McMaster University

SmartServe

Software & Mechatronics Capstone

High Level System Design

Authors:
Christopher McDonald
Harit Patel
Janak Patel
Jared Rayner
Nisarg Patel
Sam Hamel
Sharon Platkin

Professor:
Dr. Alan Wassyng

Teaching Assistants:
Bennett Mackenzie
Nicholas Annable
Stephen Wynn-Williams
Viktor Smirnov

Last compiled on December 13, 2017

Contents

List of Figures

Date	**Revision**	**Comments**	**Author(s)**
Dec 1, 2017	1.0	Main content done for all sections	Christopher McDonald

Figure 1: Revision History

1 Introduction

1.1 Project Overview

SmartServe is an autonomous table tennis training system for table tennis players with various skill levels. SmartServe aids in diagnosing and improving a player's performance over time. The system trains table tennis players by shooting table tennis balls towards the player and detects successful returns from the player. The system can further adapt to the player's weaknesses and help them overcome it through further training. Importantly, SmartServe alleviates the problems of finding and working with a coach for players, as well as coaches trying to train multiple players simultaneously. The system will be deemed a success if the table tennis players and coaches, can enjoy and see a value added by using SmartServe.

The project started at the beginning of the Fall 2017 academic term and will conclude at the end of the Winter 2018 term. In addition, the core project team consists of final year Software and Mechatronics Engineering students who are enrolled in the MECHTRON 4TB6/SFWRENG 4G06 capstone project course.

1.2 Document Overview

This document will cover the entire system from a high level point of view. This means the system will be decomposed into subsystems which each have their own responsibilities and design. The subsystems will have their intended input, expected output and description of how the module will be used. In further documentation, each will be designed in a way which is abstracted from this document's perspective. The expected use cases will also be detailed to understand the expectations of the user and how each subsystem interacts with another. A more detailed view of this including timing as a factor will be detailed in the Sequence Diagram section.

For each subsystem, its purpose will be defined with respect to the overall system. After doing so, detailed input and output parameters will be defined. The means of communication will also need to be described as these could be simple method calls to requests over some transport protocol like HTTP or TCP. The general architecture will be decided for each subsystem will be given as each needs to satisfy unique requirements and thus needs to be decided at a system level.

To finish the document, each subsystem will be responsible for implementing some set of requirements which can be found here. These requirements are outlined in the requirements document found here. In the ideal scenario, one subsystem will be the only one that is responsible for any one requirement but may not be the case for all the requirements.

1.3 Naming Conventions and Terminology

The following terms and definitions will be used throughout this document:

- **System**: encompasses both the hardware and software parts of SmartServe
- **Shooting Mechanism**: refers to the part of the system that shoots the table tennis balls towards the user side (player)
- **Team**: all team members of the core capstone project, as noted in the list of Authors
- **User Side**: the side of the table where the user (player) is standing
- **System Side**: the side of the table where the electromechanical system is placed; it is the opposite side of the User Side
- **ACID**: a database transaction which is atomic, consistent, isolated and durable
- **FPS**: frames per second
- **GUI**: graphical user interface
- **Pitch**: rotation along the y-axis; this rotation angle primarily dictates the range of the ball from the net to the edge of the table on the user side
- **Yaw**: rotation along the z-axis; this rotation angle primarily dictates the panning functionality of the shooting mechanism from the right side to the left side of the table
- **Roll**: rotation along the x-axis
- **HTTP:** hypertext transfer protocol
- **TCP:** transmission control protocol

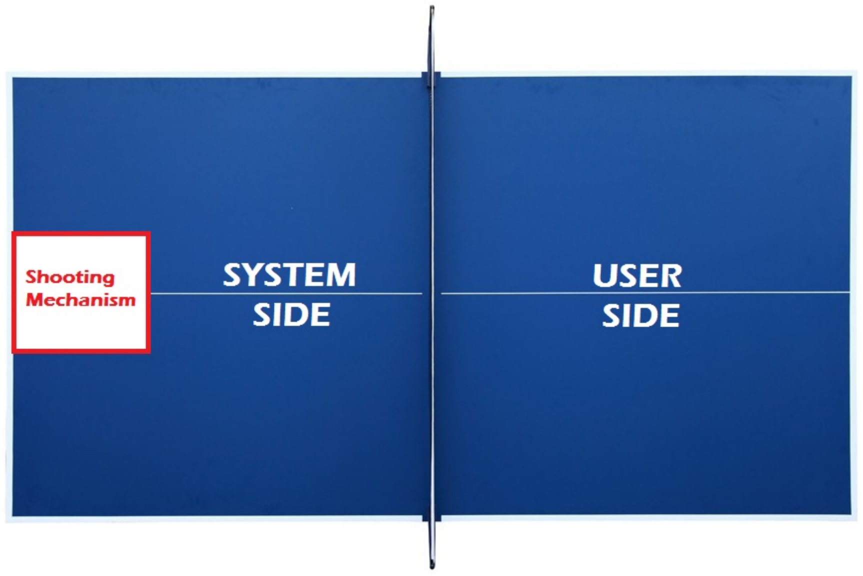

Figure 2: Top View of the Tennis Table

1.4 Project Scope

In order to make the project feasible within the time constraint imposed on the team, it will need to be scoped accordingly. One way we are scoping the project is by limiting the types of shots the system can take. A return by the system would first contact the table on the *User side* where a serve would first contact the table on the *System side*. We are limiting the system to only preform returns. The system will make no attempt at returning any shots from the user.

After the user has returned a shot from the system, the system will not make any attempts to return a shot that is likely to be returned given the user's returns. This is to give the system the ability to focus on a particular type of shot the user is the least proficient at returning. The characteristics of the shot following any return will be determined by the system's mode and the proficiency of the player if available.

2 System Description

2.1 System Architecture

The system will follow a service-oriented architecture. This means that a central subsystem will interface with several services that serve a single purpose. Some subsystems will be simply told what needs to be done and others will be asked for some return value. This is done to implement separation of concerns where one subsystem doesn't know everything about the system and only what is necessary to satisfy their requirements. It also allows for easy increments of versioning, where one subsystem can be used as long as it is functional and easily swapped out for a newer version with extra features or increased performance. Moreover, this system has heavy timing constraints and an unpredictable environment so some actions must be taken in absence of a service's response. For example, the computer vision may take longer to track a ball depending on its trajectory where the system must shoot another ball in order to keep the user engaged.

2.2 Subsystems

The system will be broken down into the subsystems with the following purposes: general managing of subsystems, computer vision to detect returns, shot recommendation, model the shot's trajectory, optimize the distance travelled by the shooting mechanism, store the data, take input and output from the user and shooting the ball toward the user. The diagram for this breakdown can be found in Figure 3.

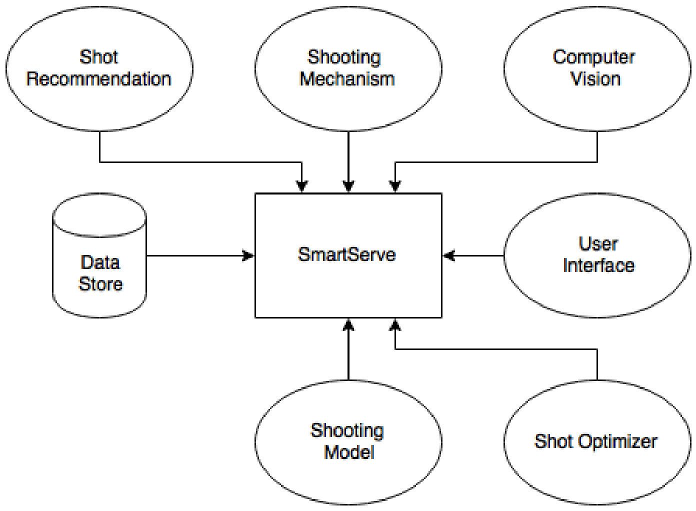

Figure 3: Subsystem Breakdown

2.3 Use Cases

The diagram including all use cases can be found in Figure 4. The user-instantiated ones will be described in detail below.

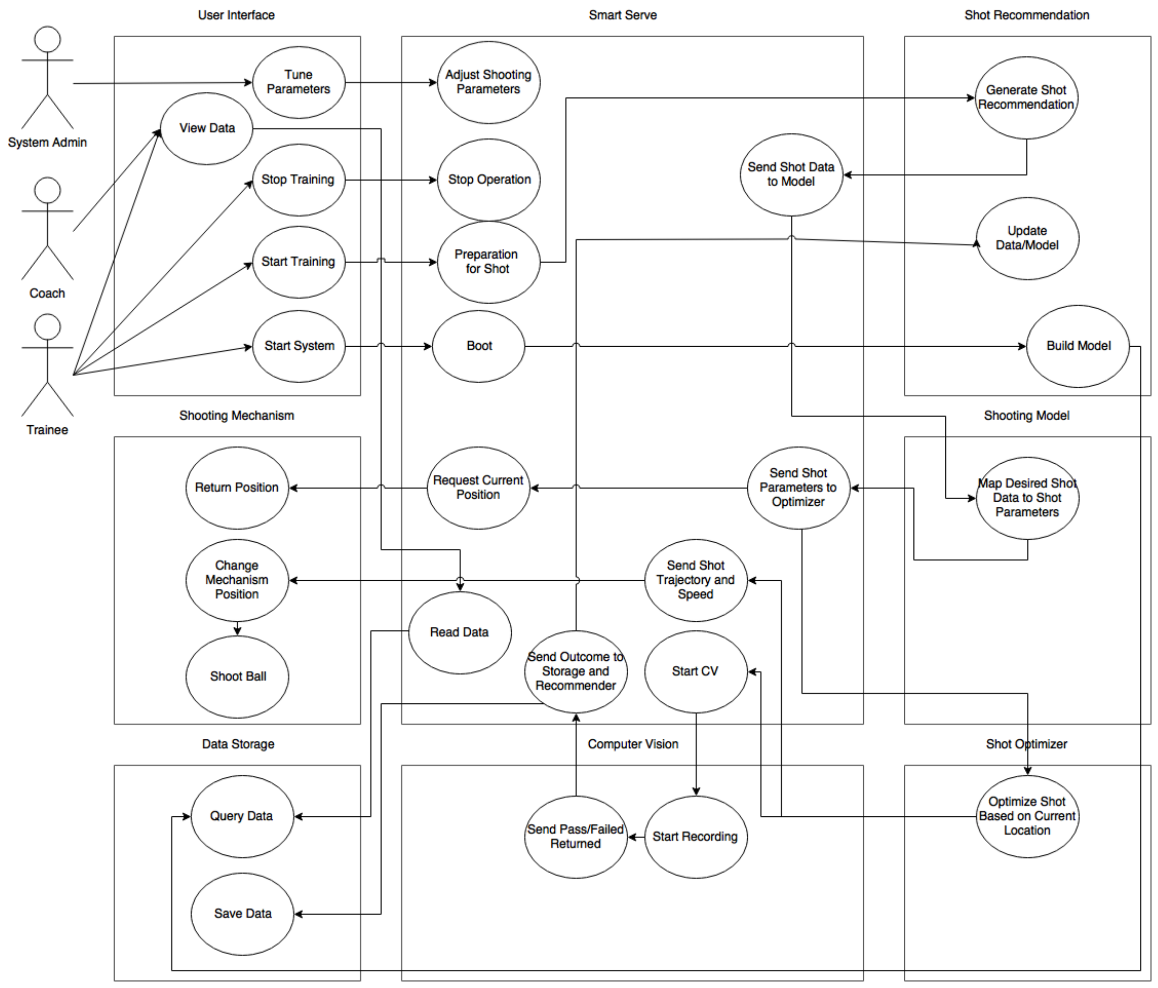

Figure 4: Use Case Diagram

Start Training

The user interface will have ways to allow the user to start this action which makes the Smart Serve subsystem prepare for a shot to fire towards the user. To do this, it needs to request a shot to use from the Shot Recommendation subsystem. This will be in the form of a desired location on the table, the speed of the shot and the angular velocity of the ball. The Smart Serve subsystem can then use the Shooting Model to translate this information into pitch, yaw and angular velocity to shoot the ball so it matches the desired shot. The Smart Serve subsystem can then use the Shooting Mechanism's current position and the desired shot to request the optimal position to shoot the ball using the Shot Optimizer. After doing so, it will instruct the shooting mechanism to shoot the ball in the desired way and start the CV subsystem to begin tracking for a successful return. Only until the CV subsystem returns the pass or failed return data, can it update the Data Storage and Shot Recommendation subsystems with this new information. The former will store the result and the data for the shot together, where the latter will update the model used to generate shot recommendations. After doing all this, it can begin preparing for a new shot.

Stop Training

In the event the user wants to cease training, the user interface will allow this and will halt the system from shooting balls. No data should be written or shots requested for the shooting mechanism to fire in order to preserve the integrity of the data the system gathers.

View Results

When the user wants to visualize the results of their performance, the user interface will allow this action to begin. It will use the Smart Serve subsystem to query data from data storage and present it in some meaningful way.

Tune Parameters

The system made need to be adjusted for various lighting, table sizes or environments. The user interface will allow a system administrator to do this in order to directly change values associated for these variables.

Start System

The user interface will allow the system to be booted where it starts the Smart Serve subsystem. This will read allow the Shot Recommendation system to build its model based on previous data for the user, if it exists.

2.4 Sequence Diagram

The sequence diagram for each user-initiated use case are listed below in their respective sections.

Start Training

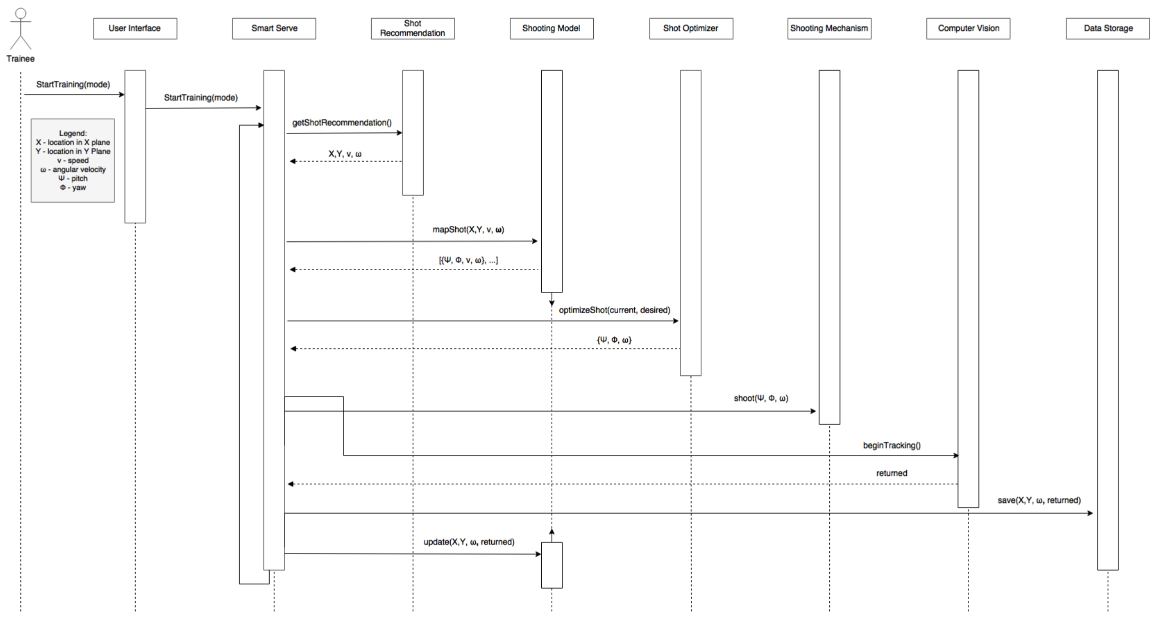

Figure 5: Sequence Diagram for Starting Training

Stop Training

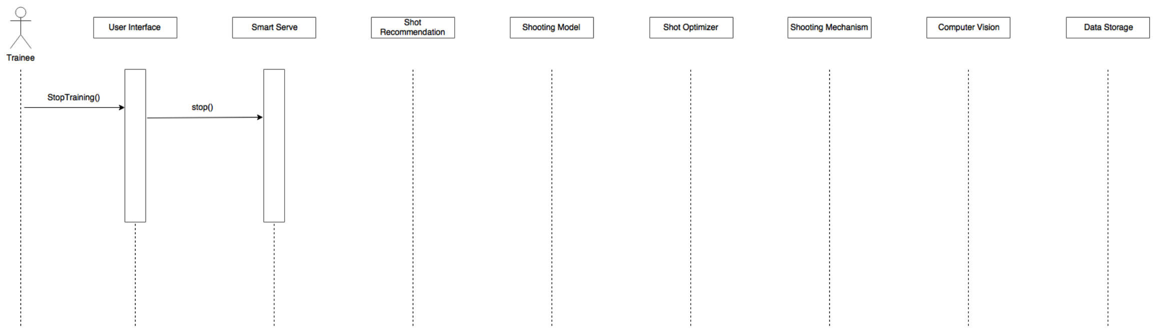

Figure 6: Sequence Diagram for Ceasing Training

View Results

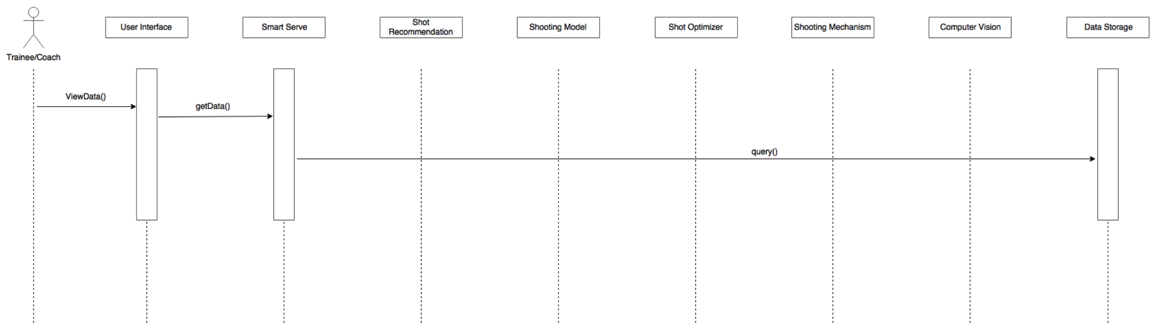

Figure 7: Sequence Diagram for Viewing Training Results

Tune Parameters

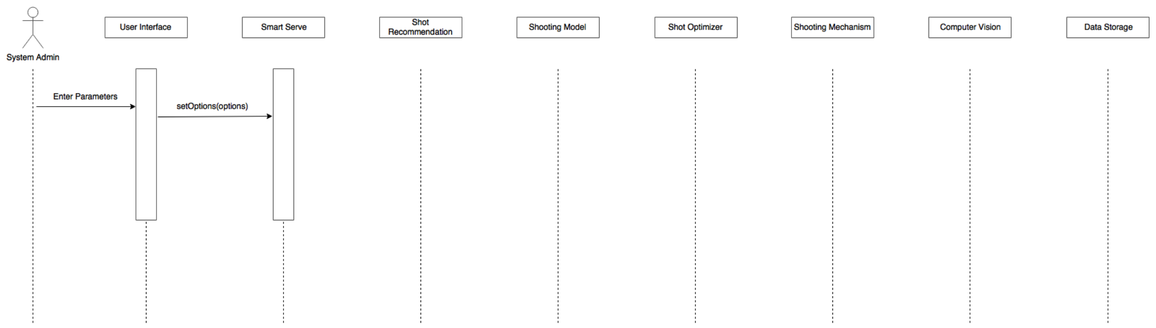

Figure 8: Sequence Diagram for Tuning Parameters

Start System

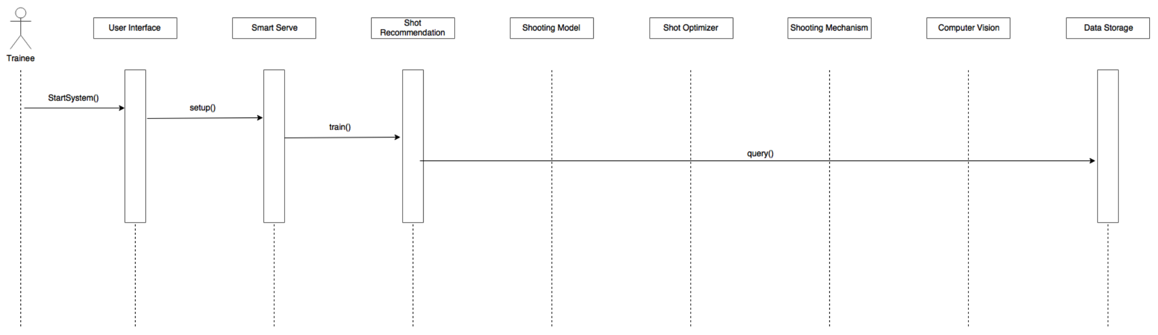

Figure 9: Sequence Diagram for Booting the System

2.5 Behaviour Description

As the behaviour of the system has been discussed previously, this section will describe the expected user behaviour and how they would interact with the system.

2.5.1 Normal Operation

The trainee would begin by starting the system from its off state. This will trigger the Start System use case and allow the Smart Serve system to boot and the Shot Recommendation System to boot and build a model to recommend shots. The user will then trigger the Start Training use case and specify a training mode. This creates the loop of Smart Serve prepping and serving shots for the user. Only until the user starts the Stop System use case will this loop stop during normal operation. Either during training or after, a user can start the View Data use case to get details on their performance.

2.5.2 Abnormal Operation

Although possible, there are some actions a user could take which would be considered abnormal. If a user hits the ball and it immediately leaves the viewable area of the computer vision, whether it never returns or does so after 1.5 seconds, it should be considered a failed return. A user could replace the table with a different one without calibrating the system to the new table. The user could also misplace the system such that it is not centre with the table or at the edge of the table. The proper placement can be found in Figure 2. The system will require the user to load the balls into the shooting mechanism's hopper which will need to be a particular size and colour to work optimally.

2.5.3 Error Handling

If the system should encounter an error, it should display the details of it through the User Interface subsystem. In addition to this, it must take measures to ensure the safety of the user and integrity of the system. This includes stopping all training and pending actions for the system to perform. For example, an error could be a jammed table tennis ball or the system has run out of balls to use.

3 Subsystems Overview

3.1 Smart Serve

The Smart Serve subsystem will provide the means of interfacing with all the subsystems and enforcing timing constraints. For all intents and purposes, it can be considered the main process and hub of the system. In the event a subsystem is taking too long to respond, this subsystem must be able to continue operation to meet timing requirements. The various states and transitions can be found in Figure 10. The FSM diagram shows each state should have an *exit option* if a service takes longer than allowed to preform an action.

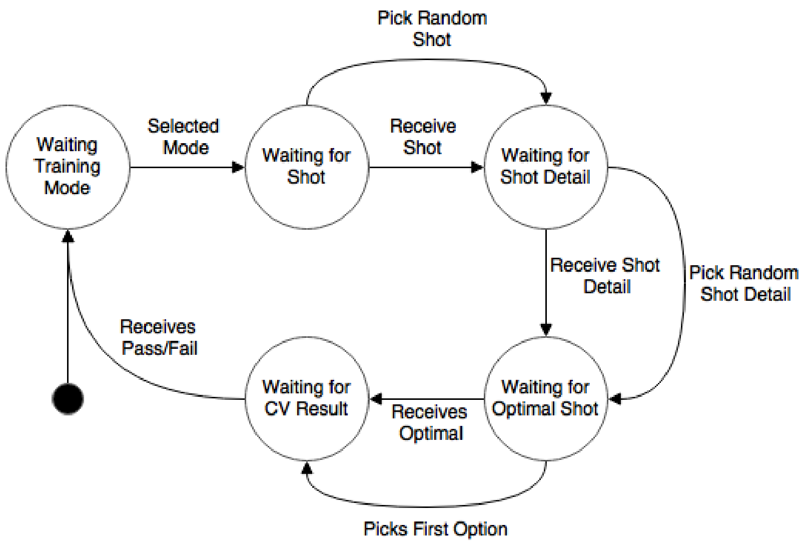

Figure 10: Smart Serve FSM

3.2 Computer Vision

The Computer Visions subsystem will be a service to the Smart Serve subsystem to determine if a shot is successfully returned. When sent a request to begin tracking the ball, it will do so and return a true or false when the ball bounces off the table or it doesn't. In the event the ball never enters frame, it will assume the return was failed after a fixed amount of time.

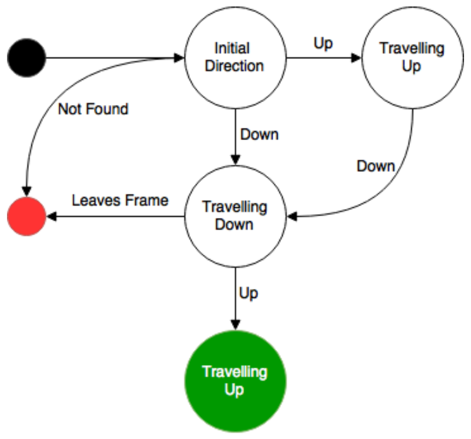

Figure 11: CV Finite State Machine

3.3 Shot Recommendation

The Shot Recommendation subsystem will be a service to the Smart Serve subsystem to determine which shot should be taken next. The way the subsystem decides on the shot will be determined by the training mode the user selects. These will include shooting the same shot every time, pseudo-randomly picking a shot or using reinforcement learning algorithms to decide the next shot. The last mode will be based on previous performance by the user.

3.4 Shooting Model

The Shooting Model subsystem will be a service to the Smart Serve subsystem which provides the means of mapping a desired shot to the details needed to take the shot. The input and output for the system can be found in Figure 12. The output will be many combinations of pitch, yaw, speed and angular velocity that satisfy the input shot details.

3.5 Shot Optimizer

The Shot Optimizer will be a service to the Smart Serve subsystem which provides means of reducing the travel time the shooting mechanism needs to do in order to shoot the

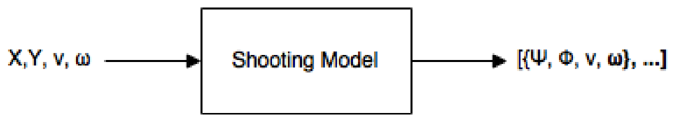

Figure 12: Shooting Model I/O

desired shot. This will simply scan each possible orientation and pick one which minimizes the distance travelled for the mechanism.

3.6 Data Storage

The Data Storage subsystem will store all the details for each shot, the user's profile and the return rates for each user. It will need to have interfaces for saving data and querying data out of the system in a variety of formats. Ideally, it will require minimal detail and configuration for the subsystem which uses it due a variety of implementations of these subsystems.

3.7 User Interface

The User Interface subsystem will be the means of translating user requests into actionable requests for the Smart Serve subsystem.

3.8 Shooting Mechanism

The Shooting Mechanism will be the subsystem which actually fires the ball towards the user. A microcontroller will be used as the point of contact and set any actuators required to fire a specified shot. It must complete this action within 1.5 seconds for the Computer Vision to detect the ball returned by the user.

4 Class Responsibility Collaboration (CRC) Cards

4.1 SmartServe

Smart Serve	
Responsibilities:	**Collaborators:**
• **F10**: pause the shooting mechanism • **F18**: shoot a ball once the previous has been returned to the system side or 1.5 seconds after the previous shot, whichever is shorter • **P2**: must include all but previous 3 shots in performance data • **P5**: must support only one user playing at one time	• Computer Vision • Shot Recommendation • Shooting Model • Shot Optimizer • Data Storage • User Interface • Shooting Mechanism

Table 1: Smart Serve CRC Card

4.2 Computer Vision

Smart Serve	
Responsibilities:	**Collaborators:**
• **F5**: detect a successful return by the user • **OE2**: functional in indoor settings with bright florescent lighting	• Smart Serve

Table 2: Computer Vision CRC Card

4.3 Shot Recommendation

Smart Serve	
Responsibilities:	**Collaborators:**
• **F7**: load a previously saved state • **MS1**: must support adding different modes to the system	• Smart Serve • Data Storage

Table 3: Shot Recommendation CRC Card

4.4 Shooting Model

Smart Serve	
Responsibilities:	**Collaborators:**
• **F13**: implements a training mode • **F14**: implements a one-shot mode	• Smart Serve

Table 4: Shooting Model CRC Card

4.5 Shot Optimizer

Smart Serve	
Responsibilities:	**Collaborators:**
• **P6**: minimize the latency between the current shooting mechanism position and the next desired position	• Smart Serve

Table 5: Shot Optimizer Card

4.6 Data Storage

Smart Serve	
Responsibilities:	**Collaborators:**
• **F6**: saves details for each shot taken by the shooting mechanism • **F8**: allows creation of a new user • **F9**: authenticate users • **P4**: must support 1000 users • **MS2**: able to add new metrics to analyze performance • **S1**: hash all passwords for user profiles • **S2**: encrypt all performance data for each user • **P1**: allow read access for coaches	• Shot Recommendation • Smart Serve

Table 6: Data Storage CRC Card

4.7 User Interface

Smart Serve	
Responsibilities:	**Collaborators:**
• **F11**: end the training session • **F12**: resume training session from a paused state • **F13**: display user's performance over a custom time range • **F16**: allows user to adjust training parameters during an active or paused session • **F17**: can be calibrated for a specific table size • **LF1**: have a minimalist design that is easy to navigate through • **UH1**: is intuitive to use • **UH2**: operable using the English language • **P1**: response time for user input must be less than or equal to 100ms	• Smart Serve

Table 7: User Interface CRC Card

4.8 Shooting Mechanism

Smart Serve	
Responsibilities:	**Collaborators:**
• **F1**: shoots the table tennis ball towards the user at various locations • **F2**: shoots the table tennis ball towards the user at various speeds • **F3**: shoots the table tennis ball towards the user at various Yaw • **F4**: shoots the table tennis ball towards the user with spin along 2 axes • **HS1**: always hit the table at least once per shot • **HS2**: not shoot the ball faster than 22 m/s • **HS3**: have no exposed electrical wiring or components • **HS4**: carries warnings around moving parts • **HS5**: has a button to cease all power to system	• Smart Serve

Table 8: Shooting Mechanism CRC Card