Proceedings of the 2015
IEEE Conference on Robotics and Biomimetics
Zhuhai, China, December 6-9, 2015

AU Ball on Plate Balancing Robot*

Ehsan Ali [1] and Narong Aphiratsakun [2]

***Abstract*— The ball on plate system is the extension of traditional ball on beam balancing problem in control theory. In this paper the implementation of a proportional-integral-derivative controller (PID controller) to balance a ball on a plate has been demonstrated. To increase the system response time and accuracy multiple controllers are piped through a simple custom serial protocol to boost the processing power, and overall performance. A single HD camera module is used as a sensor to detect the ball's position and two RC servo motors are used to tilt the plate to balance the ball. The result shows that by implementing multiple PUs (Processing Units) redundancy and high resolution can be achieved in real-time control systems.**

I. INTRODUCTION

There are two major paths to tackle the problems in control systems: (1) Derive the mathematical model of the system, and formulate the response accordingly. (2) Adopt a model-free approach, and use the feedback mechanism to calculate the response. In systems with high number of active actors the derivation of mathematical model becomes cumbersome, and in some cases impossible, therefore the second approach comes to rescue. Similar ball on plate papers were studied and the observation shows that many of them have used the mathematical model [1], [2], [3], [4] due to the fact that although ball and plate is a non-linear system but derivation of its mathematical model is very straightforward.

PID algorithm usually is run on a single micro-controller to control a given system. There are cases that computational power of a single micro-controller is not enough to cope with the real-time demands of the system. This paper attempt to demonstrated how the computational demand of various system parts can be distributed between separate processing units through simple serial UART communication pathways.

A. Related Previous Works

1) Ball Position Detection Mechanism:

- Single Camera on top of the plate, and then calculate the position by image processing techniques [3], [8].
- Two Cameras on top of the plate, and then calculate the position by image processing techniques [4], [7].
- Grid of infrared sensors.
- Attach an infrared-ultrasonic sensor to the ball and track the ball in 3D dimension.
- Resistive or capacitive touchscreen grid [2].
- Photo-transistor sensors triggered by monochromatic sharp beams of laser light [5].

*This work was supported by Assumption University of Thailand

[1]Ehsan Ali is with the Department of Computer Engineering, Assumption University, Thailand ehssan.aali@gmail.com

[2]Narong Aphiratsakun with Faculty of Mechatronics Engineering, Assumption University, Thailand nott.notty@gmail.com

2) Actuation Mechanism:

- DC motors attached to cables and pulleys.
- Servo motors attached to arms [2], [3], [5], [8].
- Linear actuators.
- Industrial Robot Arm [4], [7].

3) Control Mechanism:

- PID control on a single micro-controller [5].
- Matlab/Simulink-based real-time control prototyping application on a single micro-controller [1], [2], [3].
- PC-Based Controller, RT Linux [7].
- More than one PUs [8].

The section 2 explains the details of our system, the mechanical specifications, and Solidworks models. The section 3 covers the control method and the details of PUs involved, system flowcharts, and the specifications of the serial UART custom protocol. The section 4 provides the outcome of conducted experiments and graph results. Finally the section 5 concludes the paper.

II. AU BALL ON PLATE BALANCING ROBOT

Meanwhile there are numerous papers which have demonstrated the PID control [5], [6]. There are basically three major challenges needed to be addressed to have a functional system:

- A method to find the ball position in real time.
- The actuation mechanism to tilt the plate accordingly.
- The controller itself that sits between sensors and actuators.

First, the summarization of the methods used in the previous works to face the above challenges are provided. Second, the complete details of our own work is laid out.

A. Our System

For control mechanism a Raspberry Pi 2 board is used which equips an 700 MHz ARM1176JZF-S processor solely for image processing task and a STM32F407 board that includes an 168 MHz ARM processor solely to control the servo motors. The PID control and all other algorithms are implemented in pure C/C++ programming language which has maximized the performance of the system.

Actuation mechanism is based on RC servo motors which are connected to the beneath of the plate through arms and ball-sphere joints.

The ball position detection is done by capturing frames with a single Raspberry Pi camera module that is placed on top of the plate. Image processing techniques are used to detect the ball position per frame and send appropriate control signals to drive the servos accordingly.

978-1-4673-9675-2/15/$31.00 © 2015 IEEE

Authorized licensed use limited to: b-on: Instituto Politecnico de Braganca. Downloaded on October 23,2023 at 16:58:37 UTC from IEEE Xplore. Restrictions apply.

B. Mechanical Design

A 30cm x 30cm x 0.3cm acrylic plate fixed to a ball-sphere joint on its center. Each servo motor arm is connect to the beneath of the plate through an extended arm that ends with a ball-sphere joint attached to the plate. The servo motor 1 and 2 tilt the plate in x and y directions accordingly. That gives 2-degrees of freedom to the system. The camera module is located on top of the plate with separate support and adjustable arms let the camera position vary to get a full view of plate's surface.

1) Solidworks Modeling: The Solidworks software was used to verify the mechanical design by applying the physical constraints and check for the conflicting parts.

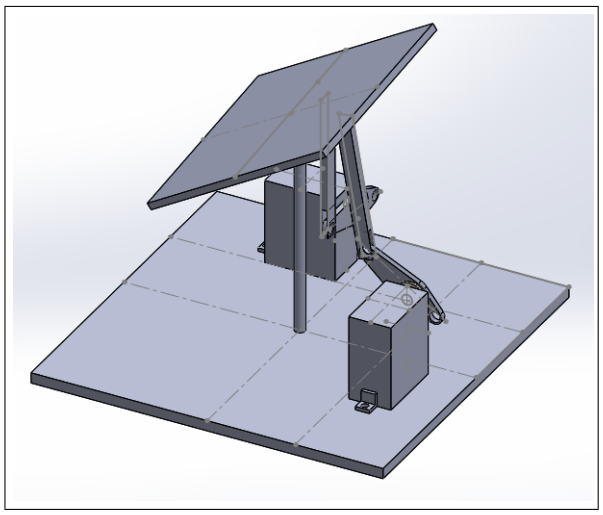

Fig. 1. CAD design, AU ball on plate balancing robot isometric view

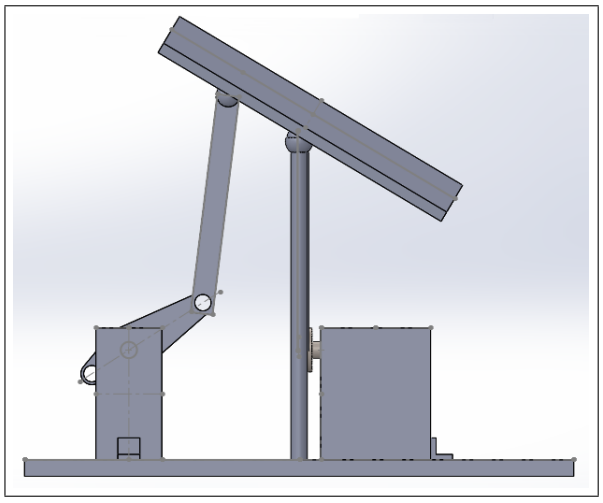

Fig. 2. CAD design, AU ball on plate balancing robot front view

Fig. 1 shows the 3D view of the system and how the plate is connected to the servo arms through ball-sphere joints. In Fig. 2 and Fig. 3 the front and left view of the design has been shown. The final assembly of all parts is shown in Fig. 4.

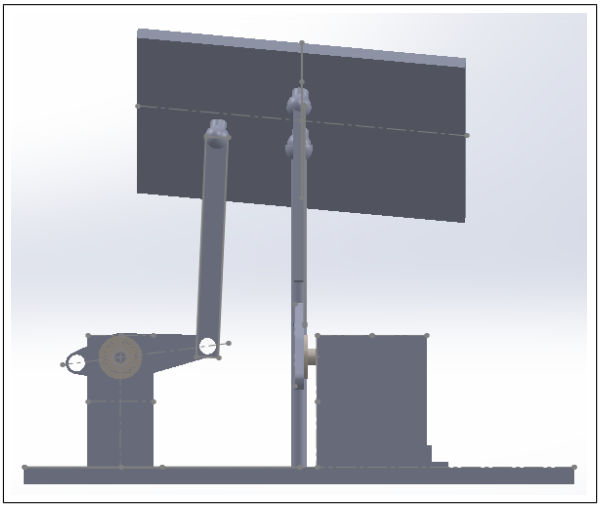

Fig. 3. CAD design, AU ball on plate balancing robot left view

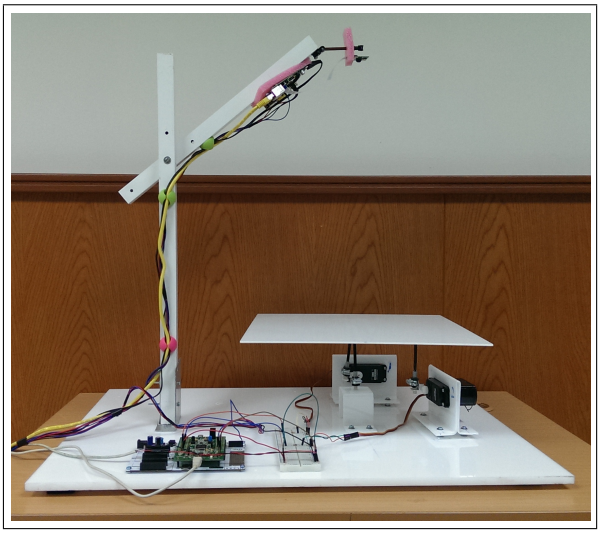

Fig. 4. The final assembly of all mechanical parts

III. CONTROL METHOD

The project is a free-model approach based on PID control. There are two PUs: (1) Raspberry Pi (2) STM32F407. Raspberry Pi has a dedicated five megapixel fixed-focus camera module that connects directly to the board through a CSI Port, and supports 1080p30, 720p60 and VGA90 video modes. OpenCV 2.0 library which is free for both academic and commercial purposes is used. The algorithm used to detect the 2D circle after capturing images from the ball is Hough transform algorithm. It is implemented in OpenCV 2.0 library under HoughCircles () function. After detecting the ball position the coordinates are transferred to STM32F407 board through a custom serial protocol.

The STM32F407 board receives the coordinates through a simple custom UART serial protocol and calculates both servo motor's arm angles considering a PID control mechanism. Next it drives the servos with standard 1.5 ms PWM (Pulse Width Module) signal. While STM32F407 is busy calculating the PID parameters, the other PU (Raspberry

Authorized licensed use limited to: b-on: Instituto Politecnico de Braganca. Downloaded on October 23,2023 at 16:58:37 UTC from IEEE Xplore. Restrictions apply.

Pi) captures another image and runs the Hough transform algorithm to detect the next ball coordinates and this cycle continues.

A. System flowchart and PUs relationship

As already mentioned the system contains two separate PUs: (1) Raspberry Pi board (2) STM32F407 board. Each board is equipped with a single core ARM processor. Image processing is completely done in Raspberry Pi module as it has a faster processor running at 700MHz. The flowchart of the major steps in Raspberry Pi module is shown in Fig. 5.

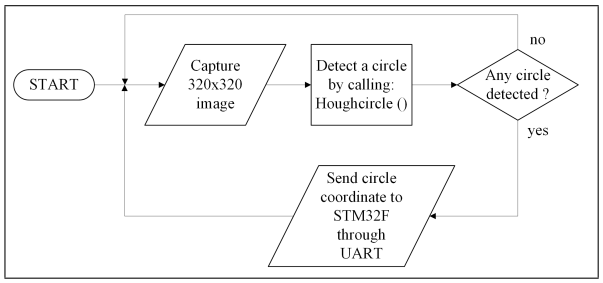

Fig. 5. Image processing flowchart in Raspberry Pi board

The system utilizes two RC servos to tilt and control the plate. In order to drive the servos the standard PWM pulses are used. A pulse of 1.5ms puts the servo arm at neutral position. The control over the servo arms are accomplished by increasing or decreasing of the PWM pulse length. The exact length of pulses are calculated by calling a function that receives the ball's coordinates as its arguments, and runs the PID control and outputs the calculated pulse length as its return value. Finally the PWM pulses will be applied to drive the RC servo motors and tilt the plate. The flowchart of all steps to drive the servos is shown in Fig. 6.

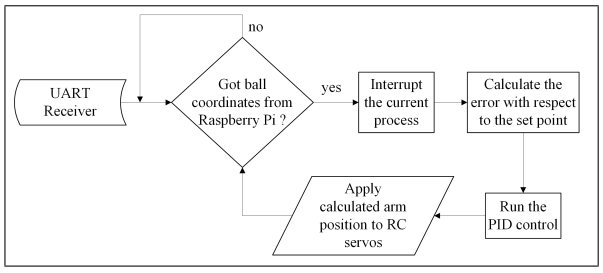

Fig. 6. PID control and servo motors drive flowchart in STM32F407 board

B. PID Control

The general PID control flowchart is shown in Fig. 7 We define the SP (Set Point) at coordinate (160, 160) which is the center of the plate. The disturbance is introduced by exerting a force on the ball alongside of an arbitrary direction. The sensor is the camera on top of the plate, and PV (Process Variable) refers to actual ball coordinate that is calculated through image processing.

To implement the above controller in C programming language the diagram needs to be converted into its mathematical formula and then an algorithm needs to be written

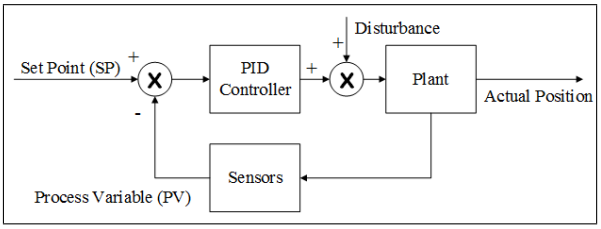

Fig. 7. General PID control block diagram

that can be fully implemented into machine code. Equation (1) shows the general PID control formula.

$$u(t) = Kp.e(t) + Ki\int_0^t e(\tau)d\tau + Kd\frac{d}{dt}e(t) \qquad (1)$$

Equation (1) consists of the summation of three terms. Each term has a coefficient with the following definition:

- Kp: Proportional coefficient that is proportional to the error.
- Ki: Integral coefficient that accounts for past errors.
- Kd: Derivative coefficient that predicts future errors.

The proportional term is just a simple multiplication that can be easily implemented into machine code using the C programming language. The integration term can be calculated using Euler integration technique by defining a small fixed time interval which is named dt. Image processing rate is at 15 FPS (Frame Per Second), it means that it approximately produces results which arrive every 66.67ms. The variable dt is set to 0.015 to ensure that upon arrival of each image processing result (as a ball coordinate) the system advances correctly. Therefore each run of the loop in PID control is considered to take 66.67ms. The area under the curve can be calculated if the instantaneous error regarding the set point is multiplied by dt. The accumulation of this value in each run of the loop gives the desired integration result with good enough precision. For the derivative term only to keep track of the previous error is sufficient. The newly calculated error is subtracted from the previous error, then the result will be divided by dt to get the rate of change of the error in each iteration of PID control loop.

C. Custom Serial UART Protocol

Raspberry Pi board has only one hardware based PWM module. That is a good example when the first PU lacks some resources and the task needs to be shifted to the next PU. The STM32F407 board in contrast supports up to four PWM counters. We have developed a simple custom protocol to be used in serial UART communication between the PUs.

The position of ball is represented as (x, y). Both x and y variables are 16-bit wide. The detail of the protocol with goal of transferring ball coordinates is shown in Table 1.

IV. EXPERIMENT AND RESULT

After plant construction and assembling all the modules, the highest achievable image processing rate is measured at 15 FPS. That gives a period of 66ms between each ball

Authorized licensed use limited to: b-on: Instituto Politecnico de Braganca. Downloaded on October 23,2023 at 16:58:37 UTC from IEEE Xplore. Restrictions apply.

TABLE I
CUSTOM SERIAL UART PROTOCOL

Definition	Byte value
Start flag (first position)	= 0xFFFF (2 byte)
x position, low byte	1 byte
x position, high byte	1 byte
y position, low byte	1 byte
y position, high byte	1 byte
Start flag (next position)	= 0xFFFF (2 byte)
x position, low byte	1 byte
x position, high byte	1 byte
y position, low byte	1 byte
y position, high byte	1 byte
Start flag (next position)	= 0xFFFF (2 byte)
and so on ...	

coordinate being sent to the second PU. The 66ms time period equals to roughly 11 million instructions for an ARM processor running at 168Mhz. This is far more than enough to let the second PU to calculate the PID algorithm, while giving the system acceptable precision to response in real time.

The PID coefficients were manually determined through trial and error. The tuning process follows the Ziegler-Nichols method. First the Ki and Kd are set to zero and Kp is increased to reach the desired proportional gain. Then Ki is increased to eliminate the oscillation about the SPs. Finally Kd is gradually increased to overcome overshooting and the fall of the ball.

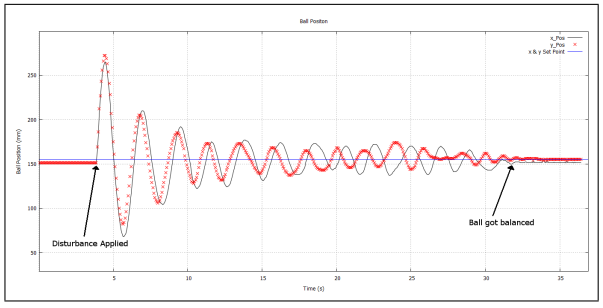

Fig. 8. PID Control Result: Ball position when Kp = 2.0, Ki = 0.2, Kd = 0.2

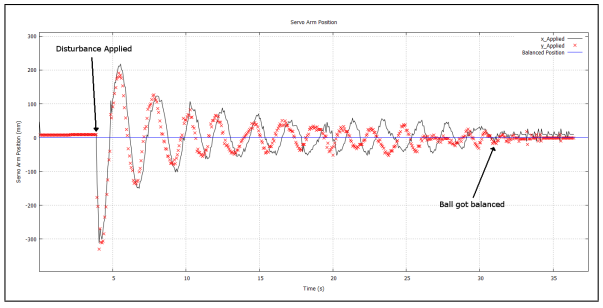

Fig. 9. PID Control Result: Servo arm position when Kp = 2.0, Ki = 0.2, Kd = 0.2

The monotonous curve in Fig. 8 refers to x and the red crosses refer to y coordinate of ball position. At first the ball is centered and then a disturbance is introduced by pushing the ball. As time progresses the system stabilizes the ball around the plate's center. In Fig. 9 the position of RC servo motors in respect to time are graphed. The monotonous curve refers to the servo motor 1 which applies force along x direction and the red crosses refer to servo motor 2 which apply force along y direction.

V. CONCLUSIONS

The overall system consists of two distinct processing units, one running the PID control and controls the RC servo motors while the other one does the image processing task to track the ball position. Both PUs are coupled via a serial UART communication channel.

The control mechanism is a free-model approach based on PID control. The PID tunning is done manually by applying Ziegler-Nichols method. The experiment is conducted by dropping a ball on the plate and waiting for the ball equilibrium around the center of the plate. A disturbance is introduced after having the ball stabilized and the motion of servo arms and ball coordinates were recorded and plotted. The image processing rate up to 15 FPS is achieved. The ball approximately stabilizes after 8 seconds and comes to stand still position after 28 seconds.

In conclusion it should be emphasized that in control systems acceptable real time performance can be achieved by decentralizing the processing unit into several PUs. It is notable to mention that with small additional piece of software and control checkpoints PU redundancy feature can be added to the plant, so in the event of physical damage and PU failure the other PU can be notified and take over the system.

ACKNOWLEDGMENT

We would like to thank Dr. Kittiphan Techakittiroj the dean of Engineering Faculty at Assumption University of Thailand for providing insights, comments and encouragement while overseeing the project. Without his precious support it would not be possible to conduct this research.

REFERENCES

[1] M. Nokhbeh and D. Khashabi, Modeling and Control of Ball-Plate System, Amirkabir Univ. of Technology, 2011.
[2] S. Awtar and C. Bernard, Mechatronic design of a ball-on-plate balancing system, Dept. Mech. Eng., Rensselaer Polytechnic Institute, Troy, USA, 2002.
[3] J. Bruce and C. Keeling and R. Rodriguez, Four Degree of Freedom Control System Using a Ball on a Plate, Southern Polytechnic State Univ., 2011.
[4] N. Wettstein, Balancing a Ball on a Plate using Stereo Vision, M.S. Thesis, Institute for Dynamic Systems and Control Swiss Federal Institute of Technology (ETH) Zurich, 2013.
[5] A. Zeeshan and N. Nauman and M. Jawad Khan, Design Control and Implementation of a Ball on Plate Balancing System, Dept. of Mech. College, National Univ. of Sci. and Tech. Rawalpindi, Pakistan, 2012.
[6] X. Dong and Z. Zhang, Applying genetic algorithm to on-line updated PID neural network controllers for ball and plate system, Forth International Conference on Innovative Computing Information and Control, 2009.
[7] P. Yip, Symbol-based Control of a Ball-on-plate Mechanical System, M.S. thesis, Dept. of Mech. Eng., Univ. of Maryland, 2004.
[8] E. Ali and N. Aphiratsakun, AU Ball on Plate Balancing System, B.Eng. in Computer Sys. senior project, Assumption Univ., Bangkok, Thailand, 2015.

Authorized licensed use limited to: b-on: Instituto Politecnico de Braganca. Downloaded on October 23,2023 at 16:58:37 UTC from IEEE Xplore. Restrictions apply.