Boston University
Electrical & Computer Engineering
EC463 Senior Design Project

First Semester Report

SwingOn

Executive Summary
— Problem Statement: Golf is an expensive sport, which leads to many potential golfers to be turned away. In addition, golf is a relatively difficult sport to be introduced to, especially without proper training.
— Final Deliverable: A smartphone app that tracks the motion of the user. Users can open the camera and analyze the swing lively. Besides, users can upload videos from the Photo library app or Files app, and get analyzed videos with feedback in real time.
— Technical Approach: Use pre-trained models to analyze and draw the user's body points. These points will be used to compute the user's centroid and provide an interpretation of the balance. Feedback will also be provided after analysis to help the user have a better insight on how to improve their swing in the future.
— Innovative Features: Most golf swing training aids focus on the trajectory of the golf ball. Although these aids are helpful, they cannot help users to solve all the problems, such as balance and consistency. The SwingOn app will focus on balance and consistency, which are necessary for a good golf swing.
Index Terms
— Video analysis, machine learning, body detection

- *Yoel Beyene is with the Department of Electrical and Computer Engineering, Boston University, MA 02215, E-mail: yoelb@bu.edu*
- *Tingru Lian is with the Department of Electrical and Computer Engineering, Boston University, MA 02215, E-mail: tinalian@bu.edu*
- *Hanlin Mai is with the Department of Electrical and Computer Engineering, Boston University, MA 02215, E-mail: hanlinm@bu.edu*
- *Jessica Martinez Marquez is with the Department of Electrical and Computer Engineering, Boston University, MA 02215, E-mail: jessmtzm@bu.edu*

1 Introduction

Golf is an expensive sport since it requires a great amount of fees for golf courses. Besides these initial costs, golf is difficult to learn, as it requires plenty of time to practice the correct swing technique as well as considerable effort for golfers to track their improvements. Most golf swing training aids focus on the trajectory of the golf ball and the static elements of the golf swing, such as the basic golf stance. These aids do not provide necessary tools to improve golfers' fundamental skills—balance and concentration.

It is essential for a golfer to maintain a good balance throughout the swing. Because it allows the golfer to have better control on the club and deliver it to the ball, which can ensure accuracy. And consistency of golf swings allow minimizing mistakes.

The goal of SwingOn is to create a golf swing training aid that analyzes golfer's balance and consistency. The app does not focus on static elements since its engineers believe that focusing on the analysis on balance and consistency will be more helpful for golfers. It aims to boost a golfer's training, either as a complement to their courses or as a stand-alone tool used during individual practice—all with the goal of maximizing the player's potential. Users can upload videos from the Photo library app or Files app, and get analyzed videos with feedback in real time.

Compared with other golf swing training aids in the market, this project focuses on the golfer's body position. Therefore, not only beginners can learn the sport in a more economical way, but also advanced players can benefit from this app by being able to improve different aspects of their golf swings. [1]

2 Concept Development

2.1 Engineering Understanding & Conceptual Approach Chosen

According to our customer, the requirement of the project would be to enable the golfer to develop good golf swings. There are three main aspects of the golf swing that relate to the golfer's body position: concentration, balance, and consistency. All of these aspects are correlative. Both concentration and balance are crucial to consistency.

2.1.1 Concentration

Concentration refers to the golfer's ability to keep one's eyes on the golf ball at the moment of impact. It is crucial in the golf swing since concentration ensures that the golfer

[1] Footnotes are the authors of the section, author orders are based on contribution. Section 1 by Tingru, Hanlin, Jessica, Yoel.

excludes all other distractions during the swing.

2.1.2 Balance

Balance refers to maintaining the stability throughout the golf swing. A good balance allows the golfer to have better control on the club during the swing and ensure the accuracy of the hit [1]. It may take a long time for beginners to play a good swing since they don't have many skills and experiences [2].

The user's balance can be calculated with body points. Our app uses machine learning models that can detect joints on the user's body. Then, the coordinates of body points will be saved and the user's centroid will be calculated by finding the average coordinates of the left and right shoulder and the left and right hip. Using a machine learning model to classify the centroid changing over time can determine whether the user is balanced or not. The model will develop a range in which all values in the range are classified as a balanced swing. Then, the app will give the user feedback on their balance and suggestions on how to improve.

2.1.3 Consistency

Consistency refers to the golfer having a similar balance level across multiple golf swings. It demonstrates that the improvement of a golfer's swing is based on their skill performance [3]. Our app will give feedback to the user about their consistency by comparing the current uploaded video with the previous one.

2.1.4 Portability

During the practice, golfers want to focus on their game rather than constantly manipulating a tool after each golf swing. Therefore, the customer requires a solution that is intuitive, portable, and with minimal manual interactions. In the following section, we will describe alternative designs and the solution chosen to fulfill this requirement.[2]

2.2 Alternative Design

Initial concepts included a desktop application with external cameras. To achieve our customers' requirements, we decided to create a mobile app that allows users to receive hands-free feedback (potentially auditory) based on their motions captured by the device's rear camera. The mobile app was chosen over the alternative designs since its portability and the ease of adoption.

2.2.1 MATLAB

MATLAB can detect human's heads and golf balls. However, it cannot be applied to a mobile application, which opposes the requirement of portability. Besides, the UI in MATLAB is not user-friendly compared with UIs in other programming languages. Thus, it is not practical to use MATLAB in our project.

2.2.2 YOLO v5

The YOLO v5 is a pre-trained model that can detect humans with high accuracy. However, it takes a long time to classify humans on each frame of the video. For our project, it is preferred to provide feedback for the user in real time. Therefore, applying a time-consuming algorithm in our app is not ideal.[3]

3 System Description

3.1 Block Diagram

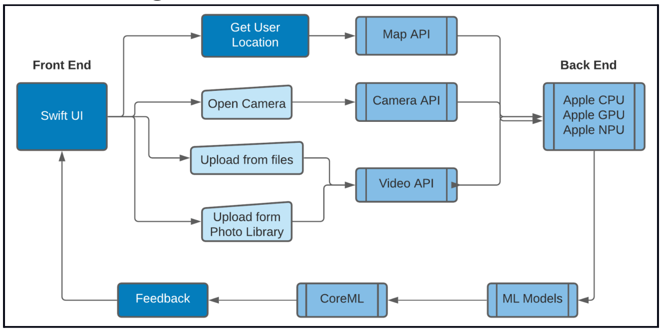

Fig. 1. Block Diagram of the SwingOn app[4]

Swift is the programming language we choose for the app. Because it can utilize the CPU, GPU, and NPU inside the Apple devices. It can also establish a more stable and more efficient connection between the user interface and machine learning models. Fig. 1. shows how the front-end connects to the beck-end.

3.2 User Interface

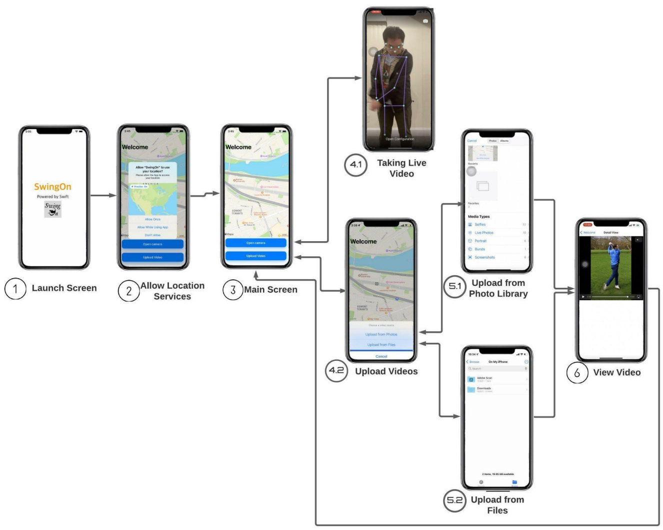

Fig. 2. Flow Chart of SwingOn app's user interface[5]

So far, the front-end consists of 8 screens. When users open the app, they can see a launch screen (Figure 2.1). Then, they will be asked if they allow the app to use their locations (Figure 2.2). There will be three options for users: "Allow Once", "Allow while using the App", and "Don't Allow". Users can also change the privacy settings of the app in Settings > Privacy > Location Services > SwingOn.

[2] Section 2.1 by Jessica, Tingru, Yoel, Hanlin.

[3] Section 2.2 by Tingru.

[4] Block diagram by Tingru.

[5] Flow Chart by Tingru, Hanlin, Jessica, Yoel.

We recommend users to authorize the app for location services which can give them a better experience. The location services can be enabled in Settings > Privacy. The framework used for this feature is called the MapKit, which displays the map within the app [4].

SwiftUI is the framework of user interface, which allows the app to present a stack of views in a navigation hierarchy. Buttons can perform navigation to different screens by linking them with the first screen. Users can navigate to their desired destinations by selecting different buttons and go back to the previous screen, which allows them to traverse a collection of views. Users have two options to choose on the first screen (Figure 2.3).

The first option is "Open Camera", which is connected to the camera API (Figure 2.4.1). The AVFoundation, which is a full featured framework for working with time-based audiovisual media, is used here. It allows access to all media functionality. The framework combines six major technology areas for capturing, processing, synthesizing, controlling, importing, and exporting audiovisual media on Apple platforms. Many tasks can be completed with this framework in our app. The AVCaptureSession is a class of the framework that manages capture activity and coordinates the flow of data from input devices to capture outputs. [5]. The AVCaptureVideoDataOutput is another class that allows A capture output that records video and provides access to video frames for processing [6].

The second option is "Upload Videos" (Figure 2.4.2). After clicking the button, users will choose whether they want to "Upload from Files" (Figure 2.5.1) or "Upload from Photo Library" (Figure 2.5.2). These buttons are connected to the video API. The uploaded videos can be viewed by users using the AVPlayerViewController class [7] (Figure 2.6). The AVPlayer class provides the interface to control the player's transport behavior, which allows playback controls, such as playing the video and pausing the video. It also supports a standard set of keyboard shortcuts to control playback.

All these APIs connect to the backend, where videos will be analyzed. The Core ML framework is used to integrate machine learning models into our app, which, in this case, we use the pre-trained model called PoseNet [8]. It takes a processed camera image as the input and outputs the information of 17 body key points (Figure 2.4.1). These key points have a confidence score between 0.0 and 1.0. The confidence score indicates the probability of a key point existing in that position. For better visualization, circles are drawn for the joints and the connected lines between the joints. The joints' confidence threshold can be changed and those visualized circles will change accordingly. The app's data storage and data analysis are still under development, which will be explained in detail in 5.0 Technical Plan.[6]

3.3 Pseudocode

```
ContentView {
        NavigationView {
        Map {
                MapViewerModel {
                        check location permissions
                }
                display user's location
        }
        Button (Let's start here) {
                Button (Open Camera) {
                        Camera API
                }
                Button (Upload from Photos) {
                        PhotoPickerModel
                }
                Button (Upload from Files) {
                        FilePickerModel
                }
        }
}
SecondView {
        display analyzed videos
}
```

Fig. 3. High-level pseudocode of how the project operates[7]

4 First Semester Progress

4.1 Alternative Design

4.1.1 MATLAB

- Video Analysis: Video is split frame by frame and the algorithms mentioned below are applied to each frame. This is used to find the location of the golfer's body and the location of the golf ball in the video. The exact frame when the golfer hits the ball can be determined.
- Cascade Object Detection: The cascade object detector uses the Viola-Jones algorithm to detect objects. It is a pre-trained model and can detect several parts of an object. The golfer's body in the video is detected using this algorithm [9].
- Circular Object Detection: This is a MATLAB built-in function for finding the golf ball in the video. By adjusting the detection sensitivity, edge threshold, and object polarity, a better result can be obtained [10].

4.1.2 Python - YOLOv5

- Video Analysis: Similar to MATLAB, video analysis in Python is done by splitting the video frame by frame and analyzing each frame separately.
- YOLOv5 Object Detection: YOLOv5 is a pre-trained model for object detection [11]. It is applied to analyze each frame of the video .[8]

[6] Section 3 by Tingru, Hanlin.

[7] Pseudocode by Hanlin.

[8] Section 4.1 by Tingru.

4.2 Swift

4.2.1 Results from first prototype testing

- The SwingOn app can successfully run on an Apple device. An iPhone 12 is used in the test.
- The content in the application is clearly displayed.
- All the buttons work properly.
- All the APIs work properly.
- 17 key-points detected: nose, left and right eye, left and right ear, left and right shoulder, left and right elbow, left and right wrist, left and right hip, left and right knee, left and right ankle.
- Increasing the number of used threads can speed up the execution. However, it will make the model use more resources and power.
- Three delegates for hardware acceleration to choose from:
  - CPU is a safe and simple choice. However, it is slower and consumes more power than other hardware accelerators.
  - GPU is the most widely-used accelerator and provides a decent performance boost.
  - NPU is similar to GPU. But instead of accelerating graphics, it accelerates neural network operations such as convolutions and matrix multiplies.
- Three models to choose from:
  - PoseNet is a pose estimation model that can detect these 17 body key points
  - Lightning is a smaller and faster model. But it is less accurate compared with the Thunder version. It can be applied to run on smartphones in real-time.
  - Thunder is a more accurate model. But it is larger and slower than the Lightning version.

4.2.2 Progress after first prototype testing

- The FilePickerModel file includes view controllers such as UIDocumentPickerViewController, which provides access to documents or destinations outside the app's sandbox [12].
- The PhotoPickerModel file includes view controllers such as updateUIViewController, which updates the state of the specified view controller with new information from SwiftUI. The makeCoordinator instance method creates a custom instance that can communicate changes from our view controller to other parts of our SwiftUI interface.
- Our project now uses the Apple Vision Framework to estimate human body poses by using VNHumanBodyPoseRequest class. It can detect 19 key body joints. Each result is filtered into 12 body joints using getBodyJointsFor as described on Section 3.2.[9]

[9] Section 4.2 by Tingru, Hanlin, Jessica.

5 Technical Plan

Schedule and Milestones

- Functional Testing: March 28, 2022
- Customer Installation: Month of April
- ECE Day: May 6, 2022

Task 1: Data Storage (Finish by Feb 1, 2022)

The SwiftUI framework will be used to control and respond to the flow of data and changes within our app's models. When the data changes (the user uploads a new video), SwiftUI will automatically update the corresponding parts of the user interface. This framework can provide tools for connecting our app's data to the user interface. The data will be stored with Core Data using the FetchRequest [13]. All of these tools will allow us to store every piece of data in the app and reduce the complexity of the code.

Task 2: Accurate Centroid Calculation (Finish by Feb 15, 2022)

Since the model we have now can find joints on the user's body, the next step of the application is to compute the centroid of the user by utilizing these points. This task will be tested by making computations and evaluating whether it reflects the actual balance of the user accurately.

Task 3: Data Analysis (Finish by March 28, 2022)

Our machine learning models will be trained to perform different tasks. These tasks include finding relationships among numerical values of body points over time and classifying whether the user performs a standard good or a bad swing. The quality of the swing will be determined based on body points and the centroid after calculations. This model will be integrated into our app by using Core ML.

6 Budget Estimate

Table 1
Budget Description and Cost

Item	Description	Cost
1	Tripod	$40
	Total Cost	$40

Table 1 shows the budget of the project. Since SwingOn is a software project and the resources we use are open-source, the only cost of our project will be the tripod. The tripod will be used to stabilize the device and ensure high accuracy. It will be purchased from Amazon.

7 Attachments

7.1 Appendix 1 – Engineering Requirements[10]

Table 2
Engineering Requirements

Requirement	Tasks
UI	Intuitive and user-friendly

[10] Section 7.1 by Tingru, Hanlin

Portability	A cross-platform app
Video Processing	Users can take and upload videos
Body Points	Points can be found accurately
Balance	Users' centroids are correctly calculated
Consistency	Detect whether golf swings are consistent
Feedback	Feedback for users to improve their golf swings

7.2 Appendix 2 – Gantt Chart[11]

Please see the last page (Fig 4).

7.3 Technical References[12]

[1] B. Hill, "The importance of balance in a golf swing," *USA Today*, Sep-29-2020, [Online]. Available [Accessed: Dec-11-2021]: https://golftips.golfweek.usatoday.com/importance-balance-golf-swing-20105.html.

[2] "Consistent golf: 5 keys for Steady rounds," *The Left Rough*, Apr-24-2020, [Online]. Available [Accessed: Dec-7-2021]: https://theleftrough.com/consistent-golf/.

[3] "How to become a more consistent golfer in 8 easy steps," *Bobby Walia Golf Pty Ltd*, Sep-8-2020, [Online]. Available [Accessed: Dec-11-2021]: https://www.bobbywaliagolf.com/more-consistent-golfer/.

[4] "MapKit," *Apple Developer Documentation*, [Online]. Available [Accessed: Dec-11-2021]: https://developer.apple.com/documentation/mapkit/.

[5] "AVCaptureSession," *Apple Developer Documentation*, [Online]. Available [Accessed: Dec-11-2021]: https://developer.apple.com/documentation/avfoundation/avcapturesession.

[6] "AVCaptureVideoDataOutput," *Apple Developer Documentation*, [Online]. Available [Accessed: Dec-11-2021]: https://developer.apple.com/documentation/avfoundation/avcapturevideodataoutput.

[7] "AVPlayer," *Apple Developer Documentation*, [Online]. Available [Accessed: Dec-11-2021]: https://developer.apple.com/documentation/avfoundation/avplayer.

[8] "Real-time human pose estimation in the browser with tensorflow.js," *The TensorFlow Blog*, [Online]. Available [Accessed: Dec-11-2021]: https://blog.tensorflow.org/2018/05/real-time-human-pose-estimation-in.html.

[9] "Cascade Object Detection," *Detect objects using the Viola-Jones algorithm - MATLAB*, [Online]. Available [Accessed: Dec-11-2021]: https://www.mathworks.com/help/vision/ref/vision.cascadeobjectdetector-system-object.html.

[10] "Circular Object Detection," *Detect and Measure Circular Objects in an Image - MATLAB*, [Online]. Available [Accessed: Dec-11-2021]: https://www.mathworks.com/help/images/detect-and-measure-circular-objects-in-an-image.html.

[11] "Pytorch," *YOLOv5*, [Online]. Available [Accessed: Dec-11-2021]: https://pytorch.org/hub/ultralytics_yolov5/.

[12] "UIDocumentPickerViewController," *Apple Developer Documentation*, [Online]. Available [Accessed: Dec-11-2021]: https://developer.apple.com/documentation/uikit/uidocumentpickerviewcontroller.

[13] "FetchRequest," *Apple Developer Documentation*, [Online]. Available [Accessed: Dec-11-2021]: https://developer.apple.com/documentation/swiftui/fetchrequest.

[11] Section 7.2 & Gantt Chart by Tingru.

[12] Section 7.3 by Tingru, Hanlin, Yoel.

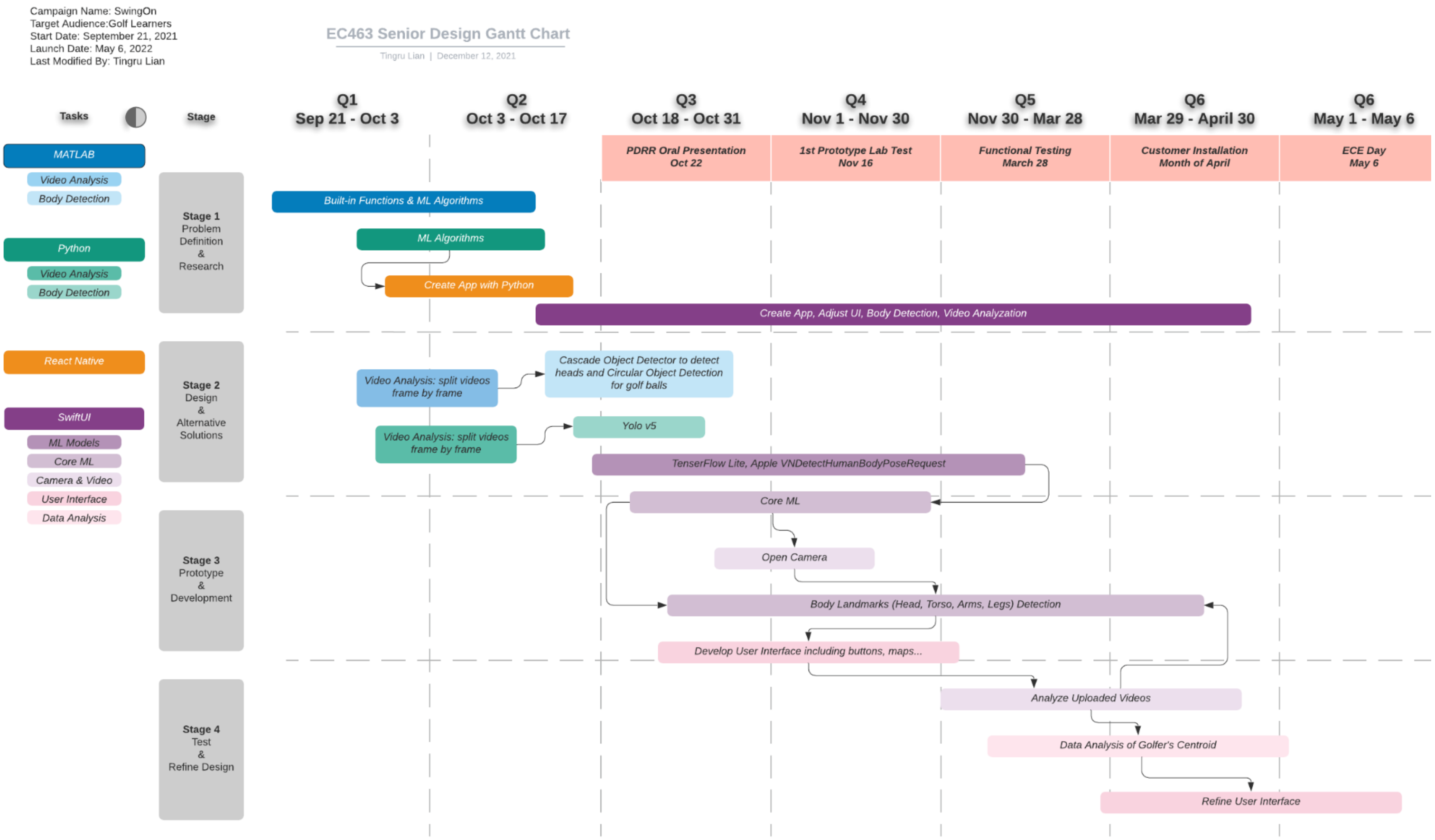

Fig. 4. Gantt Chart