**Swingman**

Advanced Baseball Swing Analysis System

Date: June 17, 2025

# Table of Contents

1. Project Overview

2. Core CV Features

3. Complete Project Structure

4. Module Descriptions

5. iOS Integration Guide

6. Installation

7. Usage

8. Configuration Options

9. Data Export

10. Implementation Status

11. Machine Learning Integration

12. Next Sprint Enhancements

13. Development Roadmap

# 1. Project Overview

Swingman\_CV is a sophisticated computer vision-based system that provides real-time baseball swing analysis. This module serves as the core CV engine that can be integrated into iOS applications or used standalone.

# 2. Core CV Features

**1. Real-Time Bat Tracking System**

* Enhanced Motion Detection: Uses OpenCV-based motion tracking with background subtraction
* Multi-Method Tracking: YOLO-based detection, manual tracking with momentum prediction, motion area fallback
* Adaptive Frame Processing: Adjusts to lighting and speed conditions
* 60 FPS Processing: Optimized for real-time performance

**2. Swing Path Analysis**

* Real-time neon trail effect with motion blur
* Swing metrics including path consistency, speed, plane type, and follow-through analysis
* Historical path storage for replay and evaluation

**3. Impact Detection System**

* Uses brightness, motion differential, and frame comparison
* Records impact point and evaluates sweet spot contact

**4. Pose Analysis Integration**

* MediaPipe-based full-body tracking
* Analyzes specific joints for baseball mechanics: head, shoulders, elbows, hips, knees, ankles
* Calculates stability scores

**5. Real-Time Analytics**

* Swing efficiency score (0–100), power, path consistency, follow-through, and pose stability
* Visual feedback with color-coded metrics

# 3. Complete Project Structure

Refer to the original README for full tree structure. Main components include:

* core/: CV & ML logic
* ui/: Interface modules
* utils/: Helper functions
* Models/: YOLO model storage
* main.py: CLI Entry Point
* swift\_bridge.py: iOS integration layer



# 4. Module Descriptions

**Core ML & CV Files**

1. enhanced\_swing\_tracker.py  
   This file serves as the primary tracking orchestrator. It integrates all computer vision and machine learning components and manages the real-time processing pipeline for swing analysis.
2. hybrid\_bat\_detector.py  
   This module combines traditional computer vision techniques with machine learning for bat detection. It includes fallback detection methods and employs adaptive tracking strategies to maintain reliability in various conditions.
3. yolo\_detector.py  
   This script handles YOLO model integration, managing the object detection pipeline and executing model inference for detecting the bat in video frames.
4. swing\_data\_manager.py  
   This module is responsible for managing session data, storing swing metrics, and providing export functionality in various formats.

### Utility Files

1. drawing.py  
   This utility contains OpenCV-based drawing functions used to generate overlays and visual effects that enhance the visualization of swing paths and metrics.
2. json\_encoder.py  
   This file includes custom JSON serialization logic. It converts complex data structures into exportable formats and ensures compatibility for data storage or sharing.

### Integration Files

1. swift\_bridge.py  
   This is the iOS integration layer that handles data conversion and provides native bridge functions to communicate between the Python-based backend and Swift-based mobile applications.
2. setup.py  
   This script manages the package configuration, dependency declarations, and installation process, making it easier to distribute or deploy the project.

### Testing & Development

1. train\_yolov8.py  
   This training script is used to train the YOLO model on a custom dataset. It also manages dataset handling and training configuration parameters.
2. test\_ml\_modules.py  
   This file includes test cases for machine learning components. It is used to validate performance and ensure successful integration with other parts of the system.
3. demo.py  
   This is a demonstration application designed to showcase the system’s features. It provides a quick and interactive way to test the core functionalities.

### Configuration

1. config.json  
   This JSON file stores global settings, model parameters, and runtime configurations used throughout the application.
2. yolov8n.pt  
   This file contains a pre-trained YOLO model, which acts as the base detection engine. It is intended for transfer learning and rapid deployment in swing tracking tasks.

# 5. iOS Integration Guide

Code snippets for Swift-based iOS usage including module initialization, frame processing, and real-time result access. For more details refer to README.md in the code files.

# 6. Installation

Clone the repository and install dependencies via pip.

```python

git clone https://github.com/yourusername/swingman-cv.git  
cd swingman-cv  
pip install -r requirements.txt

```

# 7. Usage

**Standalone:**

```python  
python main.py --camera 0 --window-size 1280x720

```

**As Python Module:**

```python  
from swingman.core import EnhancedSwingTracker  
tracker = EnhancedSwingTracker(...)  
results = tracker.process\_frame(frame)  
```

# 8. Configuration Options

Tracker can be configured using dictionary options like:

```python  
tracker.configure({  
 'pose\_detection': True,  
 'impact\_detection': True,  
 'visualization\_mode': 'full'  
})  
```

# 9. Data Export

* JSON: Full swing data
* CSV: Metrics and coordinates
* MP4: Recorded session video
* PNG: Individual frame captures

# 10. Implementation Status

MVP is 70% complete with the following components:

* ✅ CV System (100%)
* ✅ Pose Analysis (80%)
* ✅ Analytics Engine (70%)
* ✅ UI/Visualization (60%)

# 11. Machine Learning Integration

* YOLOv8 used for bat detection (ultralytics)
* MediaPipe for pose estimation
* Custom rule-based motion analysis

# 12. Next Sprint Enhancements

* Custom bat detection model
* Swing classification and form correction
* 3D visualization and real-time comparison

# 13. Development Roadmap

* Sprint 1: Core CV & ML (Current)
* Sprint 2: Enhanced analytics, new models
* Sprint 3: Full production optimization

# 14. Visuals

* Heatmap Overview:





* Swing Detection:



* Pose, Object And Impact Detection (Test on Low quality images and videos):

