

**THE INVESTIGATION OF CIRCULAR PATH GENERATION AND HAND
MOTION TRACKING PROBLEM USING 5DOF VISION BASED ROBOT**

FARAH AMIRAH BINTI RASID

**A report submitted in partial fulfillment of the requirements for the degree
of Bachelor of Mechatronics Engineering with Honours**

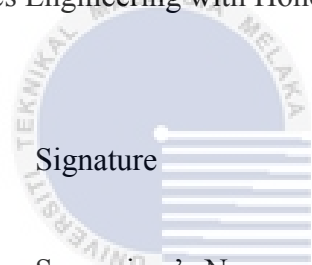


Faculty of Electrical Engineering

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

2017

“ I hereby declare that I have read through this report entitle “The Investigation of Circular Path Generation and Hand Motion Tracking Problem Using Vision Based 5DOF Robot” and found that it has comply the partial fulfillment for awarding the degree of Bachelor of Mechatronics Engineering with Honours.”



Signature

UTeM

Supervisor's Name

PM Dr. Muhammad Fahmi Bin Miskon

اونيورسيتي تيكنيكل مليسيا ملاك

Date

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

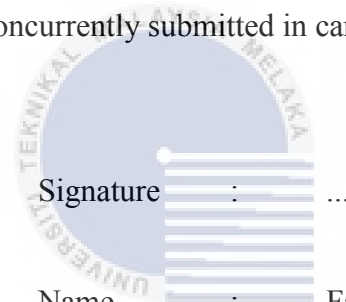

I declare that this report entitle “The Investigation of Circular Path Generation and Hand Motion Tracking Problem Using Vision Based 5DOF Robot” is the result of my own research except as cited in the references. The report has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.

Signature : _____

Name : Farah Amirah Binti Rasid

Date : _____

UNIVERSITI TEKNIKAL MALAYSIA MELAKA



اونيورسيتي تيكنيكل مليسيا ملاك



ACKNOWLEDGEMENT

Grateful to Almighty Allah for His blessings I successfully complete my final year project to complete my study for Bachelor of Mechatronics Engineering with Honours. I have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals. I would like to extend my sincere thanks to all of them.

A special gratitude I give to my final year project supervisor, Associate Professor Dr Muhammad Fahmi bin Miskon, whose contribution in suggestions and encouragement, guide me to make my project possible. I really thank him for providing me the conceptual and theoretical clarity in that enabled me to develop an understanding of this project thoroughly.

Furthermore, I would like to appreciate my parents for giving me support, love and encourage me to finish this project successfully. Special thanks go to BEKM members, who help me to assemble the parts and gave suggestion about my project.

Lastly, to University Teknikal Malaysia Melaka (UTeM), I am grateful that this institution really wanted to see its students to be skilful and knowledgeable.

ABSTRACT

Robotic rehabilitation has widely being used especially for upper limb impairment due to neurological disorder or accident. Research has shown that robotic training can also provide repetitive movement without therapist assistant. This project address two main problem in robotic rehabilitation applications which are generating a circular path in Cartesian space and the problem in identifying the positions (x,y,z) of robot's arm end effector when generating a circular motion that can vary depending on the vision based feedback of patient hand. The aim of this project is to analyse the problem of generating circular path and tracking method using a vision based robot. Secondly to develop circular path algorithm and vision based circular tracking algorithm to match with the patient hand motion and to validate the smoothness of circular path and the accuracy of a vision based robot motion when tracking and following the patient hand motion using statistical method. The idea to generate circular path is by using point to point method in Cartesian space. To get the circular motion, the trigonometry formula combined with formula of circumferences was used to generate the circle. MATLAB is used for the circular generation while the simulation for circular path is done in VREP by using LUA. For the visual part, blob detection, centroid detection (x,y) is used with the imposing of inverse kinematic to track the image. The OpenCV library was used to do blob detection. The robot simulation is done using VREP and the algorithm is developed using LUA. In generating a circular path, the smoothness of circle is important. To measure the smoothness of circle path, the number of point to point is increased. For the tracking part, the accuracy of robot is measured using the statistical method. The youBot have high accuracy in tracking part which is 97% with absolute error 0.03. In conclusion, a vision based robot is reliable for circular hand tracking problem for rehabilitation purposes. A future research can be done to make a precise robot motion by developing a new control method for the robot and add features that enable the robot to save the results of motion for therapist to follow up the patient hand condition.

ABSTRAK

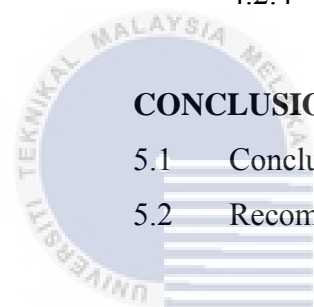
Penggunaan robot semakin luas dipergunakan untuk terapi pemulihan anggota badan bahagian atas yang rosak atau bermasalah disebabkan masalah saraf ataupun kemalangan. Kajian terbaru menunjukkan bahawa latihan menggunakan robot juga boleh memberi pergerakan secara berulang tanpa bantuan ahli terapi. Projek ini menggariskan dua masalah utama dalam aplikasi penggunaan robot untuk latihan pemulihan iaitu dalam menjana gerakan bulatan dalam ruang Cartesian dan masalah untuk mengenalpasti kedudukan (x,y,z) robot apabila mengesan pergerakan tangan pesakit yang diterima melalui visual. Tujuan projek ini ialah untuk menganalisis masalah menjana gerakan bulatan dan mengesan pergerakan tangan menggunakan robot yang mempunyai visual. Yang kedua ialah, membina algoritma gerakan bulatan dan algoritma untuk mengesan pergerakan tangan menggunakan robot visual serta mengesahkan kelancaran bentuk bulatan dan ketepatan robot mengesan gerakan tangan pesakit menggunakan kaedah statistic. Idea untuk menjana gerakan bulatan yang tepat ialah dengan menggunakan kaedah titik ke titik. Gerakan bulat, dihasilkan menggunakan persamaan bulatan. dalam MATLAB Simulasi dijalankan menggunakan VREP dan LUA. Untuk bahagian visual, teknik pengesanan tompok, pengesanan titik tengah (x,y) digunakan dengan mengenakan kinematic songsang untuk mengesan tangan. OpenCV digunakan untuk melakukan pengesanan tompok. Simulasi robot dilakukan dengan menggunakan VREP dan algoritma dibangunkan menggunakan LUA. Dalam menjana gerakan bulatan, kelancaran bulatan adalah bertambah apabila, nombor titik ke titik bertambah. Untuk bahagian pengesanan, ketepatan robot diukur menggunakan kaedah statistik. KUKA youBot mempunyai ketepatan yang tinggi di bahagian pengesanan iaitu 97% dengan ralat mutlak 0.03. Kesimpulannya, sebuah robot yang mempunyai visual boleh dipercayai untuk menyelesaikan masalah pengesanan tangan untuk tujuan rehabilitasi.

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE
	ACKNOWLEDGEMENT	v
	ABSTRACT	vi
	TABLE OF CONTENTS	viii
	LIST OF TABLES	xi
	LIST OF FIGURES	xii
	LIST OF SYMBOLS	xiv
	LIST OF APPENDICES	xv
1	INTRODUCTION	1
	1.1 Motivation	1
	1.2 Problem Statement	2
	1.3 Objectives	3
	1.4 Scope of the project	3
2	LITERATURE REVIEW	5
	2.1 Robotic Rehabilitation	5
	2.2 Generation of Circular Path	6
	2.2.1 Trajectory for Generating Circular Path	7
	2.2.2 Problems of Generating Circular Path in Cartesian Space	8
	2.2.3 Available Solution for Generation of Circular Path	11
	2.2.4 Summary for Available Solution in Generation of Circular Path	11
	2.3 Visual based robot for rehabilitation applications	12

	2.3.1	Robot Vision for hand tracking using Marker	15
	2.4	Method to Generate Circular Path and Tracking Hand Motion	18
3		RESEARCH METHODOLOGY	19
	3.1	Theoretical Concept	19
	3.1.1	Flowchart for Circular Path Generation and Tracking Hand Motion	22
	3.2	Generating circular shape motion	25
	3.3	Object Tracking Method	26
	3.4	Calculating Vision Frame Size	27
	3.5	Angle of View from the Camera	29
	3.6	Positioning the End-Effector to Make the Target in the Centre of Vision Frame	32
	3.7	Objective of simulation	34
	3.8	Simulation Equipment	35
	3.9	Simulation Setup	35
	3.9.1	Workspace of the Circular Generation and Tracking Hand Motion	37
	3.9.2	Pseudo code of Point to Point Circular Path Generation for KUKA youBot	38
	3.9.3	Pseudo code for Path	39
	3.9.4	Pseudo code of Vision Sensor	39
	3.9.5	Pseudo Code of Circular Hand Tracking for KUKA Youbot	40
	3.10	Procedure of Simulation in VREP	40
	3.10.1	Problems and Issues in Simulation	41
	3.11	Method of Analysis	42
	3.12	Consideration on the Validity of the Simulation	43
	3.13	Discussion on the reliability of data	43
4		RESULT AND DISCUSSION	45

4.1	Circular motion test in MATLAB	45
4.1.1	Circular Path Generation with Different Radius	45
4.1.2	Increasing the Number of Points to Points in the Circular Path	47
4.2	Simulation of vision based robot in VREP using LUA	49
4.2.1	Circular Path Generation	49
4.2.2	Image centre tracking based on Vision Algorithm using blob detection	50
4.2.3	Image Tracking Using Vision Based Robot	52
4.2.4	Image tracking for adjustable radius	57
5	CONCLUSION	62
5.1	Conclusion	62
5.2	Recommendation	63
	REFERENCES	64
	APPENDICES	66



LIST OF TABLES

TABLE	TITLE	PAGE
2.1	Available solution for trajectory generation	11
3.1	Position of equipment in VREP simulator	36
3.2	Initial Position for each KUKA youBout arm joints	36



LIST OF FIGURES

FIGURE	TITLE	PAGE
2.1	Robot path in PTP	6
2.2	Acceleration and deceleration of link from start and end motion	6
2.3	Trajectory generation overview	7
2.4	Cartesian-path problem type	9
2.5	Cartesian-path problem type 2 in circular path	10
2.6	Overview of vision based robot for hand rehabilitation	13
2.7	Subset of vision-based robot system for visual part	13
2.8	Block diagram of general simple machine vision system	15
2.9	Image colour detection	16
2.10	Edge detection of a car	16
2.11	Shape detection	17
2.12	Blob detection with centre of mass of the red object	17
3.1	Scenario of vision based robot for hand rehabilitation	20
3.2	Circular motion with point to point	21
3.3	Method to generate tracking and following circular	21
3.4	Flowchart of project concept	22
3.5	Flowchart for generate tracking and following circular hand Motion	24
3.6	Desire and actual circular shape	25
3.7	Image trajectory of the objet during robot arm control	27
3.8	Resolution	28
3.9	View from vision sensor	28
3.10	Top view	30
3.11	World coordinates systems	31
3.12	World coordinates is VREP	31
3.13	4x4 matric	32

3.14	Transformation matrix from original to final position	32
3.15	Transformation matrix in VREP	33
3.16	LUA display value	33
3.17	Before calculation	34
3.18	After calculation	34
3.19	Experimental set up in VREP simulation	37
3.20	Workspace for circular path tracking using vision based robot	38
3.21	Kuka youbot vision	38
3.22	Calculation of error percentage	42
4.1	Adjustable circular radius from MATLAB simulation	46
4.2	Actual and desired circular motion of robot arm	46
4.3	Error area	47
4.4	Increase number of points	48
4.5	Error versus no of point to points	48
4.6	Generation of Circular Path using Point to Point	49
4.7	Generation of Circular Path by Increasing Number of Point to Point	50
4.8	Image centre read by vision sensor for circular motion	51
4.9	Image centre read by vision sensor for adjustable circular motion	52
4.10	KUKA motion versus hand motion	53
4.11	Motion of KUKA youbot in circular path with varying velocity of target	54
4.12	Maximum error region for circular path	55
4.13	Joint position for circular path	55
4.14	Joint velocity for circular path	56
4.15	Joint acceleration for circular path	57
4.16	Motion of KUKA in swirling circular path	59
4.17	Maximum error region in swirling circle shape	59
4.18	Joint position for swirling circular path	60
4.19	Joint velocity for swirling circular shape	61
4.20	Joint acceleration for swirling circular shape	61

LIST OF SYMBOLS

x	-	Horizontal displacement
y	-	Vertical displacement
t	-	Time taken
C	-	Circumference
P	-	Position
R	-	Radius
m	-	Gradient



LIST OF APPENDICES

APPENDIX	TITLE	PAGE
A	Pseudocode for point to point circular generation using youBot	66
B	Pseudocode for circular hand tracking using youBot	68
C	Image tracking values for circular motion	71
D	Image tracking values for adjustable circular motion (swirling)	75



CHAPTER 1

INTRODUCTION

1.1 Motivation

There are many causes that lead to upper limb pain which make the patient unable to move their hand normally such as stroke diseases, sports injury and accident. Statistics has shown that 26.7% Malaysian suffer upper limb pain which causes by sports injuries, overuse of muscle arm, stroke disease and accidents [1]. Patient who suffer this kind of impairment in motor and low muscle strength will typically receive intensive, hands-on physical and occupational therapy to encourage motor recovery and strengthen the muscle. The intensive therapy that will give motivation to the patient in order to help fast recovery after stroke or injury is called rehabilitation program. However, it is impossible for the patient to receive full therapy since the number of therapist in rehab centre and rehabilitation hospital is currently not enough to help the patient since there is a statistic shown that 70 number of patient receiving treatment at one time in a rehabilitation hospital [2]. Not only that, statistic also shown that in Malaysia, the ratio of occupational therapist is 1:20000 [2]. Therefore, due to lack of resources, patients are receiving less therapy and discharged from rehabilitation hospital sooner [3].

In order to solve the problem, recent research have shown that robotic training is a considerably new technology that shows great potential for application in the field of neuro-rehabilitation as it has several advantages such as motivation, adaptability, data collection, and the ability to provide intensive individualized repetitive practice [4]

Therefore, this research is about to provide a new technology to help therapy process by using an assistive vision based robot that able to help patient do the hand exercise without the help of expert occupational therapy practitioner. The new technology using vision based robotic assistance has the ability to provide intensive repetitive practice for each patient individually. The application of computer vision on the robot is to track human hand move in correct line. This project also will analyse the potential of robot to provide accuracy of the joint to reach the target distance and can follow the human hand.

1.2 Problem Statement

The problem in hand rehabilitation exercise is that the patient is not consistent in doing their exercise due to lack of guidance especially a when doing a complicated motion such as circular. An assistive robot is needed in order to guide, track and follow the patient hand accurately to avoid demotivation from stop doing the exercise. In order to track the circular motion accurately, a vision sensor is added to the robot to improve the reliability to assist the patient to ensure that the robot can guide the hand even though the speed and radius of circle is decreasing.

This project address two main problem in robotic rehabilitation applications which are generating a circular path in Cartesian space and the problem in identifying the position (x,y,z) and orientation of robot's arm end effector when generating a circular motion that can vary depending on the vision based feedback of patient hand.

Firstly, the problem of generating a point to point circular motion in Cartesian space (x,y,z) which involved the trajectory of robot to move from initial position to final position and in multidimensional space. In this problem, there is computational burden where if the number of point to point increases, the calculation will be more. However, by increasing the number of point, it will improve the smoothness of circular path. In this project, to generate the circular path, it requires the combination of trigonometry formula with circumference formula.

Next is the problem in identifying the positions (x,y,z) and orientation of robot's arm end effector when generating a circular motion that can vary depending on the vision based feedback of patient hand. The problem involved in describing the transformation of points and coordinates systems and describing the position and orientation of one coordinate system relative to another. The present of vision sensor make the robot have to follow the input from the vision feedback. The accuracy of tracking will decreases when the speed is increases. In this project, by combining the equation of straight line and applying some image processing technique such as blob detection, centroid detection and using inverse kinematic it can tracking the image accurately.

1.3 Objective

The objectives of this projects are:

1. To analyse the problem of generating circular path and tracking method using a vision based robot.
2. To develop circular path algorithm and vision based circular tracking algorithm to match with the patient hand motion.
3. To validate the smoothness of circular path and the accuracy of a vision based KUKA youBot motion when tracking and following the patient hand motion using statistical method.

1.4 Scopes of the Project

The scopes of this project are:

- The research is focused on positioning of 5DOF KUKA youBot in circular motion.
- The generation of circular motion is validates by the equation of circle by using MATLAB.

- The generation of circle path is by using LUA and simulated in VREP.
- The generation of tracking algorithm is by using transformation matrix in LUA and simulated in VREP.
- The performance of KUKA youBot is measure in terms of error and accuracy.



CHAPTER 2

LITERATURE REVIEW

2.1 Robotic Rehabilitation

In this era, the research on robotic rehabilitation technology has widely used in biomedical engineering. The used of robotic technology for rehabilitation purpose has help to give motivation to patient with upper limb problem. Generally, in the rehabilitation therapy patient was taught with repetitive motion to improve the smoothness with the help of therapist [5]. Same goes with robotic training, the patient also will receive repetitive movement in order to motivate themselves for recovery. The technique applied in the rehabilitation able to determine the adaptability level of the patient [6].

There are several training programs for enhancing the smoothness of hand motion such diagnosis, teaching with active-assistance robot, training with passive assistance and training with no assistant [7]. Some training enables the patient to move their patient in the pathway without any force pushing it. There are also a training that require an active movement with opposing force which when the patient try to move outside the set path there force that will prevent the hand movement [6].

The research on robotic training has been gradually improved with the applications of many types of sensor and controller such as using virtual curling task, game-based training and visual servoing feedback. In this project, the robot will be the assistant to give motivation to patient by getting the feedback from patient movement via camera. Therefore, the study of visual based robot for hand rehabilitation will help me to get information and knowledge.

2.2 Generation of Circular Path

In order to generate a circular path, there are two types of space which is joint space and Cartesian space. For this project it involves Cartesian space to generate the initial and final point to move the end effector. There are more than two points needed to generate a circular path. Since this project is about rehabilitation of hand, the motion of path is free circular shape and no need a precise controlled path. Therefore PTP is suitable to describe the robot motion. In PTP the movement from point A to point B is not necessary to be in straight line as shown in Figure 2.1. There is acceleration at the starting point and deceleration at the end point as describe in the Figure 2.2 below:

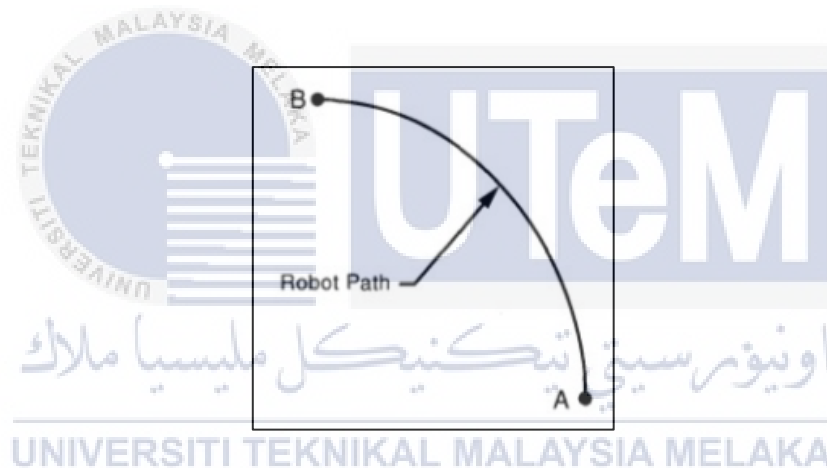


Figure 2.1: Robot path in PTP [13]

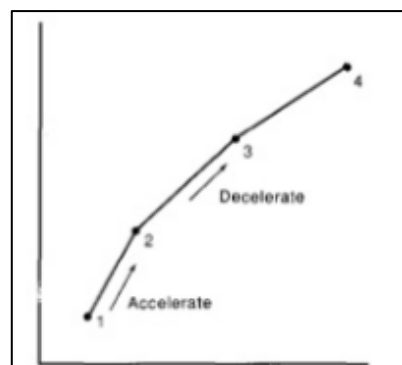


Figure 2.2: Acceleration and deceleration of link from start and end motion [13]

To reduce the acceleration and deceleration, the number of point to point can be increases and the time taken from one point to other will become short thus make the trajectory smooth.

2.2.1 Trajectory for Generating Circular Path

In this project the trajectory is to move the end effector from initial position to the desired position and to determine the smoothness of robot joint position in circular motion. Trajectory generation is related to the computation of desired motion of a manipulator to be smooth in multidimensional space. Trajectory also refers to a time history of position, velocity, and acceleration for each degree of freedom. The term trajectory generation is not only generating a path for a tool frame to be located within a tool frame, but also includes the human interface issue with the robot' path specification [11]. Trajectories are important because they enable the system to ensure feasibility where the motion can be verified to respect the dynamic constraints of lower level controllers [12]. The block diagram below shows the overview of trajectory system:

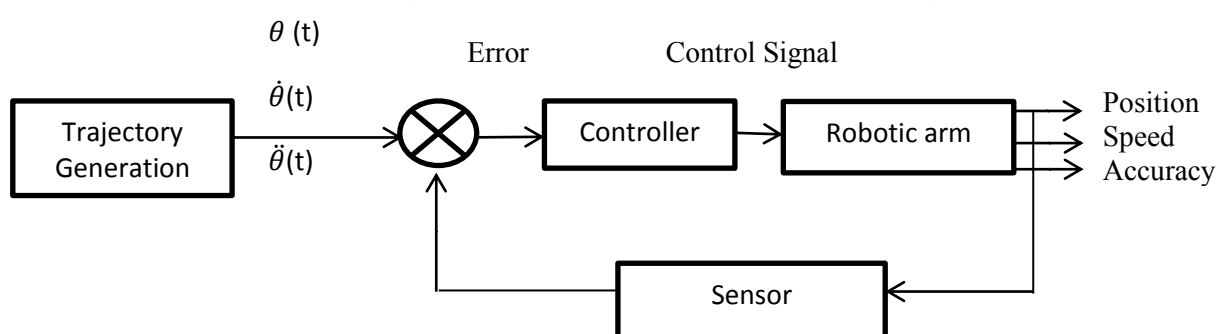


Figure 2.3: Trajectory generation overview [11]

The goal of trajectory generation is to describe the required motion of the manipulator as time sequence of joint/link/end-effector locations and derivatives of locations that is generated by interpolating or approximating desired path by a polynomial function.

Trajectory planning produced time history serve as reference input to the manipulator control system and the control system ensure that manipulator executes the planned trajectories [13].

For rehabilitation purposes, the required motion is the motion that generate by the patient with muscle problem. Therefore, the motion of joint is limited and it is generated by imposing the inverse kinematic that drives the manipulator to the desired position. The available inverse kinematic also will determine the accuracy and precision of the robot arm joint motion.

2.2.2 Problems of Generating Circular Path in Cartesian Space

In this project, the simulation is in Cartesian space since in reality the motion of patient hand is in terms of (y,z) coordinates. In Cartesian space, the velocity, position and acceleration is determine in terms of Cartesian coordinate (x,y,z) and the joint actuators are served in joint coordinates to the specific trajectory. To use this technique, user need to specify the desired end-effector path, the travelling time, and the tool orientations along the path [13]. Cartesian technique is quite complex to use compared to joint space. However, it is easy to use this technique if we wish to do straight line path or circular path because it is more accurate. Not only that, if around our workspace have obstacles, it is more suitable to use Cartesian coordinates. The limitation of this technique is singularities may occur at the manipulator. There are some challenges that related with Cartesian path such as intermediate points unreachable, high joint rates near singularity and start and goal reachable in different solutions.

However for rehabilitation purpose, the challenges that might be face in doing the trajectory in circular motion is that the problems of high joint rates near singularities start and goal reachable in different solutions. It is important to discuss this problem because the circular hand motion that is generated by patient might be too big or too small depends on their muscle strength. Therefore, there are possibility that robot might face these problems.

A. Type 1: Problem of high joint rates near singularities

In this problem, by referring to Figure 2.3.1(a) below, it shows that the planar wants to move along path point A to point B. The desired velocity is needed to be constant when moving along the straight line path. However, as it goes to the middle point the velocity of joint one is very high [11].

In the hand rehab point of view, the singularities also might happen when doing circular motion. If the velocity of the joint is high, it will make the joint unable to track human hand properly due to the sudden velocity change.

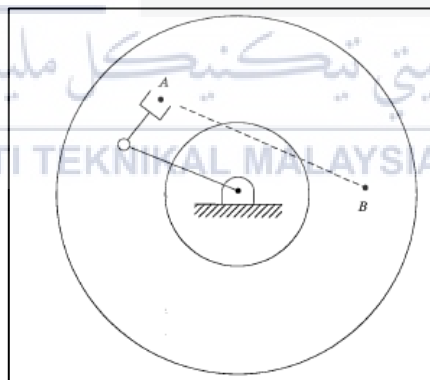


Figure 2.4: Cartesian-path problem type 1 [11]

B. Type 2: Problem of start and goal reachable in different solutions

In this problem, the planar has two-link with equal lengths. This makes the joints limits restrict the number of solutions where it can reach a given point in space. The problem is that the robot might not have same physical solution as the start point to reach goal point. In the it hand rehab view, this problem will happen if the two equal length links with a

circular motion. It seems difficult to handle the control in Cartesian space. The Figure 2.5 shows the possibility of points that can cause this problem. In rehabilitation, this problem will not occur unless if using two types of robot to generate a similar exercise.

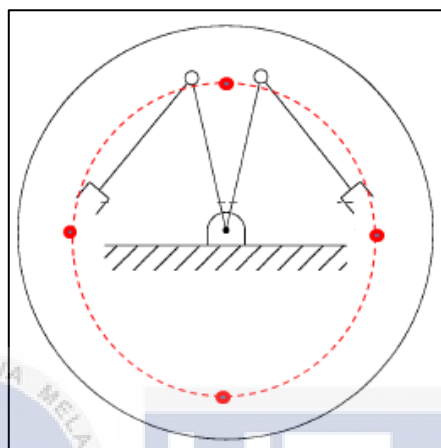


Figure 2.5: Cartesian-path problem type 2 in circular path [11]

2.2.3 Available Solution for Generation of Circular Path

There are several ways to generate a trajectory of robot in circular motion. According to Ken-ichi Yamada et al [14], they generate a circular motion for mobile robot by using control law to make the robot determine the angle at the between two closest robot by itself. The researcher used a numerical simulation to generate the circular motion. The design parameters are based on the number of mobile robots, the radius and the required speed on the circle and gain value. The position and orientation are measured by a web camera and the controlled law for each mobile robot is computed by a central controller or PC [14]. This method produced a smooth circular motion but only if the mobile robot is attached with a distance sensor.

Next, according to Byoung-Ju Lee et.al [15], they generate trajectory for soccer robot. For the project, an accurate motion trajectory control is very important to avoid obstacles. The problem of the project is the discontinuous trajectory and it is solved by a simple smoothing process. The researchers proposed using arc and line to make a continuous motion. For the motion tracking control in circular trajectory, they take radius of steering as the control parameter instead of steering radius. They used the combination of length vector and trigonometry formula to calculate the radius by determines the central point of target. Therefore, the robot will track its trajectory in circular motion. However, this method only gives accurate position control if the speed of robot is slow. This is due to the time delay limits its accuracy and it can be improved by further research on the precise control of the robot [15].

2.2.4 Summary for Available Solution in Generation of Circular Path

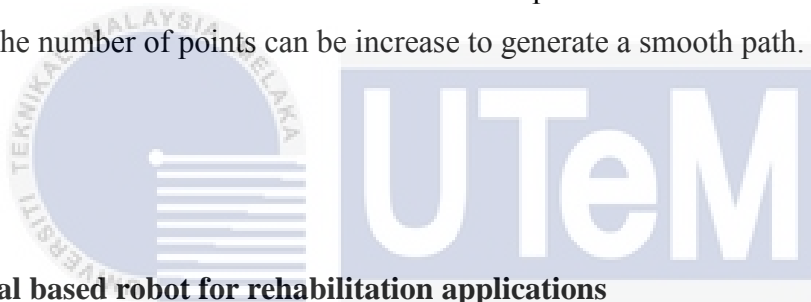
The Table 1 below shows the available solution for trajectory generation in circular motion.

Table 2.1: Available solution for trajectory generation

Technique	Problem	Comment
1. Control law	To generate a circular motion trajectory generation for mobile robot.	It requires a design of controller to control the motion of robot in circular motion. The design is depends on the linear speed and given radius.
2. Line and arc formula	To generate trajectory of a circular motion for robot soccer game.	The design is limits by the time delay. Error occurred when the speed of robot increases.

In summary, both method proposed by [14] and [15] can be used to generate trajectory in circular motion. Even though both methods are applied on a mobile robot, but the method of generating trajectory of robot in circular motion still can be referred. Both methods need to specify a radius value in order to make the robot move in circular. The method [15] is concern on the continuous velocity vector because in soccer game the robot must keep running during the whole game as same time it must avoid obstacles. In rehab point of view, the position is more concern because the accuracy of tracking the motion is important to motivate the patient hand and there is no obstacle around the scene.

In this project, the proposed method to generate circular motion is by using trigonometry formula to create the point to point to generate a continuous circular path. The movement of robot arm is free around its workspace so there is no limitation on the motion and the number of points can be increase to generate a smooth path.



2.3 Visual based robot for rehabilitation applications

There are some studies that have been conducted to improve the robotic rehabilitation technology based on visual feedback to enhance smoothness of upper limb motion for people with upper limb problem. The overall vision based robot system for this project is shown in Figure 2.6 below. The human motion will be captured by the camera and match it with robot movement. The input of the robot is the visual feedback and it will be analyse by the controller such as using MATLAB, Python, C programming or other computer languages to control the robot motion.

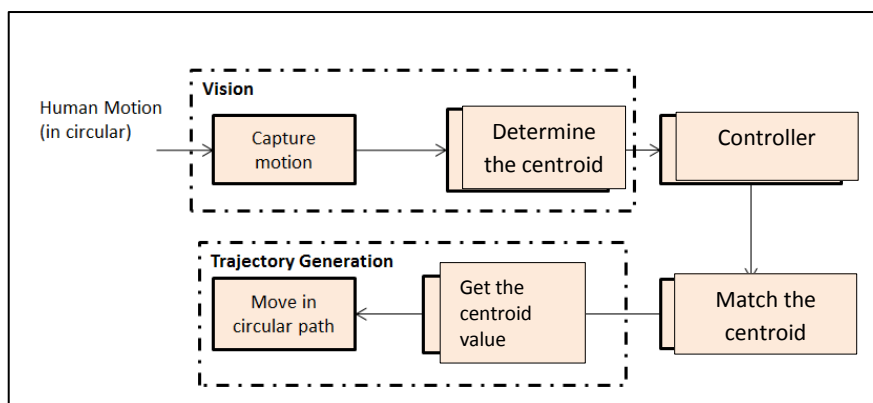


Figure 2.6: Overview of vision based robot for hand rehabilitation

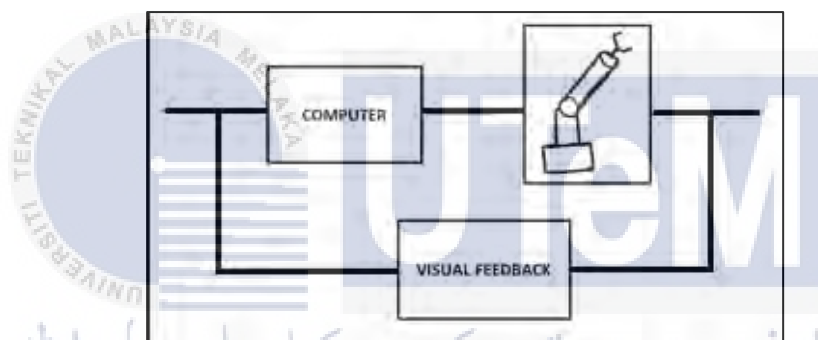


Figure 2.7: Subset of vision-based robot system for visual part

According to Won-Kyung Song et. al [6], it discussed about a robot with visual servoing will give autonomous capability to a rehabilitation robot to interact with human. There is much important information for a visual servoing such as the depth and in real-time. As we know, visual system will provide a visual feedback for a robot end-effector for closed loop control. One of the advantages of using vision sensor is it makes the robot get non-contact measurement of the environment. The configuration of camera is also important in order to get the accurate depth and in real time. The configuration is based the types of rehabilitation such as using a static robotic arm or wheelchair mounted robotic arm. The configuration also depends on the relation between space variant vision which has high speed processing and eye-in-hand camera. Therefore, this proposal gives some theory of robotic visual in order to make the vision sensor as feedback to controller to

make the end effector move according to desired circular motion according to the output from vision sensor.

Next, according to Ravipathi [9], this paper use image processing to design a vision based interface to instruct a humanoid robot doing some gestures. The application is developed using OpenCV (Open Computer Vision) libraries and Microsoft Visual C++ to process the live images which will give command to the humanoid with simple capabilities. The project is interfacing the USB-UIRT (Universal Infrared Reciever and Transmitter) with humanoid toy robot (Robosapien) to obtain the output result. The implementation for this project is the computer will acquire the single frame from the real time video capture and process the image by principle of thresholding then calculate the static position of the blob. A pattern matching algorithm is developed to ensure the robot move according to the given gesture. From this proposal, the technique of image processing using blob detection can be apply into this project where a patient can hold a marker as the blob and make the robot vision detect it and follow the motion of blob when the patient do the rehabilitation exercise.

Lastly, according to Benjamin Busam et al. [10], this paper proposed a light weight robot with a stereoscopic camera system. The camera detects a flat round reflective marker that equipped along with sleeve that wears by healthy human. The challenge is to ensure the robot can accurately track the healthy human arm in real-time. The tracking algorithm is developed based on circular marker shape to fit the geometry primitives. Since a stereo vision is use, the calculation of a sparse 3D point coordinate is needed to obtain the subpixel accurate centre. The advantage of using camera-in-hand tracking system is that it minimizes the problems of interrupting the line of sight between the camera and the markers. The contribution of this paper is it introduces a method that can train a natural gesture of hand. From this paper, the method to matching the stereo left and right image is by obtaining the coordinate value. Therefore, this method can be a reference for the hand tracking problem using a vision based robot to accurately match the position of robot arm with patient hand motion.

2.3.1 Robot Vision for hand tracking using marker

Since this project is about the investigation of tracking problem using vision based robot for hand rehabilitation application, a vision sensor (camera) will be attached on the KUKA youBot in VREP. In this case, we are not solving the vision problem but only using the available technique. However, it is still important to know the basic of machine vision knowledge. The Figure 2.8 shows the basic machine vision system.

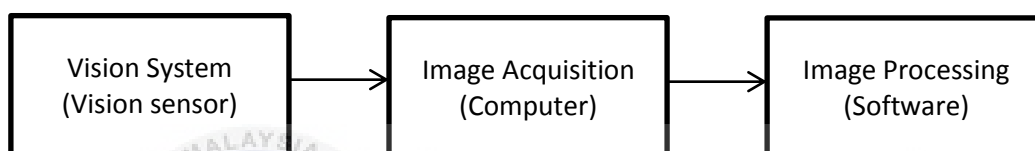


Figure 2.8: Block diagram of general simple machine vision system

In order to make the computer recognize the object, it is easy to determine the colour or shape of an object. To represent colour of image, there are two available methods which are RGB (red, green, blue) and HSL (hue, saturation, and luminance). For RGB, red green and blue are the three primary colours that represent all possible colours. RGB also defined the brightness and colour of pixels in an image. Another method that represents colour of an image is HSL. H (hue) is to define colour of a pixel such as red, yellow, green and blue or combination two of them. S (saturation) refers to amount of white added to the hue and represents the relative purity of a colour. L (luminance) is related to the brightness of image [17].

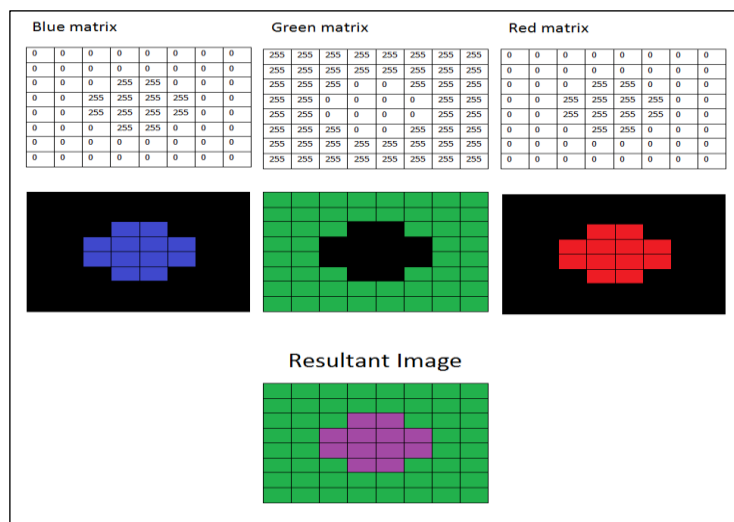


Figure 2.9: Image colour detection [17]

There are several methods to build basic algorithm of computer vision in image detection which are edge detection, shape detection or pattern recognition and blob detection. Firstly edge detection, it is a technique to define the edges of objects in a scene and this is useful to detect the corner or shape of an object. The Figure 2.10 shows the example of edge detection [17].



Figure 2.10: Edge detection of a car [17]

Secondly is shape detection. This method requires a lot of programming in order to specify the types of shape that user wished to detect such as triangle, rectangle or any polygon. The Figure 2.11 below show the example of shape detection [17].



Figure 2.11: Shape detection [17]

The simplest method is using blob detection technique. Blob detection is an algorithm used to determine whether a group of connecting pixel are related to each other or not. This technique is useful for counting numbers of object or object tracking. Therefore it is suitable to be used in this project. In this method, the centre of mass of selected coloured object will be traced. The image must be threshold by specific colour in order to find blob. The unwanted colour will be removed by using algorithm so that the computer only recognize desired object. The colour of the object will determine using RGB method. Figure 2.12 below shows the example of blob detection technique [17].



Figure 2.12: Blob detection with centre of mass of the red object [17]

To conclude, blob detection is the simplest method compared to other available method plus it is easy to find a coloured object as the marker. In rehabilitation, a patient will be given a marker and the camera track and follow the hand motion based on the marker.

2.4 Method to Generate Circular Path and Tracking Hand Motion

In summary, there are two methods could be applied in this project. Firstly, for the circular path generation, the point to point is generates in circular path in Cartesian space. By imposing the inverse kinematic, it will produce the joints value to make the end effector move to desired position. In rehabilitation exercise, the motion of human hand is not really an exact circle because of human nature cannot exactly know the radius of motion that they produce. Therefore, for the point to point method is suitable because the end effector is freely move in path in the Cartesian space due to no obstacles in the scene.

For the vision part, a simple blob detection technique could be used to track the human hand motion in order to provide accuracy of the KUKA youBot to follow the hand motion. The generation of tracking algorithm as proposed by (Ravipati, 2014) [9] could be used as the initial stage for understanding the blob detection technique before doing the simulation in VREP by using LUA syntax. In rehabilitation, rather than gesture detection, the blob detection using marker is also easy to make the vision sensor detect the hand motion. If the patient could not hold the marker, a blob ring could be used as long as there is a marker for the camera to detect it.

CHAPTER 3

METHODOLOGY

3.1 Theoretical Concept

To move the robot arm in circular motion, different task must be solved. It must have good trajectory, considering some limitations and achieve high efficiency. Generally, there are some types of constraint for motion planning such as obstacle constraints, path constraints and mechanical constraints. Overall, in this project:

- i. There is no obstacle along the circle path.
- ii. A 5DOF robot (KUKA youbot) with a camera attached on it will be used to do the circular motion using parametric equation of circle.
- iii. The visual technique used is simple blob detection.

Imagine the scenario of this project as shown in Figure 3.1, human will sit opposite of the robotic arm that attached with camera on it. The visual feedback from the camera will become the input for the robot. Patient will hold a marker for the tracking purpose. Firstly, the robot will do a circular motion to guide the patient hand. If human hand is unable to follow the robot motion, the robot will detect the hand and starts to follow the patient motion in various radius and speed. The robot will do circular motion according to patient ability. The circle radius will gradually change by reading the visual feedback from the camera. The camera track hand motion and send the output to the robot to ensure the robot follow patient hand accurately.

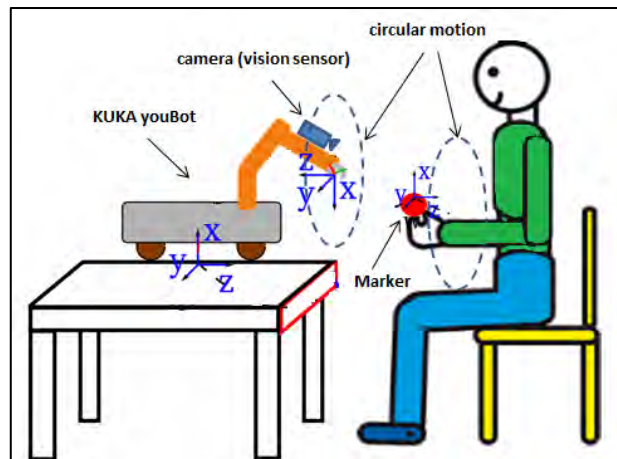


Figure 3.1: Scenario of vision based robot for hand rehabilitation

In order to generate trajectory in circular motion, a parametric equation of a circle will be used. The formula of circumference, C is:

$$C = 2\pi r \quad (3.1)$$

Where; r = radius

Next to find the point to point around the circular position to know the position, P of robot movement in terms of coordinate (x,y) , we are going to use circular trigonometry equation. Therefore, position, P is equal to:

$$P = (x, y) \quad (3.2)$$

$$x = r \cos \theta \quad (3.3)$$

$$y = r \sin \theta \quad (3.4)$$

$$P = (r \cos \theta, r \sin \theta) \quad (3.5)$$

Note that the value of θ can be found using trigonometric equation for circle using radius, r . These equations will be applied to generate a circular shape in MATLAB.

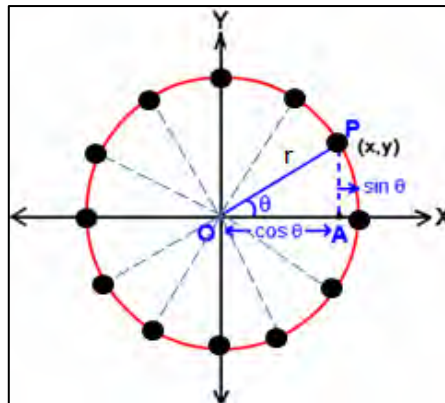


Figure 3.2: Circular motion with point to point

Next, to make the robot follow human hand, the camera will traced the centre point of the blob in order to make the circular shape. The method to get the circular motion is shown in Figure 3.3.

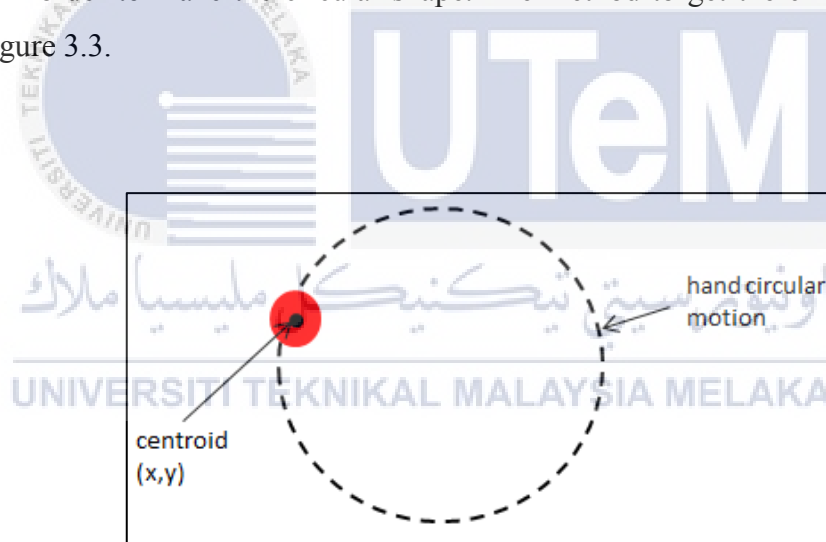


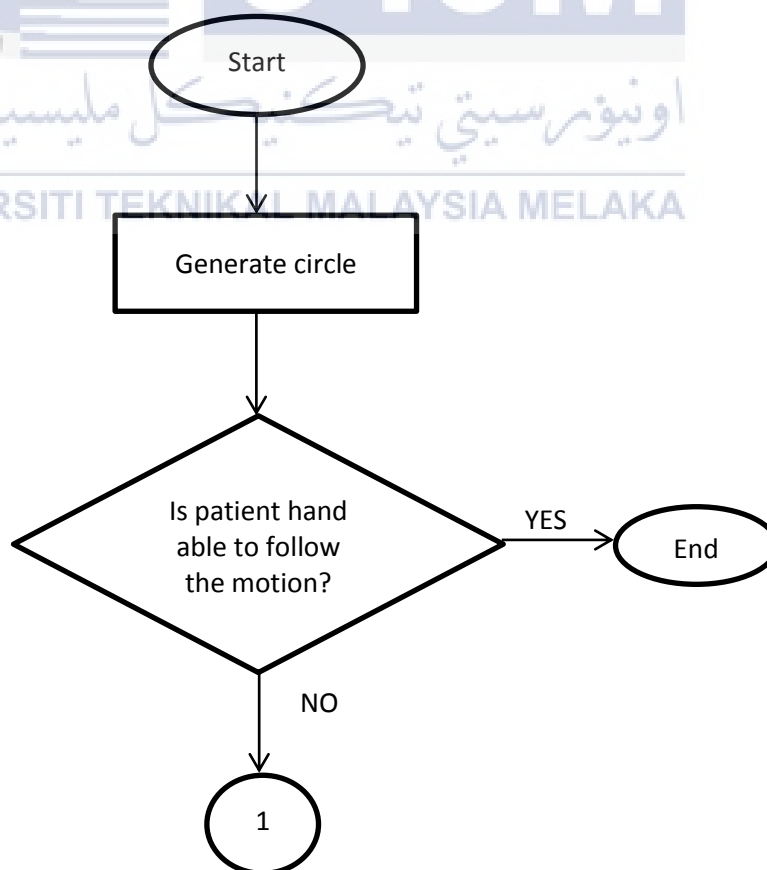
Figure 3.3: Method to generate tracking and following circular

In this motion, there are several muscles that involve motivate the hand. For the robot, there are several joints that involve and the value of theta for every joint is solved from the inverse kinematic.

3.1.1 Flowchart for Circular Path Generation and Tracking Hand Motion

Firstly, to enable the patient train their hand to follow the robot, a size of circle is set on the robot to make the motion so that the patient will follow the robot motion. The size of circle given based on the condition of muscle strength either it is weak or almost recovered. Therefore, first the robot will generate a circle as a guide for the patient hand.

Next, if the patient is unable to follow the robot motion, the robot will detect it and it will try to track the patient hand and follow the hand motion. This tracking part is to prevent demotivation of patient from stop doing the exercise. The tracking and following is according to the speed and radius of the hand motion. It means that if the patient motion speed and radius changes the KUKA youBot arm position and speed will also change. The flowchart below shows the process for the overall project concept to enhance the understanding.



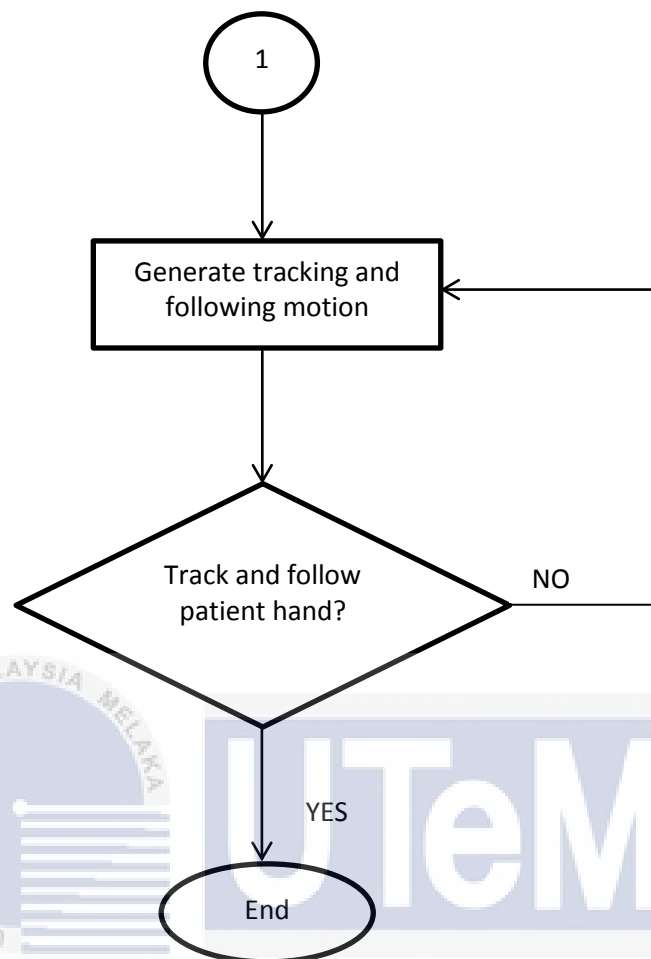


Figure 3.4: Flowchart of project concept

Next, for the tracking part, the image processing technique is needed in order to make the KUKA youBot able to track and follow the patient hand accurately. Firstly, the vision sensor will detect the present of human hand and do the image processing which are thresholding, blob detection and centroid detection. Then, some calculation is applied in order to make the end effector move according to patient motion. The flow chart in Figure 3.5 below shows the process for generate tracking and following algorithm

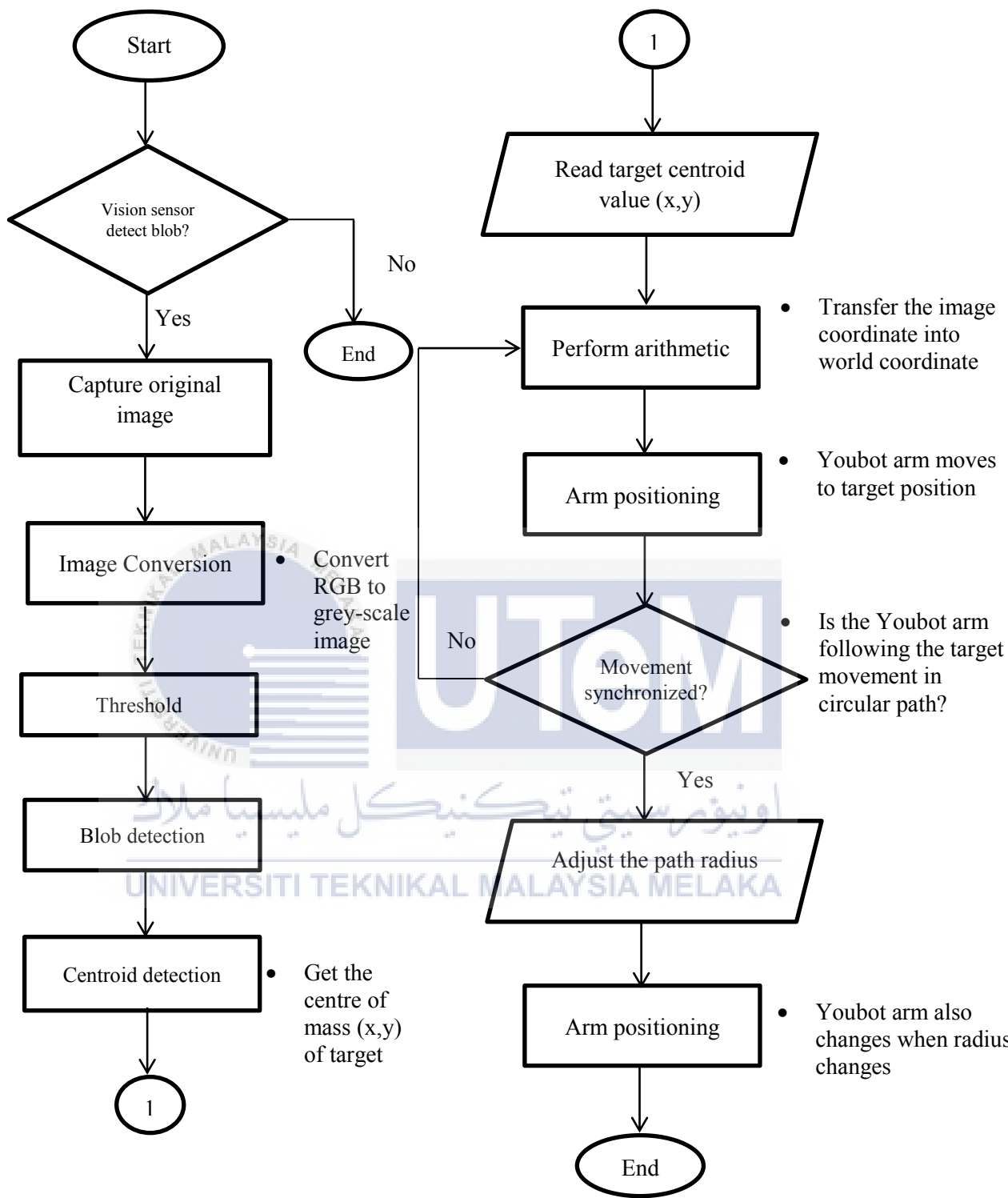


Figure 3.5: Flowchart for generate tracking and following circular hand motion

3.2 Generating circular shape motion

For simulation, the value of radius is set to minimum 12cm and maximum 20cm. These values are assumed just to validate that the equation of circle can produce adjustable circular motion using MATLAB.

However, the reality is the movement of robot is not as smooth as in Figure 3.1. It will create such a polygon shape as shown in Figure 3.6 below.

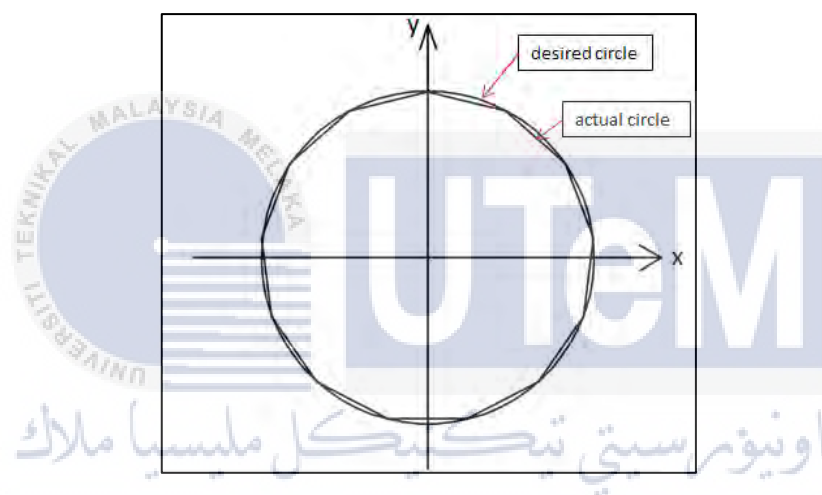


Figure 3.6: Desire and actual circular shape

In order to get the desired circle in MATLAB, a cubic spline method can be used. By getting the desired circle, we can compare the polygon shape with the smooth circle. The method is done by using some code in the MATLAB library. This method is just to compare the results of actual and desired circle for MATLAB simulation.

However, to get a similar smooth circle, the number of via points must be increased. The method will be done by adding the number of via points up until 100. The starting points will be 12, 25, 50, 75 and 100.

In this project, the position, velocity and acceleration of each joint are determine to know the smoothness of vision based robot motion in tracking and following the patient

hand. By analysing the trajectories, the conclusion can be made whether the vision based robot is reliable to be used for circular hand tracking problem rehabilitation purpose or not.

3.3 Object Tracking Method

Since this project is hand tracking using visual based robot, the simplest method that is going to be used is simple blob detection. Human will grip a marker with orange colour to make the robot enable to track it. Orange colour is chosen because the colour is bright make the camera easy to recognize it. A calculation is done by the computer to determine the centre point of an object to track their motion. The preferable method is to detect an object with HSL (hue, saturation, and luminance) value. This method is only used for openCV library with C++ because it easier to detect the object with HSL value. The algorithm is developed based on the hue of the basic colour:

- | | | | |
|------|--------|---|---------|
| i. | Orange | : | 0-22 |
| ii. | Yellow | : | 22- 38 |
| iii. | Green | : | 38-75 |
| iv. | Blue | : | 75-130 |
| v. | Violet | : | 130-160 |
| vi. | Red | : | 160-179 |

In this simulation using OpenCV, orange colour marker is chosen. This is because orange has striking colour that make the computer easy to recognize it. To recognize the object, an image thresholding method will be used so that the robot only detects the orange colour marker. The algorithm is development by using the OpenCV library with Microsoft Visual Studio C++.

In the simulation using VREP, the comparison of actual and desired value of motion using vision sensor is analyse by determine the accuracy of robot following the

human hand. The camera will capture the marker image that is hold by human and get the centre point in order to follow the motion using line tracking. Figure 3.7 below shows the method of determining the image trajectory of the object during robot arm control. The blue line is the object line tracking.

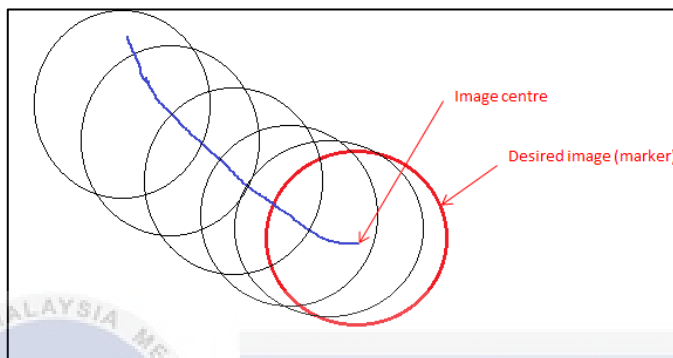


Figure 3.7: Image trajectory of the objet during robot arm control

Therefore, by using this method it easy to track the image by determining the centre point of marker object.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

3.4 Calculating Vision Frame Size

The vision frame size is calculated to obtain the image ratio where image ratio is described as proportional relationship between width and height and image. In order to calculate the vision frame, calculations in pixel coordinates are required. A red plane square shape with size (10cm x 10cm) was placed at the centre of camera coordinate. This project uses 256x256 resolution of vision sensor. The resolution represent matric (MxN) or axis ($x \times y$). Figure 3.8 below shows the representation of resolution in x and y axis.

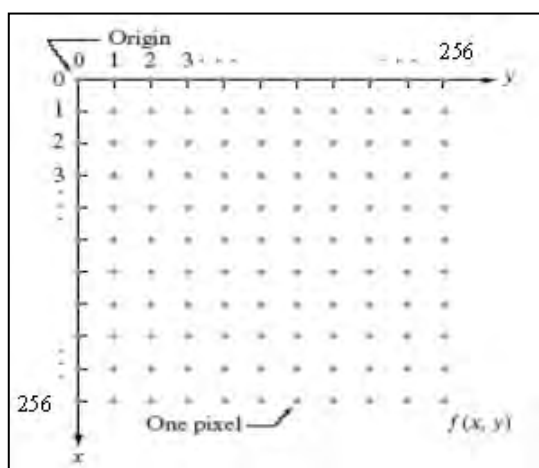


Figure 3.8: Resolution [17]

For the vision sensor frame view, the coordinate of target is in (x, y) value. Figure 3.9 below shows the illustration of frame (x, y) in VREP. The value is referring to centre of mass of object with references to vision sensor frame. Since the range values of x y axis are from 0 to 1, so the image coordinate in the centre of frame is $(0.5, 0.5)$. To calibrate the size of image, we need to know the width and height of the target. By setting the image processing filter in VREP, it gives value of (0.197×0.197) for the width and height of the plane.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

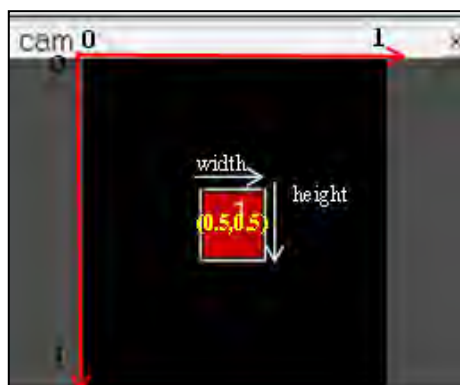


Figure 3.9: View from vision sensor

Before getting the size of vision frame, we need to find the plane size in pixel. Since square shape has same width and height values, we only use one value. The calculation of image size in pixel is as below:

$$\text{Image pixel} = \text{Width of object} \times \text{resolution} \quad (3.6)$$

$$0.197 \times 256 \text{ pixel} = 50.432 \text{ pixel}$$

Therefore, the red square plane with size (10cm x 10cm) is 50.432 pixels. To get the size of each pixel in cm is as follow:

$$10\text{cm} = 50.432 \text{ pixel}$$

$$1\text{pixel} = \frac{10}{50.432}$$

$$1 \text{ pixel} = 0.198\text{cm}$$

Therefore, to find the frame size:

$$\text{Frame size} = 256 \text{ pixel} \times 0.198\text{cm} \quad (3.7)$$

$$\text{Frame size} = 51\text{cm}$$

Therefore the image ratio for the plane is about 1:5 ratio.

For image processing, image calibration is not necessary because it is already calibrated and the camera does not have a fish eye effect plus the image is already flattened.

3.5 Angle of View from the Camera

Angle of view is angle to make the end effector move in specific position so that the target image will appear in the centre of vision sensor frame which is (0.5, 0.5) coordinate. To calculate the angle of view, we can use trigonometry formula. The Figure 3.10 below shows the position of objects by referring to world coordinate.

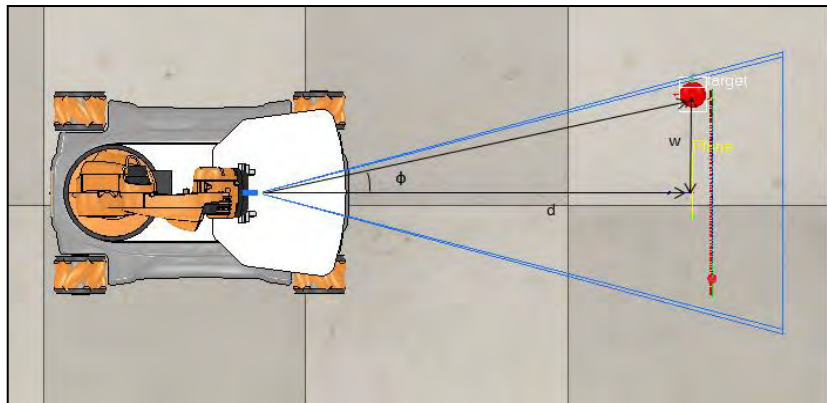


Figure 3.10: Top view

The formulas below are used to get the angle θ of view:

- i. Angle of view.

$$\tan \theta = \frac{w}{d} \quad (3.8)$$

- ii. Value of y

$$y = y_2 - y_1 \quad (3.9)$$

For the value of w , it is the value of y coordinate where it could be calculated from the value display in VREP and insert in formula 3.9. The value is 0.1899. Then, from the formula 3.8, the angle of view θ is 10.75° .

However, it is quite hard to adjust the angle of view since the robot is already in inverse kinematic (IK) mode. So, the other alternative to move the robot so that the target is in the centre of image frame is by adjusting the position of the end effector according to world coordinate. This is because the coordinates of KUKA youBot is refer to its own world frame. Therefore, a transformation matrix method was used to get the position of $[x, y, z]$ of robot according to world coordinate. Figure 3.11 below is the example coordinates systems translation and axes coordinates are defined with respect to the world coordinate system. Figure 3.11 is the coordinate of target and world.

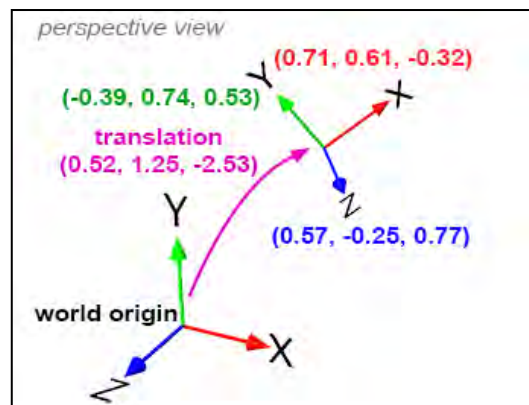


Figure 3.11: World coordinates systems [19]

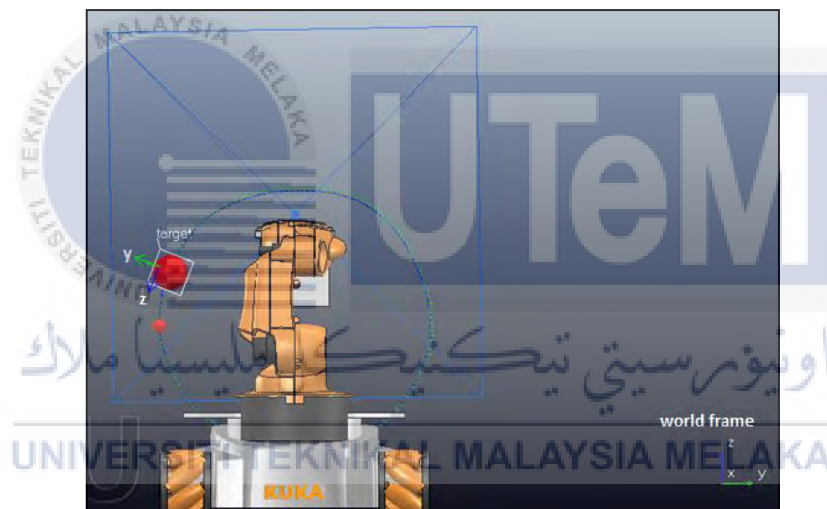


Figure 3.12: World coordinates is VREP

The transformation matrix in Cartesian coordinate is represented in 4x4 matrix form. This coordinate system is a convention used to define the coordinates $[0, 0, 0]$ in our 3D virtual space and three unit axes orthogonal to each other. Figure 3.13 below shows the 4x4 matrix for Cartesian coordinate. The first three coefficient of the fourth row (c_{30} , c_{31} , c_{32}) are the coordinates of the coordinate's system position. In LUA, the coordinate return value is $m[4]$, $m[8]$, $m[12]$. Meanwhile for the translation, it is always $[0\ 0\ 0\ 1]$.

$$\begin{bmatrix} c_{00} & c_{01} & c_{02} & c_{03} \\ c_{10} & c_{11} & c_{12} & c_{13} \\ c_{20} & c_{21} & c_{22} & c_{23} \\ c_{30} & c_{31} & c_{32} & c_{33} \end{bmatrix} \rightarrow \begin{array}{l} x\text{-axis} \\ y\text{-axis} \\ z\text{-axis} \\ \text{translation} \end{array}$$

Figure 3.13: 4x4 matrix [19]

3.6 Positioning the End-Effector to Make the Target in the Centre of Vision Frame

In order to make the image at the centre of vision sensor by moving the end effector, some calculation is needed. After the transformation matrix, the target is referred to the world frame and it is moving in y-z axis. For example, consider that the original point is (x,y,z) and the desired position is $(x + dx, y + dy, z + dz)$ [20]. The matrix will then look like Figure 3.6 below:

UNIVERSITI TEKNIKAL MALAYSIA MELAKA
اونيورسيتي تيكنيكل مليسيا ملاك

Translation matrix

$$\begin{bmatrix} 1 & 0 & 0 & dx \\ 0 & 1 & 0 & dy \\ 0 & 0 & 1 & dz \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x + dx \\ y + dy \\ z + dz \\ 1 \end{bmatrix}$$

Figure 3.14: Transformation matrix from original to final position [20]

Therefore, the position of end effector must refer to the target world. The equation of straight line was used to make the end effector move to target.

$$y = mx + c \tag{3.10}$$

From the data given in VREP, the gradient, m is 0.5208 by using the formula 3.9 below:

$$m = \frac{y_2 - y_1}{x_2 - x_1} = \frac{\Delta y}{\Delta x} \quad (3.11)$$

Then, if $x=0$, $y=0.0944$ (from LUA return value) by using formula 3.8, the value of C is -0.0492. Next, by inserting the calculated values in formula 3.8, the value of y for end effector is 0.2112.

The same method is used for z position. The negative positive sign is depends on where the robot move when given the value. In this case, the value 0.2112 must add with negative sign so the robot moves to correct position. If not the position is opposite the real goal. The results are as shown in Figure 3.15 & Figure 3.16. In figure 3.16 it is shown that the image is at centre of frame when the value of y axis is -0.2112 and z axis is 0.4479.

```
transformMatrix = simGetObjectMatrix(target,-1)
axRel=transformMatrix[4]
ayRel=transformMatrix[8]-0.0062 --equal to -0.2112
azRel=transformMatrix[12]+0.091 --equal to 0.4479
```

Figure 3.15: Transformation matrix in VREP

```
0.51145732402802    0.49460199475288
0.040130417793989  -0.21128073270321  0.44796476697922
0.51145732402802    0.49460199475288
0.040130417793989  -0.21128073270321  0.44796476697922
0.51145732402802    0.49460199475288
0.040130417793989  -0.21128073270321  0.44796476697922
```

Figure 3.16: LUA display value

Figure 3.17 and 3.18 below shows the different between before and after applying the calculation of positioning the end-effector to make the target in the centre of frame.

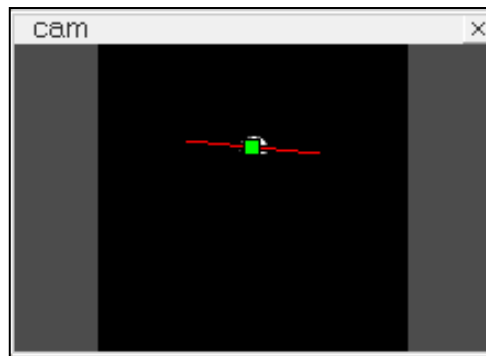


Figure 3.17: Before calculation

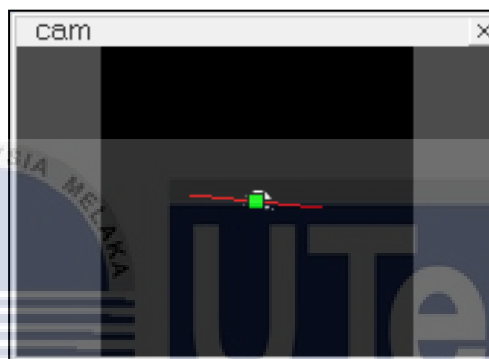


Figure 3.18: After calculation

3.7 Objective of simulation

In this project, there are two types of simulation which are using MATLAB and VREP Simulator. Therefore, there are different objectives for each simulation.

1. Objectives of simulation using MATLAB:
 - i. To validate the equation of circle to produce circular motion.
 - ii. To proof that by increasing the number of via point will make the circular motion smooth.
2. Objectives of simulation using VREP:

- i. To show the trajectory of robot from initial and final position in generating a part of circular motion.
- ii. To access the accuracy of robot motion for tracking and following target.
- iii. To determine the smoothness of each joints motion.

3.8 Simulation Equipment

The list below is the equipment used to do simulation for the project.

- i. KUKA robot arm with visual sensor attached on it.
- ii. MATLAB to develop algorithm for robot circular motion.
- iii. LUA to develop algorithm of the vision based robot for circular hand tracking.
- iv. VREP software for robot simulation.
- v. OpenCV library to develop algorithm for simple blob detection.

3.9 Simulation Setup

In the simulation, the initial position of robot is determined in (x, y, z) coordinate. The illustration for robot set up in VREP as shown in Figure 3.19 below. There are no obstacles around the robots workspace. By referring to Figure 3.1, human hand motion will be replaced with a sphere shape target in red colour attached on a circular path. The vision sensor with perspective view was attached on the end effector KUKA robot arm to detect the colour of the target. The vision sensor represents the camera while the red colour sphere shape target represent as a marker that is hold by a patient. The vision sensor is in perspective view, the sphere shape has radius 0.0.5m while the path is a circular type for circle motion and segmented type for the adjustable circular motion. The equipment position and orientation for this project is as Table 3.1. All the value is set by referring to world frame.

Table 3.1: Position of equipment in VREP simulator

Equipment	Position(m)	Orientation (θ)
KUKA youBot (Referring to base of robot)	$x = +1.2018e+00$ $y = -2.4917e-02$ $z = +9.5355e-02$	Alpha = +0.0000e+00 Beta = +8.9980e+01 Gamma = -1.8000e+02
Visual Sensor (Perspective view)	$x = +1.0533e+00$ $y = -2.3877e-02$ $z = +4.6109e-01$	Alpha = +0.0000e+00 Beta = -9.0000e+01 Gamma = -9.0972e+01
Target (red colour)	$x = +1.5086e-02$ $y = -3.6334e-02$ $z = +5.2568e-01$	Alpha = -7.9143e+01 Beta = +3.7663e-03 Gamma = -1.7998e+02
Path	$x = +1.5000e-02$ $y = +1.0000e-02$ $z = +2.8000e-01$	Alpha = +0.0000e+00 Beta = -8.9980e+01 Gamma = -1.0018e-05

Next, the initial position of each joint is as shown in Table 3.2 below:

Table 3.2: Initial Position for each KUKA youBout arm joints

Joint	Position min [deg]	Position [deg]
youBot Armjoint0	-	+8.079e-03
youBot Armjoint1	-9.000e+01	+3.091e+01
youBot Armjoint2	-1.310e+02	+5.242e+01
youBot Armjoint3	-1.020e+02	+0.000e+00
youBot Armjoint4	-9.000e+01	-5.668e-03

The initial position of equipment in this project in VREP simulator is as shown in Figure 3.19 below:

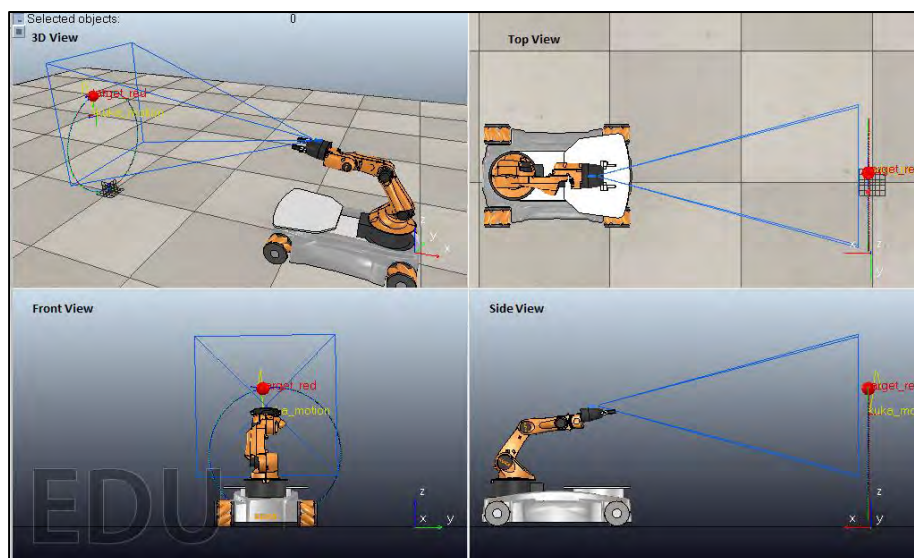


Figure 3.19: Experimental set up in VREP simulation

3.9.1 Workspace of the Circular Generation and Tracking Hand Motion

Figure 3.20 shows the workspace area for simulation using VREP. KUKA youBot is a mobile robot which can move freely according the vehicle target. However, this project involved the use of the robot arm only. Therefore the reachable workspace is only around the circular dotted line. The robot arm can only move around its workspace area if given input from vision sensor.

For the vision sensor, the type of camera is a perspective view so that it can have a wide vision area compared to orthographic view which is more suitable to focus only one image. The Figure 3.21 below shows the workspace for vision sensor. The reachable distance for the vision sensor to detect an image is called near or far clipping plane. In this project the far clipping or the maximum distance is between 1m from the vision sensor position while the near clipping which is minimum distance is 0.01m. The perspective angle or the maximum opening angle of the detection volume when the sensor is in perspective mode is set to 30° . In real applications, patient should sit between 1m from the youBot and do the circular motion not over the perspective view area. In addition, patient

should not wear red hand accessories other than holding the blob if their hand entering the visible area.

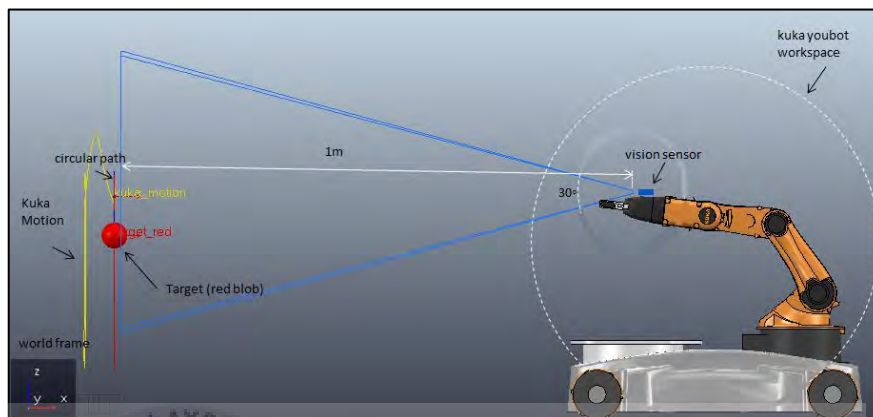


Figure 3.20: Workspace for circular path tracking using vision based robot

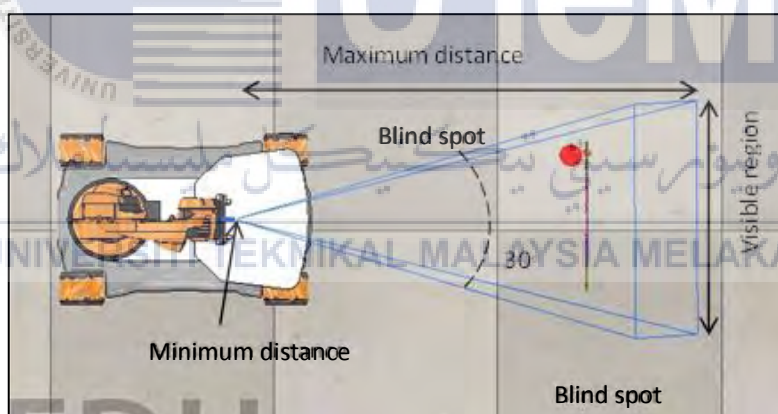


Figure 3.21: Kuka youbot vision

3.9.2 Pseudo code of Point to Point Circular Path Generation for KUKA youBot

Refer Appendix A

3.9.3 Pseudo code of Path

The pseudo code below is for the path algorithm.

1. **Program** movement of position;
2. Local maximum position, velocity and acceleration equal to start position;
3. Local target position and velocity equal to end position;
4. Local maximum velocity, acceleration and jerk equal to some value;
5. Local position handle equal to position handle;
6. Local new position velocity and acceleration equal to position velocity acceleration
7. **While** the position is true **do**
8. Local the result of new position velocity and acceleration;
9. Set the path position;
10. **If** (no result)
11. **Then** break;
12. **End if**;
13. Change to other script;
14. **End**;
15. Remove velocity acceleration and position;
16. **End**;
17. **Declare** path;
18. Move the path into certain position, velocity , acceleration and jerk.
19. **End**;

3.9.4 Pseudo code of Vision Sensor

The pseudo code below is for the vision sensor

1. Declare path;
2. Declare vision sensor;
3. **While** (simulation running) **do**
4. read the value from vision sensor;

5. **If** (there is a value from vision sensor) **then**
6. centre of mass x is equal to value;
7. centre of mass y is equal to value;
8. **Print** “centre of mass x , centre of mass y”;
9. **End.**

3.9.5 Pseudo Code of Circular Hand Tracking for KUKA Youbot

Refer Appendix B

3.10 Procedure of Simulation in VREP

For the first simulation procedure for generation of circular path is done by using point to point method. The scene for the simulation is similar to set up in Figure 3.19 and Table 3.1 and Table 3.2 but only remove the target in the scene so that the camera not detects anything. The number points are added in the environment and the algorithm is by referring pseudo code 3.9.3. The simulation is observed and the trajectory of robot motion is analysed.

Next, for the simulation of hand motion tracking, the procedures for this project simulation in VREP are as follows:

1. Set up the robot position as shown in Figure 3.19 by referring Table 3.1, Table 3.2 and workspace in Figure 3.20.
2. Develop algorithm for the path movement in associated threaded child script by referring pseudo code 3.9.3.
3. Set the filter for vision sensor by adding selective colour on work image and binary work image and trigger. Since the VREP is version 3.3.2, a blob detection filter is not suitable so it replace with binary work image and trigger. The output result is still same.

4. Develop algorithm for centroid detection in the vision sensor associated threaded child script by referring pseudo code in 3.9.4.
5. Develop algorithm for circular hand tracking motion in youBot associated threaded child script by referring 3.9.5. This part requires some calculations of machine vision and transformation matrix. Refer subchapter 3.4, 3.5 and 3.6 for the calculations.
6. Run the simulation and observe the motion of robot.
7. Analyse the circular path generation and hand tracking motion.

3.10.1 Problems and Issues in Simulation

Generally, when programming a robot there will be several issues that cause the robot unable to follow the given instruction. In VREP simulation, the issues that occurred are as follow:

1) Different coordinate frame

This problem causes the KUKA youBot end effector unable to positioning to the initial target position. This is because all the objects in the environment have their own coordinate system. The display value of target position (x,y) is in the coordinate of vision sensor not referring to target coordinate. Because in the simulator the target is floating so the changes values are y and z. The vision sensor script needs to compute the coordinate in world space of the blob. For that, it is need to perform some calculation. Basically take as input the x or y position of the blob (in image coordinates), the transformation matrix of the vision sensor, the opening angle of the vision sensor, and also the depth location of the blob. This is because the vision sensor gives several types of information, but in order to be able to process that information easily, we must to generate a point in 3D space.

2) Unreachable goal

In simulations, the robot will be given a certain values of x , y and z for inverse kinematic of joint positioning. The robot arm joint will move according to the given (x, y, z) even though it is unreachable. An unusual movement of joint will occur when robot is trying to find the position which is less or more than it reachable workspace. If this situation is happen in real life, it will cause danger to people around it. Therefore, to avoid these things happen, a calculation is needed before inserting values into joint in the inverse kinematic mode.

3.11 Method of Analysis

In order to approach the objective of this project which is to validate parametric equation for circle to create a circular motion and to access the accuracy of robot in terms of Cartesian space, the method to validate the results is by applying the percentage error and accuracy formula as followed for checking the deviation and the accuracy. In circular shape with the simulation using MATLAB, the accuracy is calculated by the difference value of radius between the actual and desired circle as shown in Figure 3.22. However, it is reminded that this accuracy is only to compare the cubic spline method and circumference equation in MATLAB. The cubic spline is to get smooth shape and compared with polygon shape that is made from circle formula.

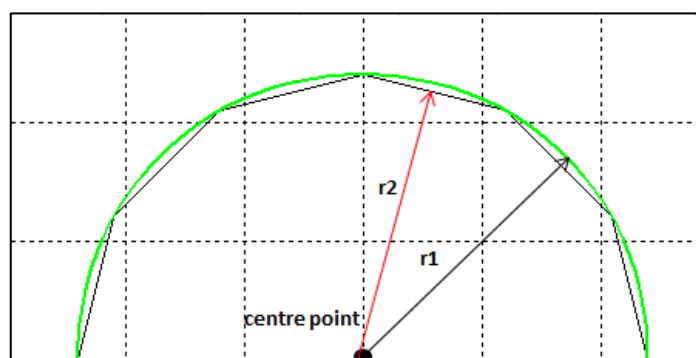


Figure 3.22: Calculation of error percentage

i. Mean:

$$\bar{X} = \frac{\sum_{i=1}^n X_i}{n} \quad (3.12)$$

Where, \bar{X} = The mean

X_i = Each value of data

n = The number of data points

ii. Absolute error:

$$error, E = X_i - X_t \quad (3.13)$$

Where; X_i = observed value

X_t = expected value

iii. Percentage accuracy:

$$\% accuracy = (1 - error) \times 100\% \quad (3.14)$$

The formula 3.12 is to determine the mean of image tracking the centroid of the target using the vision sensor in order to make the robot follow patient hand accurately. From the formula, the accuracy of image tracking can be obtained. The equations 3.13 & 3.14 above will be applied to access the accuracy of robot tracking human hand using visual sensor. It is about calculating the differences between the motion of the robot and human hand to ensure whether the robot motion match with the hand motion or not. This measurement is also to prove that a vision based KUKA youBot can follow and guide rehabilitation exercise with high accuracy.

3.12 Consideration on the Validity of the Simulation

There are several factors that may invalidate the simulation results. In this project, the factor is noise. The present of noise may effects the stability of joint when tracking the

circular motion. In the real world, noise may be occurred from the blob detection. In this project simulation using VREP, the vision sensor detects a red blob which represents the patient hand motion. The background colour of the experiment is contrast with the blob colour so there is no noise in the blob detection. For the real case, there might be other red colour around the vision sensor area of detection such the patient wear a red colour shirt or red hand accessories. Therefore, the joint stability may be decrease if the camera detects other red colour object.

3.13 Discussion on the reliability of data

The measurement of uncertainty is important and necessary in the simulation result. In this project, the uncertainty is in the tracking hand motion result. The vision sensor is the instrument that used to detect the blob. The vision sensor angle position is important because it directly affects the position of the measured image. If the vision sensor have a free angle and not fixed, it will affect the centroid value. Therefore, when the simulation is repeats it not certain that the robot move the same accurate motion. In order to solve the problems, the position and orientation of the camera should be fixed according to Table 3.1.

CHAPTER 4

RESULTS

4.1 Circular motion simulation in MATLAB

A mathematical model will describe the motions of robot trajectory. The existed trajectory equation and circular equation is simulated using MATLAB and the results which shown in Figure 4.1 and Figure 4.2 fit the actual circular motion of the robot. Since the motion is circular, the final position ends same as final position point. By referring to Figure 3.1 in the previous chapter, the radius of circle will change according to the feedback given by the visual sensor. There are two simulations that are done for circular motion which are circle with adjustable radius and smoothness of circle by increasing number of via points. An adjustable circular radius simulation is where the values of radius are assumed to ensure that the circular shape is adjustable to match with the actual radius that will send by feedback from camera. Next, the simulation of smooth circle shape by increasing the number of via points.

4.1.1 Circular Path Generation with Different Radius

The Figure 4.1 below shows a circle with adjustable radius. The maximum value of radius is set to 20cm is because it is assumed as the maximum circular motion that can be done by a patient.

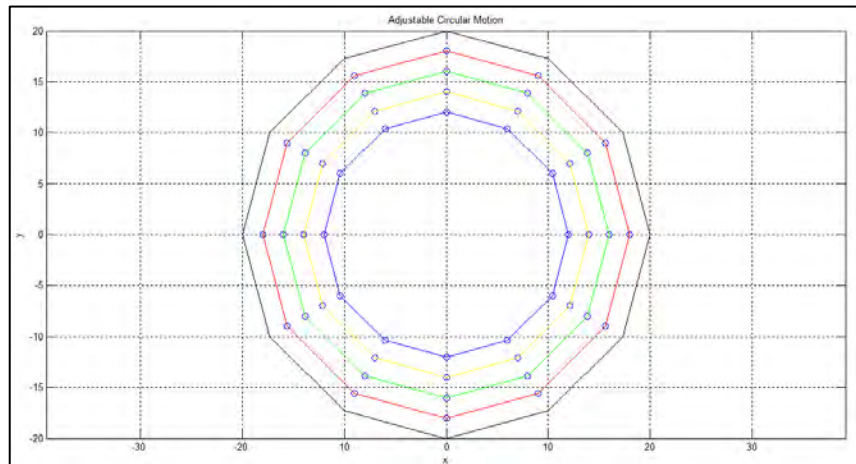


Figure 4.1: Adjustable circular radius from MATLAB simulation

The graph in Figure 4.2 is generated to show the comparison between actual and desired circle. Note that the result in Figure 4.2 below is simulated for circle at radius 12cm.

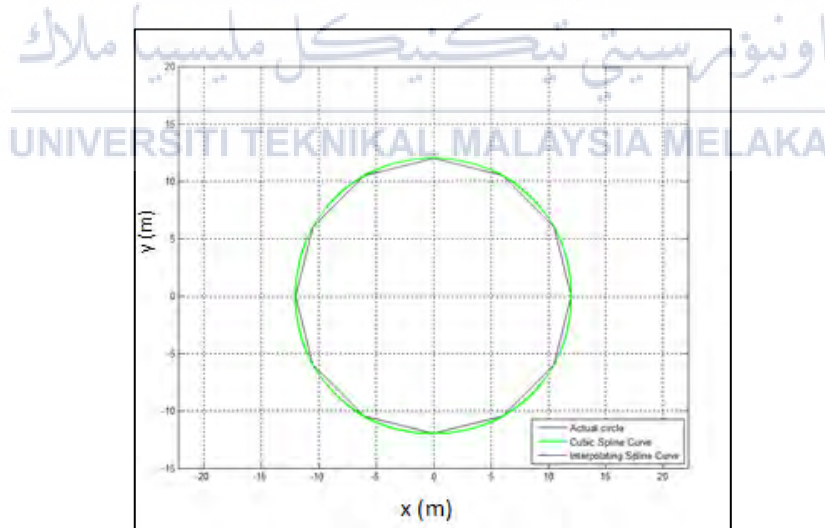


Figure 4.2: Actual and desired circular motion of robot arm

The cubic spline curve represents the desired circular motion. Therefore, the accuracy of the graph can be obtained by using statistical method. The error for each segment of polygon is same since it is circular. Segment is the distance between 2 via points. Therefore, by taking one segment, we can calculate the percentage error and accuracy. The Figure 4.3 shows how the error is determined.

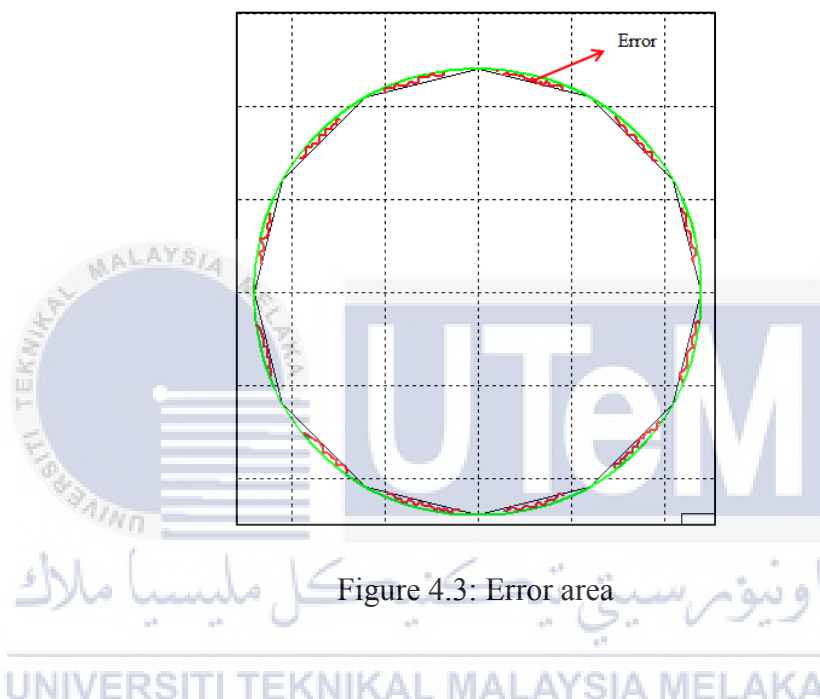


Figure 4.3: Error area

From the calculation using equation 3.13 & 3.14, the absolute error is 0.0434 while the percentage accuracy of creating a circular shape is 95.65%.

4.1.2 Increasing the Number of Points to Points in the Circular Path

In order to get the polygon shape motion smooth, the number of point to point is increased. The graph below shows the differences number of via points of to create a smooth circular motion. The calculation of error is using the equation (3.13).

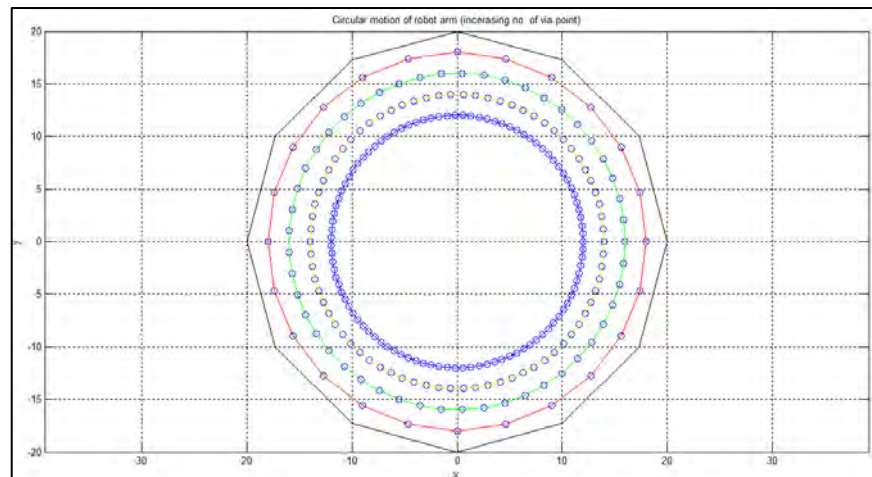


Figure 4.4: Increase number of points

Based on the Figure 4.4, the circular shape is smooth when increasing the number of points. The simulation is done with different value of radius to make us easy to see the differences. The figure 4.5 below shows the relationship between the number of points and smoothness. The error of desired and actual motion is decrease as we increase the number of points. The error calculation is using equation (3.13):

اونيور سیتی تیکنیکل ملیسیا ملاک

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

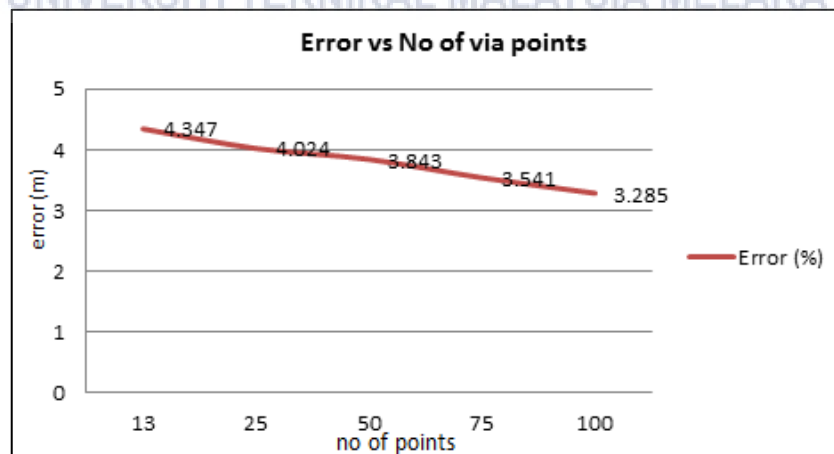


Figure 4.5: Error versus no of point to points

4.2 Simulation of vision based robot in VREP using LUA

Since VREP is already having LUA language so in this project LUA was used to build the algorithm for vision based robot. The simulation is done to determine the accuracy of KUKA youBot to track a circular path using a vision sensor.

4.2.1 Circular Path Generation

Figure 4.6 and 4.7, shows the trajectory generation of KUKA youBot to move from one point to other point. The simulation done for generate just a part of circle shape to shows that KUKA youBot able to do circular motion without tracking and following in order to make the patient exercise their hand by following the robot motion.

Based on the Figure 4.6 and 4.7, acceleration occurred when the robot start to move from the initial point and decelerate when the reaching the final point. Based on Figure 4.6, the acceleration and deceleration is high due to the number of point to point is less. Therefore the generation of a part of the circle shape is not really smooth.

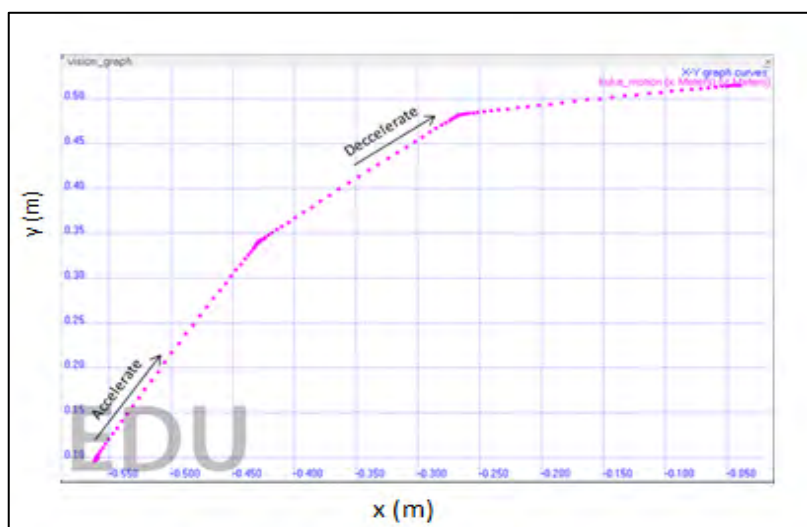


Figure 4.6: Generation of Circular Path using Point to Point

Based on Figure 4.7, the trajectory generation of circle path is smooth when the number of point increases. The acceleration and deceleration of motion is reduce due to the distance between point to point is reduced.

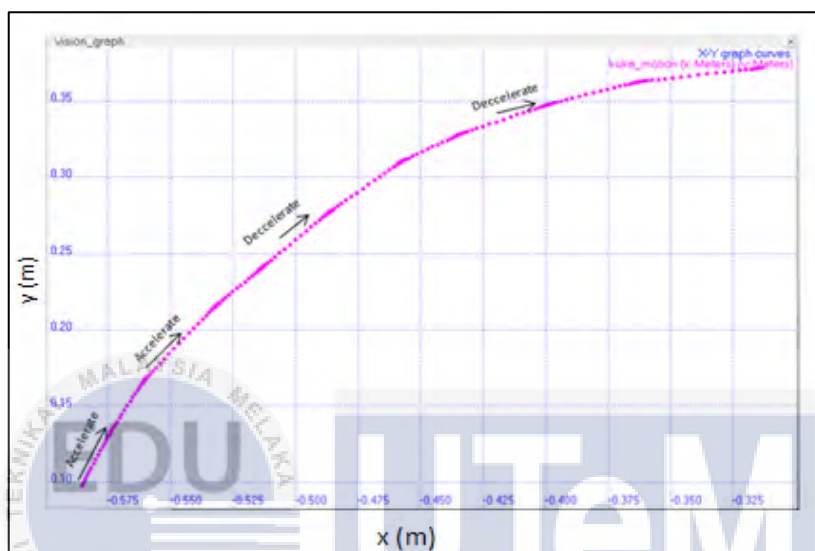


Figure 4.7: Generation of Circular Path by Increasing Number of Point to Point

Therefore, this simulation validates the simulation of circular motion in MATLAB. The KUKA youBot is able to generate a circular path for the hand rehabilitation applications.

4.2.2 Image centre tracking based on Vision Algorithm using blob detection

In this project, a blob detection method is used as a marker to make the robot detect the path. This situation was explained in chapter 3 by referring Figure 3.1. The colour chose for the blob was red colour because in LUA for the visual properties RGB colour range is easier to set. Based on simulation results, the vision sensor will detect the object centre (x, y) to ensure that the robot constantly track the blob in circular. By doing some

calculation of machine vision and apply some mathematical equations as shown in subchapter 3.5, the image centre is obtained.

The Figure 4.8 below shows the values of image centre when the target is moving in circular shape. The desired centre value is (0.5, 0.5). The simulation is valid because the mean values shown that the image is always at centre and it proved the explanation for Figure 3.3. The mean values is obtained by using formula 3.12 and the results is (0.511682291, 0.483396275) for circular motion and (0.512411317, 0.455760301) for the adjustable circular motion.

Based on the results, the image centre value is not more than (0.5906, 0.5795) and not less than (0.3793, 0.3986) for circular shape. Meanwhile, for adjustable circular motion the minimum centre is (0.4086, 0.3463) and the maximum centre is (0.622, 0.551).

From the graph in Figure 4.8 and 4.9, the error occurred in the maximum and minimum value is due to the speed changing. It shows that the end effector is adjusting its position so that the target is keep into the centre of image frame. The full values from start until stop were attached in Appendix C and Appendix D.

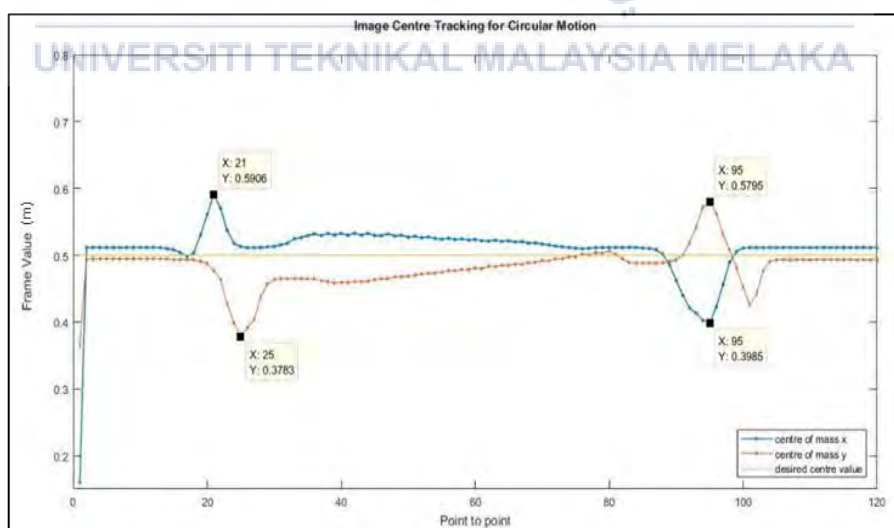


Figure 4.8: Image centre read by vision sensor for circular motion

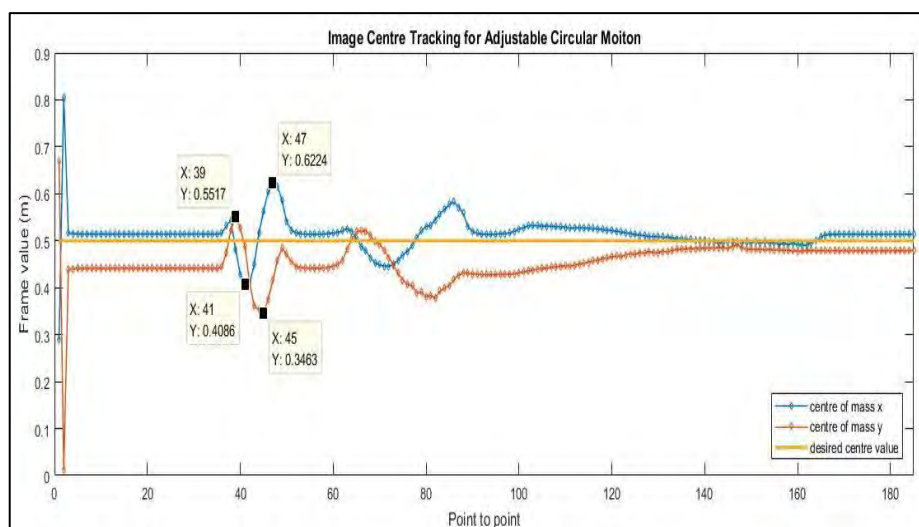


Figure 4.9: Image centre read by vision sensor for adjustable circular motion

From the results, the accuracy of circular hand tracking using KUKA youBot for rehabilitation purpose can be determine using the equation 3.13&3.14. The accuracy is 96.67% for circular motion and 91.15% for adjustable circular motion.

4.2.3 Image Tracking Using Vision Based Robot

Figure 4.10 shows the results of KUKA youBot motion and hand motion. In simulation using VREP, the path is not really an exact circle shape because it is created using segmented path. This is because if using the circular path the radius is too large and not suitable for a rehab purpose. The graphs in Figure 4.10 show that the motion of KUKA following the target is not passing through the point smoothly. If seen clearly, the KUKA youBot motion generates segmented line in circle path. The trajectory generation of via point is not necessarily passing through the point but the end effector only pass close the point.

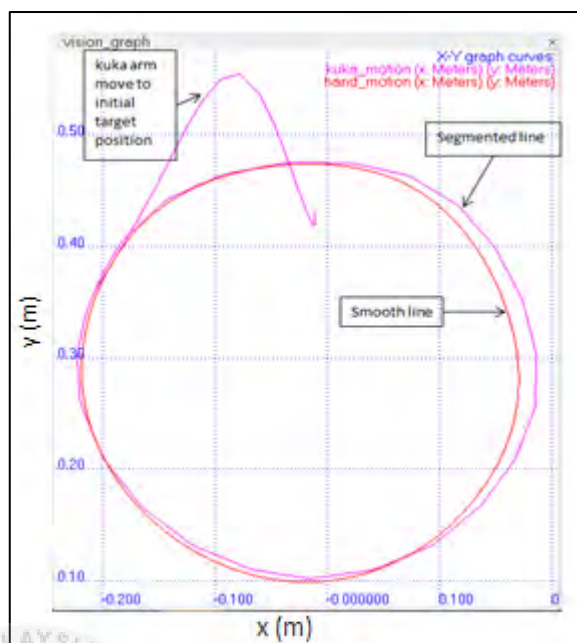


Figure 4.10: KUKA motion versus hand motion

Next, the graph in Figure 4.11 is based on the motion of end effector. The position of robot motion is not same as the desired circle motion. It is slightly different from the desired path. This is because the limitation of joint and the input from vision where the vision needs to ensure that the image is always on the centre of frame. Not only that, the movement of robot arm is due to the inverse kinematic of KUKA youBot.

The error is considered as systematic error because it is not reduced when the observation is average. This is due to the KUKA youBot is already in inverse kinematic mode by the setting in VREP. By imposing the inverse kinematic, it is not known what is the value of joint that calculated by the computer. The error may be reduced only if developing a new inverse kinematic value for the robot because we can calculate the end effector orientation and position based on the desired position. Therefore, it have slightly error in the positioning but it is not considered as a critical error because the robot is still moving in the desired circular path shape and the accuracy is still high. Next, the desired motion is in continuous motion because the end effector follows a prescribed path in three dimensional spaces and the speed of motion is varying along the path which causes the motion in segmented shape.

The Figure 4.11 below shows the results with varying velocity of the target. The motion is smooth when the velocity is slow. This simulation is to represent the motion of patient hand when doing rehab exercise. The first motions, as the hand move fast the robot keep follow it but produced a polygon motion. When the speed is suddenly slow, the robot also getting slow and guide the patient hand to keep moving. The motion produced also smooth due to the slow motion and to be as a guidance of the exercise. Then, when the strength of muscle increase little bit, the KUKA will follow it again. This situation can be seen in the VREP simulation.

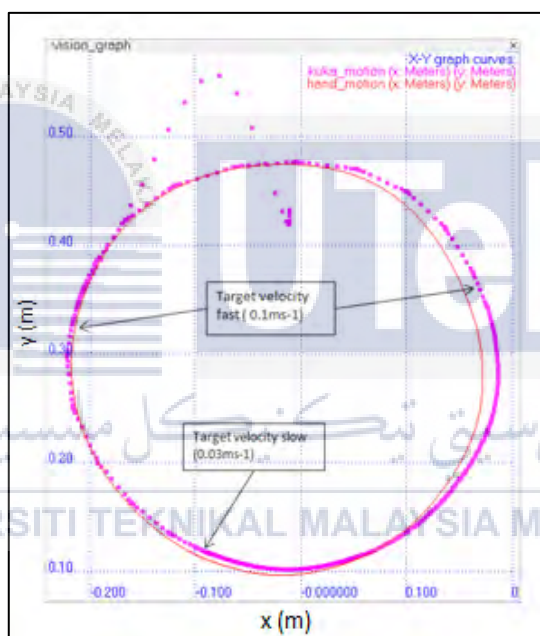


Figure 4.11: Motion of KUKA youbot in circular path with varying velocity of target

The differences error between KUKA youBot motion and hand motion can be calculated by taking the maximum differences at the cornering of path as shown Figure 4.12. The cause of error might be jerk. The calculation is based on formula 3.13 and 3.14. The absolute error is 0.03 and the accuracy is 97% of tracking a circular path hand motion.

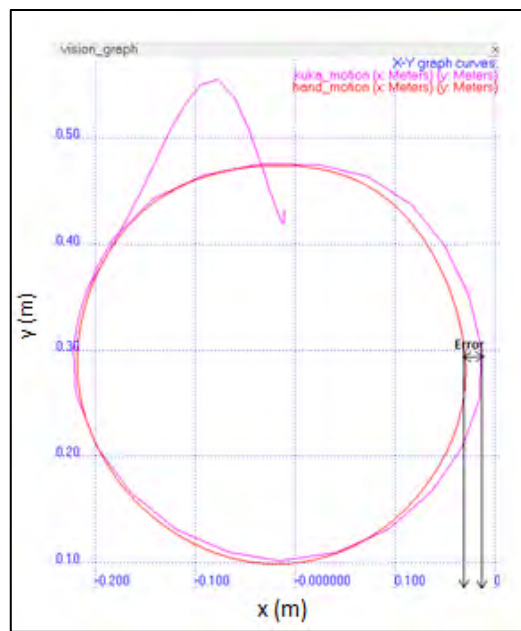


Figure 4.12: Maximum error region for circular path

Next, Figure 4.13 below shows the trajectories of each joint position for 5DOF KUKA youBot. The motion of tracking the object only require the positioning and orientation of joint 0 and joint 2. The graph shows that the joint position is smooth when the motion is slow. This is because the jerk minimizes when velocity is slow.

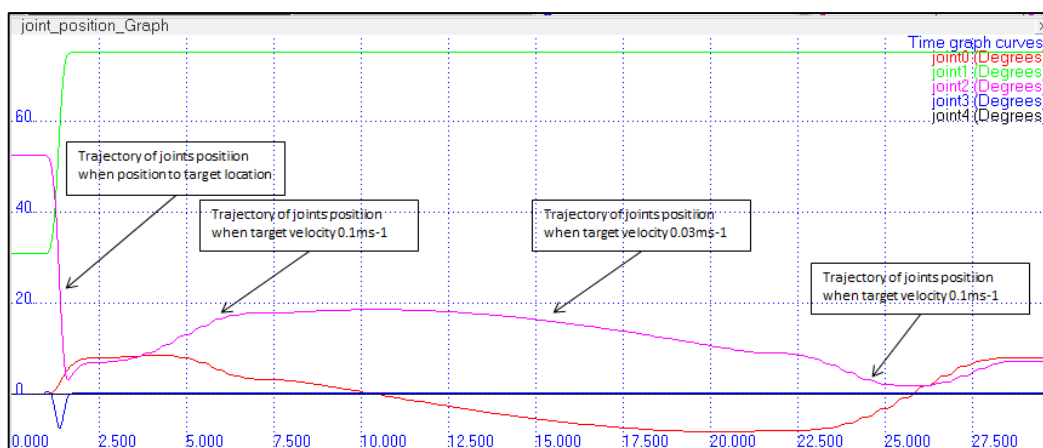


Figure 4.13: Joint position for circular path

Figure 4.14 below shows the joints velocity trajectories. The velocity is close to zero when the motion is slow. The motion is more accurate because the time is longer when the speed is slow. Velocity is needed for making the robot as guidance in the situation where the hand motion is slow due to weak muscle.

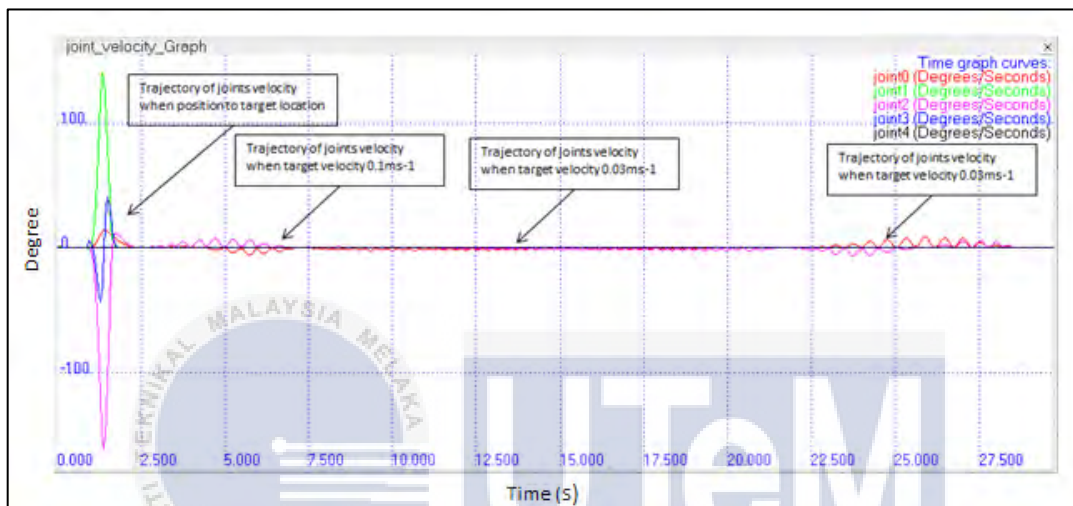


Figure 4.14: Joint velocity for circular path

Figure 4.15 below shows the joints acceleration trajectories. The accelerations can be smooth when the time at the high target velocity is longer. Since this project does not involve in grasping of pick and place, so the trajectories velocity and accelerations is not really concern. A design of new torque controller is needed if the robot needs to grasp human hand because for that case, an accurate and precise motion is needed.

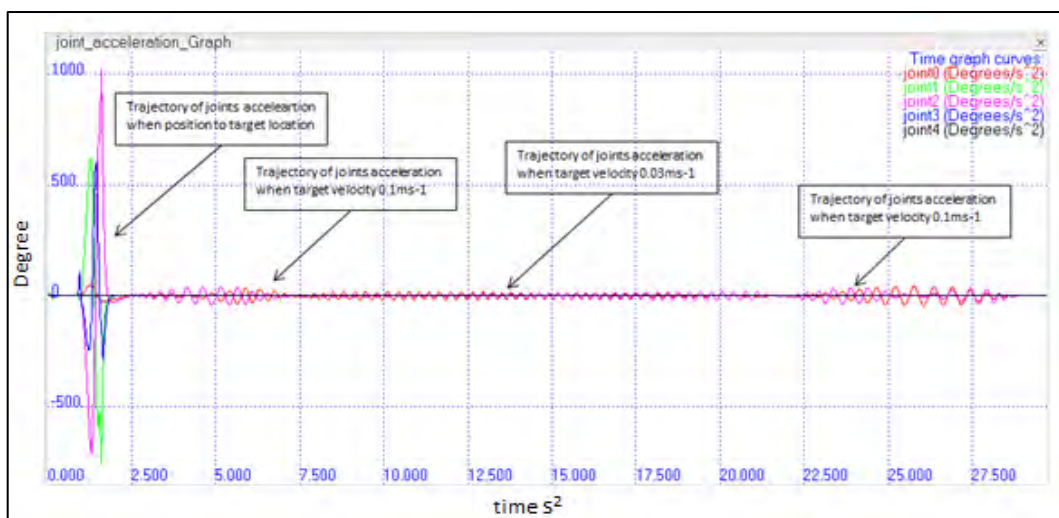


Figure 4.15: Joint acceleration for circular path

4.2.4 Image tracking for adjustable radius

In this simulation, a swirling circle shape is used to observe the motion of robot in different radius. Swirl circle is used as a path that can be adjustable because if think logically, the movement of the human hand is not necessarily in the form of a precise circle because the human eye is also unable to calculate the exact values of radius when they move their hands unless with the help of a device that can move human hand based on the radius has been set. In this project, the motion is done by rehab patients. Therefore, they may not be able to make a precise circular motion because the condition of weak muscle and they are also unable to keep the hand motion in same speed in every radius. The larger the radius, the slower the hand motion.

Thus, in this simulation there are three types of velocity of target in order to represent the ability of hand muscle motion which are 0.1ms^{-1} , 0.07ms^{-1} , and 0.03ms^{-1} . The Figure 4.16 shows the KUKA motion slightly different than the desired motion. At the first velocity 0.1ms^{-1} , the KUKA youBot motion is more to polygon shape due to the high speed and small radius create high jerk. For the second velocity 0.07ms^{-1} , the motion of KUKA is smooth compare to the first velocity. Lastly, the third velocity 0.03ms^{-1} shows

the KUKA youBot motion smooth compare to the previous velocity. The slower the velocity, the smoother the KUKA youBot motion. The slow motion of KUKA youBot will give motivation to patient to keep moving the hand. If seen in the simulation using VREP, the high speed of the target make the robot try to follow it while when speed is getting slower, the robot is trying to guide the target motion. It stops and wait the target for a while when the speed is changing.

Next, this simulation proves that KUKA youBot able to guide and motivate the patient in doing the exercise even the radius and velocity is different. For the accuracy of tracking the hand, same with the previous simulation, the error often occurred when cornering. However, the error is not considered critical because it is in small value and the accuracy is high. The calculation of error can be determined by equation 3.13 & 3.14 method shown in Figure 4.17. From the calculation it shows that KUKA youBot has accuracy of 97% with absolute error 0.03. The error shown in graph is considered as systematic error. This is because the error is not reduced even the observation is average.

In addition, since there are no obstacles within the environment for this project, the manipulator moves freely within it workspace. Figure 4.16 shows the overall planned motion of the robot in Cartesian space. The absence of obstacles makes the algorithm drive the manipulator toward the desired path and goal position. That's why the motion is similar to a human hand for rehabilitation.

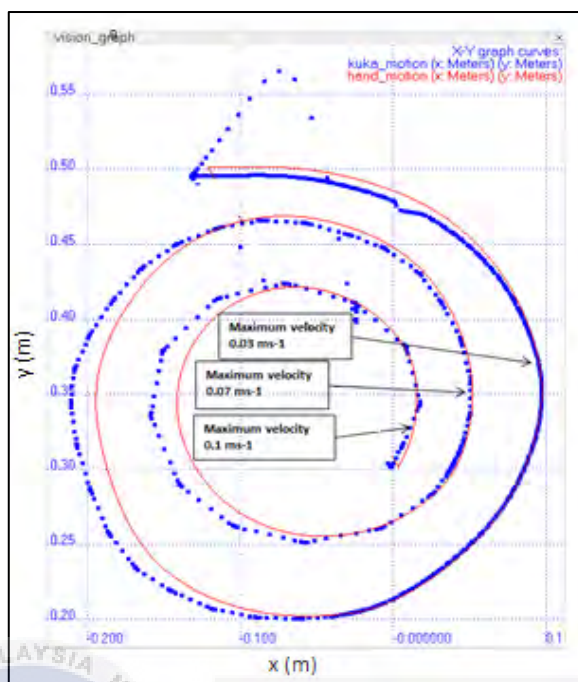


Figure 4.16: Motion of KUKA in swirling circular path

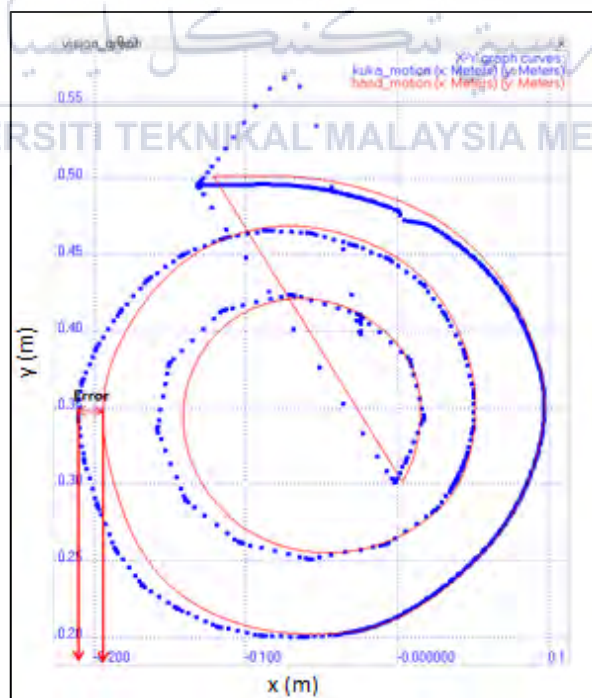


Figure 4.17: Maximum error region in swirling circle shape

The Figure 4.18 below shows the joints position trajectories. During tracking the object only joint 0 and joint 2 is involve in positioning and orientation. The graph shows that at the first velocity 0.1ms^{-1} , both joint 0 and joint 2 curves are oscillating due some jerk occurred because of the high velocity in small radius. Then the oscillation is decreases when the velocity is decrease and gives a smooth curve.

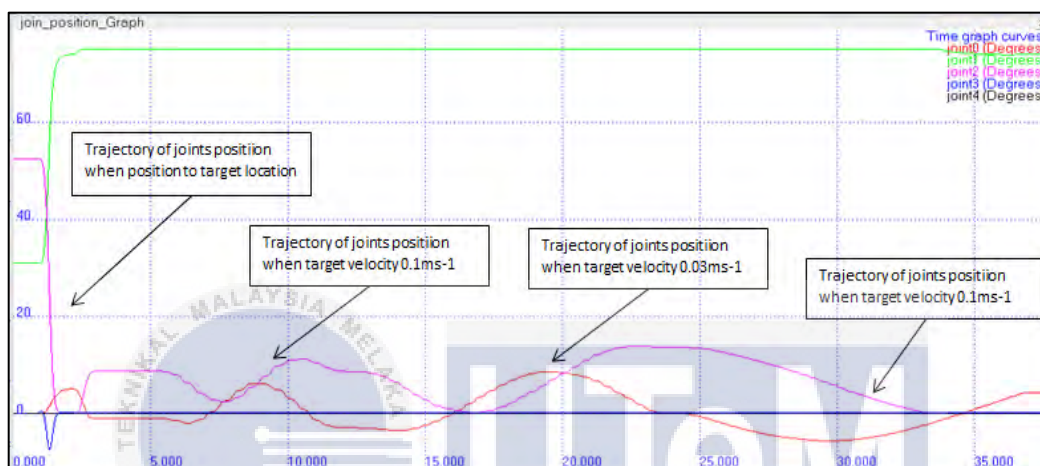


Figure 4.18: Joint position for swirling circular path

In this project, the velocity is not really concern because the robot is not coupled with the patient hand but it just track and follow only. Figure 4.19 below shows the trajectory for each joint velocity at the initial motion where the goal is the centroid of red blob. The trajectories are less smooth compared to Figure 4.13 due to the algorithm is concern on keep the image on the centre of vision sensor frame for a long path. Therefore, it cannot produce smooth trajectories even though there are no obstacles around the environment.

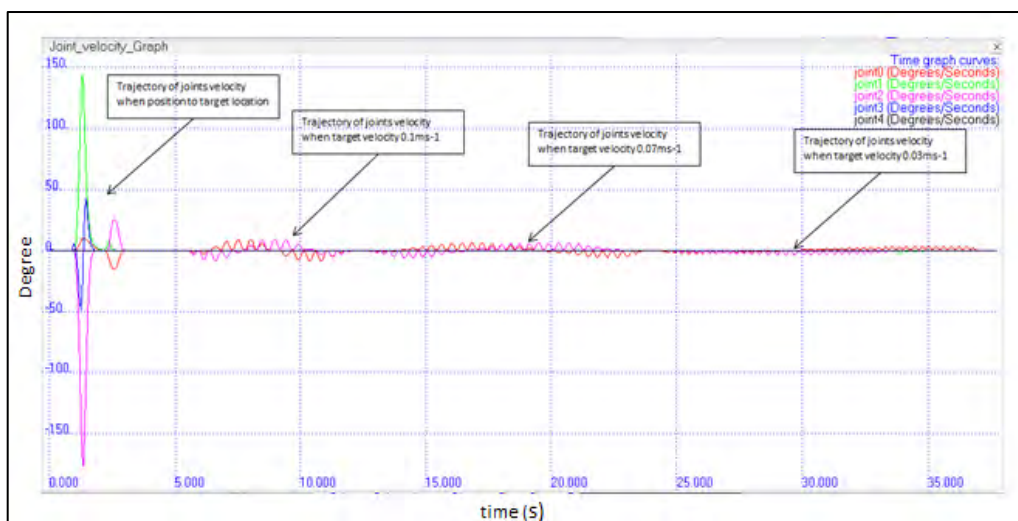


Figure 4.19: Joint velocity for swirling circular shape

Figure 4.20 below shows the joints acceleration trajectories. The accelerations can be smooth when the time at the high target velocity is longer.

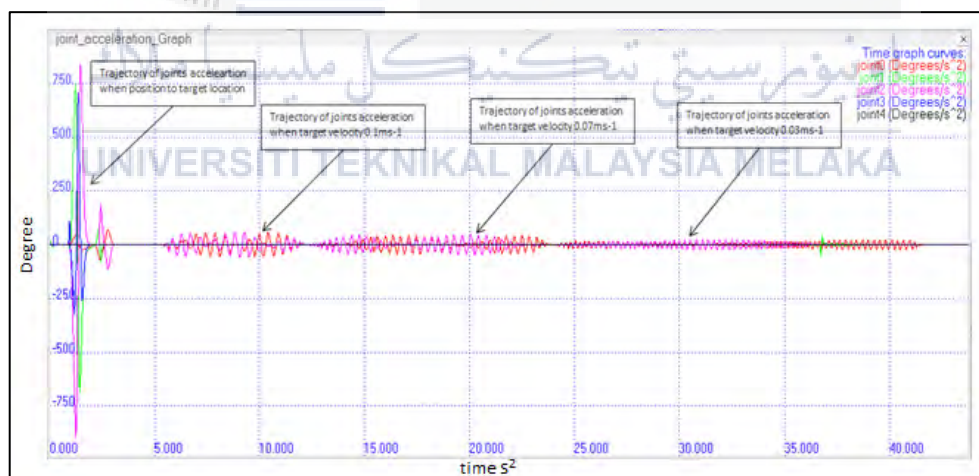


Figure 4.20: Joint acceleration for swirling circular shape

Since this project does not involve in grasping of pick and place, so the trajectories velocity and accelerations is not really concern. A design of new controller and new inverse kinematic is needed if the robot needs to grasp human hand because for that case, an accurate and precise motion is needed.

CHAPTER 5

CONCLUSION AND RECOMMENDATION

5.1 Conclusion

In this project, a vision based robot algorithm for circular hand tracking problem was developed to provide accuracy of robot to generate circular path and tracking hand motion. A point to point circular motion was developed by combining the trigonometry equations with formula of circumferences in MATLAB. Validity of these formulas was determined when the error of the desired and the actual circle is 0.04347 which give high accuracy of 95.65%. The smoothness of circular shape can be improved by increasing the number of via points. Therefore, if the number of points increases, the quality of producing a smooth circle path is high. This simulation is validated with the simulation in VREP. In VREP, it shows that the smoothness of circular path increases when the number of points increases.

For the hand tracking part, the results show that KUKA youBot is able to track and follow hand motion when varying the speed and radius. By using the blob detection and imposing the inverse kinematic, the robot achieved high accuracy when doing hand tracking. Based on the results, the accuracy of vision based robot tracking a circular hand motion in various speed and radius is 97% with absolute error of 0.03m. This error is caused by the jerk and the available inverse kinematic. It may be minimized if design a new controller and new inverse kinematic. Therefore, a vision based robot is reliable for circular hand tracking problem for rehabilitation purposes.

5.2 Recommendation

There are plenty research opportunities for future work. This approach can be extended by developing a new control method to produce a precise motion and also add an algorithm of saving the result of rehabilitation everytime the robot is used for the exercise so that the therapist can follow up the conditions of muscle strength. To further enable robotic assistance service for rehabilitation purposes, it is also important to balance trade off between computational cost, power consumption and performance of vision based hand tracking in long term use.



REFERENCES

- [1] Veerapen, K., Wigley, R. D., & Valkenburg, H. (2007). Musculoskeletal pain in Malaysia: A COPCORD survey. *Journal of Rheumatology*, 34(1), 207–213. <https://doi.org/0315162X-34-207> [pii]
- [2] Ratio of occupational therapist. Retrieved 25th September 2016 from <http://www.thestar.com.my/lifestyle/health/2014/10/16/not-enough-occupational-therapists-in-malaysia>
- [3] B. H. Dobkin, "The economic impact of stroke," *Neurol.*, vol. 45, 1995.
- [4] P. S. Lum, C. G. Burgar, P. C. Shor, M. Majmundar, and M. V. d. Loos, "Robot-assisted movement training compared with conventional therapy techniques for the rehabilitation of upper-limb motor function after stroke," *Arch Phys Med Rehabil*, vol. 83, pp. 952-959, 2002.
- [5] Robotic Rehabilitation. Retrieved 23rd October 2016 from https://en.wikipedia.org/wiki/Rehabilitation_robotics.
- [6] Won K.S, Jong S.K, Z. Bien Visual Servoing based on Space Variant Vision for Human Robot Interaction in Rehabilitation Robots. 22nd Annual EMBS International Conference, July 23-28,2000, Chicago IL.
- [7] Tanaka, Y. (2015). Robot-Aided Rehabilitation Methodology for Enhancing Movement Smoothness by Using a Human Trajectory Generation Model With Task-Related Constraints. *Journal of Human-Robot Interaction*, 4(3), 101.
- [8] Won K.S, Jong S.K, Z. Bien Visual Servoing based on Space Variant Vision for Humam Robot Interaction in Rehabilitation Robots. 22nd Annual EMBS International Conference, July 23-28,2000, Chicago IL.
- [9] D. Ravipathi, P. Kareddi, A. Patlola. Real-time Gesture Recognition and Robot control through Blob Tracking. *IEEE Students' Conference on Electrical, Electronics and Computer Science*, 2014.

- [10] B. Busam, M. Esposito, S.C. Rose, N.Navab, B Frisch. A Stereo Vision Approach for Cooperative Robotic Movement Therapy. *IEEE International Conference on Computer Vision Workshops* 2015.
- [11] John J. Craig, Introduction to Robotics Mechanical and control. 3rd Edition. United States of America: Pearson Prentice Hall, 2005.
- [12] Dong, S., Fengqi, Z., & Jun, Z. (2003). Trajectory Planning and Control for Robot Manipulator. *Simulation*, (August), 1–5.
- [13] RK Mittal & IJ Nagrath, Robotics and Control, 1st Edition. Tata McGraw-Hill, 2004.
- [14] K. Yamada, N. Hara, & K. Konishi (2011). Circular Motion Generation for Mobile Robot using Limit Cycle System Application to a Circular Formation Control. *SICE Annual Conference 2011*, September 13-18, 2011, Waseda University, Tokyo, Japan.
- [15] B. Lee, S. Lee, & G. Park (1999). Trajectory Generation and Motion Tracking Control for the Robot Soccer Game. International Conference on Intelligent Robots and Systems 1999.
- [16] Ramesh Jain, Rangachar Kasturi, Brian G. Schunck, Machine Vision, McGraw-Hill, 1995.
- [17] Computer vision. Retrieved on 16th February 2017 from http://www.societyofrobots.com/programming_computer_vision_tutorial_pt3.shtml.
- [18] Coordinate System. Retrieved on 30th April 2017 from <http://developcenter.robotstudio.com/blobproxy/devcenter/RobotStudio/html/4eac08e9-c42c-446f-bbd4-228e523dd2d5.htm>
- [19] World coordinate system. Retrieve on 18th February 2017 from <https://www.scratchapixel.com/lessons/3d-basic-rendering/computing-pixel-coordinates-of-3d-point/mathematics-computing-2d-coordinates-of-3d-points>.
- [20] Transformation Matrix. Retrieved on 30th April 2017 from <http://polymathprogrammer.com/2008/09/01/cartesian-coordinates-and-transformation-matrices/>.

APPENDIX A

Pseudocode for point to point circular generation using youBot

1. Declare Inverse kinematic function;
2. **If** (inverse kinematic is for position only) then
3. Set the object position;
4. **End if**;
5. **If** (position and orientation) then
6. handle the orientation;
7. **else**;
8. handle the inverse kinematic;
9. **End if**;
10. **for** i equal to some value **do**
11. set the joint mode
12. **end** ;
13. Set the inverse kinematic mode in position and orientation
14. **end**;
15. Declare function forward kinematic mode;
16. handle the orientation;
17. **For** i equal to some values **do**
18. set the joint mode;
19. **end**;
20. set the inverse kinematic mode in position only;
21. **end**;
22. Declare function open gripper
23. Get the gripper communication
24. **end**;
25. Declare function close gripper;

26. Get the gripper communication;
27. **end;**
28. Declare function of gripper target when vehicle is moving;
29. Set the vehicle as the reference for gripper target;
30. **end;**
31. Declare function set gripper target fixed to world;
32. Set the world as reference for gripper target;
33. **end;**
34. Declare function of kuka vision sensor;
35. Set forward kinematic mode;
36. **end;**
37. Declare gripper target;
38. Declare gripper tip;
39. Declare vehicle reference;
40. Declare vehicle target;
42. Declare armjoint value;
43. **For** i = some values **do**;
44. Declare armjoint [i + 1];
45. **end;**
46. Declare inverse kinematics;
47. Declare inverse kinematics with orientation;
48. Declare the communication for gripper;
49. Set inverse kinematic speed value;
50. Set inverse kinematic acceleration value;
51. Set inverse kinematic jerk value;
52. Set forward kinematic speed value;
53. Set forward kinematic acceleration value;
54. Set forward kinematic jerk value;
55. Set gripper target moving with vehicle
56. Set inverse kinematic mode;
57. Set inverse kinematic mode for position and orientation;
58. Let p equal to target position;
59. Move the joint to p value;
60. **End.**

APPENDIX B

Pseudocode for circular hand tracking using youBot

1. Declare Inverse kinematic function;
2. **If** (inverse kinematic is for position only) then
3. Set the object position;
4. **End if**;
5. **If** (position and orientation) then
6. handle the orientation;
7. **else**;
8. handle the inverse kinematic;
9. **End if**;
10. **for** i equal to some value **do**
11. set the joint mode
12. **end** ;
13. Set the inverse kinematic mode in position and orientation
14. **end**;
15. Declare function forward kinematic mode;
16. handle the orientation;
17. **For** i equal to some values **do**
18. set the joint mode;
19. **end**;
20. set the inverse kinematic mode in position only;
21. **end**;
22. Declare function open gripper
23. Get the gripper communication
24. **end**;

25. Declare function close gripper;
26. Get the gripper communication;
27. **end;**
28. Declare function of gripper target when vehicle is moving;
29. Set the vehicle as the reference for gripper target;
30. **end;**
31. Declare function set gripper target fixed to world;
32. Set the world as reference for gripper target;
33. **end;**
34. Declare function of kuka vision sensor;
35. Set forward kinematic mode;
36. **end;**
37. Declare gripper target;
38. Declare gripper tip;
39. Declare vehicle reference;
40. Declare vehicle target;
41. Declare vision sensor;
42. Declare target;
43. Declare armjoint value;
44. **For** i = some values **do**;
45. Declare armjoint [i + 1];
46. **end;**
47. Declare inverse kinematics;
48. Declare inverse kinematics with orientation;
49. Declare the communication for gripper;
50. Set inverse kinematic speed value;
51. Set inverse kinematic acceleration value;
52. Set inverse kinematic jerk value;
53. Set forward kinematic speed value;
54. Set forward kinematic acceleration value;
55. Set forward kinematic jerk value;
56. Set gripper target moving with vehicle
57. Set inverse kinematic mode;

58. Open the gripper;
59. **While** (simulation running) **do**
60. read the value from vision sensor;
61. **If** (there is a value from vision sensor) **then**
62. centre of mass x is equal to value;
63. centre of mass y is equal to value;
64. **Print** “centre of mass x , centre of mass y”;
65. **End.**
66. Get the transformation matrix for target relative to vision sensor;
67. Get the transformation matrix for target relative to the world;
68. Set inverse kinematic mode for position and orientation;
69. Let p equal to transformation matrix value of target relative to world;
70. Move the joint to p value;
71. **End.**



APPENDIX C

Image tracking values for circular motion

centre if mass x	centre of mass y
0.159940436	0.364619315
0.511367798	0.493942261
0.511520386	0.494430542
0.511490762	0.494613647
0.511473656	0.494613647
0.511457324	0.494601995
0.511457324	0.494601995
0.511457324	0.494601995
0.511457324	0.494601995
0.511520386	0.494601995
0.511520386	0.494601995
0.511490762	0.494430542
0.51126802	0.494430542
0.509705544	0.494232178
0.508067906	0.493942261
0.503951132	0.49347946
0.498286486	0.493499637
0.50336051	0.492941588
0.530118167	0.490761638
0.560553551	0.487889111
0.59064877	0.476577401
0.570512474	0.463922054
0.537195444	0.427107096
0.517821431	0.399159312
0.513073444	0.378255218
0.511233747	0.39162457
0.51133734	0.403433233
0.511379063	0.437552422
0.512572408	0.45608145
0.513108194	0.464121372
0.515598238	0.465192974
0.517676771	0.465144217
0.524587154	0.465144217

0.526337385	0.464986026
0.529252052	0.464752614
0.531843841	0.464272439
0.52919203	0.461642981
0.532421887	0.460767657
0.530094981	0.458684891
0.532552063	0.459430069
0.529841602	0.458984375
0.532828271	0.460439384
0.529934645	0.46042183
0.532617867	0.461157382
0.529513896	0.463461131
0.529236853	0.464517146
0.5318138	0.465184569
0.528861403	0.467283487
0.529773414	0.468225658
0.526885748	0.468788922
0.528263628	0.469816566
0.526127279	0.471998483
0.527398407	0.472313136
0.525329351	0.473569065
0.523727357	0.47471714
0.525466025	0.476722479
0.523409903	0.47645548
0.524420977	0.478856206
0.522495568	0.478556901
0.523562193	0.481054008
0.521463513	0.480343193
0.520994365	0.483327687
0.522605479	0.482742757
0.520507813	0.485076487
0.521470249	0.484623462
0.519675374	0.486928016
0.520318329	0.486441344
0.51790005	0.489835352
0.518233955	0.489037305
0.51666373	0.491619319
0.515436232	0.491319448
0.513679087	0.495088041
0.512629211	0.494436115
0.511614799	0.497549593
0.510510027	0.497895539
0.509743333	0.501799524
0.51047051	0.500725031
0.511269748	0.504154027
0.511377156	0.503067136
0.511645317	0.506325781
0.511674881	0.500935078

Statistical Analysis:

- Minimum tracking centre value = (0.398548663, 0.378255218)
- Maximum Tracking centre value = (0.59064877, 0.579518914)
- Average mean = (0.511682291, 0.483396275)



0.514139652	0.440856487
0.515644312	0.443688124
0.531606019	0.473862737
0.54366231	0.523223341
0.481461316	0.551667094
0.428925574	0.526399374
0.408593744	0.48861137
0.408690095	0.411601454
0.448830128	0.361029178
0.51636219	0.353812426
0.561703742	0.346307337
0.604154229	0.375447482
0.622395813	0.414493412
0.61623913	0.459247291
0.585830092	0.484726548
0.539672554	0.47005865
0.523232937	0.455647975
0.516187191	0.445428818
0.515173495	0.442515314
0.513952315	0.441849023
0.513877988	0.441661417
0.513900816	0.441454947
0.513877988	0.441661417
0.514274359	0.442076713
0.515194237	0.442825824
0.51597923	0.445774943
0.51786691	0.449414551
0.520774126	0.457642823
0.525558054	0.481625408
0.520122826	0.503375888
0.504972577	0.518422723
0.488270462	0.520955622
0.479067355	0.520767152
0.464127421	0.512020946
0.452846766	0.49839744
0.448753178	0.493039161
0.445814729	0.480011165
0.445478499	0.463349849
0.449627042	0.447982877
0.458608985	0.430666775
0.469654709	0.416169971
0.477499992	0.40693751
0.489195466	0.404712439
0.506372809	0.388510793
0.521344841	0.390694767
0.530217409	0.3804259
0.53271842	0.384012222
0.544544458	0.377698511

0.557579637	0.39321664
0.566617668	0.398777574
0.577268183	0.404868066
0.582289219	0.417379141
0.571268201	0.426755488
0.55852896	0.431631356
0.530862153	0.430749476
0.519457042	0.429047287
0.515478611	0.428434193
0.513658047	0.428034514
0.513658047	0.428034514
0.513584137	0.428071439
0.513658047	0.428062201
0.514849305	0.4284316
0.515136719	0.428471416
0.515993536	0.428701729
0.519465864	0.429425836
0.525537312	0.43147558
0.527334452	0.433120549
0.531905651	0.436142504
0.532558322	0.437771022
0.531894803	0.439415753
0.531418622	0.440872312
0.531116605	0.441673011
0.530556202	0.44346866
0.530103922	0.444538891
0.529301703	0.445965141
0.528751135	0.446882665
0.526311159	0.445702165
0.527609229	0.448346466
0.527460635	0.450173408
0.526945353	0.452104717
0.526257038	0.453683048
0.526116908	0.457620144
0.524552166	0.459260613
0.524033189	0.461523443
0.522769868	0.463309139
0.521604538	0.465524852
0.520065784	0.467382818
0.518369675	0.46597451
0.517063081	0.469851494
0.51486659	0.472573131
0.513143003	0.470844507
0.511749685	0.474892825
0.510972738	0.473205417
0.50870508	0.477119029
0.508556068	0.474876434
0.508381248	0.474598646

0.51380676	0.478451103
0.51380676	0.478451103
0.51380676	0.478451103
0.51380676	0.478451103
0.51380676	0.478451103
0.51380676	0.478451103
0.51380676	0.478451103
0.51380676	0.478451103
0.51380676	0.478451103
0.51380676	0.478451103

Statistical Analysis:

- Minimum centre tracking value = (0.408593744, 0.346307337)
- Maximum centre tracking value = (0.622395813, 0.551667094)
- Average Mean = (0.512411317, 0.455760301)

