Gesture Recognition Aplication based on Dynamic Time Warping (DTW) FOR Omni-Wheel Mobile Robot

Indra Adji Sulistijono, Gama Indra Kristianto

Indra Adji Sulistijono is with the Department of Mechatronics Engineering, Electronics Engineering Polytechnic Institute of Surabaya (EEPIS), EEPIS Campus Sukolilo, Surabaya 60111, Indonesia. (Tel: +62-31-594-7280 ext. 4186; Fax: +62-31-594-6114; Email: indra@eepis-its.edu).
Gama Indra Kristianto is with the Department of Mechatronics Engineering, Electronics Engineering Polytechnic Institute of Surabaya (EEPIS), Indonesia. (Email: gambrenkhom@gmail.com).

***Abstract* — This project presents of the movement of omni-wheel robot moves in the trajectory obtained from the gesture recognition system based on Dynamic Time Warping. Single camera is used as the input of the system, which is also a reference to the movement of the omni-wheel robot. Some systems for gesture recognition have been developed using various methods and different approaches. The movement of the omni-wheel robot using the method of Dynamic Time Wrapping (DTW) which has the advantage able to calculate the distance of two data vectors with different lengths. By using this method we can measure the similarity between two sequences at different times and speeds. Dynamic Time Warping to compare the two parameters at varying times and speeds. Application of DTW widely applied in video, audio, graphics, etc. Due to data that can be changed in a linear manner so that it can be analyzed with DTW. In short can find the most suitable value by minimizing the difference between two multidimensional signals that have been compressed. DTW method is expected to gesture recognition system to work optimally, have a high enough value of accuracy and processing time is realtime.**

***Keywords* : dynamic time warping , gesture recognition, omni-wheel robot**

I. INTRODUCTION

Some models of mobile robots have been developed with the specific navigation capabilities. Some of them as a vehicle capable of climbing stairs, assist assembly, inspection, and work in hazardous industrial environments, the mapping space for the robot maid / cleaning, Indonesia Intelligent Robot Contest (KRCI), and so on. Omni-wheel mobile robot is currently more popular, and many applications (Helder, 2006). For example, in contests robots and industrial automation. Mobile robot has two types, non-holonomic and holonomic. Non-holonomic robot is a robot that could do to change the direction of movement directly. This type of robot must perform a complex set of movements to change direction. While the holonomic robot capable of performing movements in all directions directly and do not require complex movements to achieve a particular goal.

Great effort has developed an intelligent and natural interfaces between users with computer systems. This is done by utilizing the information mode (visual, audio, pen, etc.) either used separately or combined. At the end of this project, focused on the visual sensory information to recognize human activity in the form of hand movements. The method used is the Dynamic Time Warping is used for time alignment and normalization by computing a temporal transformation allowing matching of the two signals (Andrea C, 2001). One important aspect of the mobile robot is controlling the movement or the ability to navigate an automated and reliable. Research on mobile robot navigation has been widely performed, both reactive navigation and global path planning. A number of other researchers have also been conducting research on mobile robot control by using Dynamic Time Warping. In particular control systems using Dynamic Time Warping is developed to obtain matching data with mengukut eucledian distance between the input signal and the signal database (Bima S, 2009).

II. CONSTRUCTION

A. Mechanical Construction

OMR is a classification of a mobile robot that can be modeled by holonomic and non-holonomic. In this paper, OMR modeled by holonomic systems. When the robot is said to be holonomic because travel every direction under any orientation. This capability is widely known as omnidirectional mobility. A variety of omnidirectional design have been developed. They Provide excellent mobility, especially in areas with static

or dynamic conguested obstacles, Such as offices, workshops, warehouses and hospitals.

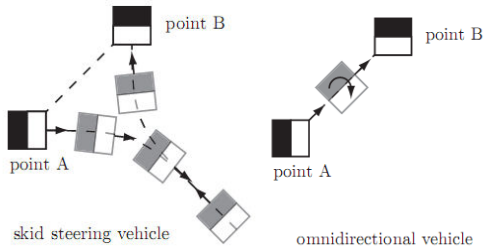


Figure 1. Non-holonomic mobility versus omnidirectional mobility

Omnidirectional Mobile Robot is designed to have three wheels that have the same angle between each other.

Figure 2. Design OMR without frame

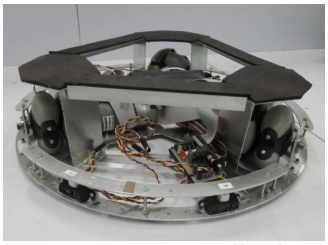


Figure 3. OMR Realization

If we want to prescribe the robot's movements in the environment, we need to know how these variables relate to the primary variables we can control: the angular positions and velocities of the wheel shafts. Therefore, a kinematical model of the robot has to be developed.

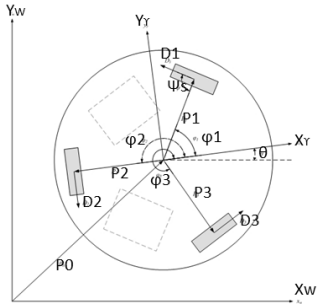


Figure 4. Geometry used for deriving OMR kinematic model\

For the reason of simplification, the first step is to determine the relationship between the individual wheel velocities and the robot velocity. A vector v_w containing the translational velocities of the wheels is defined:

$$Vw = [V1 \quad V2 \quad V3]^T \tag{1}$$

Vw is the velocity of each individual wheel.

To derive the transformation matrix between wheel velocities and robot velocity, the wheel speed is expressed by the rotation of a unit vector in x_r direction.

$$Vi = \begin{bmatrix} cos(\varphi i + \psi) & sin(\varphi i + \psi) \\ sin(\varphi i + \psi) & -cos(\varphi i + \psi) \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} cos(\varphi i + \psi) \\ sin(\varphi i + \psi) \end{bmatrix} \tag{2}$$

$$\begin{bmatrix} \omega 1 \\ \omega 2 \\ \omega 3 \end{bmatrix} = \frac{1}{r} \begin{bmatrix} V1 \\ V2 \\ V3 \end{bmatrix} = \frac{1}{r} \begin{bmatrix} -sin(\varphi 1) & cos(\varphi 1) & R \\ -sin(\varphi 2) & cos(\varphi 2) & R \\ -sin(\varphi 3) & cos(\varphi 3) & R \end{bmatrix} \begin{bmatrix} \dot{x}r \\ \dot{y}r \\ \Omega \end{bmatrix} \tag{3}$$

Determining robot velocities from wheel velocities can be done by inverting the matrix. The following section will treat the relationship between the wheel velocities and the velocity of the robot expressed in world coordinates.

Specification robot :

- Mass = 5.35 kg
- Torque Motor = 11.7 kg.cm
- Speed Motor = 195 rpm
- Wide of Robot = 400 mm
- L1 = L2 = L3 = 182 mm
- Wheel Diameter = 80 mm

Transformation between wheel velocity and world velocity

$$\omega i = \frac{(-sin(\varphi i) \times \dot{x}r) + (-sin(\varphi i) \times \dot{y}r)}{r} \tag{4}$$

$$\begin{bmatrix} cos(\theta + \varphi i + \psi) & sin(\theta + \varphi i + \psi) \\ sin(\theta + \varphi i + \psi) & -cos(\theta + \varphi i + \psi) \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} cos(\theta + \varphi i + \psi) \\ sin(\theta + \varphi i + \psi) \end{bmatrix} \tag{5}$$

The cosine part is carried out by rotating the wheel. The other parts are carried out by moving the wheel sidewards. Hence, by including the tangential rotational velocity Ω of the robot (which is similar in robot and world frames), using the above equations and the fact that $\psi = \frac{\pi}{2}$, the velocity and angular velocity of each wheel is given by:

$$\begin{bmatrix}\omega 1\\ \omega 2\\ \omega 3\end{bmatrix} = \frac{1}{r}\begin{bmatrix}V1\\ V2\\ V3\end{bmatrix} = \frac{1}{r}\begin{bmatrix}-sin(\theta+\varphi 1) & cos(\theta+\varphi 1) & R\\ -sin(\theta+\varphi 2) & cos(\theta+\varphi 2) & R\\ -sin(\theta+\varphi 3) & cos(\theta+\varphi 3) & R\end{bmatrix}\begin{bmatrix}\dot{x}r\\ \dot{y}r\\ \dot{\Omega}\end{bmatrix} \quad (6)$$

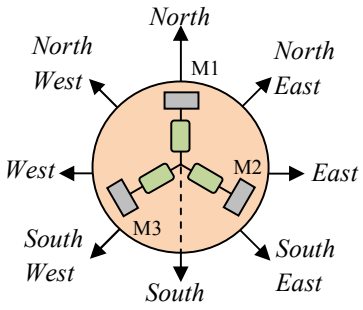


Figure 5. Initialization Movement of OMR

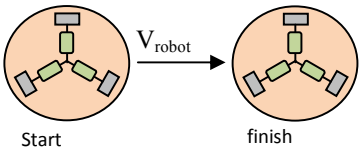


Figure 6. Move to the East

Then it will try the calculation with fixed robot speed is 50. From the kinematic formula of OMR it will get the speed of each motor. In the figure 6. Implemented OMR moving towards the East.

So,

$$Vm1 = 50$$

$$Vm2 = -\frac{\sqrt{3}.(50)}{2} = -25\sqrt{3}$$

$$Vm3 = -\frac{\sqrt{3}.(50)}{2} = -25\sqrt{3}$$

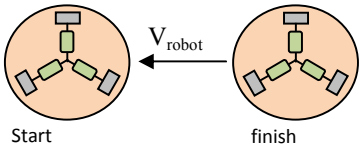


Figure 7. Move to the East

So.,

$$Vm1 = -50$$

$$Vm2 = -\frac{\sqrt{3}.(-50)}{2} = 25\sqrt{3}$$

$$Vm3 = -\frac{\sqrt{3}.(-50)}{2} = 25\sqrt{3}$$

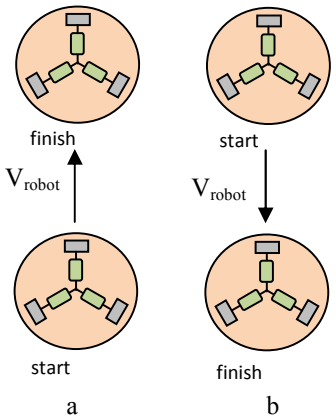


Figure 8. a) Move to the North b) Move to the South

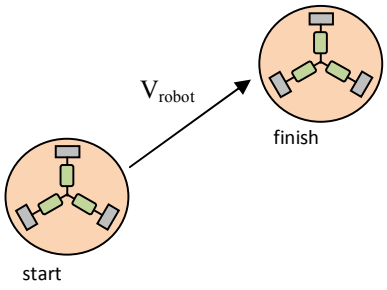


Figure 9. Move to the North East

Table 1. Velocity Each motor with V_{robot} = 50

Vx	Vy	Vm1	Vm2	Vm3	Direction
0	50	0	-25	25	*North*
35.35	35.35	35.35	-48.39	-12.939	*North East*
50	0	50	-43.3	-43.3	*East*
35.35	-35.35	35.35	-12.939	-48.39	*South East*
0	-50	0	25	-25	*South*
-35.35	-35.35	-35.35	48.39	12.939	*South West*
-50	0	-50	43.3	43.3	*West*
-35.35	35.35	-35.35	12.939	48.39	*North West*

B. *Electronic Design*

To control the movement of the OMR, electronic controls designed to distinguish between motor control of behavior detection by the camera coordinates. There are a Low Level Control for controlling the three actuators and High Level Control for calculating the kinematic equation. PC used to process the webcam.

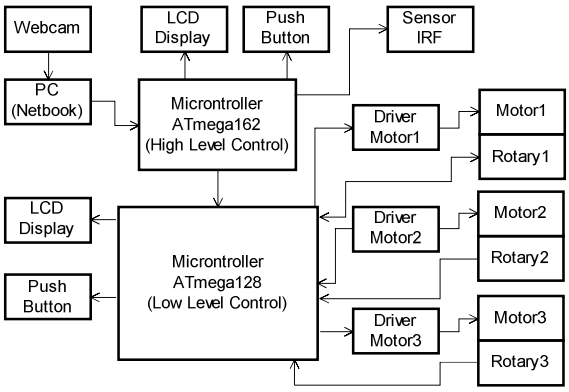


Figure 10. Block Diagram System

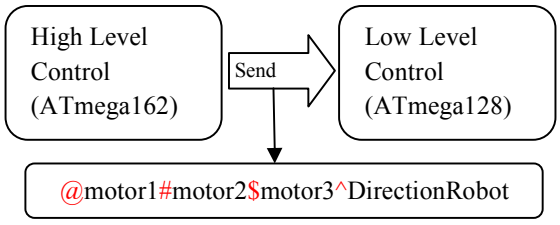


Figure 11. Sending Serial data from HLC to LLC

PC will process the gesture that would be obtained coordinates of the object. Coordinates of the object will be converted into speed robots and sent serially to the HLC. Inside HLC inverse kinematics of the equation there OMR will gain the speed of each motor. The speed of each motor will be sent to the LLC, which simply compute LLC PID and process the wheel speed sensor

PID Controller

For motor control, using the PD control in accordance with the plan that is accelerating the motor response and decrease overshoot when the error soared.

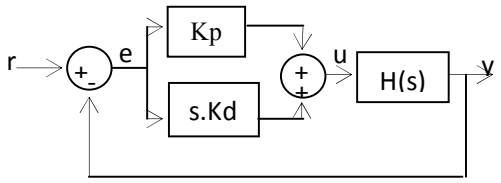


Figure 12. PD Control

$$u = Kp \times e + Kd \times \frac{\Delta e}{\Delta t} \quad (7)$$

Characteristics of Control Derivative (Kd) is able to predict the error values that will happen. If Kp is too large then the system has overshoot and Kd are able to make the system better when overshoot.

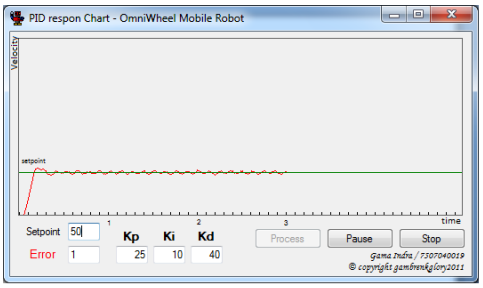


Figure 13. GUI for setting PD Control

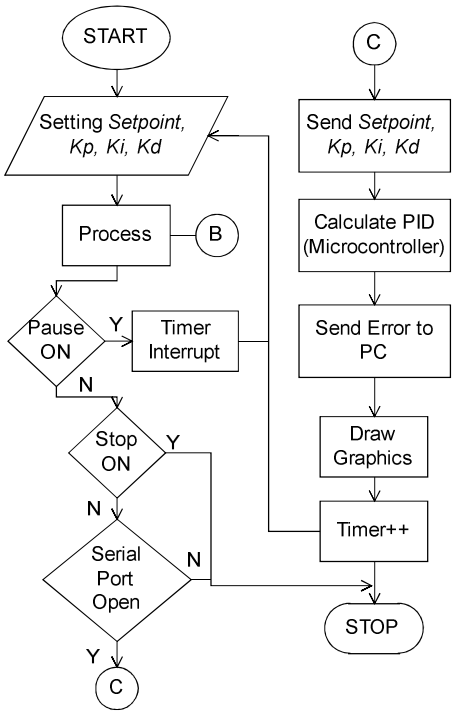


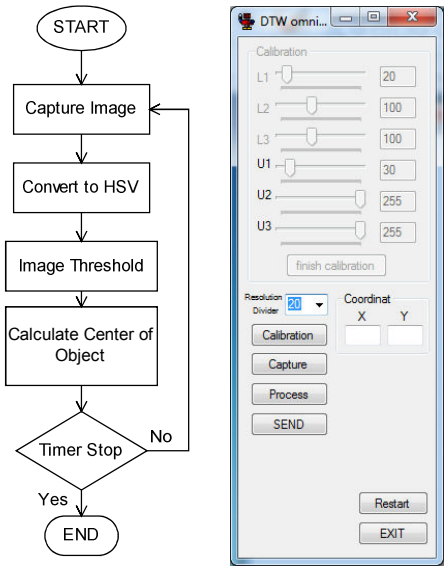
Figure 14. Flowchart GUI PID control

$$Kp = 25$$
$$Kd = 45$$
$$Setpoint = 50$$

C. Color Detection

Image Processing using the OpenCV library has been integrated with Visual C + + programming. PCs activate webcams and capture objects. Results of capture is converted from RGB to HSV. Then the threshold process can detect the color in accordance with the plan. Then by calculating the center of the Object.

For GUI Image Processing, designed like figure above. There is a calibration process to the process of setting a variable threshold with Upper Limit Lower Limit RGB and RGB for the threshold. So we could be setting up what color will be detected by the system when the indoor and outdoor experiments.



a b

Figure 15. a Flowchart Color Detection b Setting Upper and Lower Limit RGB threshold

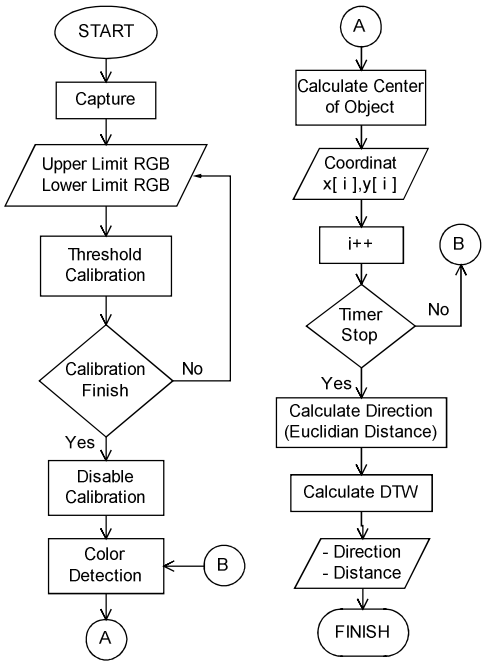


Figure 16. Flowchart GUI Image Processing

D. Dynamic Timer Warping

Given two sequence S of Length N and T of length M, a straghtforward distance measure between S and T. Dist (S,T), can be given with$L \overset{def}{\longleftrightarrow} \min\{N, M\}$ as the sum of the absolute distance below :

$$Dist(S,T) = \sum_{i=1}^{L}\|Si - Ti\| + \begin{cases} \sum_{i=L+1}^{N} \|Si\|, if N > M \\ 0, if\ N = M \\ \sum_{i=L+1}^{M} \|Ti\|, otherwise \end{cases} \quad (8)$$

This is rigid measiure like the humming distance, the number of symbols that disagree. In contrast, DTW offers greater flexibility in measuring similarity (or distance between a given pair of patterns S and T. This is similar in spirit too the edit distance and the Levenshtein distance). Such a flexible distance measure is often meaningful in real-world application

Euclidian Distance

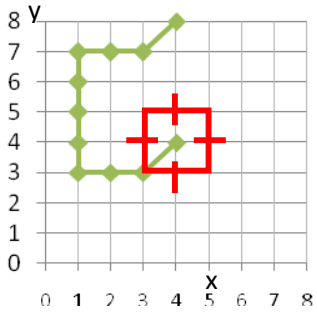
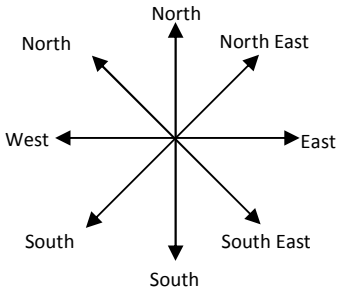


Figure 17. Example path from color detection tracking

$$Vextor\ x = x_n - x_{n-1}$$
$$Vextor\ y = y_n - y_{n-1}$$



Direction	$\vec{x}$	$\vec{y}$
N	0	+
NE	+	+
E	+	0
SE	+	-
S	0	-
SW	-	-
W	-	0
NT	-	+

Figure 18. Mapping Direction

Equation of **Euclidian Distance :**

$$|d(n,n-1)| = \sqrt{(x_n - x_{n-1})^2 + (y_n - y_{n-1})^2} \quad (9)$$

We can find distance from (x_{n-1}, y_{n-1}) to (x_n, y_n). And then calibrate the distance with field of robot. As in the table, suggesting that if vextor x is positive and the vector y are positive then the robot is intended to move toward the Northeast.

III. EXPERIMENTAL

E. PID control

Table 2 Konstanta PD each motor

Motor 1			Moto 2			Motor 3		
Kp	Kd	Respon (ms)	Kp	Kd	Respon (ms)	Kp	Kd	Respon (ms)
50	0	300	50	0	300	50	0	300
1	0	600	1	0	600	1	0	600
60	0	450	60	0	450	60	0	450
50	50	300	50	50	300	50	50	300
50	10	300	50	10	300	50	10	300
10	90	1300	10	90	1300	10	90	1300
10	10	300	10	10	300	10	10	300
50	30	350	50	30	350	50	30	350

As in the table, a good motor response obtained when the value of Kp = 50 and Kd = 50. Motor capable of achieving steady state within 300 ms. When the value of Kp is too large then the system will be oscillating. To eliminate this required Kd capable of predicting the error value is generated, however if the Kd is too large then the system oscillations

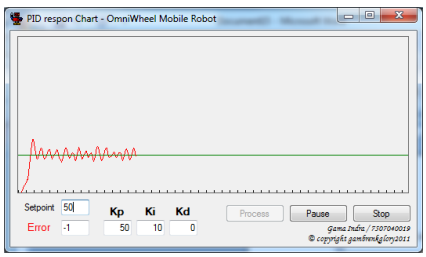


Figure 19. Respon motor 1 Kp = 50 and Kd = 0

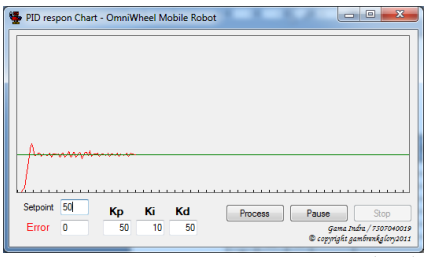


Figure 20. Respon motor 1 Kp = 50 and Kd = 50

F. Color detection

start *finish*

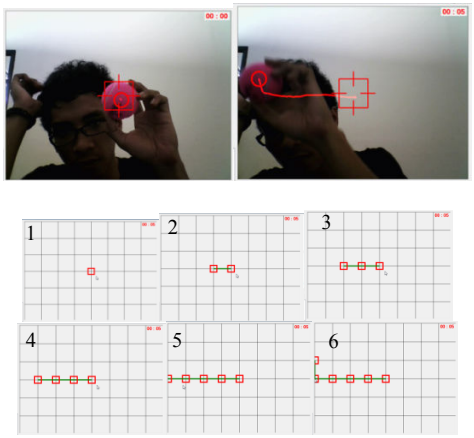


Figure 21 Experimental 1 gesture recognition

The experiment was carried out by moving the object to the left of the camera, or on a system called the west. The camera captures objects and obtain the center of the object. Time for a camera captures the object is 5 seconds. From this experiment obtained satisfactory results because the system can calculate the path formed by the gesture and sends the data via serial

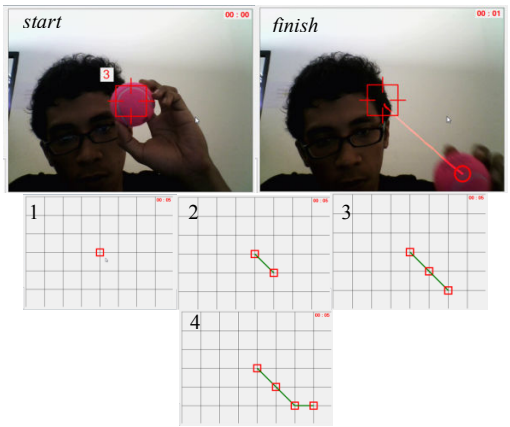


Figure 22. Experimental 2 gesture recognition

The experiment was carried out by moving the object into the right corner of the camera, or on a system called the south east. The camera captures objects and obtain the center of the object. Time for a camera captures the object is 5 seconds. 4 pieces obtained motion path formed by the gesture and can transmit data via the serial

G. Movement robot from object tracking from camera

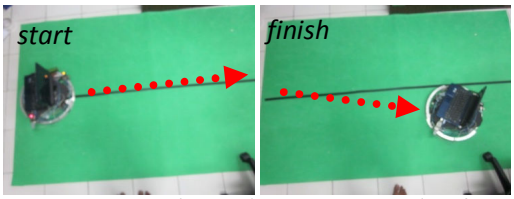


Figure 23. Experimental 1 movement robot from object tracking from camera

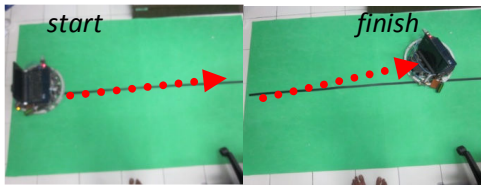


Figure 24. Experimental 2 movement robot from object tracking from camera

On the integration of robots with this gesture GUI figure 2.4, given that a simple gesture toward the South. So to translate the position of an object into a position that will be sent to the robot, the robot is simply get the command to move to the south. The robot is able to follow objects, but there are still tracking the position error caused by wheel slippage factor and the response of each motor.

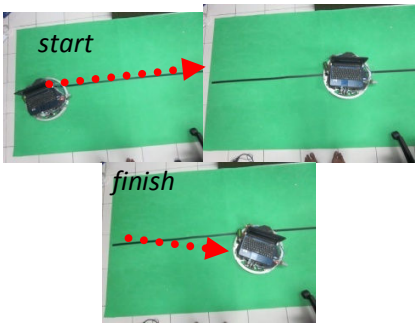


Figure 24. Experimental 3 movement robot from object tracking from camera

Wheel slip, causing the robot is not on track. But the robot to get the appropriate command. Another factor that causes the robot out on the tracks is the response of each motor is not the same, causing delays in one of the motor at start.

IV. CONCLUSION

From the experiments that have been done then some conclusions can be drawn as follows:

1. PID control of each motor can increase the rise time so as to accelerate the response of the motor.(Kp=50 and Kd=50)
2. For example, moving the robot toward the east, or other movements that require third rotation motorcycles, it takes a very fast motor response. If one of the motor response was not good, it might change the direction of movement
3. Wheels to skid when the trajectory is smooth and the rough trajectory experienced slippage but not so large
4. For object detection, threshold calibration feature is very useful when experiments are carried out in different places

REFERENCES

[1] Mizutani, E., “The Dynamic Time Warping Algorithms”, Mechanical Engineering Seminar, Tokyo Metropolitan University, 2006

[2] Corradini, Andrea. 2001. "Dynamic TimeWarping for Off-line Recognition of a Small Gesture Vocabulary". Proceedings of the IEEE ICCV Workshop on Recognition, Analysis, and Tracking in Real-Time Systemsm,2001: Germany

[3] Niels, Raiph. (2004). Dynamic Time Warping; Nijmegen, Netherlands

[4] Watanabe, K. (1998). Control of an omnidirectional mobile robot. In KES'98, 2th International Conference on Knowledge-Based Intelligent Electronic Systems.

[5] S. Kim and S. Lee. " Robust Velocity Estimation of an Omnidirectional Mobile Robot Using a Polygonal Array of Optical Mice ". International Journal of Control, Automation and System, 2008

[6] Y. Liu, J. J. Zhua, R. L. Williams II, J. Wu. 2008. " Omni-directional mobile robot controller based on trajectory linearization". Journal of Robotics and Autonomous Systems (56) (2008)