

**AUTONOMOUS MOBILE ROBOT TRACKING ACROSS
MULTIPLE CAMERA VISION**

BY
LAI HONG PEI

A REPORT
SUBMITTED TO
Universiti Tunku Abdul Rahman
in partial fulfillment of the requirements
for the degree of
BACHELOR OF INFORMATION TECHNOLOGY (HONOURS)
COMPUTER SCIENCE
Faculty of Information and Communication Technology
(Kampar Campus)

JANUARY 2022

REPORT STATUS DECLARATION FORM

Title: AUTONOMOUS MOBILE ROBOT TRACKING
ACROSS MULTIPLE CAMERA VISION

Academic Session: January 2022


I LAI HONG PEI
(CAPITAL LETTER)

declare that I allow this Final Year Project Report to be kept in

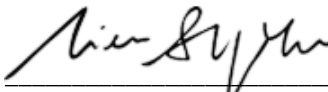
Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

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:

5, Lorong Bukit Minyak Utama 4
Taman Bukit Minyak Utama
14000, Bukit Mertajam, Pulau Pinang

Liew Soung Yue
Supervisor's name

Date: 17 APR 2022

Date: 22 APR 2022

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY
UNIVERSITI TUNKU ABDUL RAHMAN

Date: 21/04/2022

SUBMISSION OF FINAL YEAR PROJECT /DISSERTATION/THESIS

It is hereby certified that *Lai Hong Pei* (ID No: *18ACB01409*) has completed this final year project entitled “ *Autonomous Mobile Robot Tracking Across Multiple Camera Vision* ” under the supervision of Ts. Dr. Liew Soung Yue (Supervisor) from the Department of Computer and Communication Technology, Faculty of Information and Communication Technology.

I understand that University will upload softcopy of my final year project in pdf format into UTAR Institutional Repository, which may be made accessible to UTAR community and public.

Yours truly,



Lai Hong Pei

DECLARATION OF ORIGINALITY

I declare that this report entitled “**AUTONOMOUS MOBILE ROBOT TRACKING ACROSS MULTIPLE CAMERA 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: LAI HONG PEI

Date: 17 APR 2022

ACKNOWLEDGEMENTS

This is to express my overwhelming gratitude and appreciation towards all whom have offered me much guidance and assistance in accomplishing this project. First of all, I would like to thank my supervisor, Ts. Dr. Liew Soung Yue, for not only his wisdom and ideas throughout the duration of this project, his help in acquiring resources to achieve some of the ideas we discussed, but most importantly, for letting me take part in this interesting project that might make a change in the current industry.

Besides that, I would like to thank Mr. Ch'ng Chee Henn for his guidance for matters relating to software and code. Next, I would also like to express my thanks towards the Head of Department, Ts. Dr. Ooi Boon Yaik, and also the FICT FYP Laboratory for approving my requests to lend a number of equipment. Without these, I am sure that I would not have been able to carry out detailed implementation and testing for this project.

Last but not least, I would like to thank my family and friends for all the love, support and encouragements you have given me throughout the course of this project. Even though there were many ups and downs, thank you for listening and becoming my pillars of support so that I had the determination to complete this project.

ABSTRACT

There has been a marked increase of interest in automation in many industries, ever since the COVID-19 pandemic has made it difficult for too many employees to be on-site in positions that require a labour-driven workforce. One such industry is warehouses or fulfilment centres, where processes such as sorting, packaging and shipping are considered labour-intensive. The restrictions placed on the number of employees allowed at the workplace has affected the efficiency of these operations negatively, causing many parcels to be delayed in shipping. Hence, it would be beneficial for these warehouses to implement an autonomous sorting system to speed up operations, allowing faster and more accurate sorting of parcels.

An efficient warehouse localization system is the first step to achieving this. Localization plays the most important role in the efficient navigation of autonomous robots. To localize a mobile robot in a warehouse is to determine its position in the deployment area by using information gathered by external sensors. Previously deployed autonomous sorting robots in the market right now such as Amazon Robotics are costly because of the need to implement navigation sensors in each robot. Since the objective of this project is to reduce the cost of an autonomous sorting system, hence in project 1, we attempt to use a single webcam to perform detection of a moving object and display the coordinates on the screen. After that, we will use image stitching to combine multiple camera views to track the moving object across areas captured by different cameras.

TABLE OF CONTENTS

TITLE PAGE	i
REPORT STATUS DECLARATION FORM	ii
FYP THESIS SUBMISSION FORM	iii
DECLARATION OF ORIGINALITY	iv
ACKNOWLEDGEMENTS	v
ABSTRACT	vi
TABLE OF CONTENTS	vii
LIST OF FIGURES	x
LIST OF TABLES	xi
LIST OF SYMBOLS	xii
LIST OF ABBREVIATIONS	xiii
CHAPTER 1 INTRODUCTION	
1.1: Problem Statement and Motivation	1
1.2: Background Information	3
1.3: Objectives	5
1.4: Proposed Approach	6
1.5: Contributions	6
1.6: Report Organization	7
CHAPTER 2 LITERATURE REVIEW	
2.1: Mobile robot self-localization system using single webcam distance measurement technology in indoor environments	8
2.2: Mobile robot localization via indoor fixed remote surveillance cameras	10

2.3: Localization and navigation using QR code for mobile robot in indoor environment	12
2.4: Mobile robot localization system using multiple ceiling mounted cameras	14
2.5: Robot localization using overhead camera and LEDs	16
CHAPTER 3 SYSTEM DESIGN	
3.1: System Design / Overview	18
3.2: Detailed System Process	20
3.2.1: Importing Libraries	20
3.2.2: Image Stitching	21
3.2.3: Detecting the ArUco Markers	22
3.2.4: Calculating Coordinates	23
CHAPTER 4 DESIGN SPECIFICATION	
4.1: Tools to Use	24
4.1.1: Software	24
4.1.2: Hardware	24
4.1.2.1: Machine	24
4.1.2.2: Webcam	25
4.1.2.3: Toy Cars	26
4.1.2.4: Aruco Markers	26
4.2: Requirement	27
CHAPTER 5 IMPLEMENTATION AND TESTING	
5.1: Implementation	28

5.2: Testing	29
5.2.1: ArUco Marker Dictionary	29
5.2.2: ArUco Marker Size	30
5.2.3: Webcam	30
5.2.4: Height of the Cameras	31
5.3: Verification Plan	32
5.4: Experiment	32
5.5: Analysis	34
CHAPTER 6 CONCLUSION	35
BIBLIOGRAPHY	36
WEEKLY LOG	38
POSTER	46
PLAGIARISM CHECK RESULT	47
PLAGIRAIISM CHECKLIST	49
FYP 2 CHECKLIST	50

LIST OF FIGURES

Figures	Page
Figure 1.1	1
Figure 2.1.1	9
Figure 2.2.1	10
Figure 2.3.1	12
Figure 2.4.1	15
Figure 2.5.1	16
Figure 3.1.1	19
Figure 3.2.1.1	20
Figure 3.2.2.1	21
Figure 3.2.2.2	21
Figure 3.2.3.1	22
Figure 3.2.3.2	22
Figure 3.2.3.3	22
Figure 3.2.4.1	23
Figure 4.1.2.3.1	26
Figure 4.1.2.4.1	26
Figure 5.1.1	28
Figure 5.2.1.1	29
Figure 5.2.1.2	29
Figure 5.2.2.1	30
Figure 5.2.3.1	30
Figure 5.2.3.2	31
Figure 5.2.3.3	31
Figure 5.4.1	33
Figure 5.4.2	33

LIST OF TABLES

Tables	Page
Table 2.3.1	13
Table 4.1.2.1.1	24
Table 4.1.2.2.1	25
Table 4.1.2.2.2	25
Table 4.1.2.2.3	25
Table 5.3.1	32

LIST OF SYMBOLS

Symbol	Meaning
\$	United States Dollar (USD)
°	Degree (Angle)

LIST OF ABBREVIATIONS

2D	Two-Dimensional
AGV	Automated Guided Vehicle
AMR	Autonomous Mobile Robot
ArUco	Augmented Reality library from the University of Cordoba
CNN	Convolutud Neural Network
COVID-19	Coronavirus Disease
GPS	Global Positioning System
IDE	Integrated Development Environment
LED	Light Emitting Diode
QR code	Quick Response code
RFID	Radio Frequency Identification

CHAPTER 1: INTRODUCTION

1.1: Problem Statement and Motivation

In year 2020, the COVID-19 pandemic had caused many disruptions to daily life as we know. After experiencing various versions of movement restriction orders, most people have accepted the concept of social distancing as the new norm. Consumers have turned to online shopping to make their purchases instead, and this had led to a sharp increase in the demand for package shipments. However, most traditional warehouses that are still mostly operated by humans are not equipped to deal with such a massive influx of shipments, and they were unable to effectively sort these packages which inadvertently led to pile-ups and shipping delays.

The solution to this issue lies in the automation of sortation centres to meet the demands of sorting. A successful example of a company that had managed to capitalize on this upsurge of online purchasing is Amazon. In 2020, Amazon has reported net revenue of almost \$386 billion compared to \$280 billion last year 2019, and this figure continued to rise in 2021 up to \$470 billion [1]. One of the main reasons that Amazon has succeeded where others have failed is because of its efficient distribution system – highly automated sortation centres where thousands of autonomous robots are deployed to sort packages according to zip codes [2][3].



Figure 1.1: Autonomous mobile robots deployed in Amazon sortation centres

Taking a closer look at the automation of robots, we would discover that the core component that enables this automation is localization. The localization phase comes after the perception phase, where these robots analyse the information of the surrounding environment that was collected using external sensors such as cameras or radio frequency identification (RFID). The robot will be able to estimate its current coordinates based on those information,

and the coordinates will be passed on to the cognition phase to decide the next steps it should take, and to the motion control phase to move the robot according to the planned trajectory.

However, there is a reason why most warehouses have yet to implement autonomous sorting systems – it is because current existing systems such as Amazon Robotics are costly. In these methods, an on-board sensor such as a 2D camera or a laser is installed on each robot unit for them to be able to navigate the warehouse. This is not very cost-effective as the user will have to spend extra cost to replace the sensors on the robots if they become faulty or damaged over time. Besides that, programming modules need to be integrated in each unit to make decisions based on the information received from the sensor so that they do not collide with other robots or cause a deadlock.

Referring to the problem statement above, more businesses may be encouraged to employ autonomous mobile robots in their warehouse for the increased sorting efficiency if a cheaper alternative for the automation can be found and implemented.

This will only happen if the return of investment of implementing this system is higher than the cost of hiring human labour over the long term. Besides that, it will lessen the amount of human errors while performing repetitive tasks, like sorting a package wrongly, as autonomous robots have the ability to consistently extract information on the packages and then sort them precisely. At the same time, this eliminates the need for human labour, since the robots are able to independently function without the need for human intervention. This means that warehouses can transition their workforce to other less repetitive tasks, or reduce the number of employees needed.

Hence, these are the motivations to explore this project, and to see if it is possible to implement a cheap alternative by stitching multiple overhead camera views for a wider floor plan coverage and to locate and track the moving sortation robots, paired with a central path planning algorithm to address the movement of the robots and prevent them from crashing into one another or enter a deadlock situation. The path planning aspect will not be covered in this project.

1.2: Background information

The use of robotics paired with autonomous systems has slowly increased in both the business sector and in our daily lives in the last two decades. But what are autonomous robots? An autonomous robot is defined as a robot that is programmed to mimic the ability of humans to make decisions based on the conditions it receives from the environment, and then perform an action connected to the decision [4]. One of the best examples of an autonomous robot that is now widely used is the Roomba, a vacuum cleaner released by iRobot back in year 2002. Earlier generations of Roomba robots had used algorithms such as wall following and spiralling to navigate the room. However, starting from the seventh generation, Roomba robots employed an upwards facing camera and a downward facing infrared floor tracking sensor to automatically map and navigate a room, increasing its efficiency and ensuring a more complete floor coverage without going through the same area multiple times.

The relationship between the use of autonomous robots and warehouse package sorting is that autonomous robots can speed up the sorting process and also increase the accuracy of the sorting. This can be done by incorporating object recognition, visual tracking, and indoor positioning to coordinate the movement of multiple robots at the same time inside the warehouse.

Object recognition refers to a broad array of tasks, which include image classification, object localization, object detection, etc. Image classification extracts information from an image then predict the class that the object belongs to. Usually, image classification involves the use of convoluted neural networks (CNN) to extract low-dimensional features that could be used to identify an object class. Nowadays, image classification is used for various purposes in many industries.

Object localization is the task of locating an instance of an object category in an image, and showing the result by bounding the instance in a box. The CNN that is used for image classification is also used for object localization. The main difference of object localization from object detection is that the latter will search all objects in the image and further classify the objects, meaning that it is an extension of object localization. In this project, object detection will be utilized to detect and identify multiple robots to track their movement across the warehouse.

The two most common techniques of autonomous robot localization are probabilistic approaches and autonomous map building. Probabilistic approaches include Markov localization and Kalman filter; Markov localization will be able to predict the location of a robot even though it starts moving from an unknown position by using a probability distribution of possible locations, so that after the robot moves, the additional information gathered will help to determine the actual position of the robot. Kalman filters differ in that it usually requires the starting position of the robot, and uses a Gaussian distribution to estimate the position. Autonomous map building on the other hand can be used when the environment is large or dynamic. Simultaneous localization and mapping (SLAM) algorithms are used here instead, although it may also use the Kalman filter as the learning method [5].

Visual tracking takes object detection a step further by taking the initial set of object detections, assigning an ID for each object and then continuously tracking them as they move around, while retaining the ID that was assigned to the objects. Centroid tracking is a simple algorithm that can be implemented to achieve object tracking, and it works by using the Euclidean distance between existing object centroids and new object centroids in the next frame in a video. The algorithm assumes that the distance between the centroids of the same object will be smaller than the distance between other objects. Therefore, in the subsequent frame, the algorithm will associate centroids that have the minimum distance between each other, and update the coordinates saved under each ID to the coordinates of the new centroid [6].

Lastly, having an indoor positioning system is important for determining the location of the robot inside the warehouse. Since GPS cannot be used effectively indoors due to the weakening of signal strength as it passes through construction material, indoor positioning relies on a wide range of other technology such as Wi-Fi based positioning system (WPS), Bluetooth, radio frequency identification (RFID), and even computer vision.

1.3: Objectives

The main objective of this project is to reduce the cost of an autonomous sorting system. Existing autonomous sorting systems are costly because of the need to implement navigation sensors in each robot. This can be solved by using overhead cameras instead, which will reduce the number of cameras needed to cover the floor area, and in extension the cost of equipment required to sense the environment and detect objects.

At the end of this project, the expected deliverable is a software capable of detecting multiple objects from videos, identify each object uniquely, and store their coordinates along with the timestamp. First, the software should be able to receive multiple video feed from multiple overhead cameras in real time. Then, it should be able to calculate the homography matrix and stitch the video feeds into one single stream. Lastly, object detection may be performed on the video stream and return the positions of multiple objects inside each area. Hence, the software should be capable of keeping track of objects when they pass through the boundaries of one camera area to another area.

The software developed should help to reduce the hardware cost for the implementation of autonomous mobile robot sorting systems. If more companies are willing to invest to convert to this system, it can improve the sorting efficiency of incoming parcels, and in extension prevent delayed package delivery, which will improve customer satisfaction and retain and attract more customers to purchase through this company.

It is also imperative to find a solution to the problem of losing track of AMRs as it moves across the boundary of a camera view to the next camera view. Existing papers such as on this subject have explored applying the Markov model algorithm to track human motion across multiple fixed cameras [7]. However, I would like to propose a simple, intuitive solution instead: since we would have been using unique identifiers to track each AMR, the few frames where the AMR moves across the camera video feed boundaries can be considered negligible, since the algorithm will be able to locate the AMR again after it has passed, and still be able to identify the AMR by virtue of the unique identifiers mounted on top of the AMR.

1.4: Proposed Approach

An experiment to test out the ideas discussed above has been carried out. Within the test area, the cameras are mounted on top of stands and rotated so that they were facing downwards 90° straight towards the floor. Each of the camera will have some overlapping field of vision to carry out video stitching, which will be explained in detail in Chapter 3. The toy cars in place of AMRs will be placed on the floor, and taped on top of the toy cars will be unique ArUco codes from the same class for identification.

The toy cars will be controlled to move in different directions, and also across the whole stitched video stream to show that the program is capable of performing detection to recognize objects and continuously track the detected objects across multiple camera areas. The program is also capable of showing the coordinates of the ArUco codes, allowing the information to be sent and utilized for path planning algorithms.

1.5: Contributions

The software that is developed in this project is confirmed to be able to identify and derive the coordinates of moving objects in real-time, using multiple overhead cameras to cover a wide area. Firstly, thanks to the application of homography matrix calculation, the field of vision of each camera can be stitched together to create a complete map of the floor, proving that the use of multiple overhead cameras is indeed feasible. Next, the ArUco codes used in this project to uniquely identify the robots allowed the program to precisely perform localization and tracking. Lastly, thanks to

When deployed in a warehouse, this software will be able to help detect and localize the mobile robots, further using the information in a navigation system to help increase the efficiency of the robots moving to and from sorting points and hence the throughput. Besides that, using autonomous robots means that there will be a much lesser need to employ a large number of staff in the warehouse, so the warehouse can continue operations no matter the time of the day.

1.5: Report Organization

The report is organized into 6 chapters. Chapter one above talks about the basis of this project. In Chapter 2, background information about object localization and detection, use of multiple cameras and unique identifiers and also techniques such as computation of homography were reviewed. Chapter 3 lays out the detailed explanations for the system design of this project. In Chapter 4, we discuss the design specifications, which covers the choices and reasons for the use of such specifications. Chapter 5 will show the implementation and testing of this project. And lastly, Chapter 6 is the conclusion segment for this project.

CHAPTER 2: LITERATURE REVIEW

In the following section, some current practices and solutions that are connected to the problem of indoor robot localization are reviewed. This includes different ways to detect and identify the position of the robot, some different methods to uniquely identify multiple robots if an external camera is used instead of equipping each robot with its own vision sensor, as well as the advantages and limitations of each solution.

2.1: Mobile robot self-localization system using single webcam distance measurement technology in indoor environments

This paper by Li et al. discusses the use of two simple algorithms to locate a patrolling autonomous robot using fixed and low-cost webcams [8]. The algorithms are used for calculating the distance between the centre of the robot and the webcam, and also the distance between the centre of the robot and the wall, which will be then be converted into coordinates of the robot and, as the robot moves, will be marked on the map as the captured path.

Before the localization technique can be used, the image captured by the camera must be processed. Firstly, the camera is corrected to fix the tangential distortion of image capture by calculating the distortion coefficient vector and the camera conversion matrix. Next, image segmentation is carried out by subtracting a binary image from the original binary background image to extract the position of the robot inside the image. A few morphological image processing operations are applied to remove excessive noise from the image extracted. Lastly, connected-component labelling is carried out to determine groups based on the connectivity of the extracted pixels. Using this processed image, the algorithms can be applied to calculate the coordinates of the robot in the image.

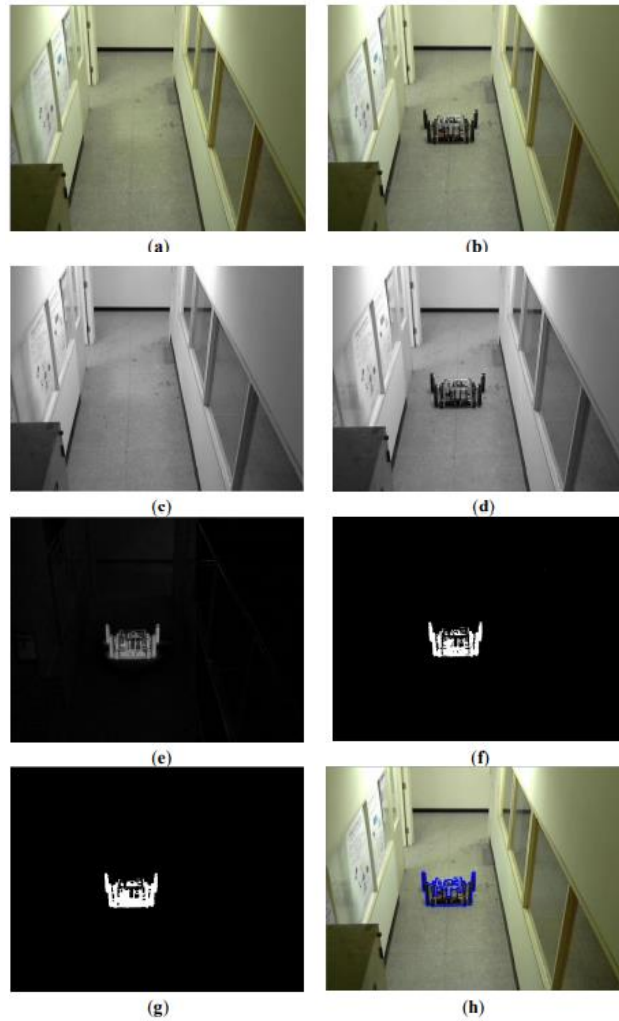


Figure 2.1.1: (a) Background image, (b) Captured image, (c) Grayscale of background image, (d) Grayscale of captured image, (e) Background subtraction, (f) Binarization processing, (g) Morphological processing, (h) Connected components labelling

There are three main advantages of this proposed method – firstly, the computational burden is lessened through the use of simple formulas to estimate the coordinates of the robot. Next, the method is easy to implement, as there are only three simple parameters that need to be defined. Lastly, cheaper low-resolution cameras can still be used instead of requiring more expensive high-resolution cameras.

The weakness of this method is the spike in error when the robot first moves into frame. It is noticed that whenever the robot first appears in the view of the camera, the two front arms of the robots caused the model to detect only one of the arms and hence mark down the wrong coordinates. Besides that, as the robot moves further away from the camera, the measurement errors increase, but this can be solved by using cameras with higher resolution.

2.2: Mobile robot localization via indoor fixed remote surveillance cameras

In the project by Shim & Cho, two cameras are mounted on opposite ends of a corridor at an angle [9]. This is to ensure that the object can be detected by the camera on the other side if it was occluded on one side. The image captured is converted by homography to get an even width and length of the floor area, which is called an air view image, and used as a two-dimensional map. If multiple cameras are used, the air view images of the two cameras can be combined into a single image to use as the map. This also solves the distortion of camera view at the same time. For object detection, the images captured are converted to binary images, and the two-dimensional maps are then obtained through homography. The maps will be combined and will show the common area of the binary image, which is the object.

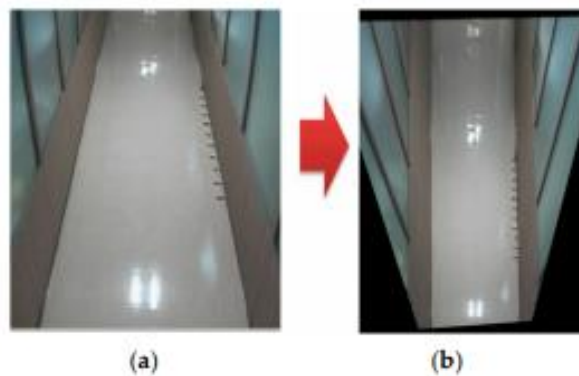


Figure 2.2.1: (a) Original image, (b) Air view image of (a)

The project also proposed using the thinning algorithm for path planning. It will generate a path that is safe for the autonomous robot to take without colliding into any obstacles. After that, the nodes and edges that cannot be traversed by the robot are removed so that only a traversable path remains. The A* algorithm is then applied to determine the shortest path from the starting node to the target node.

The strength of this solution is the ability to combine the use of multiple camera vision at the same time. This increases the area of vision for tracking the robot and is versatile as shown in the paper that it can localize and control robot path as the autonomous robot moves through different corridors. Besides that, the robot successfully avoided the obstacles and reached the target location without the need of extra sensors. This proves that it is possible to use an external vision to control the robot without the need to install environmental sensors on every robot, which is costly.

However, there are also limitations to this solution as well. The most glaring weakness is that only one robot was shown to be localized in this project, but an ideal solution should also consider the localization of multiple robots. This can be solved by simply adding more robots to the environment, then inspecting the performance of the model, and propose and implement additional changes if new problems arise.

2.3: Localization and navigation using QR code for mobile robot in indoor environment

The use of QR codes as artificial landmarks containing label and location information is explored in this project by Zhang et al [10]. The QR codes are placed at regular intervals on the ceiling, so that the robots are able to capture and read the QR codes while moving, and hence calculate their own location by taking advantage of the positional relationship between the QR code and itself.

The camera is first mounted on top of the robot facing the ceiling, and calibrated using the software Halcon to increase the accuracy of the localization result. Each QR code that is generated is unique and encoded to identify its location. The QR codes are placed in a grid pattern with the distance between each other fixed so that the robot can always capture at least one QR code to determine its position. Besides that, the orientation of the captured QR code can also be decoded to determine the orientation of the robot itself. This is done by using the three corners in a QR code, which contain the finder pattern and will be used to determine the orientation of the QR code and, in extension, the orientation of the robot.

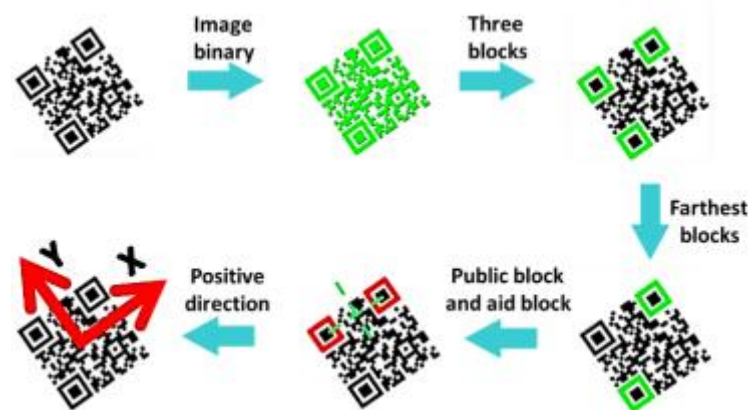


Figure 2.3.1: QR code orientation to determine direction of robot movement

The advantages of this project is that the system is able to recognize QR codes even when the robot moves at high speeds since the recognition algorithm is fast. The localization error is much lower compared to a previous project by Okuyama, Kawasaki & Kroumov. Besides that, QR codes are cheap to manufacture and apply, and they are able to store additional information such as its coordinates, so the use of QR codes as artificial landmarks is a feasible proposal. The weakness of this project is that the QR codes must be visible to the robot, so this will not work in environments where the lighting will change.

Localization Error			
<i>Method used</i>	<i>X-axis[cm]</i>	<i>Y-axis[cm]</i>	<i>Maximum[cm]</i>
Method in [12]	6.29	6.29	32.9
Our Proposed Method	6.02	6.02	13.0

Table 2.3.1: Localization error in project by Okuyama et al. compared to current project [14]

2.4: Mobile robot localization system using multiple ceiling mounted cameras

A more efficient mobile robot localization system is suggested in this paper by installing multiple cameras on the ceiling vertically above the floor area, instead of at an angle on walls where images captured would need to be converted into an air view [11]. This allows a simpler image calibration process than in the previous paper. The camera views are first calibrated to correct the image distortion using a calibration grid template, so that the object tracking is as accurate as possible.

After that, the calibrated image is used to track the robots with the use of a pattern matching algorithm. The algorithm is previously trained with different pattern templates. The pattern templates are placed on top of the robots as visual trackers. Then, the algorithm is applied to captured images to find patterns which have been fed earlier. If a match is found, the pixel coordinates and orientation of the pattern within the image is returned and converted to real world measurements so that the coordinates of the pattern, and in extension the robot, in real world is obtained. Multiple patterns can be identified within the same image since the patterns are unique.

This proposed solution has many strengths, and the main advantage is the ability to localize multiple autonomous robots at the same time. It also increases the area that is tracked as the images from multiple cameras can be combined to obtain the position of every robot in a single coordinate map. Besides that, the proposed solution is less costly compared to systems where the vision sensors have to be mounted on individual robots. This is because the cameras mounted on the ceiling can effectively cover more area while still being able to track multiple robots, hence less cameras are needed and operating costs can be lowered. Not only that, computational costs are also lowered as image processing is simplified by fixing the cameras vertically above the floor area being tracked.

However, this solution also has its limitations as well, but most of the limitations have been solved in the paper being reviewed. Firstly, cameras with wide angle lens will capture images that have some amount of distortion. This is called the fish eye effect and it has been solved by using calibration grids. Another problem is camera perspective error caused by the height of the robot, where the marker is elevated slightly above ground as it is attached to the top of the robot. This will cause the perceived position of marker to be further than the actual

position of the marker. To fix this, trigonometry functions can be used to correct the perceived position, since the height of the camera and the robot is constant.

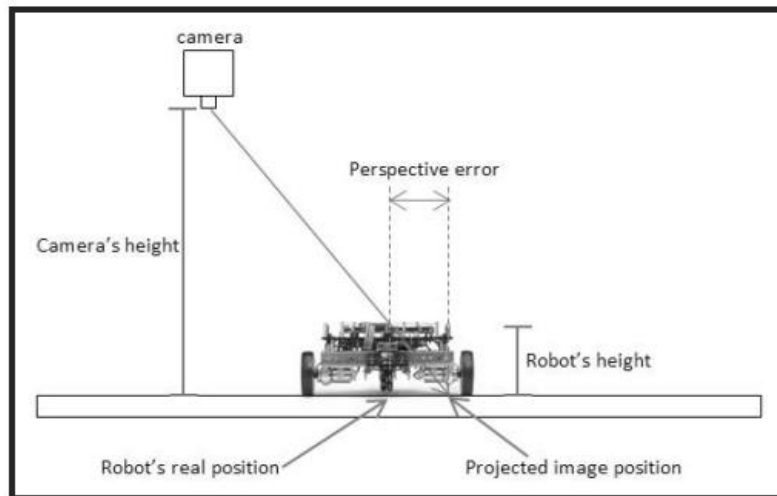


Figure 2.4.1: Pose estimation error due to height of robot

2.5: Robot localization using overhead camera and LEDs

In this project, Johnson, Olson & Boonthum-Denecke suggested using light emitting diodes (LEDs) to replace 2D bar codes as markers for smaller autonomous robots [12]. This is because the barcode stickers are too large to fit on the smaller robots, and reducing the size of the barcodes made it unreadable for the overhead camera, thus making the system unable to process the barcode.

LEDs are suggested because they are also cheap, which allowed the research team to spend less on the equipment. Three LEDs are fitted on the robot in this project, and these LEDs are programmed to blink a unique binary sequence which would indicate its identity. The overhead camera would capture and decode the binary pattern to identify the robot that is detected. This is achieved by first converting the buffered images into 2D arrays of pixel objects. A filter identifies points that have RGB values that are within the range for LEDs, and then remove all pixels other than these from the array. The pixels left behind will be used to find the midpoint of the LEDs.

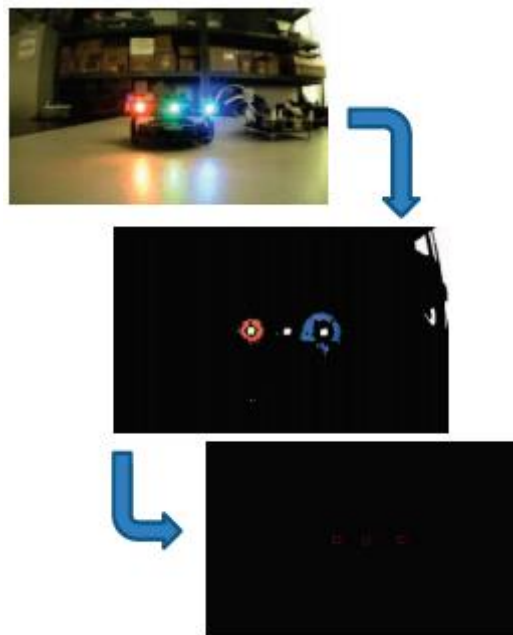


Figure 2.5.1: Image segmentation to identify midpoint of LEDs

The advantage of this solution is that LEDs can be used on robots of all sizes, unlike the barcode which needs to be sufficiently large enough for the overhead camera to successfully capture and decode. Other than that, LEDs are also considerably cheap so it would not be a huge increase in equipment cost when they are used for replacing barcodes which are printed on paper. Nonetheless, the project also has its weaknesses. One of it is the problem of data error when capturing the sequence of blinking lights. This was solved by implemented lexicographical code to the LED blinking pattern for error detecting and error correcting. Another problem is that the system might not work well in an environment where the lighting might fluctuate throughout the day, which limits the implementation of this system to a closed environment.

CHAPTER 3: PROPOSED METHOD/APPROACH

3.1: System Design/Overview

This system will be developed and compiled using PyCharm, with Python as the programming language. This is because Python is so far the best language as it supports an extensive collection of libraries, modules and packages. Besides that, the OpenCV library is utilized to perform tasks related to real-time computer vision such as image processing. The OpenCV library offers functions such as image processing, video capture and analysis, which are the main tasks in this project. Other than that, the ArUco library is also used to perform detection and pose estimation for the unique identification of the AMR.

Toy cars will be used in this project to simulate the AMRs in the warehouse. Unique ArUco markers from the same class will be affixed on top of these toy cars, to enable the program to detect and recognize each unit individually. Lastly, two webcams with a small area of overlapping field of vision will be mounted on tripods and angled 90° downwards facing the floor to simulate the view of cameras mounted on ceilings in the warehouse.

Next, we will then briefly run through the system overview. The program will start by accepting two input video streams for processing, in this case two real time video streams from two webcams. The program will grab the first frames captured, which will be passed to the Stitcher class in `panorama.py` for image stitching. This is done by detecting and extracting features from the frames from the two cameras, which will then be used for matching and lining up the two images. If successfully found more than 4 matching features from the two frames, then the program can use these points to compute the homography matrix. Note that the homography matrix is only computed once at the beginning of the program, and will be cached for the next iterations. This is because there is no need to recalculate the homography matrix for every frame, which would be inefficient, so the program uses the cached matrix from the first frame to stitch the two camera views in the next iterations and returns the stitched frame, which we will name as result.

The result – the stitched frame – will then be used to perform ArUco marker detection. The ArUco module that was imported from the OpenCV library allowed the program to detect the ArUco markers in the input frame based on the specific ArUco dictionary specified by the user. After successfully detecting ArUco markers, the program will draw a bounding box over the borders of the markers that are detected. The program will continuously track the markers once detected. Besides that, the program will also calculate the coordinates of each individual

marker by using the two of the four corners of the marker and finding the centroid. The information, with the addition of the current datetime, can be both displayed and appended into a text file, so that it can be viewed and utilized for further calculations. The program will continue running in loop, processing the video stream frame by frame, until the user terminates it by tapping the escape key “Q”.

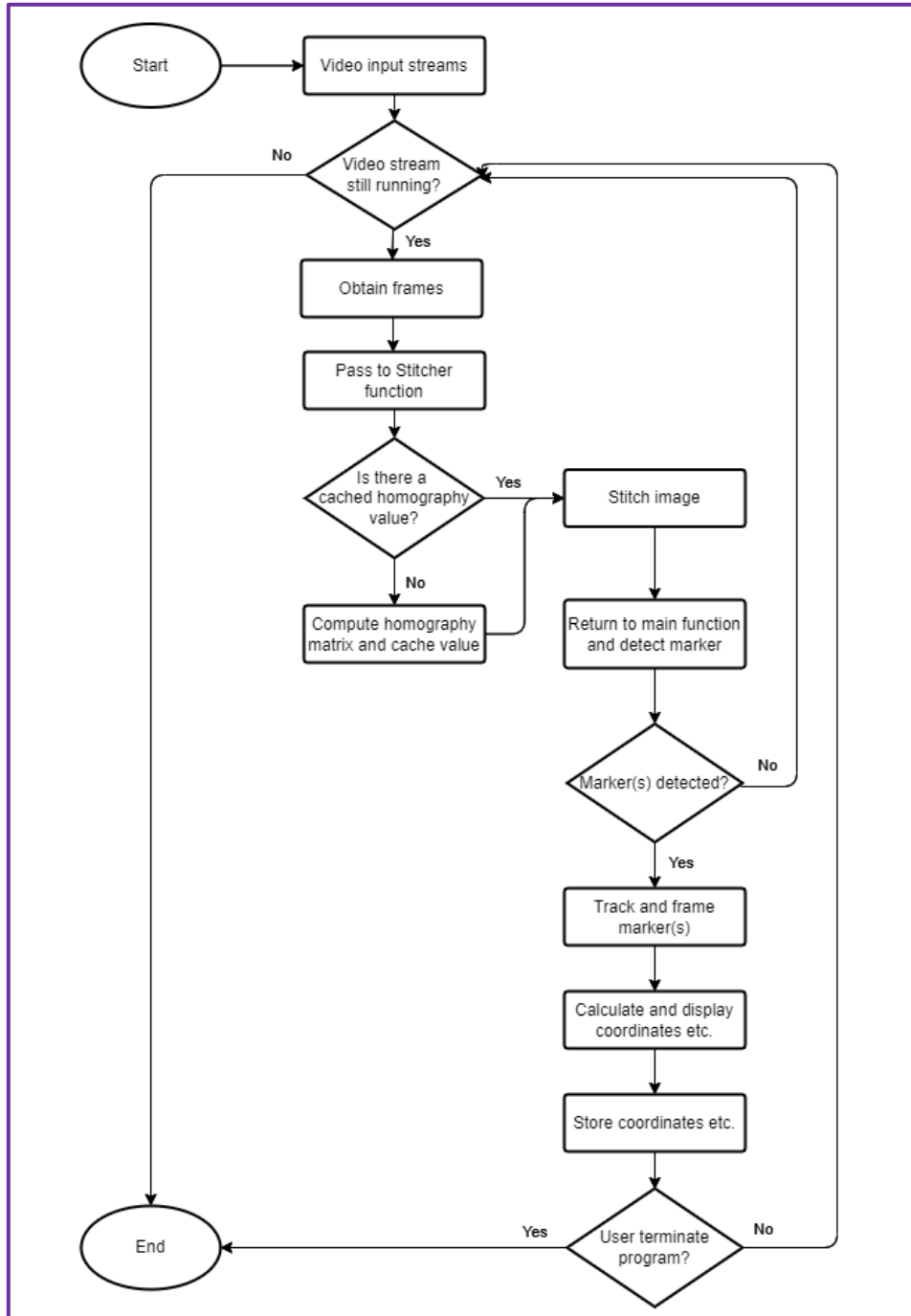


Figure 3.1.1 Flowchart of the system design

3.2: Detailed System Process

3.2.1: Importing Libraries

The libraries that will be used in the program must be imported before the project is able to run. In this program, the OpenCV library is crucial, since it is an open source library and widely used for real-time computer vision tasks. It has many modules that supports the functionality that will be needed in the program.

The first one is the time library, which we use the *.sleep()* function to minutely delay a few executions of the thread – such as only reading the frame for marker detection every 0.01 seconds. Besides that, the datetime library is used to print the current date and time along with marker ID and coordinates. The ArUco library is also important to detect and display the location of the markers within the frame. Lastly, the imutils library is used to resize the input video streams to the same width.

However, the OpenCV library will need to be installed manually by the user, as it is not available by default in PyCharm. It is important to note that it is the contrib version of OpenCV that is required, because it includes extra modules such as the ArUco library. Besides that, the user will need to manually check that they have not installed other versions of OpenCV on their machine before downloading the contrib version, as having both installed might cause conflicts or errors. The following commands are used to install OpenCV contrib:

```
pip uninstall opencv-python
```

```
pip install opencv-contrib-python
```

```
C:\WINDOWS\system32>pip install opencv-contrib-python
Requirement already satisfied: opencv-contrib-python in c:\users\hp\appdata\local\programs\python\python39\lib\site-packages (4.5.5.62)
Requirement already satisfied: numpy>=1.19.3 in c:\users\hp\appdata\local\programs\python\python39\lib\site-packages (from opencv-contrib-python) (1.19.5)
```

Figure 3.2.1.1 Output if opencv-contrib-python is installed successfully

3.2.2: Image stitching

The two camera feeds will have a small section of overlapping view. Image stitching may be done by detecting and extracting features from the frames from the two cameras. Firstly, the images will be converted to grayscale, then the SIFT feature detection function will be used to detect keypoints, which will be stored in a tuple and then used for matching by calling *DescriptorMatcher* class from OpenCV. If more than 4 matching features are successfully found between the two frames, then the program can use these points to compute the homography matrix.

The homography matrix will only be computed once at the beginning of the program, and will be cached so that the program may use the cached matrix to stitch the two camera views in the following iterations. The stitched frame is returned to the main function.

```
# if the cached homography matrix is None, then we need to apply keypoint matching to construct it
if self.cachedH is None:
    # detect keypoints and extract
    (kpsA, featuresA) = self.detectAndDescribe(imageA)
    (kpsB, featuresB) = self.detectAndDescribe(imageB)
    # match features between the two images
    M = self.matchKeypoints(kpsA, kpsB, featuresA, featuresB, ratio, reprojThresh)
    # if the match is None, then there aren't enough matched keypoints to create a panorama
    if M is None:
        return None
    # cache the homography matrix
    self.cachedH = M[1]
# apply a perspective transform to stitch the images together
# using the cached homography matrix
result = cv2.warpPerspective(imageA, self.cachedH, (imageA.shape[1] + imageB.shape[1], imageA.shape[0]))
result[0:imageB.shape[0], 0:imageB.shape[1]] = imageB
# return the stitched image
return result
```

Figure 3.2.2.1 Partial code for computing homography matrix

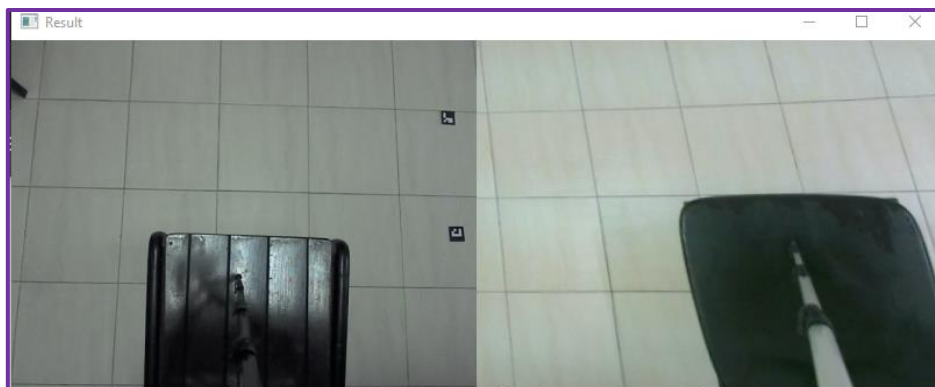


Figure 3.2.2.2 Result of image stitching using the homography matrix computed

3.2.3: Detecting the ArUco Markers

The ArUco markers should have been generated and printed out prior to the experiments. The markers should be from the same dictionary, and should be specified in the program as the `detectMarkers()` function will take the specified dictionary and detect if there any markers from this dictionary in the frame. The detection process consists of two steps: first, the frame is analysed to detect square shapes that might be marker candidates using adaptive thresholding. These shapes are filtered to remove those that are not square, too small, too close to each other etc. Then, the marker bits of each marker will be extracted and analysed to determine if the marker belongs to the specific dictionary.

After that, the function will return three values: `corners`, which is a list containing the (x, y)-coordinates of the detected markers; `ids`, which are the IDs of the detected markers; and `rejected`, which are the marker candidates that were filtered and discarded. Lastly, the `cv2.line()` function will use all the corners to draw the bounding boxes on all the markers inside the frame.

```
(corners, ids, rejected) = cv2.aruco.detectMarkers(result, arucoDict, parameters=arucoParams)
```

Figure 3.2.3.1 Detecting ArUco markers

```
cv2.line(result, topLeft, topRight, (0, 255, 0), 2)  
cv2.line(result, topRight, bottomRight, (0, 255, 0), 2)  
cv2.line(result, bottomRight, bottomLeft, (0, 255, 0), 2)  
cv2.line(result, bottomLeft, topLeft, (0, 255, 0), 2)
```

Figure 3.2.3.2 Drawing bounding boxes



Figure 3.2.3.3 Marker detected and bounded
(*the other two markers are from a different class and so is discarded)

3.2.4: Calculating Coordinates

The coordinates of the four corners are extracted from the list *corners*. The corners are always stored clockwise, starting from the topLeft corner which is marked with a small red square. Hence, by summing the x and y coordinates of the topLeft and bottomRight corners and halving to find the centre, and then converting the values to integers, we can calculate the centroid of the marker and this value is effectively the coordinate of the marker. This same process will be looped through all of the *ids* that have been detected, so all coordinates of all markers will be calculated, shown and stored.

```
cX = int((topLeft[0] + bottomRight[0]) / 2.0)
cY = int((topLeft[1] + bottomRight[1]) / 2.0)
cv2.circle(result, (cX, cY), 4, (0, 0, 255), -1)
```

Figure 3.2.4.1 Calculate the coordinates

CHAPTER 4: DESIGN SPECIFICATION

4.1: Tools to use

4.1.1: Software

The program is coded in Python. This is due to the many existing and open source libraries that can be utilized to ease the development process. The Python IDE that was used to develop this program was PyCharm, as it is easy to import and install required modules using the simple interface provided by PyCharm. The community version of the IDE already offers many features such as powerful debugging UI, smart code editor and navigation, as well as easy refactoring to avoid code rot which can be caused by programming discrepancies.

4.1.2: Hardware

4.1.2.1: Machine

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

Platform	Windows 10, x64-bit operating system
Processor	Intel(R) Core(TM) i7-4510U CPU @ 2.00GHz 2.60 GHz
RAM	8.00 GB
Graphics Card	NVIDIA GeForce 840M

Table 4.1.2.1.1: Machine specifications

4.1.2.2: Webcam

USB webcams were chosen as the input stream provider as the video stream from the cameras can be easily transferred to a computer for developing and testing the program by using the USB cable connected to the webcam. Three webcams were tested for the quality and framerate of the video streams taken. The specifications of each webcam are as shown in the tables below.

Model	Creative labs Senz3D
Resolution	HD 720p (1280 x 720)
Frame Rate	Up to 30 fps
Field of View	74°

Table 4.1.2.2.1: Webcam 1 specifications

Model	AVerMedia Live Streamer CAM 513
Resolution	UHD 4K / 1080p
Frame Rate	Up to 30 fps / Up to 60fps
Field of View	94°

Table 4.1.2.2.2: Webcam 2 specifications

Model	Logitech C615
Resolution	1080p / 720p
Frame Rate	Up to 30 fps / Up to 30fps

Table 4.1.2.2.3: Webcam 3 specifications

4.1.2.3: Toy Cars

Toy cars were used in the experiment. They can be controlled using remote controller. The image shows the toy cars that were used.



Figure 4.1.2.3.1: Toy cars

4.1.2.4: ArUco Markers

The aruco module provides the *Dictionary* class to represent a dictionary of markers. The dictionary can be manually generated, but it is not recommended to do so. The next option is automatic dictionary generation, where the user may specify the desired number of markers as well as the 2D bit size, and the *generateCustomDictionary()* function will take a few seconds to generate the dictionary. However, the easiest way to select a dictionary is still by utilizing the few predefined ArUco dictionaries. For example, in this project, the ArUco dictionary used was DICT_5X5_100, which contains markers with 5x5 bits and a total of 100 markers.

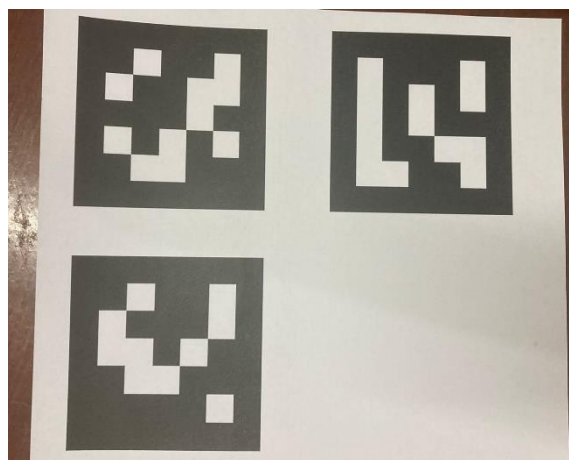


Figure 4.1.2.4.1: DICT_5X5_100 ArUco markers

4.2 Requirements

4.2.1: User requirements:

- Users are able to read the real time coordinates of the robots
- Users are able to clearly see all the robots on the area detected by the program

4.2.2: System performance definition:

- The program must perform the task of localization well in the testbed environment, which will simulate a small section of the warehouse. The program should be able to perform detection and tracking of AMRs on the real time video source. The program should also be able to return the coordinates of the AMR in real time so that the user can read and the information may be utilized for path planning.

CHAPTER 5: IMPLEMENTATION AND TESTING

5.1: Implementation

The experiment was set up in an indoor environment. The cameras were mounted so that they faced the floor vertically, and were calibrated prior to the testing. The specifications of the webcams used are stated in 4.1.2.2. The cameras are set to a height of 1.5m above the ground, and the area of the experiment is around 2m x 1m (length x width).

The webcams are connected to the machine via USB so that the program will receive the input streams live in real time. There will be a small overlapping area between the video input from the two cameras, where two ArUco markers from a different class is placed so that they could be used as keypoints to compute homography matrix to stitch the two video feeds into one. The toy cars are placed on the floor and will be moved around or stay stationary, but only within the area of experiment.

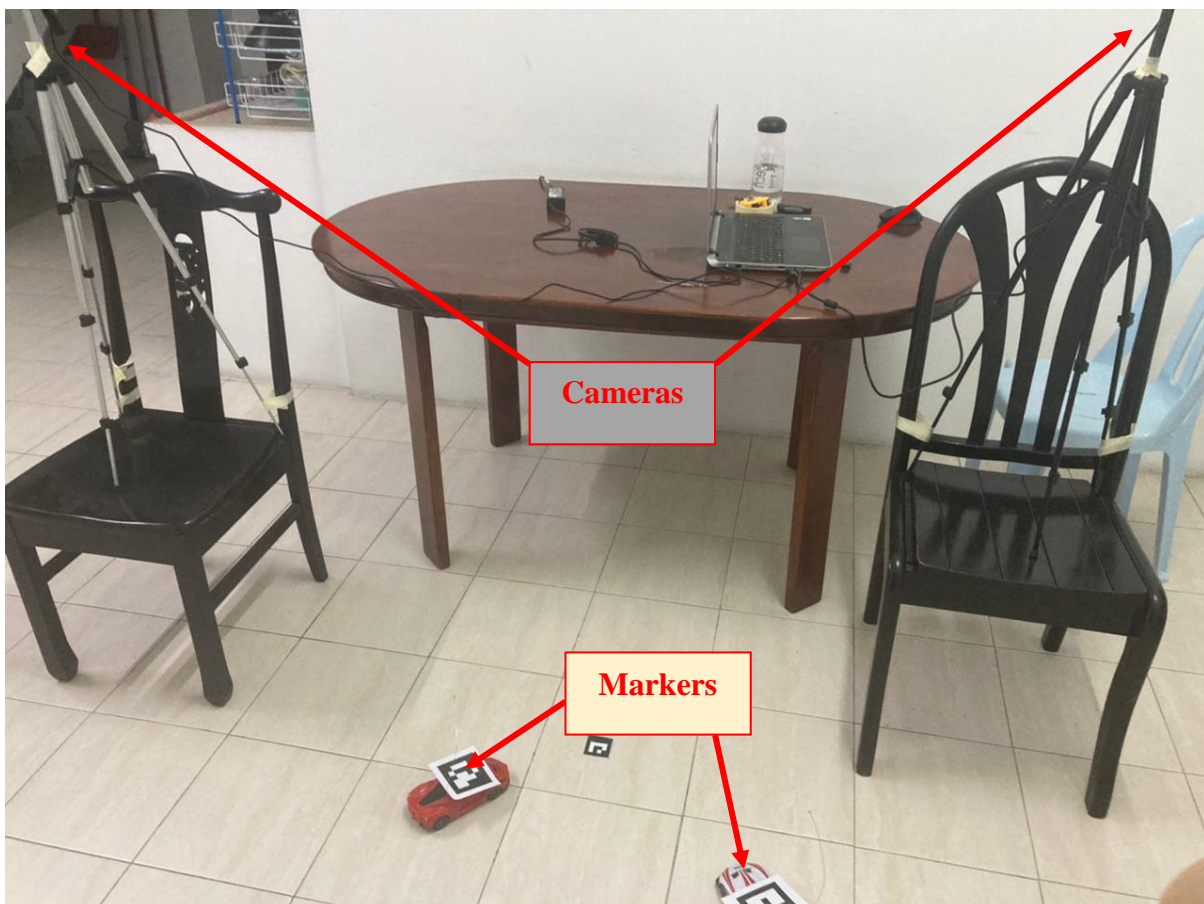


Figure 5.1.1: Setup of the experiment

5.2: Testing

5.2.1: ArUco Marker Dictionary

There are a few predefined ArUco dictionaries, as shown in the image below.

```
ARUCO_DICT = {  
    "DICT_4X4_50": cv2.aruco.DICT_4X4_50,  
    "DICT_4X4_100": cv2.aruco.DICT_4X4_100,  
    "DICT_4X4_250": cv2.aruco.DICT_4X4_250,  
    "DICT_4X4_1000": cv2.aruco.DICT_4X4_1000,  
    "DICT_5X5_50": cv2.aruco.DICT_5X5_50,  
    "DICT_5X5_100": cv2.aruco.DICT_5X5_100,  
    "DICT_5X5_250": cv2.aruco.DICT_5X5_250,  
    "DICT_5X5_1000": cv2.aruco.DICT_5X5_1000,  
    "DICT_6X6_50": cv2.aruco.DICT_6X6_50,  
    "DICT_6X6_100": cv2.aruco.DICT_6X6_100,  
    "DICT_6X6_250": cv2.aruco.DICT_6X6_250,  
    "DICT_6X6_1000": cv2.aruco.DICT_6X6_1000,  
    "DICT_7X7_50": cv2.aruco.DICT_7X7_50,  
    "DICT_7X7_100": cv2.aruco.DICT_7X7_100,  
    "DICT_7X7_250": cv2.aruco.DICT_7X7_250,  
    "DICT_7X7_1000": cv2.aruco.DICT_7X7_1000,  
    "DICT_ARUCO_ORIGINAL": cv2.aruco.DICT_ARUCO_ORIGINAL  
}
```

Figure 5.2.1.1: Predefined ArUco dictionaries

A few dictionaries were tested in this project, which are the DICT_5X5_100, DICT_6X6_100 and DICT_7X7_100 dictionaries. Unsurprisingly, the dictionary that showed the best performance is the DICT_5X5_100, as the smaller the dictionary, the higher the inter-marker distance, which will reduce the inter-marker confusion rate [13]. Theoretically, DICT_4X4_50 will have the best performance, however the decision for the dictionary that will be used in this project is still DICT_5X5_100, since we would like to test the average performance so it is wise to find the middle ground between the quantity of markers available and the performance.

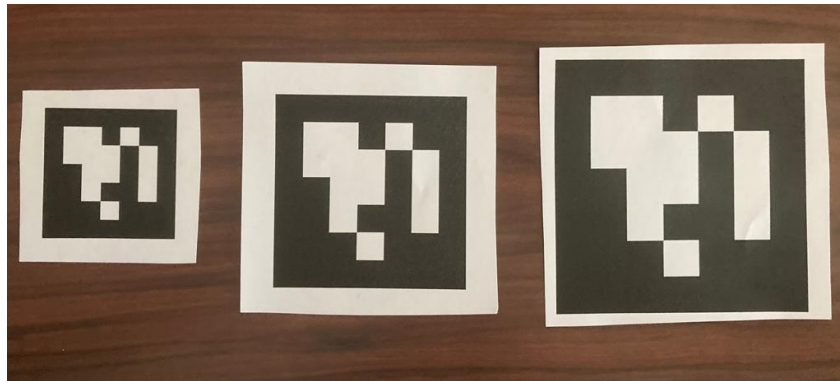


Figure 5.2.1.2: ArUco Dictionaries

(left to right: DICT_4X4, DICT_5X5, DICT_6X6, DICT_7X7)

5.2.2: ArUco Marker Size

The size of the ArUco marker can be adjusted according to the needs of the user. As a rule, the higher the height of the camera above the ground, the size of the marker should also increase to compensate. Hence, the final size of the marker that was used was 7.5cm x 7.5cm.



*Figure 5.2.2.1: Same ArUco marker, different side length
(left to right: 5cm, 7.5cm, 10cm)*

5.2.3: Webcam

Referring to the specifications in 4.1.2.2, sample output images from each camera were taken and compared. After comparison and testing, Creative labs Senz3D and AVerMedia Live Streamer CAM 513 webcams were used in the project, as Logitech C615 webcam produces lower quality, blurred image, and often failed to detect the markers even when the markers were stationary.

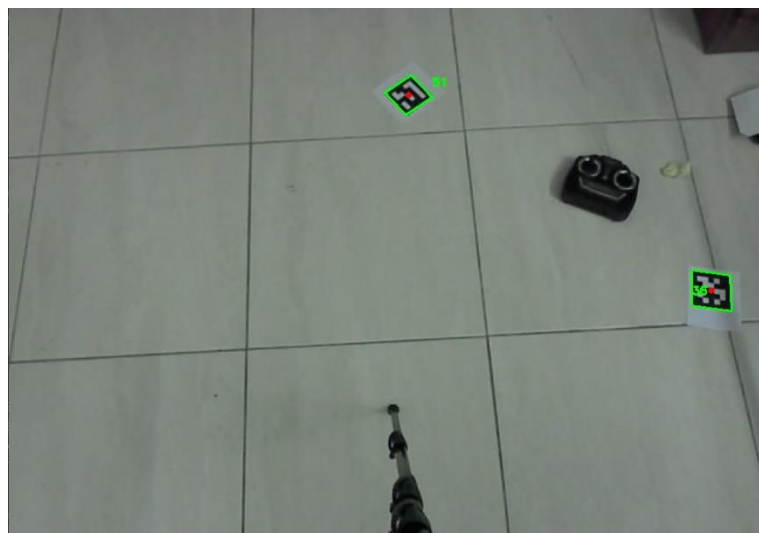


Figure 5.2.3.1: Creative labs Senz3D sample output



Figure 5.2.3.2: AVerMedia Live Streamer CAM 513 sample output



Figure 5.2.3.3: Logitech C615 sample output

5.2.4: Height of the Cameras

Several heights of the webcams from the ground has been tried and tested during the phase of project. First, the webcams were just mounted on top of camera stands, and placed on the ground. The height during this setup is around 1m tall. However, the area coverage is quite small. Hence the webcams were placed on top of chairs to increase height. Now the height is 1.5m tall and the area captured is increased. The size of the markers had to be increased as well to 7.5cm so that the program is able to detect easily.

5.3: Verification Plan

The verification plan outlines the methods used to test the system.

Description	Test Method
ArUco marker cannot be detected	<ol style="list-style-type: none">i. Change to a smaller ArUco dictionaryii. Increase the size of ArUco markersiii. Decrease the distance between camera and markeriv. Check resolution of camerav. Decrease the speed of toy car moving
Quality of images not clear	<ol style="list-style-type: none">i. Change to another camera with different specificationsii. Change environment brightness and lighting
Homography could not be computed	<ol style="list-style-type: none">i. Check if there are enough keypointsii. Add keypoints

Table 5.3.1: Verification plan

5.4: Experiment

Figure 5.1.1 shows the setup of the experiment, where two webcams are placed side by side and will each capture a section of the floor, with a small area between them overlapping, which is where we will place some markers as keypoints. From the figure, the webcam on the top right is actually responsible for capturing the left area, while the webcam on the top left is capturing the right area. This is because the order of the video inputs into the *Stitcher* class should be from left to right, viewing from the perspective of standing behind the webcams.

In figure 5.4.1, the top left and top right images shows the original video input from the left and right area respectively. The window titled “Result” shows the result of the stitched image, along with the detected markers, the id number of the markers, and their centroids. It may be observed that although there are four markers in the figure, only two markers are detected and bounded; this is because the other two markers are from a different dictionary than the one specified in the program, and in this case they were used as keypoints to calculate homography, so they will not be identified and displayed by the program.

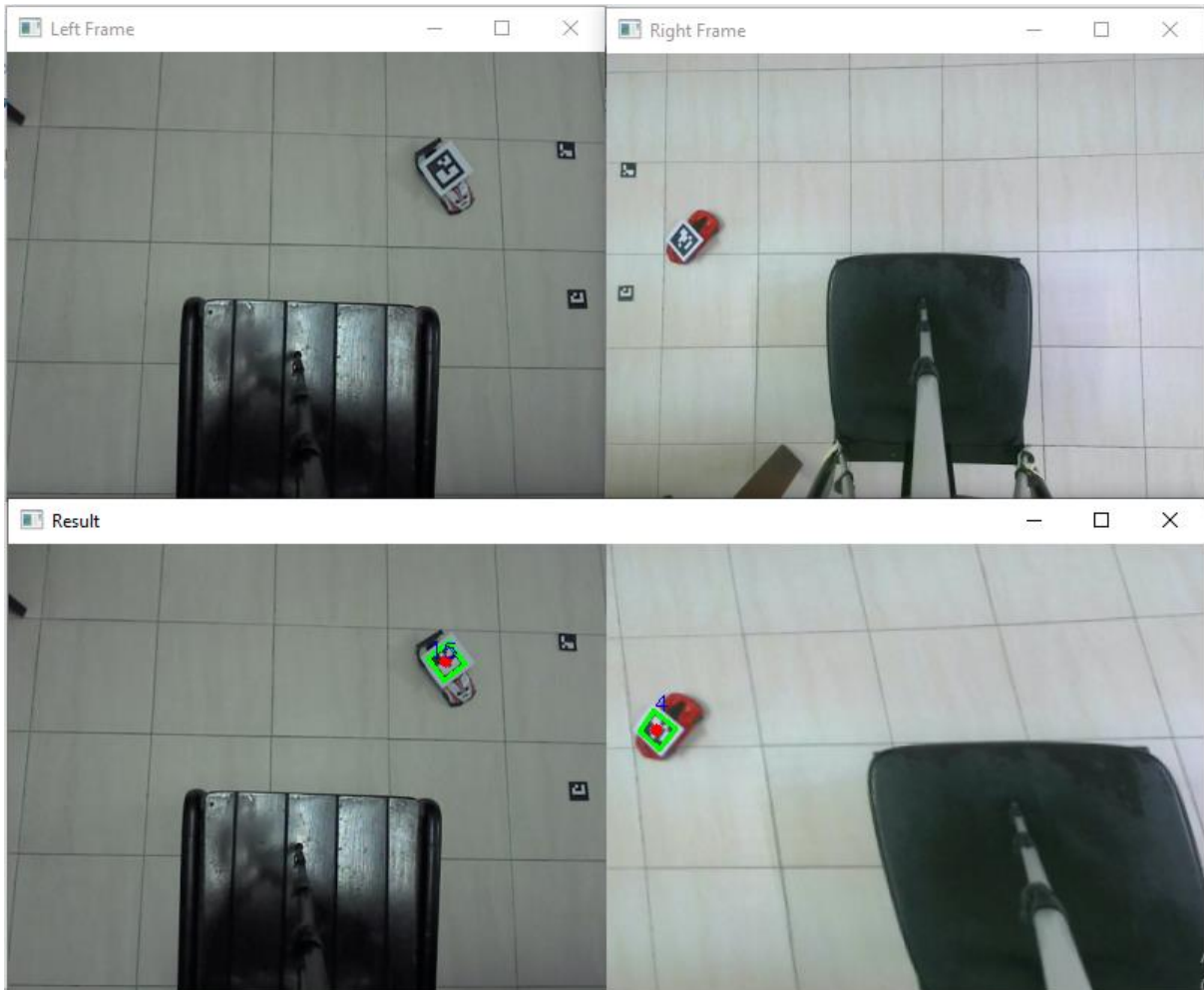


Figure 5.4.1: Screenshot of video output

On the console terminal of the PyCharm IDE at the same time, the information will also be printed out. The order of the information shown is markerID, (x, y)-coordinates, date and time. The user can use the information printed out on the console paired with the video output to keep track of all the markers detected inside the area of vision. Lastly, at the same time the information is printed on the console, the program will also continuously append the information into a txt file to store.

```
4 433 124 Thursday 21 April 2022 08:22:17PM
15 292 78 Thursday 21 April 2022 08:22:17PM
4 433 124 Thursday 21 April 2022 08:22:18PM
15 292 78 Thursday 21 April 2022 08:22:18PM
```

Figure 5.4.2: Screenshot of console output

5.5: Analysis

The demo was carried out using the conditions where the program has performed the best in previous observations. During the first phase of the program, which is computing the homography matrix to stitch the image, the keypoints must be placed so that both cameras can pick them up, or else there would be no points of similarity. Other than that, it is noticed that the cameras needed to be placed very specifically so that the floor area captured line up as much as possible. This is because otherwise, when stitching the image using the video input from the left stream as a base, the right part of the image will become terribly skewed or distorted.

Besides that, the lighting of the environment also affects the performance of the program. For one of the webcams, while testing at night, the artificial lighting of fluorescent tube light will cause flickering in the captured video due to frame rate and shutter speed, degrading the performance and sometimes even resulting in the homography matrix unable to be computed. In addition, during low light conditions (for example not switching on fluorescent lights at late evening), the video streams captured are also of bad quality, and the program often struggles to detect the markers in the hazy, grainy frames. For these two problems, further testing should be carried out using cameras with better specifications or with night vision.

The marker size used in the demo is 7.5cm x 7.5cm, when the heights of the cameras are approximately 1.5m. The tests show that at this marker size, the program has a steady performance and can detect the markers, at both stationary and while moving at slow speeds of around 0.5ms^{-1} . One exception is the few frames during the marker passes between the “boundary” of the stitched frame in any direction – however, the program is able to recover and track the marker again once it passes through the boundary.

It is also a shame that we had not been able to mount the cameras higher, and hence the field of vision captured was smaller, and there was also some occlusion in the form of the chairs elevating the height of the cameras.

CHAPTER 5: CONCLUSION

In the past decade, the field of robotics have advanced and evolved so much thanks to the widespread digitization of information, increased computing power, data storage, highly efficient electronics and even more powerful integrated circuits. The modern robot is a culmination of many technological fields, both hardware and software, and these robots are capable of performing various tasks such as locomotion, object recognition, mapping, sorting operations – the possibilities are endless.

There are several types of autonomous mobile robots that we commonly see in warehouses nowadays. One of them is a robotic arm; they are able to perform pick and put operations, replacing manual labour for repetitive tasks in a fixed workstation. There's also the AGV; they require guidance systems to move around, but they can help to move around packages. AMRs serve a similar purpose, with the exception that it uses sensors to detect its surroundings and has built in path planning system to move around [15]. AMRs have many advantages over AGVs, such as more flexibility for path movement, easy to remap routes, easy to install as there is no need for the extra guidance infrastructure... The trouble is, they are more expensive than AGVs due to the hardware requirements for accurate sensors and to run the path planning software efficiently.

Hence, the motivation of this project is to lower those costs so that AMRs can be used widely. The proposed solution is to lower the hardware cost by removing the need for built in sensors, which are expensive. Instead, this project proposed that all the AMRs are localized using overhead cameras which cover the entire field, thus eliminating the need for self-localization.

In conclusion, the project went well and has successfully met the project objectives that were set at the beginning of the proposal. The program is able to take in real time video streams from multiple cameras, then stitch into one single frame and perform marker detection upon the frame, and lastly localize the marker by calculating the centroid. Unfortunately, there were also limitations as mentioned in the analysis section in 5.5. In future works, cameras with higher specifications and night vision should be tested out to review the performance of the program at other conditions.

BIBLIOGRAPHY

- [1] “Revenue for Amazon (AMZN)”. *CompaniesMarketCap.com*. [Online], Available: <https://companiesmarketcap.com/amazon/revenue/> [Accessed 17 April 2022]
- [2] Bowles, R., “Organized chaos: behind the scenes of amazon’s inventory management system”, *Logiwa*, 2021. [Online], Available: <https://www.logiwa.com/blog/amazon-inventory-management-system> [Accessed 4 April 2021]
- [3] Simon, M., 2019. “Inside the amazon warehouse where humans and machines become one”, *Wired*. [Online], Available: <https://www.wired.com/story/amazon-warehouse-robots> [Accessed 6 April 2021]
- [4] Walker, J., n.d. “What are autonomous robots? 8 applications for today’s AMRs”, *Waypoint Robotics Insight*. [Online], Available: <https://waypointrobotics.com/blog/what-autonomous-robots/> [Accessed 4 April 2021]
- [5] Panigrahi, P.K. & Bisoy, S.K., 2021. “Localization strategies for autonomous mobile robots: A review”. *2021 Journal of King Saud University - Computer and Information Sciences*, ISSN 1319-1578, doi: <https://doi.org/10.1016/j.jksuci.2021.02.015>
- [6] Rosebrock, A., 2018. “Simple object tracking with OpenCV”, *pyimagesearch*, 2021. [Online], Available: <https://www.pyimagesearch.com/2018/07/23/simple-object-tracking-with-opencv/> [Accessed 4 April 2021]
- [7] Cai, Q and Aggarwal, J.K., 1996. “Tracking human motion using multiple cameras”, *Proceedings of 13th International Conference on Pattern Recognition, Vienna, Austria*, pp. 68-72 vol.3, doi: 10.1109/ICPR.1996.546796
- [8] Li, I.-H. et al., 2014. “Mobile robot self-localization system using single webcam distance measurement technology in indoor environments”. *Sensors*, 14(2), pp. 2089–2109, doi:10.3390/s140202089
- [9] Shim, J. & Cho, Y., 2016. “A mobile robot localization via indoor fixed remote surveillance cameras”. *Sensors*, 16(2), p. 195, doi:10.3390/s16020195
- [10] Zhang, H.J., Zhang, C.N., Yang, W. & Chen, C., 2015. “Localization and navigation using QR code for mobile robot in indoor environment”. *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 2501-2506, doi: 10.1109/ROBIO.2015.7419715

- [11] Visvanathan, R. et al., 2015. “Mobile robot localization system using multiple ceiling mounted cameras”, *2015 IEEE SENSORS*, Busan, Korea (South), pp. 1-4, doi: 10.1109/ICSENS.2015.7370454
- [12] Johnson, E., Olson, E. & Boonthum, Chutima., 2012. “Robot localization using overhead camera and LEDs”. *Proceedings of the 25th International Florida Artificial Intelligence Research Society Conference, FLAIRS-25*. pp. 524-526
- [13] Garrido-Jurado, S.; Muñoz-Salinas, R.; Madrid-Cuevas, F.J.; Marín-Jiménez, M.J. (2014). “Automatic generation and detection of highly reliable fiducial markers under occlusion”. *Pattern Recognition*, 47(6), 2280–2292. doi:10.1016/j.patcog.2014.01.005
- [14] K. Okuyama, T. Kawasaki and V. Kroumov. “Localization and position correction for mobile robot using artificial visual landmarks”, *Advanced Mechatronic Systems (ICAMechS), 2011 International Conference on. IEEE*, 2011, pp. 414-418
- [15] CrossCo. n.d. “The Difference Between AGVs And Mobile Robots”, *CrossCo*. [Online], Available: <https://www.crossco.com/resources/articles/the-difference-between-agvs-and-mobile-robots> [Accessed 17 April 2022]

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	Study week no.: 3
Student Name & ID: Lai Hong Pei 18ACB01409	
Supervisor: Ts. Dr. Liew Soung Yue	
Project Title: Autonomous Mobile Robot Tracking Across Multiple Camera Vision	

1. WORK DONE

Research on AprilTags

2. WORK TO BE DONE

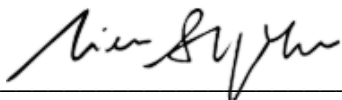
Reasearch on possible alternatives for fiducial markers other than AprilTag.

3. PROBLEMS ENCOUNTERED

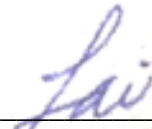
Having trouble installing and importing the AprilTag library in Visual Studio Code.

4. SELF EVALUATION OF THE PROGRESS

Need to figure out fast which alternative marker can replace AprilTags, which was the original fiducial marker discussed to be used early on even during the proposal.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	Study week no.: 4
Student Name & ID: Lai Hong Pei 18ACB01409	
Supervisor: Ts. Dr. Liew Soung Yue	
Project Title: Autonomous Mobile Robot Tracking Across Multiple Camera Vision	

1. WORK DONE

Research on possible alternatives for fiducial markers other than AprilTag. Found ArUco markers which can be successfully installed and imported in PyCharm.

2. WORK TO BE DONE

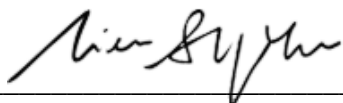
Start learning how to use ArUco markers.

3. PROBLEMS ENCOUNTERED

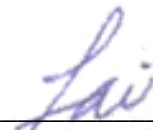
After many unsuccessful attempts at importing opencv-contrib modules again in Visual Studio Code, made the switch to using PyCharm as an IDE and finally successfully access the ArUco library within the opencv-contrib modules.

4. SELF EVALUATION OF THE PROGRESS

Need to thank Mr. Ch'ng for suggesting to switch to using PyCharm as IDE. Easier to import and install necessary packages.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	Study week no.: 5
Student Name & ID: Lai Hong Pei 18ACB01409	
Supervisor: Ts. Dr. Liew Soung Yue	
Project Title: Autonomous Mobile Robot Tracking Across Multiple Camera Vision	

1. WORK DONE

Successfully learned to use the basic functions within the ArUco library.

2. WORK TO BE DONE

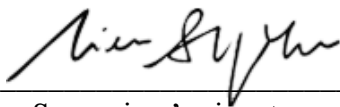
Modify code to add in coordinate calculation and output.

3. PROBLEMS ENCOUNTERED

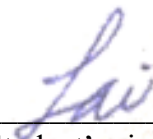
Found that marker cannot be detected if toy car moves too fast. Added friction to the bottom of toy car to decrease speed.

4. SELF EVALUATION OF THE PROGRESS

Not spending as much time as should have, busy with another assignment.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	Study week no.: 6
Student Name & ID: Lai Hong Pei 18ACB01409	
Supervisor: Ts. Dr. Liew Soung Yue	
Project Title: Autonomous Mobile Robot Tracking Across Multiple Camera Vision	

1. WORK DONE

Modified code so that the program can both output and save the coordinates of the marker.

2. WORK TO BE DONE

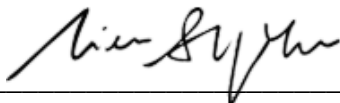
Start research on image stitching.

3. PROBLEMS ENCOUNTERED

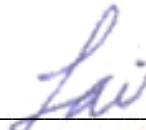
None.

4. SELF EVALUATION OF THE PROGRESS

Still busy week working on other assignments and studying for tests so did not spend much time on this project.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	Study week no.: 8
Student Name & ID: Lai Hong Pei 18ACB01409	
Supervisor: Ts. Dr. Liew Soung Yue	
Project Title: Autonomous Mobile Robot Tracking Across Multiple Camera Vision	

1. WORK DONE

Research on image stitching.

2. WORK TO BE DONE

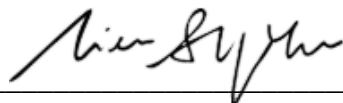
Test out the result of image stitching on ArUco marker detection.

3. PROBLEMS ENCOUNTERED

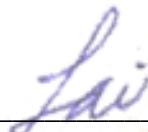
Setup of the cameras need to be very specific. Or the image stitching result frames will have high distortion. Best method so far is try to line up the tiles on the floor captured by first and second webcam.

4. SELF EVALUATION OF THE PROGRESS

Should be fair progress.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	Study week no.:10
Student Name & ID: Lai Hong Pei 18ACB01409	
Supervisor: Ts. Dr. Liew Soung Yue	
Project Title: Autonomous Mobile Robot Tracking Across Multiple Camera Vision	

1. WORK DONE

Can now perform ArUco marker detection within the stitched image. Testing different environment lightings and setups on the performance of detection.

2. WORK TO BE DONE

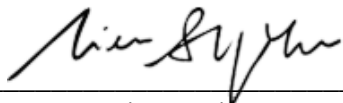
Start writing report.

3. PROBLEMS ENCOUNTERED

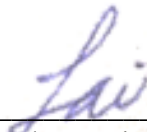
ArUco markers cannot be detected when passing through the border of stitched frames. ArUco marker size too small to be detected confidently when camera height increased; solution was to increase marker size.

4. SELF EVALUATION OF THE PROGRESS

Need to think of more situations to test the program and also record findings.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	Study week no.: 12
Student Name & ID: Lai Hong Pei 18ACB01409	
Supervisor: Ts. Dr. Liew Soung Yue	
Project Title: Autonomous Mobile Robot Tracking Across Multiple Camera Vision	

1. WORK DONE

Still testing different variables for the setup such as speed of car, cameras with different specifications, etc.

2. WORK TO BE DONE

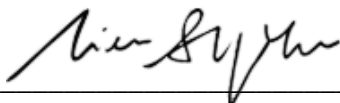
Finish writing report using information collected.

3. PROBLEMS ENCOUNTERED

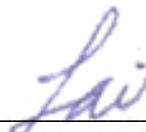
Toy car speed hard to control. Occlusion by chair in the latest setup. But nowhere else safe enough to fix the cameras to.

4. SELF EVALUATION OF THE PROGRESS

Need to hurry up. Spent half of the week completing another assignment.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	Study week no.: 13
Student Name & ID: Lai Hong Pei 18ACB01409	
Supervisor: Ts. Dr. Liew Soung Yue	
Project Title: Autonomous Mobile Robot Tracking Across Multiple Camera Vision	

1. WORK DONE

Progress in writing report.

2. WORK TO BE DONE

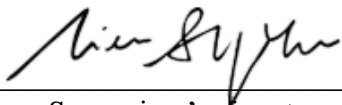
Finish writing report

3. PROBLEMS ENCOUNTERED

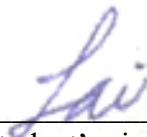
None. Just writing the report using information already collected

4. SELF EVALUATION OF THE PROGRESS

Cutting a bit tight on the time but should be able to finish before the deadline.



Supervisor's signature



Student's signature

POSTER

AUTONOMOUS MOBILE ROBOT TRACKING ACROSS MULTIPLE CAMERA VISION

By

LAI HONG PEI

OBJECTIVES:

- Reduce the cost of implementing Autonomous Sorting Systems.
- Optimize the throughput of autonomous robots by deploying an efficient detection and tracking system.
- Detect and track autonomous robots in real time and show coordinates

Bachelor of Information Technology (Hons) Computer Science
Faculty of Information and Communication Technology (Kampar Campus)

PLAGIRAIISM CHECK RESULT

AUTONOMOUS MOBILE ROBOT TRACKING ACROSS MULTIPLE CAMERA VISION			
ORIGINALITY REPORT			
8%	7%	3%	1%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS
PRIMARY SOURCES			
1	eprints.utar.edu.my Internet Source	3%	
2	www.mdpi.com Internet Source	1%	
3	docs.opencv.org Internet Source	<1%	
4	Li, I-Hsum, Ming-Chang Chen, Wei-Yen Wang, Shun-Feng Su, and To-Wen Lai. "Mobile Robot Self-Localization System Using Single Webcam Distance Measurement Technology in Indoor Environments", Sensors, 2014. Publication	<1%	
5	Shim, Jae, and Young Cho. "A Mobile Robot Localization via Indoor Fixed Remote Surveillance Cameras †", Sensors, 2016. Publication	<1%	
6	community.element14.com Internet Source	<1%	
7	www.hindawi.com		

Internet Source	<1 %	15 Submitted to Swinburne University of Technology Student Paper	<1 %
8 link.springer.com Internet Source	<1 %	16 Submitted to University of Sheffield Student Paper	<1 %
9 utpedia.utp.edu.my Internet Source	<1 %	17 hdl.handle.net Internet Source	<1 %
10 Aman Ahmed, Prateek Bansal, Atiya Khan, Neha Purohit. "Crowd Detection and Analysis for Surveillance Videos using Deep Learning", 2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC), 2021 Publication	<1 %	18 www.wifi-highpower.co.uk Internet Source	<1 %
11 docksci.com Internet Source	<1 %	19 doku.pub Internet Source	<1 %
12 en.m.wikipedia.org Internet Source	<1 %	20 vfrdyky.blogspot.com Internet Source	<1 %
13 Huijuan Zhang, Chengning Zhang, Wei Yang, Chin-Yin Chen. "Localization and navigation using QR code for mobile robot in indoor environment", 2015 IEEE International Conference on Robotics and Biomimetics (ROBIO), 2015 Publication	<1 %	21 "Universal Access in Human-Computer Interaction, Ambient Interaction", Springer Nature, 2007 Publication	<1 %
14 repository.tudelft.nl Internet Source	<1 %	22 A.Y. Hassan. "A wide band QAM receiver based on DFE to reject ISI in wireless time varying fading channel", AEU - International Journal of Electronics and Communications, 2015 Publication	<1 %
		23 Advances in Intelligent Systems and Computing, 2015. Publication	<1 %
		24 Mulas, Marcello, Nicolai Waniek, and Jörg Conrad. "Hebbian Plasticity Realigns Grid Cell Activity with External Sensory Cues in Continuous Attractor Models", Frontiers in Computational Neuroscience, 2016. Publication	<1 %
		25 academic.oup.com Internet Source	<1 %
		26 pure.rug.nl Internet Source	<1 %
		27 www.frontiersin.org Internet Source	<1 %
		28 www.researchgate.net Internet Source	<1 %
		29 www.scitepress.org Internet Source	<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.: 1 of 1



FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Full Name(s) of Candidate(s)	LAI HONG PEI
ID Number(s)	18ACB01409
Programme / Course	BACHELOR OF COMPUTER SCIENCE (HONOURS)
Title of Final Year Project	AUTONOMOUS MOBILE ROBOT TRACKING ACROSS MULTIPLE CAMERA VISION

Similarity	Supervisor's Comments (Compulsory if parameters of originality exceeds the limits approved by UTAR)
Overall similarity index: <u> 8 </u> % Similarity by source Internet Sources: <u> 7 </u> % Publications: <u> 3 </u> % Student Papers: <u> 1 </u> %	
Number of individual sources listed of more than 3% similarity: <u> 0 </u>	
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 <i>Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.</i>	

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: Ts. Dr. Liew Song Yue

Name: _____

Date: 22/4/2022

Date: _____



UNIVERSITI TUNKU ABDUL RAHMAN

FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY (KAMPAR CAMPUS)

CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	18ACB01409
Student Name	Lai Hong Pei
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.
N/A	Front Plastic Cover (for hardcopy)
√	Title Page
√	Signed Report Status Declaration Form
√	Signed FYP Thesis Submission Form
√	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
N/A	Appendices (if applicable)
√	Weekly Log
√	Poster
√	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)
√	I agree 5 marks will be deducted due to incorrect format, declare wrongly the ticked of these items, and/or any dispute happening for these items in this report.

*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: 21/04/2022