KU LEUVEN

FACULTY OF ENGINEERING TECHNOLOGY

GROUP T LEUVEN CAMPUS

Implementing an Automated Table Tennis System

GAME_7

Leo CAERS

Tuur COLIGNON

Hugo FACHE

Emilio JACQUIER

Yusuf HUSSEIN

Coach: Jit Chatterjee

Paper submitted in partial fulfillment of the requirements of Engineering Experience 3 - Electronics and ICT Engineering

Academic Year 2024-2025

KU LEUVEN

Faculty of Engineering Technology Group T Leuven Campus — Paper submitted in partial fulfillment of the requirements of Engineering Experience 3 - Electronics and ICT Engineering
Academic Year 2024-2025

Implementing an Automated Table Tennis System

GAME_7

Hussein Yusuf, Jacquier Emilio, Fache Hugo, Colignon Tuur, Caers Leo

Bachelor in Engineering Technology: Electronics and ICT Engineering, Faculty of Engineering Technology, Group T Leuven Campus, Andreas Vesaliusstraat 13, 3000 Leuven, Belgium

Coach: Jit Chatterjee

Engineering Technology: Electronics and ICT Engineering, Faculty of Engineering Technology, Group T Leuven Campus, Andreas Vesaliusstraat 13, 3000 Leuven, Belgium, jit.chatterjee@kuleuven.be

ABSTRACT

This paper proposes a computer-vision based system to automate a ping pong system capable of competing in real-time with a human player. The system will incorporate a simple upright gantry mechanism for precise and efficient 2D control. Tracking the ball is achieved via a dual-camera setup with a top and side view, to collect and compute 3 dimensional positional information for the ball. By integrating predictive algorithms to analyze the ball's trajectory and speed, the paddle is able to respond with accuracy and return the ball. The proposed system is cost-effective, using readily available components such as consumer grade servos, a solenoid, and stepper motors. Furthermore, a supplementary companion app is provided which provides relevant user data, tracking information, and game information.

1 INTRODUCTION

Table tennis, colloquially known as ping-pong, is a dynamic sport with rapid ball movements, complex spin mechanics, and the need for precise timing. The increasing sophistication of robotics and computer vision systems has created opportunities for designing autonomous systems capable of participating in such high-speed, interactive environments. Among these innovations are robotic ping-pong players, which serve as a compelling testing environment for the advancing motion control, real-time object tracking, and predictive modeling.

Several commercial and research-driven systems have attempted to tackle the challenges of creating robotic ping-pong players. For instance, Omron's Forpheus employs stereo cameras and advanced trajectory prediction to deliver high-level gameplay and even assess player skill to offer personalized challenges.Kyohei et al. (2019) However, such systems often rely on expensive mechanical arms and high-performance sensors, making them inaccessible for low-cost, consumer-grade applications.

In this paper, we propose a cost-effective robotic ping-pong system designed to replicate the core functionalities of these high-end solutions while maintaining simplicity in design and affordability in materials. The system leverages a dual-camera setup to track the ball's position in three dimensions and employs predictive algorithms to calculate its trajectory. A rail-and-belt mechanism with a mounted paddle responds in real time to intercept and return the ball to the player. The simplicity of the hardware design, combined with the adaptability of the vision-based tracking system, ensures ease of deployment and operation for a wide range of users.

The research aims to address two core challenges: the optimization of vision-based tracking for real-time gameplay and the balance between mechanical speed and control precision for consistent performance. By exploring these aspects, the project contributes to the broader field of robotics by demonstrating how accessible technology can deliver effective solutions in high-speed, interactive scenarios.

1.1 Related Works

A lot of companies have already created their own ping pong robot, the mechanical arm seems to be a necessity for an efficient 3D movement for the robot. The camera disposition differs between the projects. The Forpheus built by Omron, Kyohei et al. (2019), uses stereo camera to be able to see all three dimensions and make an accurate prediction of the ball position. Additionally, it uses another camera to track the racket and player's movement to be able to determine the effect on the ball and predict where the ball will land.

This system is very high budget, and as a result our implementation will not achieve such results. The mechanical arm is out of reach for this prototype with the budget and time given, but it is an effective example for the robot movements. The same can be said about the cameras. The top view with an angle provides a 3D visual of the ball movement only using two cameras.

The most relevant article is Acosta et al. (2004), as it goes through the whole procedure up to the fully realized prototype. Even though technology advancement has largely improved since then, the book provides many insights into possible issues and solutions for them.

Our main idea is to detect the ball and return it using a rigid rectangle cardboard and a solenoid. Even though, the design of the book Yu et al. (2012) is more complex, it provides some ideas on real-time control setups and how it can be incorporated into the design.

The tracking of the ball will probably be one of the biggest issues, a fuzzy image, camera positioning, depth track, or even the effect the ball could have from the player. All of this makes having an accurate prediction of the ball position very challenging. A lot of scientific articles discuss these possible issues we might encounter.

Depending on the quality of the cameras, the image might be blurry, or due to timing and noise issues, the data might be affected. This can be solved using the Kalman filtering algorithm Lu (2020).

A mathematical solution using a K-means algorithm, Fourier series, EM and others, can be used to have an accurate prediction on the ball position if some effect was applied Zhao et al. (2017).

Hitting the ball with a correct trajectory will be found through trial and error. The article Trasloheros et al. (2014). explains how they manage to properly hit the ball only using three degrees of freedom whereas most other companies use at least 4. This could facilitate our task and give us some good ideas about the proper angle to return the ball.

1.2 Requirements analysis

The system targets individuals seeking a challenging ping-pong opponent, regardless of their skill level or the availability of human players.

The reason for such a broad demographic is because the ping-pong is very straightforward, and easy to play. It can

be a 90-year-old grandpa or his 4-year-old granddaughter that plays against the robot. Players benefit from the challenge of playing against a robot and whilst exercising at the same time.

A finished prototype, with every component working, should be able to handle a back and forth game of ping pong with ball speeds up to at least 15 m/s consistently. To achieve this all of these working components will be required to be reliable and fast.

The railing system must achieve rapid positioning while maintaining stability, avoiding excessive acceleration that could compromise the system's performance.

The image tracking software should be able to accurately track the ball's position at different heights and angles. Ideally the camera will always be mounted at the exact same position. However, in reality this is never the case. That is why the software that handles the ball tracking will need to be calibrated every time you set up the system. This should all happen automatically using reference points on the table to ensure a fast, and user friendly setup of the prototype.

Factors like unfamiliar lighting or human interference could make the ball tracking unreliable at times. However, this will not be taken into account when making the prototype.

The prediction model of the system is never going to be a perfect representation of the real world's physics. That is why it should be able to update its prediction with each frame of new information it gets from the tracking software. This way, any inaccuracy in the prediction model can be reduced and the racket will be able to be repositioned if any unexpected behaviour were to show up.

2 DESIGN AND MATERIALS

In this section we will discuss the design of our mechanical, electronic, and software systems and the materials used.

2.1 Mechanical design

The mechanical design consists of three main components: the horizontal movement, the vertical movement, and the rotation of the racket. The materials utilized are a combination of purchased components such as linear rails, timing belts, bolts, and nuts alongside custom 9MM MDF parts cut using a laser cutter in FabLab.

The two 1.5-meter-long linear rails are positioned vertically, parallel to each other, along the table's edge to ensure stability and prevent tipping of the system. Three linear rail blocks, which slide along these rails, support the vertical structure of the design. These rail blocks have screw holes, allowing them to be securely connected in a triangular formation using an MDF plate. The vertical structure is mounted on this plate.

A stepper motor is mounted at one corner of the table, fixed to the edge of the rails using MDF. This motor drives a 60-tooth GT2 gear. At the opposite corner, a shoulder bolt with a gear is similarly secured using MDF. A 6mm-wide GT2 timing belt spans between these two gears, and is attachted to one of the horizontal rail blocks, allowing horizontal movement by rotating the stepper motor.

The vertical structure is mounted on the horizontal MDF plate, attached to the horizontal linear blocks. It includes a 0.5-meter-long aluminium pipe and a vertical MDF plate that stabilizes the pipe. A gear is placed on top of the vertical plate and pipe, held in place with a shoulder bolt. A vertical rail block is positioned on the pipe, allowing vertical movement. Another stepper motor is attached at the bottom of the horizontal MDF plate, with a gear connected to it. A timing belt spans between these two gears and is attached to the vertical rail block, allowing vertical movement of the block by rotating the stepper motor.

To rotate the racket, a servo motor is attached to the vertical rail block using MDF. A solenoid is mounted on the rotating components of the servo motor, and the racket is attached to a small MDF piece connected to the solenoid.

This mechanical design enables the racket to move left and right, up and down, rotate 90 degrees around its axis, and push the ping-pong ball.

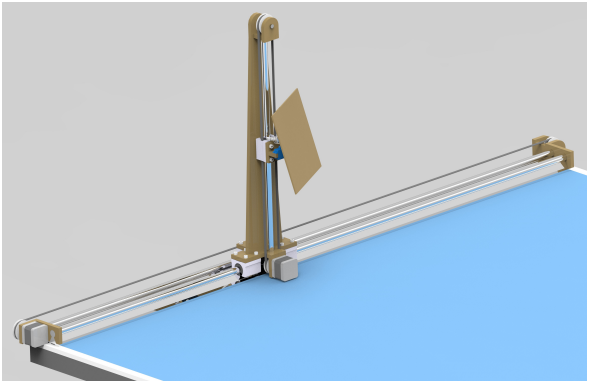

Figure 2.1: A render of the mechanical design from the front.

2.2 Block Diagram

Below is a block diagram showing the general control flow of the entire system and all its components.

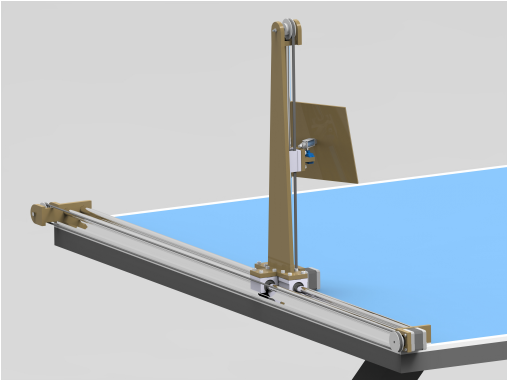

Figure 2.2: A render of the mechanical design from the back.

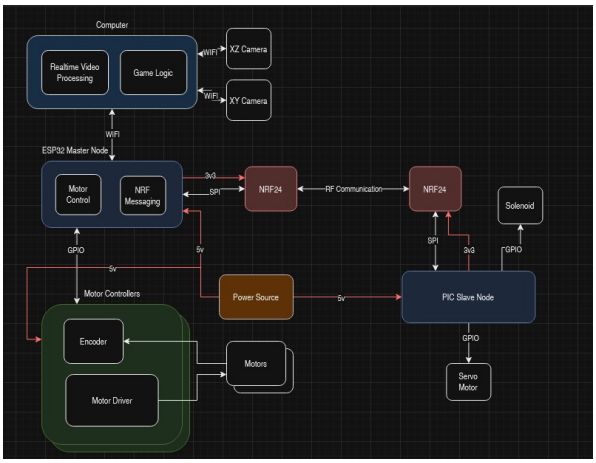

Figure 2.3: System Design Block Diagram

2.3 Electronic Components and Controllers

The system mainly relies on two mobile phones which are used for collecting the position information of the ping pong ball with three degrees of freedom. The mobile phones are configured with an app to act as IP cameras, which are then accessed over Wi-Fi on the control device, which is a laptop running our software.

The ESP32 microcontroller is connected to two stepper motor driver modules (DRV8825), which are in turn connected to two bipolar stepper motors which control the gantry. The ESP32 acts as a master to a PIC18F, though a connected radio frequency communication module (NRF24), which is used to transmit data to another NRF24 module, connected to the PIC18F slave microcontroller. This data consists of information on when to ”fire” the racket to hit the ball, by activating the optocoupled 12V solenoid, and information on what angle the racket should be at, by controlling the 5V servo motor controlling the racket.

2.4 Software Design

The main software which controls the system is ran on a separate computer, due to the speed constraints of small microcontrollers. The code connects to the IP camera phones, reads the streamed video data, and process it using a ball detection algorithms to determine the position of the ping pong ball in 3D space. The software sends a requested gantry position, a requested racket orientation, and when the racket should ”fire” over Wi-Fi to the ESP32 microcontroller, which then sends a part of this data over radio using the NRF24 to the PIC18F.

The software additionally maintains the game state through a simplified ruleset, allowing the score and current ball position to be visualized in real time. The software determines the ball’s position and additionally applies a predictive model to determine where the gantry should move to.

2.5 Bill of Materials

Item	Amount	Unit Price (€)
MDF (60x30)	1	4
5meter 6mm Timing Belt	1	6.62
GT2 60-tooth gear	4	3.19
0.5meter M16 Aluminium pipe	2	5.9
1.5meter SBR16 Linear rail	2	20.46
D5 M4 50mm Shoulder bolt	2	2.94
Servo Motor	1	3.5
Solenoid	1	6.9
ESP32	1	6.9
PIC18F	1	2.4
NRF24	2	6.68
Stepper Motor	3	-
Stepper Driver DRV8825	3	13.5
Optocoupler	1	-
Transistors	-	-
Resistors	-	-
Capacitors	-	-

3 IMPLEMENTATION

3.1 Hardware implementation

TEXT HERE

3.1.1 Printed Circuit Boards

Our design additionally consists of two Printed Circuit Boards (PCBs), one for each used microcontroller. The

first PCB designed is for the PIC18F slave microcontroller. It contains connections for the solenoid, and its control circuit which consists of an optocoupler and a Darlington NPN bipolar transistor (TIP120). It also contains connections for the servo motor and its control circuit which is a simple MOSFET transistor circuit and contains connections for the NRF24 module.

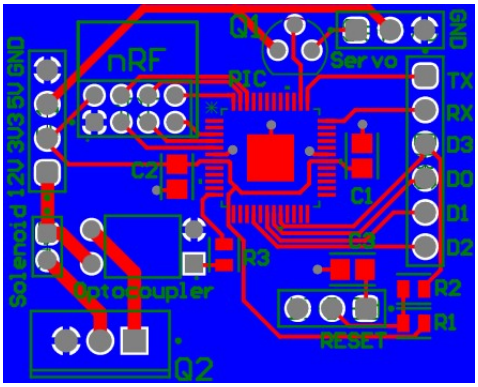

Figure 3.1: PIC18F PCB Design

The second designed PCB is for the ESP32 master controller. This PCB consists of headers for the ESP32 controller itself, and headers for each of the DRV8825 modules. It has a switch for on board adjustment of the drivers' control pins. It also contains connections for the NRF24 module.

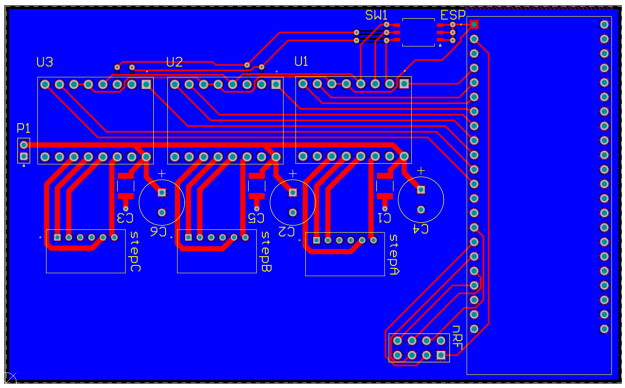

Figure 3.2: ESP32 PCB Design

3.2 Software implementation

The embedded software system is developed for an ESP32-based real-time application involving motion control and wireless communication. The design follows a modular, task-based architecture utilizing the FreeRTOS operating system, allowing concurrent execution of multiple subsystems and ensuring responsiveness to real-time events.

3.2.1 Architectural Overview

The system architecture is composed of three primary subsystems: a communication subsystem, a motion control subsystem, and a simulation/testing subsystem. These subsystems interact through well-defined interfaces and communicate via FreeRTOS message queues, allowing asynchronous data exchange and decoupling between components.

3.2.2 Communication Subsystem

The communication subsystem is responsible for receiving external control commands via UDP over WiFi and transmitting actuator signals through an NRF24L01 radio interface. The UDP client manages socket creation, data reception, and error handling, converting received commands into structured messages. These messages are placed into dedicated queues for consumption by the motion and actuator controllers. The NRF24L01-based radio communication provides an additional, low-latency communication channel for controlling remote actuators, contributing to the system's flexibility and scalability.

3.2.3 Motion Control Subsystem

The motion control subsystem handles the operation of two stepper motors along orthogonal axes. Motor control is implemented using hardware timers configured to trigger interrupts at precise intervals. Within the interrupt service routines, step signals are toggled, positions are updated, and step frequencies are adjusted according to desired momentum and target positions. The use of easing functions ensures smooth acceleration and deceleration, minimizing mechanical stress and enabling precise control. Position and speed adjustments are driven by messages received from the communication subsystem, allowing dynamic and responsive actuation.

3.2.4 Simulation and Testing Subsystem

To facilitate development and validation in the absence of hardware, the system incorporates a simulation and testing subsystem. Mock tasks generate synthetic messages simulating both UDP communication and actuator commands. This feature enables rigorous testing of system behavior, ensuring robustness and correctness prior to hardware deployment.

3.2.5 Inter-task Communication and Decoupling

The system employs three dedicated message queues for horizontal stepper control, vertical stepper control, and racket actuator commands. These queues enable asynchronous, thread-safe communication between tasks. The use of structured message formats for stepper and actuator commands maintains consistency and simplifies message handling logic, supporting scalability and modular expansion.

3.2.6 Modularity and Configurability

The firmware includes compile-time configuration flags to enable or disable subsystems such as UDP communication, radio communication, stepper control, and mock simulation tasks. This configurability allows the firmware to be adapted for various development and deployment scenarios, including partial module testing, hardware-in-the-loop simulations, or full production operation.

3.2.7 Design Rationale

The overall system design reflects a deliberate emphasis on separation of concerns, real-time responsiveness, scalability, and testability. The modular task-based architecture and asynchronous queue-based messaging enable the system to maintain predictable real-time behavior while remaining adaptable to future expansions or modifications. The inclusion of simulation capabilities addresses the need for verification and debugging in complex embedded systems where hardware access may be limited during certain development phases.

3.2.8 Conclusion

In conclusion, the software architecture presented here demonstrates an effective approach for real-time motion control and communication in an embedded environment. Through its modular design, task-based concurrency, and flexible configuration, the system achieves a balance between robustness, maintainability, and adaptability, making it well-suited for both research and applied deployment in embedded control scenarios.

4 EVALUATION AND VALIDATION

Important: do not introduce new designs, materials or implementation methods in this section.

Table 4.1: An example table

Day	Min Temp (°C)	Max Temp (°C)
Monday	11	22
Tuesday	9	19
Wednesday	10	21

5 DISCUSSION

Critical reflection is important in this chapter.

6 CONCLUSION

[TEXT]

ACKNOWLEDGEMENTS

[TEXT]

LIST OF SYMBOLS

Symbols

λ_{ex}	Excitation wavelength
λ_{em}	Emission wavelength
$N(\dots)$	Noise process

Acronyms

ADC	Analog to Digital Converter
CCD	Charge-Coupled Device

BIBLIOGRAPHY

Acosta, L., Rodrigo, J., Mendez, J. A., Marichal, G., and Sigut, M. (2004). Ping-pong player prototype. *Robotics & Automation Magazine, IEEE*, 10:44 – 52.

Kyohei, A., Masamune, N., and Satoshi, Y. (2019). The ping pong robot to return a ball precisely trajectory prediction and racket control for spinning balls.

Lu, C. (2020). Kalman tracking algorithm of ping-pong robot based on fuzzy real-time image. *Journal of Intelligent & Fuzzy Systems*, 38:3585–3594. 4.

Trasloheros, A., SebastiÃ¡n, J. M., Torrijos, J., Carelli, R., and Roberti, F. (2014). Using a 3dof parallel robot and a spherical bat to hit a ping-pong ball. *International Journal of Advanced Robotic Systems*, 11(5):76.

Yu, X., Zhao, Y., Fu, C., and Chen, K. (2012). Research and development of a ping-pong robot arm. In Lee,

G., editor, *Advances in Automation and Robotics, Vol.1*, pages 65–71, Berlin, Heidelberg. Springer Berlin Heidelberg.

Zhao, Y., Xiong, R., and Zhang, Y. (2017). Model based motion state estimation and trajectory prediction of spinning ball for ping-pong robots using expectation-maximization algorithm. *Journal of Intelligent & Robotic Systems*, 87(3):407–423.

APPENDICES

APPENDIX A : SOME EXTRA INFO

Delete the appendix if not needed.