Real-Time Control of a Two-Wheeled Inverted Pendulum Mobile Robot

S. W. Nawawi, M. N. Ahmad, and J. H. S. Osman

Abstract—The research on two-wheeled inverted pendulum (T-WIP) mobile robots or commonly known as balancing robots have gained momentum over the last decade in a number of robotic laboratories around the world. This paper describes the hardware design of such a robot. The objective of the design is to develop a T-WIP mobile robot as well as MATLAB interfacing configuration to be used as flexible platform comprises of embedded unstable linear plant intended for research and teaching purposes. Issues such as selection of actuators and sensors, signal processing units, MATLAB Real Time Workshop coding, modeling and control scheme will be addressed and discussed. The system is then tested using a well-known state feedback controller to verify its functionality.

Keywords—Embedded System, Two-wheeled Inverted Pendulum Mobile Robot.

I. INTRODUCTION

INVERTED pendulum has been the subject of numerous studies in automatic control since the forties. In this paper, the development of the theme inspired by the well known Segway robot is described. This kind of robots has induced a lot of interest recently [1]. This is due to the fact that it provides rich opportunities for application of control design, signal processing, distributed control systems and consideration of real time constraint for implementation issues. A scaled down prototype of a Digital Signal Processor controlled two–wheeled vehicle called JOE based on the inverted pendulum with weights attached to the system to simulate a human driver was considered in [2]. A linear state space controller utilizing sensory information from a gyroscope and motor encoders is used to stabilize the system. With small carts built utilizing this technology allows humans to travel short distances in a small area or factories instead of using car or buggies which is more polluting [3].

The trajectory control for this type of robot was successfully implemented in two dimensional by [4]. The proposed trajectory control algorithm can make the robot move autonomously, although quite slowly. The work was further extended in [5] where the robot is assumed to receive similar input on both wheels and the mathematical modeling is represented in one dimensional plane system. Based on successful result in [5], similar work has been reported using LQR controller [6]. The dynamic model of a two-wheeled inverted pendulum is analyzed from controllability and feedback linearizability point of view by [7]. The strong accessibility condition is checked and the maximum relative degree of the system is found. A general discussion on motion control of T-WIP is done by [8].

The attitude is difficult to detect by simply using the signals from sensor. The integration of rate gyroscope signal has contributed the problem of drift. The problem solve using an observer that considering internal dynamic of the system [9]. The estimation and control algorithm of the posture using the adaptive observer is proposed by [10]. The presented algorithm did not consider movement on the two dimensional plane and the robot also was not autonomous and connected by wires with ground computer. A variable structure controller for two wheels inverted pendulum mobile robot is proposed for tracking to desire trajectories by [11]. However this work is consider the robot has castor to support the robot body from falling down. As a view point of tilt-up control, [12] doing research on gyroscopically in stabilized the robot, and [13] have studied inverted systems. The research is still open to explore for new controller development. A common sense solution to the stability problem and applied it to human postural problem is proposed by [14]. However the approximate profile of terrain can be an uncertain.

The remaining of the paper is organized as follows: Section II treats on hardware design, particularly in the selection of sensors and their associated algorithms Section III describes the mathematical modeling of the robot. Section IV addressed the controller design and their simulation results. Section V presents the MATLAB interfacing configuration. Finally the verification of the controller using real time implementation is shown experimentally. Results of the real time implementation will be illustrated in details.

Fig. 1 The prototype T-WIP mobile robot
II. HARDWARE DEVELOPMENT

Fig. 1 shows the prototype design of the robot. The robot is equipped with two servo drives for actuation, a Gyroscope for measuring angle and angular velocity of pendulum body, and encoders for measuring the position of the wheels. Signal processing and control algorithm are distributed among the three microprocessors. Two of them are used for servo drives while the other one is used for stabilizing control. Although this kind of layout enables hierarchical control design, it also complicates implementation, since processor communication must also be considered. The T-WIP mobile robot is composed of a chassis carrying a DC motor coupled to a planetary gearbox for each wheel, the DSP board used to implement the controller, the power amplifiers for the motors, the necessary sensors to measure the vehicle’s states. The battery is bolted inside the body casing and it significantly represents 30% of the total robot mass. The wheels of the vehicle are directly coupled to the output shaft of the gearboxes. The robot is control by applying a torque $C_R$ and $C_L$ to the right and left wheels respectively. Fig. 2 illustrates the block diagram of the control architecture of the system.

Both geared servo motor needs to generate a very high torque. To achieve this, the IPM100 is used as the motor driver. It is basically a 36V, 3A fully digital intelligent servo drive based on the DSP controller technology. It is also embedded with the high level Technosoft Motion Language (TML) and therefore offers a flexible, compact and easy to implement solution for single or multi-axis applications with brushless and DC motors.

To provide information about T-WIP states for control purposes, two incremental encoders and a rate gyroscope are interfaced together as shown in Fig. 4. All the interfacing is based on control structure of embedded system seen in Fig. 2 above. Straight line position and speed as well as yaw angle and rate can be determined from the angle rotation of the two wheels ($\theta_{RR}$ and $\theta_{RL}$) with respect to the gravity. The relation of these angles with pitch angle, $\theta_p$ and the body angle, $\theta_{RPL}$ and $\theta_{PRI}$ can be referred in Fig. 3. The embedded controller has task to monitor all feedback coming from incremental encoder. Then process the feedback to make sure T-WIP is balance at it equilibrium point. The command to embedded controller is given by IPC using C language interfacing in the Real Time Workshop of MATLAB. The IPC is running online or at the same time with system to show the real time result according to the output response needed.
III. MATHEMATICAL MODELING

The dynamic performance of a balancing robot depends on the efficiency of the control algorithms and the dynamic model of the system [17]. By adopting the coordinate system shown in Fig. 5 using Newtonian mechanics, it can be shown that the dynamics of the T-WIP mobile robot under consideration is governed by the following motion equations (1-15) [2],[18]. The coordinate system for the robot is depicted in Fig. 5.

For the chassis, the equations:

\[
\begin{align*}
\dot{x}_p &= \dot{\theta}_p L \sin \theta_p \\
\dot{y}_p &= \dot{\theta}_p L \cos \theta_p + \dot{x}_{RL} + \dot{x}_{RR} \\
\delta &= \frac{\dot{x}_{RL} - \dot{x}_{RR}}{2f}
\end{align*}
\]  

where \(H_{TL}, H_{TR}, H_{L}, H_{R}, V_{TL}, V_{TR}, V_{L}, V_{R}\) represent reaction forces between the different free bodies. The robot parameters are as tabulated in Table I.

**TABLE I**

<table>
<thead>
<tr>
<th>Parameter of T-WIP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Symbol</td>
</tr>
<tr>
<td>(x_r)</td>
</tr>
<tr>
<td>(\theta_p)</td>
</tr>
<tr>
<td>(\delta)</td>
</tr>
<tr>
<td>(J_{RL}, J_{RR})</td>
</tr>
<tr>
<td>(M_r)</td>
</tr>
<tr>
<td>(\dot{M}<em>{RL} = M</em>{RR} = M_r)</td>
</tr>
<tr>
<td>(J_p)</td>
</tr>
<tr>
<td>(J_{p\beta})</td>
</tr>
<tr>
<td>(M_r)</td>
</tr>
<tr>
<td>(R)</td>
</tr>
<tr>
<td>(L)</td>
</tr>
<tr>
<td>(D)</td>
</tr>
<tr>
<td>(y_r)</td>
</tr>
<tr>
<td>(x_p)</td>
</tr>
<tr>
<td>(g)</td>
</tr>
<tr>
<td>(C_L, C_R)</td>
</tr>
</tbody>
</table>

Equations (1)-(11) can be represented in the state-space form as:

\[
\dot{x}(t) = f(x) + g(x)u
\]

where \(x \in \mathbb{R}^n\), \(u \in \mathbb{R}^m\) are respectively the state and the control. \(f(x)\) is nonlinear dynamic function matrix and \(g(x)\) is
nonlinear input function matrix. The state, $x$ of the system is defined as:

$$\begin{align*}
\dot{x} &= [x, \dot{x}, \dot{\theta}, \dot{\delta}]'
\end{align*}$$

(13)

Modifying the equations above and then linearizing the result around the operating point ($\theta_p=0, x=0, \delta=0$) and de-coupling, the system’s state space equations can be written in matrix form as:

$$\begin{align*}
\begin{bmatrix}
\dot{x} \\
\dot{x}_c \\
\dot{\theta}_p \\
\dot{\delta}
\end{bmatrix}
= & \begin{bmatrix}
0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
x \\
\dot{x}_c \\
\theta_p \\
\delta
\end{bmatrix}
+ \begin{bmatrix}
0 \\
B_2 \\
B_4 \\
B_6
\end{bmatrix}[C_L + C_R]
\end{align*}$$

(14)

$$\begin{align*}
\begin{bmatrix}
\dot{\delta} \\
\dot{\delta}
\end{bmatrix}
= & \begin{bmatrix}
0 & 1 \\
0 & 0
\end{bmatrix}\begin{bmatrix}
\delta \\
\dot{\delta}
\end{bmatrix}
+ \begin{bmatrix}
0 \\
B_6
\end{bmatrix}[C_L - C_R]
\end{align*}$$

(15)

where

$$\begin{align*}
A_2 &= g \left(1 - \frac{4}{3} \frac{L}{g} \frac{M_p}{X} \right) \\
A_3 &= \frac{g M_p}{X} \\
B_2 &= \frac{4L}{3X} - \frac{1}{M_p} \\
B_4 &= \frac{Y}{X} \\
B_6 &= \frac{(9M_p + M_p)R}{6D}
\end{align*}$$

and,

$$\begin{align*}
X &= \frac{1}{3} M_p \left( \frac{M_p + 6M_p}{2} \right) \\
Y &= \frac{M_p}{(M_p + 3M_p)R} + \frac{1}{L}
\end{align*}$$

For simplicity, the details of equation (14) and (15) are not shown here and can be found elsewhere [19]. The T-WIP balancing model, namely equation (14) will be used throughout this work consequently.

IV. CONTROLLER DESIGN

System performance (i.e. reaction to disturbance forces, tracking of driver input, etc.) is driven by the pole placement controller. In order to test the T-WIP performance, pole-placement controllers with different poles has been applied.

For a chosen pole placement, the controller’s gains were calculated and implemented on the embedded board. T-WIP was then tested with the configuration and the response is then recorded by the control system. One of the tests conducted consist of an impulse disturbance force applied to a position above the center of gravity. The energy transmitted with a falling weight amounted to about 1.2 J.

Issues like damping ratio and settling time could be clearly identified on the recorded responses and permitted an efficient fine-tuning of the system. Fig. 6 shows the system’s response to the above mentioned test with the initial pole placement chosen at pole $[-1.5-1.5-0.5-3i-0.5+3i]$. Note the pronounced oscillation of the system which indicates too weak damping. Increasing the damping ratio when change the pole to $[-1.5-i,-1.5+i,-3.5-5i,-3.5+5i]$ give the result as shown in Fig. 7. It can be seen that the response improves significantly. Now it has a harmonious catching of the disturbance force. When the force hit T-WIP, it causes the pendulum to fall forwards ($\theta_p<0$).

The control system accelerates the wheels in a positive direction to catch this fall and ultimately makes the pendulum fall in the other direction. A negative torque is then applied, moving the vehicle back to its original position and getting the pendulum back in an upright position. The controller task is to make sure that:

$$\theta_p \in \left[ \theta_m < \theta_m < \pi / 2 \right], \text{ for a given } \theta_m > 0$$

(16)

This is a physically meaningful problem because using these reference commands, one can safely follow a motion plan. The essential idea to use the pitch angle $\theta_p$ as a gas pedal for vehicle and use it to accelerate and decelerate until the specified speed is attained.

![Fig. 6 Initial pole placement of the “pendulum” system and associated response to an impulse disturbance force (energy transmitted $\approx 1.2$ J) applied to the pendulum](image)

![Fig. 7 Improved pole placement of the “pendulum” system and associated response to an impulse disturbance force (energy transmitted $\approx 1.2$ J) applied to the pendulum](image)
Another issue that has been addressed during testing is drivability. In order to successfully improve drivability, it was characterized based on two criteria. First criteria are readouts of the system’s reaction to a ramp shaped speed input and second criteria are the way different drivers felt about T-WIP handling.

Combining the driver’s feelings with the readouts of system behavior allowed further improvement of T-WIP control system. Fig. 8 shows the system’s response to a velocity ramp input with the final pole placement chosen. Note that the maximum acceleration possible is lower than the maximum deceleration. Due to the motor’s speed-current characteristics, a high torque cannot be obtained when operating at high speeds. However, this is exactly what is necessary to get the vehicle back into an upright position at the end of the acceleration phase. Deceleration demands maximum torque at low speeds so a steeper ramp is therefore possible.

Increasing performance with the pole placement chosen can be achieved by moving the poles further to the left, thus making the system faster [20]. Backlash as well as the maximum torque that can be transmitted to the ground (grip) prevent tuners from moving the poles past a certain limit. The used of an adaptive pole placement and nonlinear controller (depending on the system’s state) would enable further improvements to the system.

V. MATLAB INTERFACING DESIGN

The embedded control system in this work is design based on real-time workshop of MATLAB. Hence it makes the interfacing protocol between embedded controller card and MATLAB is the most pivotal. T-WIP real-time balancing will be fully carried out in the Real Time Workshop of MATLAB. Therefore it has an advantages of be intellectualized to observe the real-time results and performance of controller when be integrated with actuator. Furthermore, the controller can be redesigned expediently and repetitiously until users get a satisfactory result.

In the RTW (Real Time Workshop) of MATLAB, the special real-time kernel model replaces message processing via windows. Hence the capability of real-time mode to gets better response is good enough.

RTW builds applications from Simulink diagrams for prototyping, testing, and deploying real-time systems on a variety of target computing platforms. Users of Real-Time Workshop can direct it to generate source code that accommodates the compilers, input and output devices, memory models, communication modes, and other characteristics that their applications may require.

First step of configuration setup is to install MATLAB with Real-Time Windows Target and Visual C/C++ software. Then by using some command in Matlab, the Real-Time Windows Target kernel is activated together menu to select C compiler in MATLAB. A sampling demo of the sensor will be presented based on MATLAB RTW. In way to design an S-function block written in C Language, in which an S-function parameter for index of ad channels and an output should be defined. The fractional source code of GetAD.c is as follows,

```c
#define S_FUNCTION_NAME   GetAD
#define NUM_PARAMS                 (1)
#define AD_CHANNEL  _PARAM  (ssGetSFcnParam(S,0))
#define AD_CHANNEL (real_T)mxGetPr(AD_CHANNEL_PARAM)[0])
```

After coding for GetAD.c, by using a command line in MATLAB with no errors occur, the GetAD S-function then finished generating. Then the block parameter for ad channel is configured to channel 1. Secondly another system target file GetAD.tlc need to be design which is will be saved in the same directory with GetAD.c. The main source code flow chart of GetAD.tlc is as follows:
For the reference purpose, the command “outp” and “inp” in the target file GetAD.tlc is used for sending and receiving data between embedded controller and MATLAB. The command ctrl_byte=oxff, is used in GetAD.tlc file to make sure the DA card will stop find online data on the PC104 bus.

GetAD.tlc then compiled and be confirmed there is no error occurs.

In simulation parameter properties of GetAD.c, category need to build as Target configuration with system target file of rtwin.tlc. This configuration is used as setting for Real-time workshop in Matlab. Then the solver options and fixed step size configuration is set with sampling time 5ms.

The file is compiled after selecting “External” mode. The output of the sensor signal can be shown in Fig. 11.

VI. EXPERIMENTAL RESULT

In order to verify the developed T-WIP hardware system, the pole-placement algorithm as designed in previous section implemented as the controller. Fig. 13 show that the responses of the system closely match the simulation result in Fig. 6 which demonstrates the complete T-WIP system is functioning well in close-loop system. From the Fig. 14, it can be shown that the output response for position, velocity, inclination angle and angular velocity of inclination is following the shape of simulation result. The steady state error of position and velocity is approximately zero. It also shows that the values of steady state error are about zero of the inclination angle and its velocity. The pole placement controller seems to be capable in term of steady state error and settling time.
For further research the T-WIP will be tested with the various type nonlinear controllers. Among of them are State feedback Linearization and Sliding Mode Control.

VII. CONCLUSION

In this paper, the development of a T-WIP mobile robot system is presented. The issues of dynamical modeling, selection of actuators and sensors, MATLAB based interfacing and configuration of the embedded controller, as well as the implementation of pole placement control strategy has been addressed. The embedded control system using real-time workshop of MATLAB is confirm working well and all sensors give a good feedback signal based on the response getting from the experimental work. The results from this work show that the proposed embedded design architecture based on MatLab is capable of delivering the desired outcome.

ACKNOWLEDGMENT

The authors gratefully acknowledged the contributions of Kaustubh Pathak for work on the original version of this document.

REFERENCES