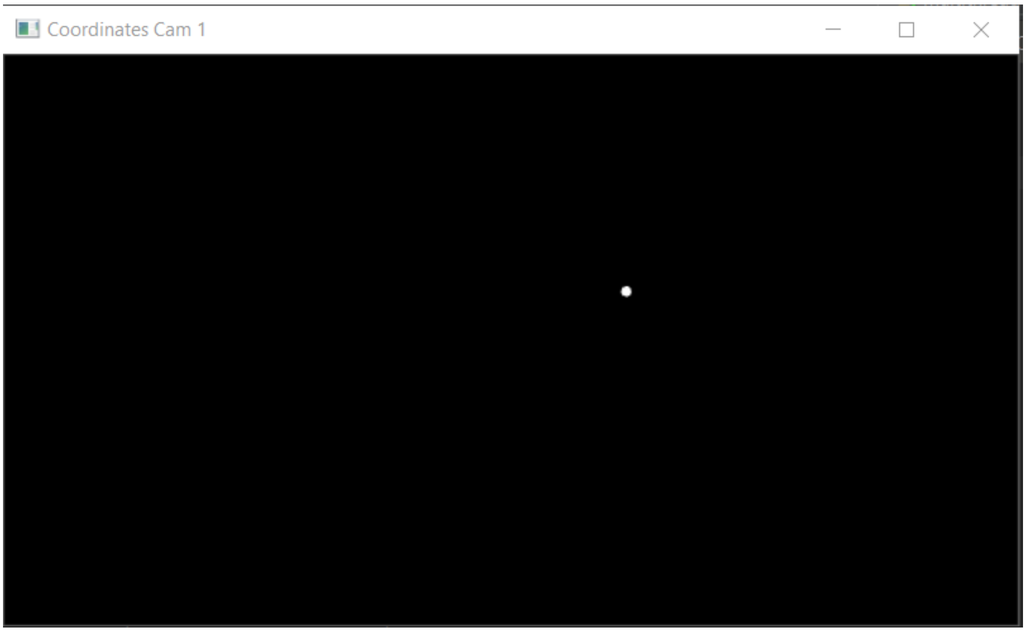

Figure 3.2.3.3: Black window for visualizing

Then, it will check if there are any QR codes to be detected and decode in the video stream. If there are, the program will detect all the QR codes in the particular frame, at ‘ for barcode in decode()’ line. This is where the program can detect multiple QR codes in a frame. In the next line, it will decode the QR code to extract the information of the QR code. For example, ‘CAR 1’, ‘CAR 2’, and etc. Then the next line ‘barcode.rect’ will return the x-coordinates, y-coordinates, width, and height information of the detected QR code. The returned information will then use in the next line to print out the text on top of the detected QR code. Then in next line, the rectangle method is used to draw a bounding rectangle around the QR code. Next, the timestamp when detecting a QR code is printed out in the console.

3.2.4 Calculating Orientation

3.2.4.1 Finding Contours

In Figure 3.2.4.1.1, the first line of code will crop the square image of QR code to be used to calculate the orientation of toy car later as shown in Figure 3.2.4.1.2. Then a copy of the cropped image will be assigned to ‘ori’ for later use. In the next line of code, it will convert the colour of the cropped image from BGR colour space to HSV colour space. In this proposed system, the 3 corners of the QR code has been replaced with red colour squares. The variables ‘lower’ and ‘upper’ are the lower and upper threshold for the red colour in HSV colour space. Then in the next line of code, it will get the masked image of the cropped QR code as shown in Figure 3.2.4.1.3. The white coloured square boxes are the areas detected with red colours. Then in the next line, it will find the contours in the masked image. Next, it will check if the number of contours is 3. The program will continue to process another QR code if the number of contours detected are not 3. Else, it will continue to calculate the orientation.

```
cropped = self.frame[qrroi[1]:qrroi[1] + qrroi[3], qrroi[0]:qrroi[0] + qrroi[2]]
ori = cropped.copy()

image = cv2.cvtColor(cropped, cv2.COLOR_BGR2HSV)
lower = np.array([161, 80, 70], dtype="uint8")
upper = np.array([180, 255, 255], dtype="uint8")
mask = cv2.inRange(image, lower, upper)

cnts = cv2.findContours(mask, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
cnts = cnts[0] if len(cnts) == 2 else cnts[1]

if len(cnts) != 3:
    break
```

Figure 3.2.4.1.1: Finding contours

Figure 3.2.4.1.2: Cropped detected QR code

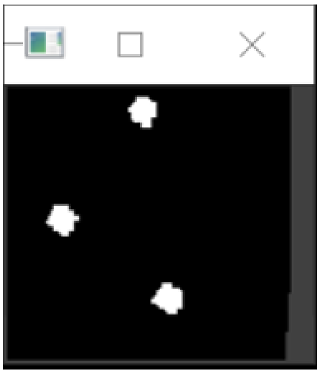

Figure 3.2.4.1.3: Masked image

3.2.4.2 Calculate Coordinates of Contours and QR code

After getting the contours from the previous section, the 3 contours and the ‘ori’ image will be feed into the function ‘threemidpoints’ as shown in Figure 3.2.4.2.2.

```
yellowmidpoints = []
yellowmidpoints = threemidpoints(cnts,ori)

combine = itertools.combinations(yellowmidpoints, 2)
checkdistidx = []
for i in combine:
    checkdist = math.hypot(i[0][0] - i[1][0], i[0][1] - i[1][1])
    checkdistidx.append(checkdist)
maxdistidx = np.argmax(checkdistidx)  # Check with group in combine that has the highest distance
```

Figure 3.2.4.2.1: Calculate coordinates part 1

```
def threemidpoints(contours, image):
    temparray = []
    for c in contours:
        x, y, w, h = cv2.boundingRect(c)
        xmid = round(x + w / 2)
        ymid = round(y + h / 2)
        cv2.rectangle(image, (x, y), (x + w, y + h), (36, 255, 12), 2)
        cv2.circle(image, (xmid, ymid), 2, (0, 0, 255), 4)
        coords = tuple([xmid, ymid])  # Assign midpoints to a tuple
        temparray.append(coords)  # Append points into list
    return temparray
```

Figure 3.2.4.2.2: Find midpoints function

First, it will create an empty array. Then in the for loop, it will find all the 3 pairs of midpoints of the detected contours, then return the array. The array will be returned like this: ([x1, y1], [x2, y2], [x3, y3]). In the next line, it utilize the itertools library to do the combinations for the 3 pairs of coordinates.

Let's simplify in this way, let's say the variable 'yellowmidpoints' are like this (a, b, c), where a,b,c represents [x1, y1], [x2, y2], [x3, y3] respectively. The variable 'combine' would look like this after the combination method: (a, b), (a, c), (b, c).

Figure 3.2.4.2.3: Combination of coordinate pairs

As shown in Figure 3.2.4.2.3, the line of code will return tuple of distance between Point A and B, distance between Point A and C, and also distance between Point B and C. Then in the for loop in Figure 3.2.4.2.1, it will calculate the distance between each pair of points. From the example given, the greatest distance would be pair of Point A and C. In the next line, it will return the index of the greatest distance of points pair. In this example, it would be the second pair, which is the Point (A, C), and assign '1' to variable 'maxdistidx'. Number 1 is returned as array is counted from 0, then the second index will be '1'.

```
# Swap(Unparticipated points in group will be swapped)
if maxdistidx == 0:
    yellowmidpoints[2], yellowmidpoints[1] = yellowmidpoints[1], yellowmidpoints[2]
elif maxdistidx == 2:
    yellowmidpoints[0], yellowmidpoints[1] = yellowmidpoints[1], yellowmidpoints[0]

for p1, p2 in zip(yellowmidpoints, yellowmidpoints[1:]):
    cv2.line(ori, p1, p2, (0, 0, 255), 2)
```

Figure 3.2.4.2.4: Calculate coordinates part 2

Next, it will check whether the points that are not 'participate' in the greatest distance of points pair are in the middle position. In the example, it is the point B, which is correct. If it is not, it will switch the places. For easy understanding, below are scenarios that would occur:

Let say the 0th pair of points is the hypotenuse:

As shown in Figure 3.2.4.2.5 and Figure 3.2.4.2.6, if the 0th pair of points is the hypotenuse, the box labelled '2' is the one that is not 'participate' in the hypotenuse. When this happens, the box labelled '1' and '2' should be swapped place so that it forms the right angle among the 3 boxes. The first 2 lines of code in Figure 3.2.4.2.4 will be doing the swapping of points.

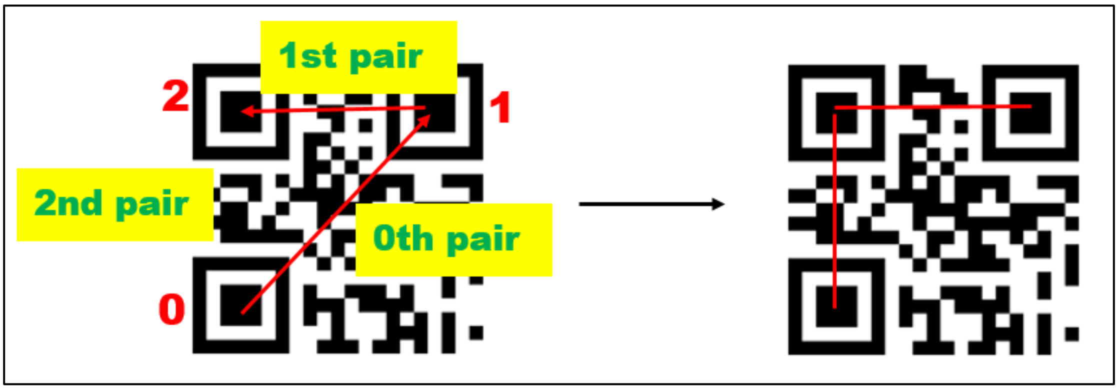

Figure 3.2.4.2.5: 0th pair hypotenuse scenario 1

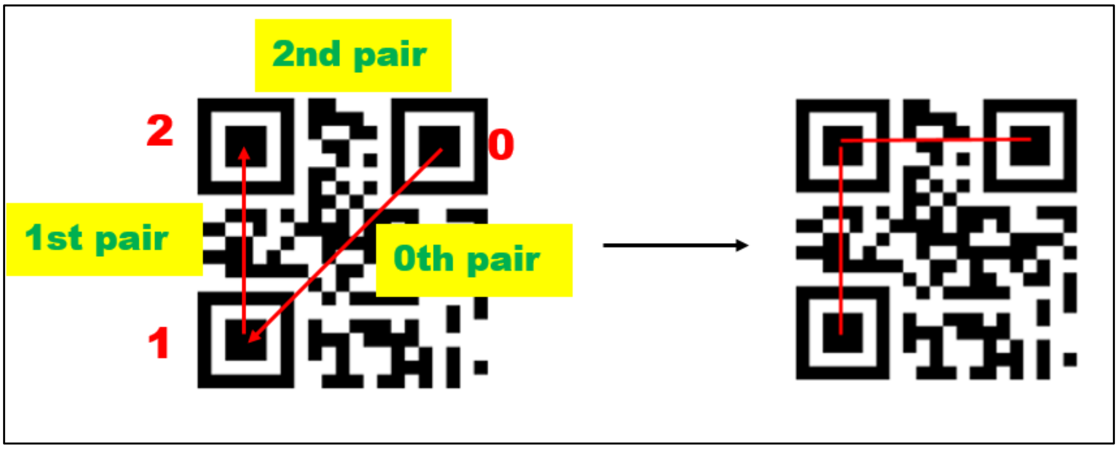

Figure 3.2.4.2.6: 0th pair hypotenuse scenario 2

How about if 2nd pair of points is the hypotenuse?

As shown in Figure 3.2.4.2.7 and Figure 3.2.4.2.8, if the 2nd pair of points is the hypotenuse, the box labelled '0' is the one that is not 'participate' in the hypotenuse.

When this happens, the box labelled ‘0’ and ‘1’ should be swapped place so that it forms the right angle among the 3 boxes. The 3rd and 4th lines of code in Figure 3.2.4.2.4 will be doing the swapping of points.

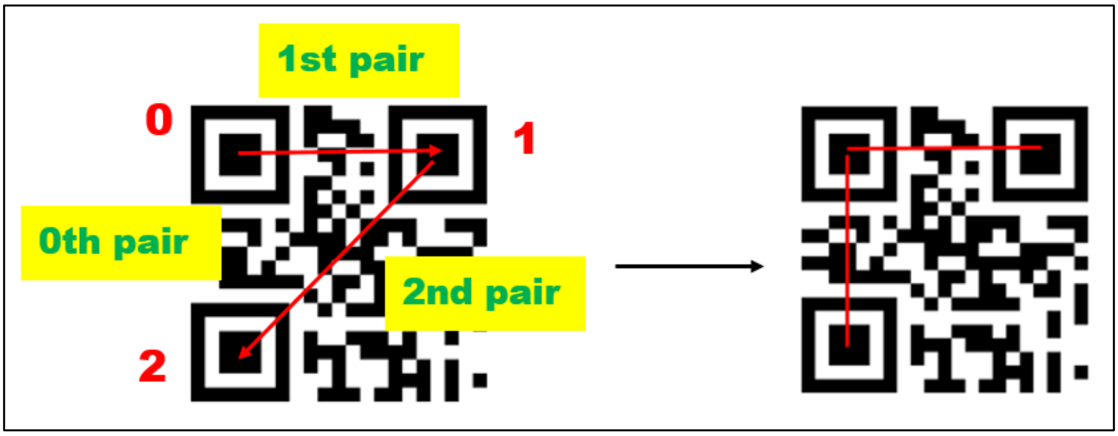

Figure 3.2.4.2.7: 2nd pair hypotenuse scenario 1

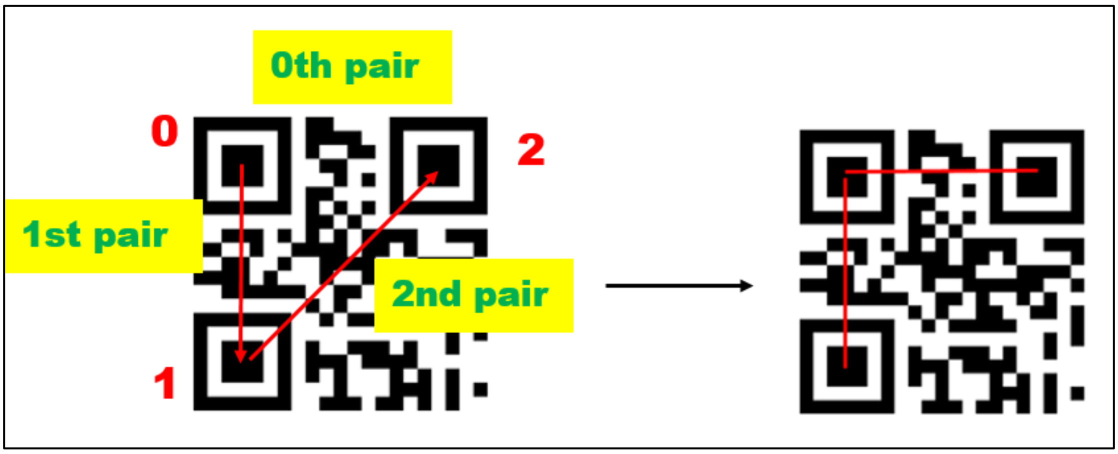

Figure 3.2.4.2.8: 2nd pair hypotenuse scenario 2

Then the remaining 2 lines of code in Figure 3.2.4.2.4 will just draw the lines connecting the 3 boxes as shown in Figure 3.2.4.2.9.

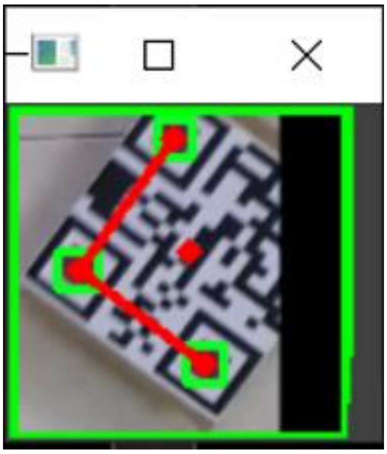

Figure 3.2.4.2.9: Connected lines

In Figure 3.2.4.2.10, the first 2 lines will calculate the midpoint formed by the hypotenuse and draw it as shown in Figure 3.2.4.2.9. Figure 3.2.4.2.11 shows the function ‘calmidpoint’ that calculates the midpoint given 2 set of points. Then the last line is to draw the coordinate of the midpoint, which is the localize coordinates of the toy car onto the black window defined earlier as shown in Figure 3.2.4.2.12.

```
# Calculate midpoint of the hypotanuse of the triangle formed by the 3 orientation square
midpoint = calmidpoint(yellowmidpoints[0], yellowmidpoints[2])
cv2.circle(ori, (midpoint[0], midpoint[1]), 2, (0, 0, 255), 4)
cv2.circle(coordinates, (round(qrroi[0] + (qrroi[2] / 2)), round(qrroi[1] + (qrroi[3] / 2))), 3, (255, 255, 255), 5)
```

Figure 3.2.4.2.10: Calculate coordinates part 3

```
def calmidpoint(p1, p2):
    return (round((p1[0] + p2[0]) / 2), round((p1[1] + p2[1]) / 2))
```

Figure 3.2.4.2.11: Calculate midpoints function

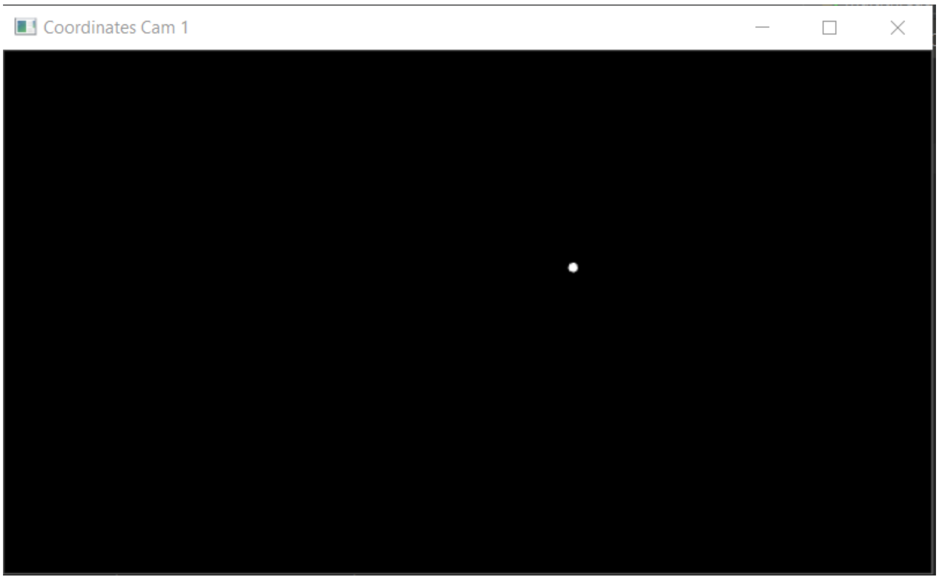

Figure 3.2.4.2.12: Coordinate on the black window

3.2.4.3 Calculate Angle

```
localanglepoint = (yellowmidpoints[1][0] - midpoint[0], midpoint[1] - yellowmidpoints[1][1])

A = (0, 0)
B = (localanglepoint[0], localanglepoint[1])
qrAngle = angle_between(B, A)

C = (0, 0)
D = (-1, 1)
referenceAngle = angle_between(D, C)

angleRotated = referenceAngle - qrAngle
if angleRotated < 0:
    angleRotated += 360
print(carname + " with rotated angle of ", round(angleRotated, 6), u"\u00b0")
```

Figure 3.2.4.3.1: Calculate angle

The first line in Figure 3.2.4.3.1 will calculate the localize angle point. As shown in Figure 3.2.4.3.2, the coordinate's plane in Python is measuring the x-axis from left to right, y-axis from top to bottom. Whereas in normal coordinate's plane, the x-axis is measuring the x-axis from left to right, y-axis from bottom to top. So, the code in first line will calculate the localize angle point such that the point is extended from the origin (0, 0) as shown in Figure 3.2.4.3.3.

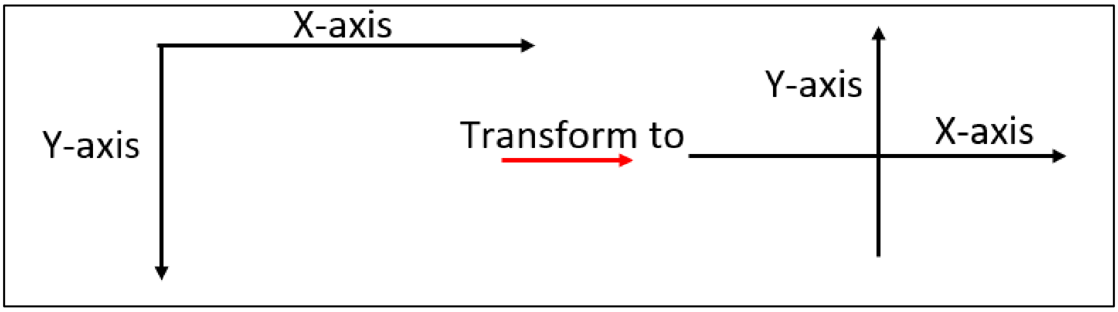

Figure 3.2.4.3.2: Transforming to normal coordinate's plane

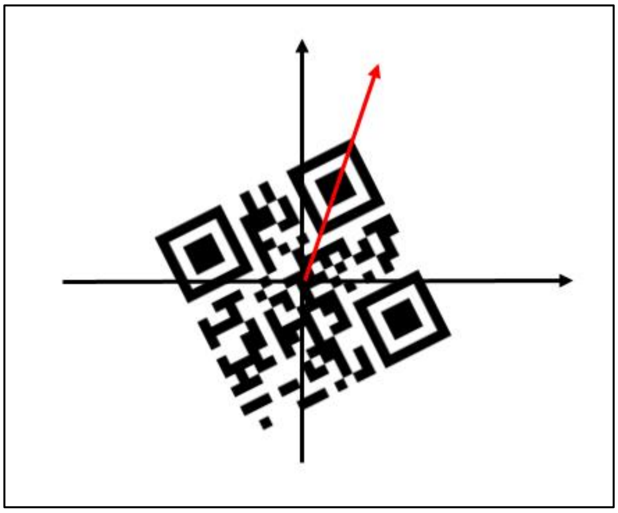

Figure 3.2.4.3.3: Localize angle point

Then, the 6 lines of code below will calculate the angle rotated of the QR code. As shown in Figure 3.2.4.3.4, it is a function to calculate angle between 2 points.

```
def angle_between(p1, p2):
    ang1 = np.arctan2(*p1[::-1])
    ang2 = np.arctan2(*p2[::-1])
    return np.rad2deg((ang1 - ang2) % (2 * np.pi))
```

Figure 3.2.4.3.4: Calculate angle between 2 points

Given example in Figure 3.2.4.3.5, the local angle is calculated between the origin and the point on the arrow shown in small yellow dot. The reference angle, as shown in Figure 3.2.4.3.6 is the angle to compare with the rotated angle by the QR code. The reference angle is pointing 135˚ as this is the normal position of a QR code. When comparing the local angle and the reference angle, the rotated angle can be calculated out as shown in Figure 3.2.4.3.7.

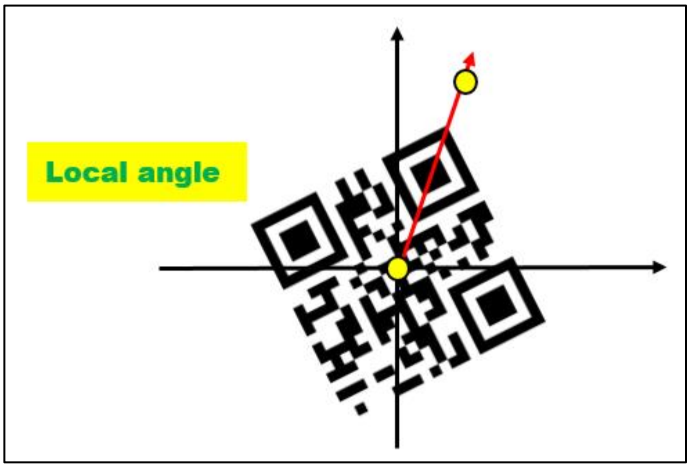

Figure 3.2.4.3.5: Local angle

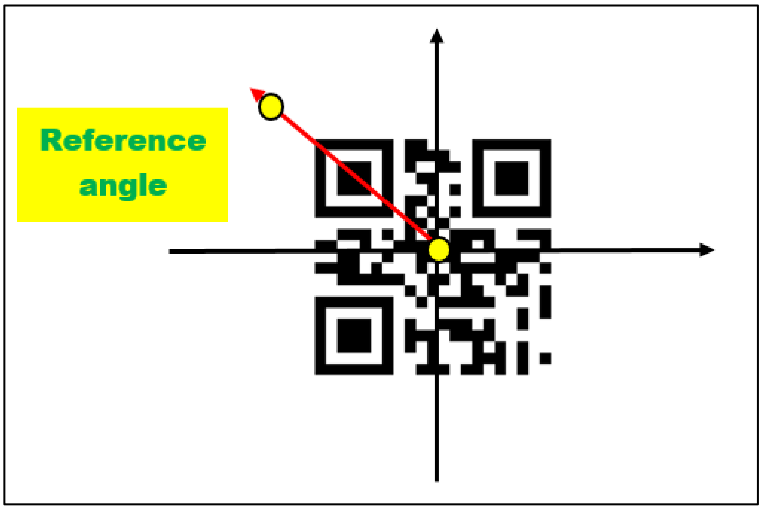

Figure 3.2.4.3.6: Reference angle

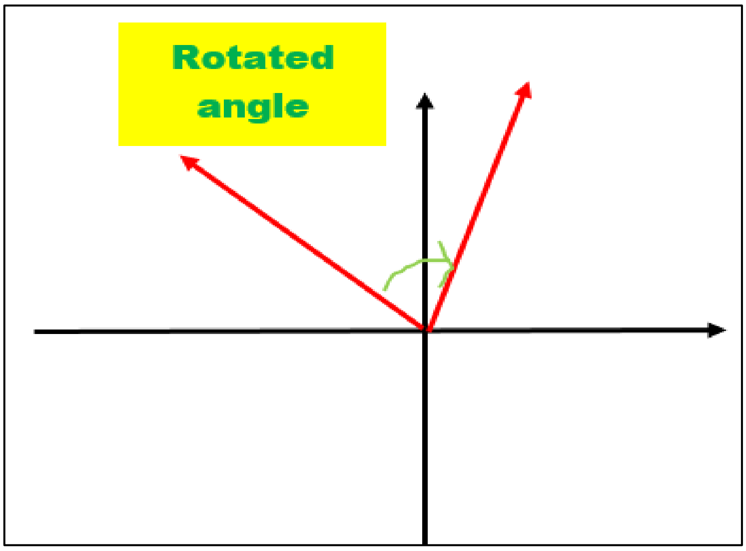

Figure 3.2.4.3.7: Final rotated angle

In the last portion of code in Figure 3.2.4.3.1, if the rotated angle is less than 0, then add a 360˚ degrees to it to make it a positive angle.

3.2.5 Showing Frame and Print Information

```
            cv2.imshow('mask', mask)
            cv2.imshow('original', ori)
            cv2.imshow("ROI", cropped)

    elif not decode(self.frame):
        pass

    cv2.imshow('Coordinates '+ framename, rescale_frame(coordinates))
    frame60 = rescale_frame(self.frame)
    cv2.imshow(framename, frame60)
```

Figure 3.2.5.1: Show frame code snippet 1

As shown in Figure 3.2.5.1, the first 3 lines of code will show the window of the masked image, original image, and also cropped image mentioned before. Figure 3.2.5.2 shows the 3 small windows shown.

Figure 3.2.5.2: Ori, cropped, and mask

Then in the next line of 'elif not decode', this line of code will check whether there are any QR code to be detected and decode. If there are no QR codes to be detect, then it will do nothing and get back to the if statement in Figure 3.2.3.2. For the next line of code, it will show the black window with coordinates of QR codes as shown in Figure 3.2.4.2.12. The next 2 lines will rescale the window showing the recorded videos. Figure 3.2.5.3 shows the function to rescale the windows.

```
def rescale_frame(frame,percent=50):
    width = int(frame.shape[1]* percent/100)
    height = int(frame.shape[0]* percent/100)
    dim = (width, height)
    return cv2.resize(frame, dim, interpolation=cv2.INTER_AREA)
```

Figure 3.2.5.3: Rescale window function

```
key = cv2.waitKey(1)

global counter
global start_time
global fps
counter += 1
if (time.time() - start_time) > x:
    fps = counter / (time.time() - start_time)
    print("FPS: ", counter / (time.time() - start_time))
    print("------------------------------------------------------")
    counter = 0
    start_time = time.time()

if key == ord('q'):
    self.capture.release()
    cv2.destroyAllWindows()
    exit(1)
```

Figure 3.2.5.4: Printing out information

The first line of code in Figure 3.2.5.4 is to get the input from user to terminate the program. In the last portion of code, if the user pressed button ‘q’, the camera stream will be release and then the program will terminates. The user can terminate the program at any time. In the centre portion of code, the program will calculate the frame per second when the programs runs. This information is printed out to monitor the performance of the computer handling this system.

CHAPTER 4 DESIGN SPECIFICATION

4.1 Tools to Use

4.1.1 Software

This project was written in Python programming language. There were many libraries and modules to be used in the Python language, so this ease the progress in developing. All the libraries can be imported easily in just a line of code. Next, the PyCharm Python IDE was used to develop this project. The features like intelligent coding assist, developer tools, and ease in using increase the efficiency in developing this project.

4.1.2 Hardware

4.1.2.1 Machine

In this project, the following machine was used to develop and done the demo of the project. Table 4.1.2.1.1 shows the specifications of the machine.

Platform	Windows 10 x64-based
Processor	Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz 2.70 GHz
RAM	8GB
Graphics	NVIDIA® GeForce® 940MX

Table 4.1.2.1.1: Machine specifications

4.1.2.2 IP Camera

The IP cameras were used to do the capturing of video for processing. Table 4.1.2.2.1 shows the specifications of the cameras.

Model	Tp-Link Tapo C200
Field of View	360˚ Horizontal, 114˚ Vertical
Video Definition	1080p (1920 × 1080 px)

Table 4.1.2.2.1: IP camera specifications

4.1.2.3 Mobile Phone

Besides using IP cameras to do the experiment, mobile phones were also used to test which device was better in doing the video streaming. The mobile phone used was having 16MP of camera. But the recorded video was in 1080 × 720 px.

4.1.2.4 Toy Cars

Different toy cars were used in the testbed experiment. All the toy cars can be controlled remotely. Figure 4.1.2.4.1 shows the images of the toy cars used.

Figure 4.1.2.4.1: Toy cars

4.1.2.5 QR codes

QR codes were one of the most important part in this project. QR code of version 1 was used in this project. In the experiment, different types of QR codes were used. Version 1, 2 and 3 were experimented but the Version 1 of QR code was chosen due to its better performance compared to others. In the experiment, QR code of size 21cm × 21 cm was used.

4.2 Requirement

The requirement describes on what the software will do and also how the system perform. In this subsection, will be discussing on the functionality of this project that needs to be fulfilled this final year project.

The following are the user requirement of the software:

- User shall be able to get the current coordinates of the toy cars
- User shall be able to monitor the toy cars' position represented by a dot
- User shall be able to receive all the information regarding the environment and also the QR codes

With that said, the ability of this project was to perform the localization of the toy cars in the environment. This project shall be able to perform the localization and calculate the orientation of the toy cars precisely. Then, this project shall also be able to detect and track the QR codes detected in the video stream. This project shall also be able to perform in real-time.

CHAPTER 5 IMPLEMENTATION AND TESTING

5.1 Implementation

This testbed based experiment has been carried out in the indoor environment with the help of cameras to capture the frame as shown in Figure 5.1.1. Due to uncertainties and technical problems encountered, the cameras used were mobile phones' cameras instead of the IP camera. Further explanation will be in Section 5.2. The cameras has been set up to point down to the ground 90˚ degrees upright. The height of the cameras placed were around 2m from the ground. The testbed area was around $4m^2$. Instead of streaming using the phones, videos were recorded and let the program to process it. Each of the camera was capturing their own area, and there were no overlapping of areas among the cameras. The toy cars were move in random direction from one area to another area and only allowed to move in the testbed area. Videos were took for 4 cars, 3 cars, and 2 cars in each experiment.

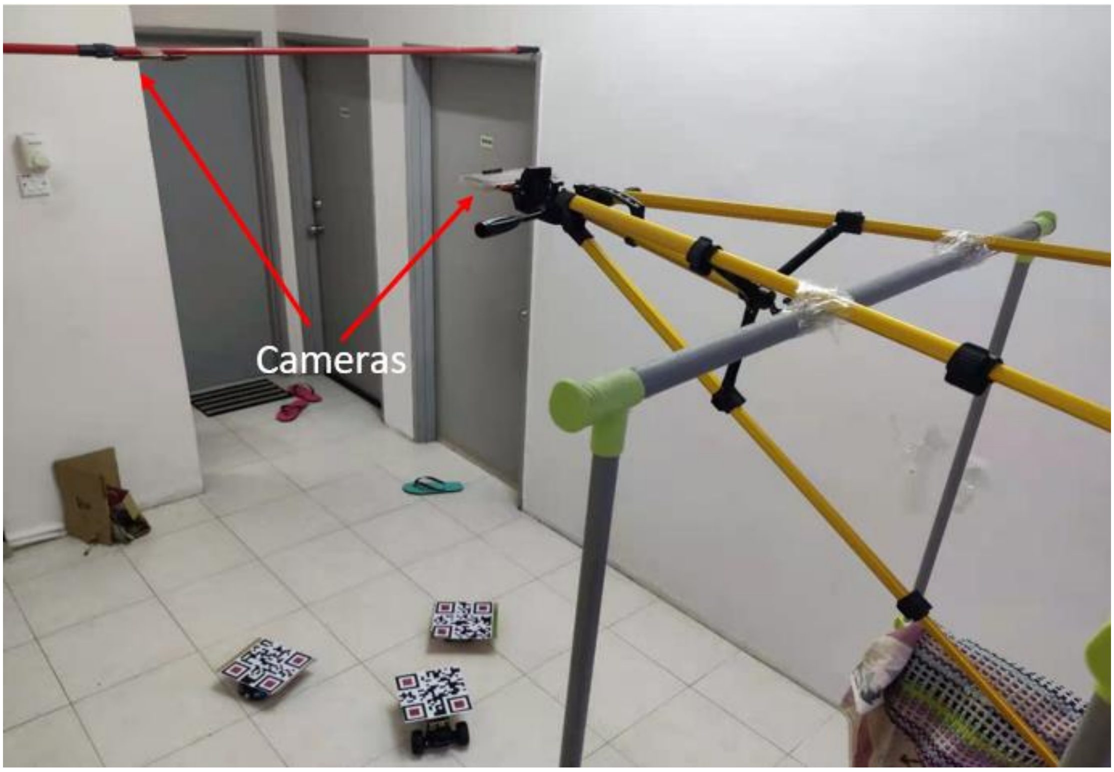

Figure 5.1.1: Testbed experiment

5.2 Testing

5.2.1 Operating System

At first, this project was developed in Windows platform, but then due to the performance concern, this project was switched to run in Linux platform in Virtual Machine. In the Linux environment, some settings has been modified to run this program. However, after tested it in the Linux environment, there seems no significantly better than in Windows platform. The environment in the Virtual Machine was also not smooth to use. Thus, Windows platform was still to use to perform this experiment.

5.2.2 Type of QR

Different types of QR codes were used to find the best settings for this experiment. Version 1, 2, and 3 of QR code were used to test whether they can be detected. Surprisingly, the Version 1 of QR codes performed better and can be detected by the program more easily compared to other versions. Figure 5.2.2.1 shows the 3 versions of QR codes.

This experiment was using the version 1 QR code with size of 21cm × 21cm. Another version 1 QR code with size of 42cm × 42cm was experimented. With the larger dimension of the QR code, it was proved that the distance between the cameras and the QR codes can be further away. The bigger QR code that was 2× larger was used and the maximum distance that the QR code can be detected was also 2× further, using the same setting. Figure 5.2.2.2 shows the smaller and bigger size of QR code.

Figure 5.2.2.1: 3 versions of QR code

Figure 5.2.2.2: Different sizes of QR codes

5.2.3 Cameras

At first, 2 IP cameras were used to conduct the experiment in real-time streaming. Due to uncertainties like wireless connection were unstable, low resolutions, and also distorted field of view, the IP cameras were not used to conduct the testbed experiment. The mobile phones' cameras were used to record the videos and fed it into this system to demonstrate the localization of toy cars. Figure 5.2.3.1 shows the IP cameras used.

Figure 5.2.3.1: IP camera

5.2.4 Colours of the 3 Corners Box of QR code

This system proposed to use the red colour as the positioning detection markers for the QR codes. At first, yellow colour was used. Due to its similarity of colour with the white colour when viewed from far, the cameras could not detect it as a proper QR code. Thus, a darker coloured was used to replace the yellow coloured boxes. As shown in Figure 5.2.4.1, the yellow coloured QR code could not be identified as a proper QR code when it was viewed at a higher range, but the red coloured QR code can still be recognized as a proper QR code at higher range.

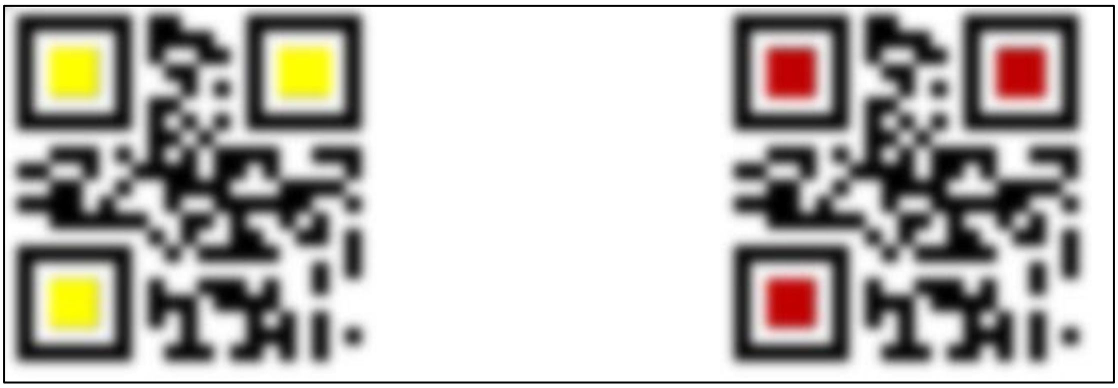

Figure 5.2.4.1: Conditions of QR codes

5.2.5 Height of the Cameras

Different heights of the cameras from the ground has been experimented using different settings. Settings that tuned and tested including type of cameras, height of cameras from the ground, type of QR codes, size of QR codes, as well as the colours of the positioning detection markers. Different settings were tuned to see which settings can have the higher height of cameras from the ground.

5.3 Verification Plan

The verification plan will outline the different verification method to test the system.

Description	Test Method
Performance of the system is not good	(a) Switch operating system (b) Switch programming language
The cameras perform badly	(a) Change settings of the cameras (b) Switching to other types of cameras
QR codes could not be detected	(a) Change the version of QR codes (b) Print the QR code blacker and clearer (c) Increase the size of the QR code (d) Decrease the distance between the cameras and the QR codes (e) Decrease the speed of the car
Orientation of QR code could not be calculated	(a) Switch to another environment with different light condition (b) Change the colours of the positioning detection markers

Table 5.3.1: Verification plan

5.4 Experiment

As shown in Figure 5.4.1, the 2 left part of the windows were the videos captured by each of the camera. Top left was the video recorded by camera 1, and bottom left was the video recorded by camera 2. The camera 1 and camera 2 were capturing their own areas and did not have any area of overlapping between both of the cameras. The middle 2 black windows showed the QR codes detected from the videos. The coordinates of the QR codes were represented by white dots in the black windows. Although there were many QR codes can be seen in the frame, but the QR code were not completely shown up to the camera, thus the system did not recognized it as a proper QR code, thus there were only 3 proper QR code. The right 3 small windows were image that drew with lines, original cropped QR image, and also masked image of the detected contours, respectively.

Both the videos were shown up simultaneously and user could monitor the console as there were related information printed out. As shown in Figure 5.4.2, the frame per second of the system was only around 4 frames per second. There were different QR codes detected from both the cameras in the particular second. The dotted lines separated out the information obtained in 1 second interval, meaning that there were 8 cars detected in one second from both the cameras. The orientation of the QR codes was also printed out for every QR code, precisely. On the other hand, the localize coordinates was also printed out, by calculating the centroid point of the bounding boxes of the QR codes.

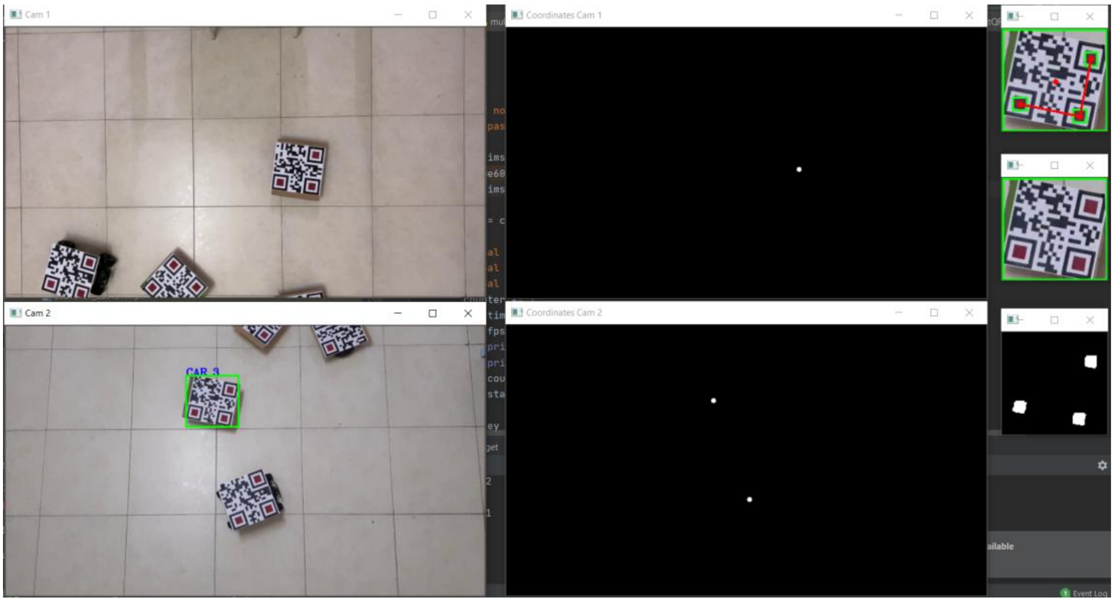

Figure 5.4.1: Demo of system

```
FPS:  3.767215698109276
-------------------------------------------------------
74   CAR 5   2021-04-16 07:51:52.056948   (648.5, 463.5)            Cam 2
CAR 5 with rotated angle of  160.709954 °
75   CAR 3   2021-04-16 07:51:52.082809   (552.0, 201.0)            Cam 2
CAR 3 with rotated angle of  189.833564 °
76   CAR 1   2021-04-16 07:51:52.342129   (780.0, 377.5)            Cam 1
CAR 1 with rotated angle of  187.373766 °
77   CAR 5   2021-04-16 07:51:52.557896   (648.5, 463.5)            Cam 2
CAR 5 with rotated angle of  160.709954 °
78   CAR 3   2021-04-16 07:51:52.584330   (552.0, 201.0)            Cam 2
CAR 3 with rotated angle of  189.833564 °
79   CAR 1   2021-04-16 07:51:52.789642   (781.0, 376.5)            Cam 1
CAR 1 with rotated angle of  185.440332 °
80   CAR 5   2021-04-16 07:51:53.021001   (648.0, 463.5)            Cam 2
CAR 5 with rotated angle of  157.693795 °
81   CAR 3   2021-04-16 07:51:53.047363   (551.5, 201.0)            Cam 2
CAR 3 with rotated angle of  190.175511 °
FPS:  4.1713407690662905
-------------------------------------------------------
```

Figure 5.4.2: Related information printed out

5.5 Analysis

The demo of the system has been shown in previous subsection, and that was the best setting so far after tuning for different settings. For the performance matter, the average frames per second of the system was only about 4 to 5, which was pretty bad. This is due to the processing power of the machine that limits the performance of the system.

For the detection part, the sizes of the QR code was only 21cm × 21cm. If were to use it in a real warehouse environment, the cameras will sure be much higher than this. The height of the cameras to the QR codes was 2m. If in the real warehouse environment, a bigger QR code could be used. However, the QR code has to be place on top of the robot to get detected, how if there were goods placed on top of the QR code? Thus, in a scenario where the goods will covered up the QR code, the QR code has to be small and put it aside with the goods. This results in the usage of bigger QR codes was technically impossible. For the positioning detection markers, as stated at above section, a darker coloured boxes would be more preferred compared to light coloured ones. This was because at a further range, the cameras could not detect the QR codes properly.

For the orientation part, the lightning condition of the environment was also important. Few experiments were done and can conclude that a dark environment will make the cameras difficult to detect and track the QR code. If the environment is dark, even if the system could detect the QR code, the system may not find any red coloured contours of the positioning detection markers, thus the orientation of the QR codes could not be determined.

Regarding the speed of the toy car, the maximum speed it could ran is 1 ms^{-1}. With that said, theoretically:

Frame Rate = 4 frames/s

Frame Interval = 0.25

Assume loss of approximately 2 frames -> Detect car at every 0.5s

Assume car speed = $1ms^{-1}$

1/0.5 = 2

Thus, distance loss to detect/track car = 1/2 = 0.5m

Theoretically, the system could only detect and track toy cars in every 0.5m movement of the toy cars in each frame interval. However, the performance of the system was better than this as shown in the demo.

CHAPTER 6 CONCLUSION

As in the project title stated, the indoor positioning of toy cars has been done successfully and proved that ceiling cameras could be used to detect multiple mobile robots in a closed environment. This project also proved that using multiple overhead cameras to detect the mobile robots could reduce the cost, as compared to install one camera or sensors into each of every robots. Furthermore, the usage QR codes in this proposed system could lower down the cost in developing, as compared to other approaches. Thus, the low cost concept has been proved once again. Next, this proposed system wanted to prove the concept of this method was working, so that this system could be implemented in the real warehouse. This system was also succeed to outperform the current development that has issues in localizing the mobile robots. This project was also succeed to get almost none average localization error compared to others. A small scale testbed based experiment was also done to test the system to mimic the real world scenario.

There were some problems encountered to complete this final year project. One of the problems encountered was the lacking of knowledge in the computer vision field. Due to not familiar with the idea on how to deal with this project, it took some time to learn. However, this was a great experience for me as this was the first time I discovered into this computer vision field. Nevertheless, this project also prepares me well to know about the current industries. Another problem encountered was the lacking of computation resources. This project was quite resources intensive, thus the performance of the system when ran on this machine was terrible. A machine with optimized hardware would be good and may push the performance of this to the limit, and even in real time. Another problems encountered was the set up for testbed environment. Due to lack of budget, the testbed rack could not be bought, thus the cameras were only supported by a rod. The cameras have to be stick firmly onto the rack. It took quite some time to set up the testbed environment.

One of the contributions of this project was to confirm that the ability of using multiple cameras were able to perform the detection, tracking, as well as localization. In this project, it proved that low cost equipment were able to capture the frame and perform localization. What low cost means was that the cost of using overhead cameras were cost less than when installing cameras on each of the robots. Another contribution was the average localization error was near to zero in this project. Calculation of orientation of QR codes were also proposed in this project by calculating the coordinates of the positioning detection markers.

This proposed system was not perfectly done, it still had some limitations. The QR codes cannot be detected during bad lighting condition. Nevertheless, the red coloured boxes on the positioning detection markers could also not detected under terrible lightning condition, thus the orientation will not be calculated, even the QR code can be detected. For future work, since the QR codes had some limitations when the distance of QR code scanner was very far away, another approach like neural network to perform detection will be proposed. Currently, the machine to run the system was not ideal, as the performance of the system was terrible, which had only average of 4 frames per seconds. A computer with optimized hardware will be used to test this system. Same goes to the quality of the cameras. Currently, the cameras used to capture the videos were not the best. To max out the performance of this system, a higher resolution cameras will be proposed so that QR code can be detected even the cameras were mounted on the ceiling.

BIBLIOGRAPHY

DF Automation & Robotics. 2020. *How An Autonomous Mobile Robot (AMR) Can Help To Fight COVID-19? - DF Automation & Robotics*. [online] Available at: <https://www.dfautomation.com/how-an-autonomous-mobile-robot-amr-can-help-to-fight-covid-19/> [Accessed 6 September 2020].

Epicor.com. 2020. *What Is Industry 4.0—The Industrial Internet Of Things (Iiot)?*. [online] Available at: <https://www.epicor.com/en-ae/resource-center/articles/what-is-industry-4-0/#1> [Accessed 6 September 2020].

Faulkner, C., 2020. *Amazon Doubled Its Profit During A Pandemic*. [online] The Verge. Available at: <https://www.theverge.com/2020/7/30/21348368/amazon-q2-2020-earnings-covid-19-coronavirus-jeff-bezos> [Accessed 5 September 2020].

Huang, W., Tsai, C. and Lin, H., 2012. Mobile robot localization using ceiling landmarks and images captured from an RGB-D camera. *2012 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*.

Johnson, E., Olson, E., Boonthum-Denecke, C., 2012. ‘Robot Localization Using Overhead Camera and LEDs’, **Florida Artificial Intelligence Research Society Conference**, North America. Available at: <https://www.aaai.org/ocs/index.php/FLAIRS/FLAIRS12/paper/view/4455/4861>. [Accessed 9 September 2020].

Kamarudin, K., Mamduh, S., Shakaff, A. and Zakaria, A., 2014. Performance Analysis of the Microsoft Kinect Sensor for 2D Simultaneous Localization and Mapping (SLAM) Techniques. *Sensors*, 14(12), pp.23365-23387.

Kroumov, V. and Okuyama, K., 2012. Localisation and position correction for mobile robot using artificial visual landmarks. *International Journal of Advanced Mechatronic Systems*, 4(2), p.112.

Lan, G., Wang, J. and Chen, W., 2016. An improved indoor localization system for mobile robots based on landmarks on the ceiling. *2016 IEEE International Conference on Robotics and Biomimetics (ROBIO)*,.

Lee, S., Tewolde, G.S., Lim, J. and Kwon, J. (2015). Vision based localization for multiple mobile robots using low-cost vision sensor. *2015 IEEE International Conference on Electro/Information Technology (EIT)*.

López Fernández, J., Watkins, C., Pérez Losada, D. and Díaz-Cacho Medina, M., 2013. Evaluating different landmark positioning systems within the RIDE architecture. *Journal of Physical Agents (JoPha)*, 7(1), pp.3-11.

logisticsIQ. 2020. *Warehouse Automation Market*. [online] Available at: <https://www.thelogisticsiq.com/research/warehouse-automation-market/> [Accessed 6 September 2020].

Marr, B., 2020. *What Is Industry 4.0? Here's A Super Easy Explanation For Anyone*. [online] Forbes. Available at: <https://www.forbes.com/sites/bernardmarr/2018/09/02/what-is-industry-4-0-heres-a-super-easy-explanation-for-anyone/#527e2c7d9788> [Accessed 6 September 2020].

Oh, J., Kim, D. and Lee, B., 2014. An Indoor Localization System for Mobile Robots Using an Active Infrared Positioning Sensor. *Journal of Industrial and Intelligent Information*, 2(1), pp.35-38.

Sabattini, L., Secchi, C., Fantuzzi, C. and Stefani, A., 2010. Bird's–Eye View Image for the Localization of a Mobile Robot by Means of Trilateration. *IFAC Proceedings Volumes*, 43(16), pp.223-228.

Surgade, A., Bhavsar, N., Thale, K., Deshpande, A. and Mohite, D. (2017). Real-time navigation for indoor environment. 2017 International Conference on Advances in Computing, Communication and Control (ICAC3).

Tarjan, Laslo, et al. “A Readability Analysis for QR Code Application in a Traceability System.” Computers and Electronics in Agriculture, vol. 109, Nov. 2014, pp. 1–11, 10.1016/j.compag.2014.08.015. Accessed 14 Mar. 2020.

Visvanathan, R., Mamduh, S., Kamarudin, K., Yeon, A., Zakaria, A., Shakaff, A., Kamarudin, L. and Saad, F., 2015. Mobile robot localization system using multiple ceiling mounted cameras. *2015 IEEE SENSORS*.

Zhang, H., Zhang, C., Yang, W. and Chen, C., 2015. Localization and navigation using QR code for mobile robot in indoor environment. *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*,.

POSTER


Objectives:
To perform localization of mobile robots using overhead cameras
Why is it so important?
ROUTING DECISION
The importance is that after localization of the mobile robots, the information collected is useful for routing decision by the system.
How does it work?
- Detect the QR codes
- Track the QR codes
- Get the coordinates and orientation of the object
- Saved the collected information for further use
Indoor Mobile Robots Localization
A testbed experiment will be carried out to demonstrate the system developed. See you there!
April 2021
Bachelor of Computer Science (Hons)
Faculty of Information and Communication Technology, UTAR Kampar
Zhun Yee Ng

PLAGIARISM CHECK RESULT

Processed on: 16-Apr-2021 14:37 +08 ID: 1560743989 Word Count: 11139 Submitted: 1	INDOOR-POSITIONING FOR WAREHOUSE MOBILE ROBOT... By Zhun Yee Ng	Similarity Index 7%	Similarity by Source Internet Sources: 2% Publications: 6% Student Papers: 1%

exclude quoted include bibliography excluding matches < 8 words mode: quickview (classic) report

Change mode print download

2% match (publications)
R. Visvanathan, S. M. Mamduh, K. Kamarudin, A. S. A. Yeon, A. Zakaria, A. Y. M. Shakaff, L. M. Kamarudin, F. S. A. Saad. "Mobile robot localization system using multiple ceiling mounted cameras", 2015 IEEE SENSORS, 2015

2% match (publications)
Wen-Tsai Huang, Chun-Lung Tsai, Huei-Yung Lin. "Mobile robot localization using ceiling landmarks and images captured from an RGB-D camera", 2012 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM), 2012

<1% match (publications)
Seokju Lee, Girma Tewolde, Jongil Lim, Jaerock Kwon. "Vision Based Localization for Multiple Mobile Robots Using Low-cost Vision Sensor", International Journal of Handheld Computing Research, 2016

<1% match (publications)
Springer Tracts in Advanced Robotics, 2016.

<1% match (student papers from 22-Apr-2020)
Submitted to Universiti Tunku Abdul Rahman on 2020-04-22

<1% match (Internet from 04-Dec-2020)
https://ieeexplore.ieee.org/document/7419715

<1% match (Internet from 26-Oct-2018)
http://www.riskreductionafrica.org

<1% match (Internet from 07-Oct-2020)
https://helpx.adobe.com/air/help/air-gamepad-protocol.html

<1% match (Internet from 13-Apr-2021)
https://www.mobile-industrial-robots.com/en/insights/get-started-with-amrs/agv-vs-amr-whats-the-difference/

Match Overview

7%

1	R. Visvanathan, S. M. ...	Publication	2%
2	Wen-Tsai Huang, Chun-...	Publication	2%
3	Seokju Lee, Girma Tew...	Publication	<1%
4	Springer Tracts in Adva...	Publication	<1%
5	Submitted to Universiti ...	Student Paper	<1%
6	ieeexplore.ieee.org	Internet Source	<1%
7	www.riskreductionafric...	Internet Source	<1%
8	helpx.adobe.com	Internet Source	<1%
9	www.mobile-industrial-...	Internet Source	<1%

All Sources

Match 1 of 26

Wen-Tsai Huang, Chun-...	Publication	2%
R. Visvanathan, S. M. ...	Publication - 2 publications	2%
Seokju Lee, Girma Tew...	Publication	1%
Seokju Lee, Girma S. Te...	Publication	1%
Lee, Seokju, Girma S. T...	Publication	1%
Seokju Lee, Girma Tew...	Publication	1%
Abu Sadat Mohammed...	Publication - 5 publications	<1%
Springer Tracts in Adva...	Publication	<1%
"Field and Service Rob...	Publication - 2 publications	<1%
Submitted to Universiti ...	Student Paper	<1%

Universiti Tunku Abdul Rahman			
Form Title : Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date: 01/10/2013	Page No.: 1of 1

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Full Name(s) of Candidate(s)	NG ZHUN YEE
ID Number(s)	17ACB04671
Programme / Course	BACHELOR OF COMPUTER SCIENCE
Title of Final Year Project	INDOOR-POSITIONING FOR WAREHOUSE MOBILE ROBOTS USING COMPUTER VISION

Similarity	**Supervisor's Comments (Compulsory if parameters of originality exceeds the limits approved by UTAR)**
Overall similarity index: 7 % **Similarity by source** Internet Sources: 2 % Publications: 6 % Student Papers: 1 %	Within the required range
Number of individual sources listed of more than 3% similarity: 0	Within the required range
Parameters of originality required and limits approved by UTAR are as Follows: **(i) Overall similarity index is 20% and below, and** **(ii) Matching of individual sources listed must be less than 3% each, and** **(iii) Matching texts in continuous block must not exceed 8 words** ***Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.***	

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.

Signature of Supervisor	Signature of Co-Supervisor
Name: Liew Soung Yue	Name:
Date: 16/4/2021	Date:

UNIVERSITI TUNKU ABDUL RAHMAN

FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY (KAMPAR CAMPUS)

CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	17ACB04671
Student Name	NG ZHUN YEE
Supervisor Name	Ts. Dr Liew Soung Yue

TICK (√)	**DOCUMENT ITEMS** Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
✔	Front Cover
✔	Signed Report Status Declaration Form
✔	Title Page
✔	Signed form of the Declaration of Originality
✔	Acknowledgement
✔	Abstract
✔	Table of Contents
✔	List of Figures (if applicable)
✔	List of Tables (if applicable)
✔	List of Symbols (if applicable)
✔	List of Abbreviations (if applicable)
✔	Chapters / Content
✔	Bibliography (or References)
✔	All references in bibliography are cited in the thesis, especially in the chapter of literature review
✔	Appendices (if applicable)
✔	Poster
✔	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)

*Include this form (checklist) in the thesis (Bind together as the last page)

I, the author, have checked and confirmed all the items listed in the table are included in my report. ____________________ (Signature of Student) Date: 16-APR-2021	Supervisor verification. Report with incorrect format can get 5 mark (1 grade) reduction. ____________________ (Signature of Supervisor) Date: 16/4/2021