2016 International Conference on Information Technology Systems and Innovation (ICITSI)
Bandung – Bali, October 24 – 27, 2016
ISBN: 978-1-5090-2449-0

Three-wheeled Omnidirectional Robot Controller Implementation

Gusti Arif Hanifah Pawitan
Autonomous Vehicle Research Group
SEEI Institut Teknologi Bandung
Bandung, Indonesia
arif.pawitan@gmail.com

Kusprasapta Mutijarsa
Autonomous Vehicle Research Group
SEEI Institut Teknologi Bandung
Bandung, Indonesia
soni@stei.itb.ac.id

Widyawardana Adiprawita
Autonomous Vehicle Research Group
SEEI Institut Teknologi Bandung
Bandung, Indonesia
wadiprawita@stei.itb.ac.id

***Abstract* – This paper presents a controller design and its implementation for an omnidirectional robot. Due to the holonomic nature of an omnidirectional robot, the mobile robot are able to follow any planned trajectory. The path are calculated on the world frame and then transformed using inverse kinematic to each of the mobile robot motor speed. Therefore a kinematic model for the robot are derived and a PID controller is utilized to control the speed of each actuator. MATLAB® System Identification Tools® is used to identify the transfer function of each actuator and then simulated using the MATLAB® SIMULINK® platform to design a suitable PID control.**

Index Terms— Mobile Robot, Omnidirectional, Kinematics Model, PID Speed Control.

I. Introduction

Omnidirectional mobile robot is a robot that is able to move to any direction without any reorientation[1]. Therefore these type of robot is suitable to perform tasks in tight and complex spaces. Their manuverability is due to the use of omnidirectional wheel that is capable to slide parallel to the rotor axis whilst retaining their traction normal to the rotor axis. This makes omnidirectional robot to perform better than non-holonomic robot in dynamic environment aplications, for example, in the Ekshibisi Robot Sepak Bola Beroda of Kontes Robot Indonesia (KRI).

The robot used in this work is a three-wheeled omnidirectional mobile robot which is designed for two dimensional motions. Robot location are represented by (x,y) coordinates and the orientation of the robot is specified by the angle of rotation (phi). This robot is able to control the three degree of freedom (x, y, phi) independently by controlling each actuators. With invers kinematic, the speed of each actuators can be derived from the trajectory of the robot.

In this paper, the kinematic model of the constructed robot is derived along with the transfer function of each actuator used.The kinematic model will be the base of the invers kinematik algorithm while the transfer function is used to calculate the PID constant. Calculation of the PID constant is done using the PID Tuning toolbox in MATLAB® SIMULINK® to obtain a controlled actuator plant with the desired specification. The whole system is then simulated on MATLAB® SIMULINK® and implemented in hardware.

II. Mobile Robot Platform

A. *Robot Architecture*

The mobile robot platform used in this work have 3 actuators mounted 120° apart each equipped with a 4 inch omnidirectional wheel. The base of the robot took a triangle shape which is made from a machined 3mm thick aluminum plate with a diameter of 38cm. With all its component mounted, the weight of the robot is about 2kg. The kinematic structure and the hardware construction of the robot is shown in Fig. 1.

The platform itself consist of 3 DC motors equipped with a microcontroller for each motor, a master microcontroller that communicate with I2C, a single board computers as the main controller, and a lithium polymer battery. For feedback purpose, the motor used for the robot each are equipped with two encoder counting 1920 pulse/turn. The two encoder is

This work was supported in part by the Institut Teknologi Bandung Student Organization.

90° out of phase so it is possible to tell the direction by observing the transition of each encoder.

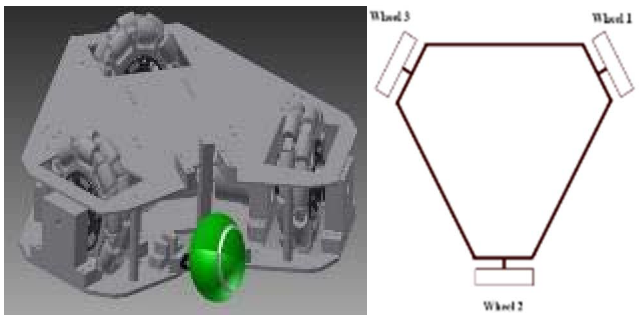

Fig. 1. The designed mechanical structure of the robot (left) and the kinematic structure of the robot (right)

The platform have a three-level control architecture. The main control is conducted by a single board computer which is connected to a master microcontroller via serial communication. The master microcontroller is connected to three slave microcontroller via I2C communication. The block diagram of the control architecture is shown in Fig.2.

The main controller is a Cortex A15 single board computer working at 2GHz with 2GB of RAM responsible for trajectory planning and PID control calculation. The Master microcontroller is an Arduino Pro Micro working at 16MHz with 2.5KB of RAM responsible for collecting encoder input and sending them to the main controller while receiving speed control instruction from the main controller and send it through to the right motor. The slave microcontroller is responsible to read encoder counts with interrupt, sending them to the master on request, and implement the speed control instruction received from the master.

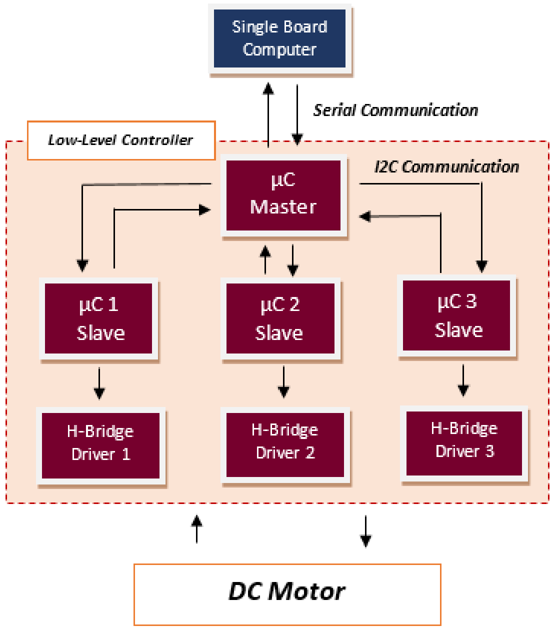

Fig. 2. Block Diagram of the Control Architecture

B. *Kinematic Model*

It can be seen from Fig. 3 that the fixed world coordinates is represented as x_wOy_w and the moving coordinates is represented as x_mOy_m where both of the coordinates share the same origin point but the moving coordinates rotate with the robot. The angle of the moving coordinates with respect to the world coordinates is represented as phi. F_T is defined as the traction force of the ground on the wheel where the positive direction is as shown in the figure.

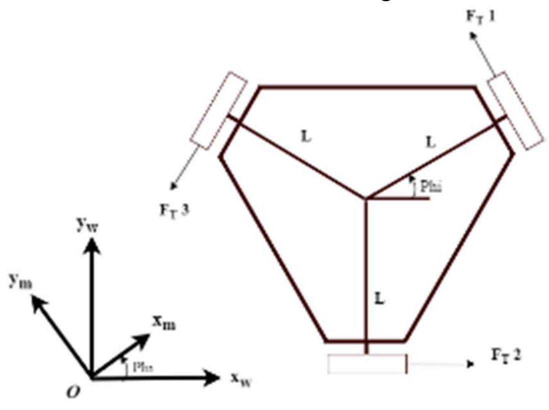

Fig. 3. Block Diagram of the Control Architecture

The relationship between the traction forces with the force of the moving coordinate can be expressed as ${}^{m}F = BF_T$ where ${}^{m}F = [F_{mx}\ F_{my}\ F_{Tz}]^T$, $F_T = [F_{T1}\ F_{T2}\ F_{T3}]^T$, and B a geometrical matrix defined as [2]:

$$B = \begin{bmatrix} 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \\ 1 & -\frac{1}{2} & -\frac{1}{2} \\ L & L & L \end{bmatrix} \tag{1}$$

where L is the distance between the wheel and the center of the robot which is 20cm. The force in the world coordinate then can be derived as [2]:

$$ {}^{W}F^{T}.{}^{w}\dot{X} = {}^{m}F^{T}.{}^{m}\dot{X} = F_T^T.r\dot{q}_L \tag{2}$$

where ${}^{w}\dot{X} = [\dot{x}_w\ \dot{y}_w\ \dot{Phi}_w]$ and ${}^{m}\dot{X} = [\dot{x}_m\ \dot{y}_m\ \dot{Phi}_m]$ are the velocity of the robot in world coordinates and in moving coordinates respectively. $r\dot{q}_L$ represents the velocity of each wheel where $\dot{q}_L$ is the angular velocity of the wheel defined as $\dot{q}_L = [\dot{q}_3\ \dot{q}_2\ \dot{q}_3]$.

The velocity kinematic equations of the omnidirectional mobile robot is expressed in world coordinates as follows [2]:

$$ {}^{m}\dot{X} = (B^T)^{-1}r\dot{q}_L \tag{3}$$
$$ {}^{w}\dot{X} = {}^{w}_{m}R\ {}^{m}\dot{X} = {}^{w}_{m}R\ (B^T)^{-1}r\dot{q}_L \tag{4}$$

where ${}^{w}_{m}R$ is the rotation matrix which transform each vector from the moving coordinates to the world coordinates defined as [2]:

$$ {}^{w}_{m}R = \begin{bmatrix} \cos(phi) & -\sin(phi) & 0 \\ \sin(phi) & \cos(phi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5} $$

C. *Obtaining Actuator Transfer Function*

The motor used for the robot is a Pololu 37Dx68L mm 12V Gearmotors with a gear ration of 30:1 shown in Fig.4. With no load, the motor speed is 360RPM and the stall torque at12V is 110 oz-in. This motor is equipped with two encoders, encoder A and encoder B where encoder A is 90° out of phase with encoder B. The transfer function estimation can be obtained by a series of step.

First of all, make a program to read the motor encoder output and store it periodically every 0.01s. In this step, it doesn't matter which encoder is being used so it can be either encoder A or encoder B. Give the motor a step input and collect its encoder output after for 2-3s. The data acquired is then processed with MATLAB® System Identification Tools®. The system identification was carried out on the motors in the discrete time domain with a sampling time of 0.01s.

Fig. 4. Pololu 37Dx68L mm Gearmotors with Encoder (left)

III. Control Implementation

A. *PID Control Design*

With the transfer function estimation acquired, the PID control can be defined and simulated with MATLAB® SIMULINK® platform. The simulation plant of the system is shown in Fig. 5. First the plant spesification must be present, in this case, the plant must have an overshot less than 20%, rising time less than 0.01s, and respons time less than 0.01s. The spesification above is decided with consideration of the encoder sampling time. After optimisation, the plant cannot satisfy all the specification mentioned above (±0.005s).

The PID that is implemented in the system uses Backward Euler equation. That equation can be derived to [3]:

$$ u[k] = -\frac{a_1}{a_0}u[k-1] - \frac{a_2}{a_0}u[k-2] + \frac{b_0}{a_0}e[k] + \frac{b_1}{a_0}e[k-1] + \frac{b_2}{a_0}e[k-2] \tag{6} $$

where u[k] is the plant output with b_0, b_1, b_2, b_0, a_0, a_1, and a_2, respectively are [3]:

$$ b_0 = K_p(1 + NT_s) + K_iT_s(1 + NT_s) + K_dN \tag{7} $$
$$ b_1 = -(K_p(2 + NT_s) + K_iT_s + 2K_dN) \tag{8} $$
$$ b_2 = K_p + K_dN \tag{9} $$
$$ a_0 = (1 + NT_s) \tag{10} $$
$$ a_1 = -(2 + NT_s) \tag{11} $$
$$ a_2 = 1 \tag{12} $$

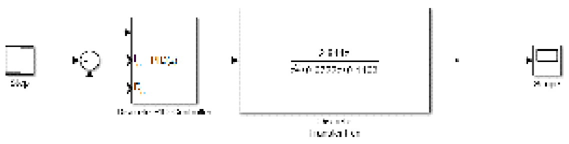

Fig. 5. PID Control Simulation Block in MATLAB® SIMULINK®

B. *System Simulation in MATLAB®*

The simulation of the proposed control system is done using the MATLAB® SIMULINK® platform. The simulation block is shown in Fig. 5. There are three major MATLAB function block that is used for the simulation. The first block is the path planner, where the line equation is used as a trajectory. The next one is the invers kinematic block to transform the trajectory to the speed of each motor where the motor PID control is already implemented. The last function block is the forward kinematic function. The forward kinematic function is used to check the output of the controlled system and compare it to the input. The compared result shown in Fig. 6 shows that the system simulated is able to follow the planned trajectory.

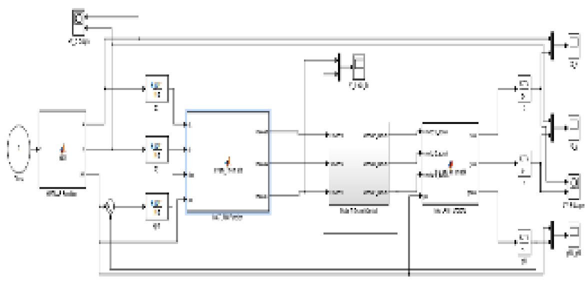

Fig. 5. The system simulation block

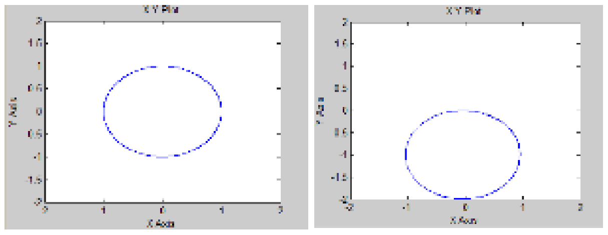

Fig. 6. Simulation result: Planned Trajectory (left) and the forward kinematic output plot (right

C. System Implementation in Hardware

A C language program is made for the hardware implementation of the designed control system. The high level program will collect the feedback data from the microcontroller and send the computed output via serial communication. A circle trajectory, the same as in the simulation, is used for the mobile robot platform test.

The result of the test is the robot are able to move as planned in the trajectory with an error less than 10cm from the desired path. This is due to the slippage of the wheel that is not calculated to simplify the control design and simulation. The omnidirectional mobile robot used for the test is shown in Fig.7.

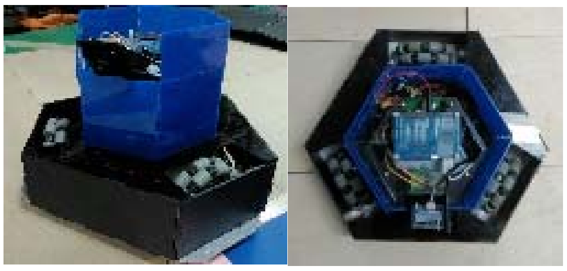

Fig. 7. The mobile robot used for hardware implementation

Robot with the implemented control system was also used to compete at the Kontes Robot Indonesia (Indonesia Robot Contest) in Politeknik Elektronika Negeri Surabaya on 1-4 July 2016. The omnidirectional mobile robot used for the competition and the robot during the match is shown in Fig.8 and Fig.9 respectively.

Fig. 8. The mobile robot used for the competition

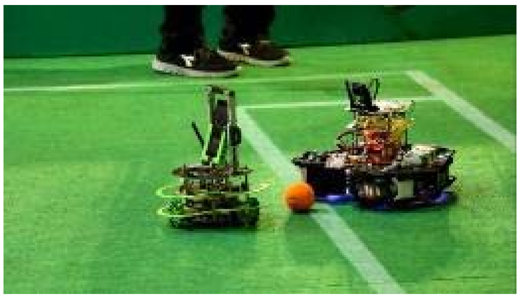

Fig. 9. The mobile robot during one of the match on the day of the competition

IV. Conclusion

In this work, the implementation of a PID controller for a three-wheeled omnidirectional mobile robot is implemented. The mobile robot kinematic model is derived and the actuator transfer function estimation is obtained. The proposed control system has been simulated and then implemented for the robot. The upshot shows that the mobile robot can be controlled to follow the inputted trajectory.

V. References

[1] R. Comasolivas, J. Quevedo, T. Escobet, A. Escobet and J. Romera, " Low level control of an omnidirectional mobile robot," in *Proc. 2015 23rd Mediterranean Conference on Control and Automation (MED).*, pp. 1160-1166.

[2] W. Jianhua, "Dynamic Path Planning of an Omni-Directional robot in a Dynamic Environment," Ph.D. dissertation, Dept. of Integrated Engineering, Univ. Ohio, Ohio, 2005.

[3] V. Toochinda, "Discrete-time PID Controller Implementation", Scilab Ninja, 2014. [Online]. Available: http://scilab.ninja/discrete-time-pid-controller-implementation/. [Accessed: 11- May- 2016].