Volleyball Training System

Kelsie Morris, Electrical Engineering

Project Advisor: Dr. Tony Richardson

April 17, 2019

Evansville, IN

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

I. INTRODUCTION

A common problem for volleyball players is the ability to practice by oneself or with few people. Having someone there to toss the ball for practice is not always convenient, and can sometimes be a lot of effort for the person tossing. A volleyball training system would take away this problem for both the player and the coach. This project was designed and constructed at an affordable price and will give student athletes the chance to practice on their own time; during practice it will put less stress on the coach and allow him/her to focus on adjusting the athlete's form rather than focus on tossing. This project was aimed to be a fully functional system, but in prototype form.

II. PROBLEM DEFINITION

There are some volleyball training systems already on the market, but those require manual loading and some require aiming. The "best volleyball throwing machine on the market today" is shown in Figure 1.

Figure 1: Volleyball throwing machine sold on the market today. [1]

The cost of these machines run anywhere from $1300-$4000, which is not affordable by many schools or athletes. The machines also require a separate cart to hold the volleyballs, and are not programmed or designed to run on their own.

This new training system is more affordable for customers so that more players can have access to it. It can be loaded with multiple balls and is installed with different types of programs and settings. This eliminates the need for someone to constantly be at the machine一why purchase something like that when it still requires constant interaction? The training system has different settings for passing. The user can choose what types of “serves” for the machine to perform: the first selection is between the speed modes and the second selection is between the direction modes; both selections offer the option of “uniform” or “variable.” Since this system is a prototype, ping pong balls are used instead of volleyballs.

III. CLIENT REQUIREMENTS

- ○ User interface
- ○ Ability to launch ball across the net
- ○ Ability to change horizontal angle
- ○ At least two settings
- ○ Ability to hold 10+ balls

IV. PROJECT DESIGN

A. OVERVIEW

A significant design constraint was the easy mobility of the system. Wood was chosen as the enclosure material due to its easy alteration, the tools available to the designer, and its potential to be lightweight. PVC was chosen as a vessel/track for the ping pong balls; piping was available for purchase at the exact diameter needed. When thinking of launch, a method was chosen that was more electrical-based rather than mechanical. The ping pong balls roll into spinning rubber wheels that shoot the balls out of the end of the piping, with the wheel speeds controlled by a microcontroller. To add more work and capabilities, the base of the system rotates in order to change the ball's direction; the system is attached to a servo motor follows commands of the microcontroller. Another key design to the system is the user interface, involving two buttons for selection of practice modes.

B. HARDWARE

The ARM Cortex M4 Nucleo Board (Figure 2) is used to control the user interface, both DC motor speeds, and both servo motors. The Nucleo was chosen over other microcontrollers because of its low cost ($20) and ability to perform every task needed. The Nucleo is connected to the user interface, which is two buttons. These are used as the method for the user to access the system. The buttons represent two different practice modes. The first mode has uniform direction with varying speed; the second mode has varying direction with varying speed. Based on the user's selection, the system rotates to the desired position. The rotation is performed by a servo motor that is stationed so that the arm of the servo is just above the surface of the base of the box, as shown in Figure 3. A 3D printed piece sits directly on top of the arm in order to attach it to the base of the turntable. A lazy susan is used to assist the servo when rotating.

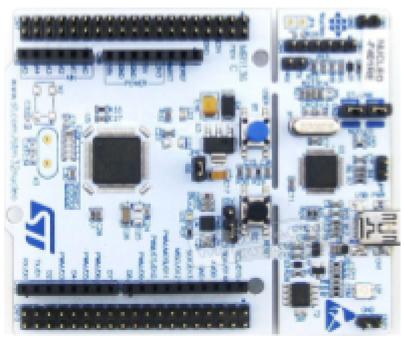

Figure 2: STM32 Nucleo [2]

Figure 3: Base Servo Motor [3]

Once the selection is made, the system gives the user time to get in position and the DC motors that are placed on either side of the PVC start up for the launch. After a set time has passed, a single ball is released by a micro servo motor that is fastened to the outside of the PVC with a hose clamp. The PVC is cut so that the micro servo arm can rotate in and out of the piping. Once a ball is released it travels inside the PVC pipe and into the wheels attached to the motors. Two slots are cut out near the end of the pipe for the wheels to connect with the ball. The DC motors for the wheels were chosen based on the RPMs needed to launch the ball across the court. A small 3D printed piece was designed to attach the wheels to the motors.

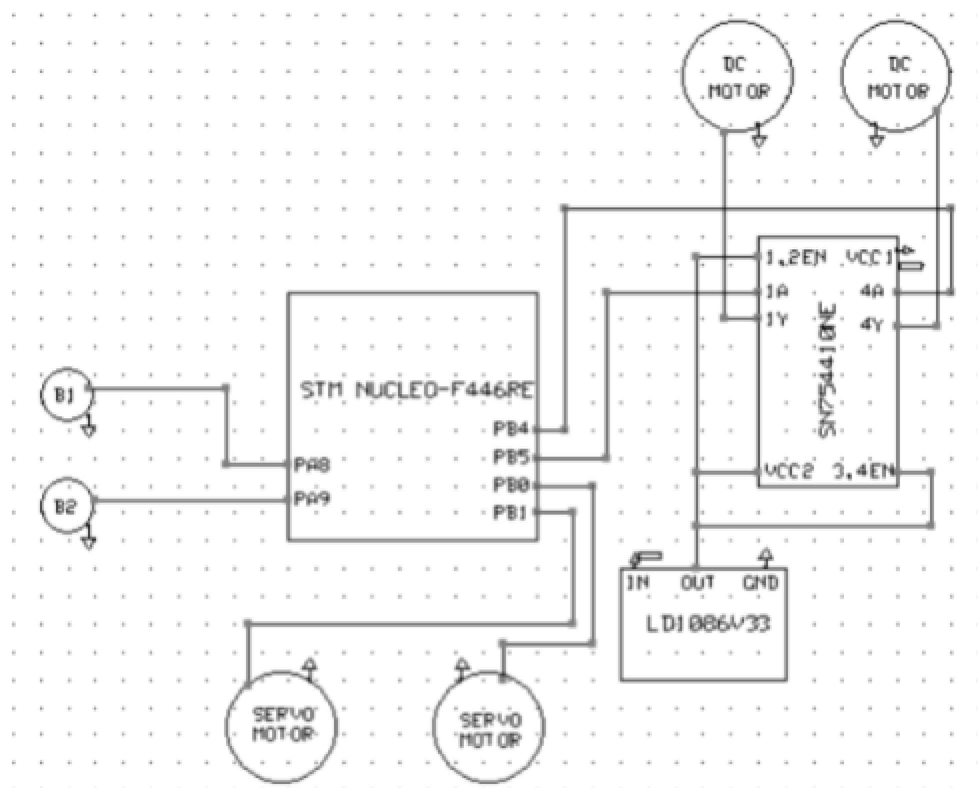

Figure 4: Hardware Schematic

Since this project is a prototype, it does not require high voltage or high current. The actual model, however, would involve those so this project is built with that in mind. A rechargeable battery is used to power the system; this increases the range of mobility and is more eco-friendly than using regular batteries.

C. SOFTWARE

Each button represents a practice mode. After one is pressed, a function is called for the DC motors to begin running. The micro servo is programmed to rotate and release a ball periodically to give the user time to react and get back in position. The program is set so that the servo does this 10 times, since that is the max that the vessel can hold. Once this is completed, the program resets and waits for the user to select another practice mode. If practice mode 1 is chosen, the base servo motor rotates to the straight forward position and remains there throughout the practice mode. If practice mode 2 is chosen, then the PWM sent to the base servo is randomized for each launch. A brief pseudocode follows:

```
MODE 1 - UNIFORM
        DC MOTORS START
        NO CHANGE IN DIRECTION
        BALL RELEASE x10
        MOTORS STOP
MODE 2 - VARIABLE
        DC MOTORS START
        RANDOM DIRECTION CHANGE x10
        BALL RELEASE x10
        MOTORS STOP
```

D. TESTING

Testing for this system began with the DC motors to ensure the H bridge and regulator functioned properly. The DC motors were then tested along with the switches to ensure they ran when and how they were supposed to. The next step was testing the base of the system to see how far it needed to turn in either direction and to ensure that it was stable enough to support the system on top of it. Once this stage was completed, the base servo was added into testing with the switches and DC motors. The motorized wheels were then attached and tested to determine their ideal placement and how they should be attached to minimize error.

E. DESIGN STEPS

The first task was drawing a model of the training system. CAD was not used since the project was not intended to be 3D printed. Once satisfied with the design, the individual parts were ordered/purchased. The PVC section of the system was built first. This is the largest part of the system, so it allowed everything else to be built around it. Once this was completed, the turntable base was constructed and the PVC section was attached to it with an adjustable pipe hanger.

The enclosure was then cut and put together, leaving two sides open to complete the work inside. Five holes were then created: one on top to give access to PVC that holds the balls; two on the front for the user interface and power switch, one on the back for the balls to shoot out of, and one on the bottom for the servo motor to sit in. The servo was positioned and caulk was filled in around it to stabilize it. A lazy susan was screwed into the base of the enclosure, with the servo centered in the middle of it. The turntable base was then attached to the lazy susan with bolts and nuts. Metal brackets were then constructed to attach the DC motors to the top of the turnable.

A scrap sheet of PVC was then cut and used to create a surface in which to attach the entire user interface (LCD and button switches). This was fastened to the front of the enclosure, placed over the hole made previously.

Codes were then written for testing of each area of the system. The first was written for the turning of the base. Once it was stabilized and running properly, the motorized wheels were tested and then put into place and tested again to ensure that the wheels were positioned correctly to perform an effective launch. Once that all was functioning properly, code was written for the user interface to control the commands to the other components.

F. COSTS

The price of this system was fairly inexpensive due to it being a prototype. However, the goal was to design this in the same way a full-size system would be designed so the prices should be somewhat to scale with the size. In order for this system to be available to anyone (schools or students) it had to be significantly cheaper than the ones on the market. The system designed consists of wood and PVC ($55), hinges and a hook ($5), a lazy susan ($6), 9g servo motor ($4), 20kg servo motor ($18), microcontroller ($20), DC motors ($4), rubber wheels ($5), battery ($20), and ping pong balls ($7), buttons ($9), PCB and wires ($7), which totals to $160. The cost of mass-producing a fully developed system would likely decrease the cost. The bill of materials is summed up in Table 1.

Table 1: Bill of Materials

Part	Quantity	Cost	Item Total
Wood & PVC		55	55
Hinges & Hook	1	5	5
Lazy Susan	1	6	6
9g Servo Motor	1	4	4
20kg Servo Motor	1	18	18
Microcontroller	1	20	20
DC Motors	2	2	4
Rubber Wheels	2	2.50	5
Rechargeable Battery	1	20	20
Ping Pong Balls	12	.58	7
Buttons	2	4.50	9
PCB & Wires		7	7
		Total:	160

V. RESULTS

The system satisfies all of the client requirements. Many minor inconveniences were exposed throughout the design work, but no major changes had to be made. The original estimated cost was about $141 which is very close to the final total of $160. The final results are pictured in Figures 5-8.

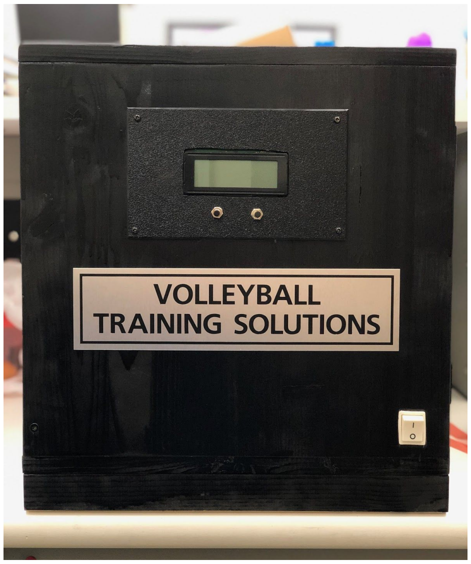

Figure 5: Front of System

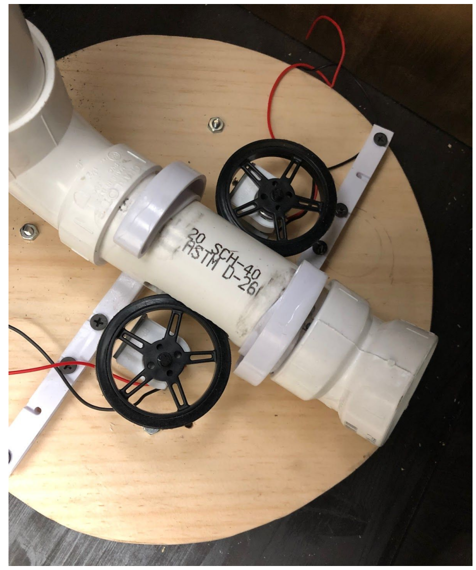

Figure 6: Overhead View of Inside

Figure 7: Top View & Ball Vessel

Figure 8: Inside View

VI. CONCLUSION AND RECOMMENDATIONS

The project functions as desired for the minimum requirements made. It was realized that a 3D printed enclosure would have been much more compact and efficient, but that was realized too late. A microswitch could have been used to detect whether a ball was in the vessel instead of using a preprogrammed servo.

VII. REFERENCES

[1] "Total Attack Volleyball Machine." Total Attack Volleyball Machine - AchillionSports, www.achillionsports.com/total-attack-volleyball-machine

[2] "NUCLEO-F446RE ." STM32F0DISCOVERY - Discovery Kit with STM32F051R8 MCU - STMicroelectronics, www.st.com/content/st_com/en/products/evaluation-tools/product-evaluation-tools/mcu-eval-tools/stm32-mcu-eval-tools/stm32-mcu-nucleo/nucleo-f446re.html#samplebuy-scroll.

[3] "ANNIMOS 20KG Digital Servo High Torque Full Metal Gear Waterproof for RC Model DIY, DS3218MG,Control Angle 270°." Amazon. Amazon. 29 Apr. 2019 <https://www.amazon.com/gp/product/B076CNKQX4/ref=ppx_yo_dt_b_asin_title_o05_s00?ie=UTF8&psc=1>.

APPENDIX A

```
/* KELSIE MORRIS

MODE1 PB0
MODE2 PB1
BALL RELEASE SERVO PC7
BASE SERVO PC6
DC MOTORS PA6 & PA7

*/

#include "stm32f446.h"
#include "stdlib.h"

void Init_Tim3(void);
void Init_GPIO(void);
void Start_Motors(void);
void Stop_Motors(void);
void CheckButton(void);
void Mode1(void);
void Mode2(void);
void Delay(unsigned int z);
void RandDirection(void);

int ButtonPressed = 0;
int ButtonPicked = 0;
int LaunchDirection;

int main()
{
  Init_GPIO();
  Init_Tim3();

  while(1)
  {
        while (ButtonPressed != 1)
        {
                CheckButton();
        }
                if (ButtonPicked == 1)
                {
                        Mode1();
                }
                else if (ButtonPicked == 2)
```

```
                {
                        Mode2();
                }
 }

}

void Init_GPIO(void)
{
 RCC_AHB1ENR |= 0x3F;
 GPIOC_MODER |= 2<<(2*7);          //PC7 Set to Alternate Function
 GPIOC_AFRL |= 0x20000000;         //PC7 Set to Alt. Funct. 2 - TIM3 Ch2
 GPIOC_OTYPER |= 1<<7;             //PC7 open drain
 GPIOC_MODER |= 2<<(2*6);          //PC6 Set to Alternate Function
 GPIOC_AFRL |= 0x2000000;          //PC6 Set to Alt. Funct. 2 - TIM3 Ch1
 GPIOA_MODER |= 1<<(2*5);          //PA7 output to dc motor
 GPIOA_MODER |= 1<<(2*4);          //PA6 output to dc motor
 GPIOB_PUPDR |= 0X05;              //PB0 & PB1 PULL-UP
}

void Init_Tim3(void)
{
 RCC_APB1ENR |= 1<<1;              //Timer 3 clock enable
 TIM3_CCMR1 |= 0x6C6C;   //PWM mode output compare 1, preload and fast enable for Ch1,2
 TIM3_CR1 |= (1<<7);               //ARPE Pg 526
 TIM3_PSC |= 15;                   //16 Mhz/15+1 = 1 MHz
 TIM3_ARR |= 19999;                //PWM Period = (19999 + 1) * (1/1Mhz) = .02Sec
 TIM3_CCR1 |= 0;                   //Duty cycle starts at 0
 TIM3_CCR2 = 900;                  //Start_Motors servo in way of balls
 TIM3_CCER |= 0x1111;              //Capture/Compare 1 output enable for Ch1,2,3,4
 TIM3_EGR |= 1;                    //Update generation
 TIM3_CR1 |= 1;                    //Counter enabled
}

void Mode1(void)
{
        TIM3_CCR2 = 950;
        Start_Motors();
        Delay(50000);//Delay to let motors get up to speed
        TIM3_CCR1 = 750; //Base straight forward
        Delay(100000);
        for(int x; x<10; x++)
        {
                TIM3_CCR2 = 950;
                Delay(40000);
```

```
			TIM3_CCR2 = 650;
			Delay(750);
		}
		TIM3_CCR2 = 950;
		Stop_Motors();
		Delay(50000);
		ButtonPicked = 0;
		ButtonPressed = 0;
}

void Mode2(void)
{
		TIM3_CCR2 = 950;
		Start_Motors();
		Delay(3000);
		for(int x; x<5; x++)
		{
			TIM3_CCR2 = 900;
			Delay(50000);
			RandDirection();
			Delay(5000);
			TIM3_CCR2 = 700;
			Delay(3000);
		}
		Stop_Motors();
		Delay(50000);
		ButtonPicked = 0;
		ButtonPressed = 0;
}

void Start_Motors(void)
{
		GPIOA_ODR |= (3<<4);
}

void Stop_Motors(void)
{
		GPIOA_ODR &= ~(3<<4);
}

void CheckButton(void)
{
		if ((GPIOB_IDR && (1<<0)) == 0)
		{
			ButtonPicked = 1;
```

```
			ButtonPressed = 1;
		}
		else if (GPIOB_IDR && (1<<1) == 1)
		{
			ButtonPicked = 2;
			ButtonPressed = 1;
		}
		else
		{
			ButtonPicked = 0;
			ButtonPressed = 0;
		}
}

void Delay(unsigned int z)   //Function for variable delay based on the unsigned int that is sent
{
		unsigned int x;
		int y;
		for(x=0;x<z;x++)
			{for(y = 0;y < 256; y++);
			}
}

void RandDirection(void)
{
		LaunchDirection = rand();
		if(LaunchDirection >= 0 && LaunchDirection <=268435455)
		{
			TIM3_CCR1 = 925;
		}
		else if (LaunchDirection >= 268435456 && LaunchDirection <= 536870911)
		{
			TIM3_CCR1 = 900;
		}
		else if (LaunchDirection >= 536870912 && LaunchDirection <= 805306367)
		{
			TIM3_CCR1 = 850;
		}
		else if (LaunchDirection >= 805306368 && LaunchDirection <= 1073741824)
		{
			TIM3_CCR1 = 800;
		}
		else if (LaunchDirection >= 1073741825 && LaunchDirection <= 1342177280)
		{
			TIM3_CCR1 = 700;
```

```
	}
	else if (LaunchDirection >= 1342177281 && LaunchDirection <= 1610612735)
	{
		TIM3_CCR1 = 650;
	}
	else if (LaunchDirection >= 1610612736 && LaunchDirection <= 1879048191)
	{
		TIM3_CCR1 = 600;
	}
	else if (LaunchDirection >= 1879048192 && LaunchDirection <= 2147483647)
	{
		TIM3_CCR1 = 575;
	}
}
```