Sport Scientist Interview

Tennis Hits & Bounces

Welcome to our interview exercise! We specialize in consulting and advanced modeling to enhance sports performance. Our mission is to push the boundaries of innovation by leveraging data, technology, and creative thinking.

For this exercise, we want to assess your ability to think critically, approach problems from unique perspectives, and apply data-driven solutions to real-world challenges in sports. We face complex problems daily, and we're looking for interns who can bring fresh insights and a methodical approach to tackle them.

At Roland Garros 2025, we have elaborated a full computer vision code to detect the ball on a Tennis Court during a point, we gathered all the data of the Final of Roland Garros 2025 on the following drive:

https://drive.google.com/drive/folders/1YbrujfBn4kqhrSSqZqG7cpuYs8k4sb2v

Exercise:
Hit & Bounce Detection in Roland-Garros 2025 Final (Computer Vision & ML)

In this challenge, you will build two methods — one **unsupervised** and one **supervised** — to detect tennis ball ***hits*** and ***bounces*** using ball-tracking data extracted from the 2025 Roland-Garros Final.

You are provided with:

1. Video Data

- A full video of the Roland-Garros 2025 Final (don't thank us, it is on Youtube already)

2. Ball-Tracking Dataset

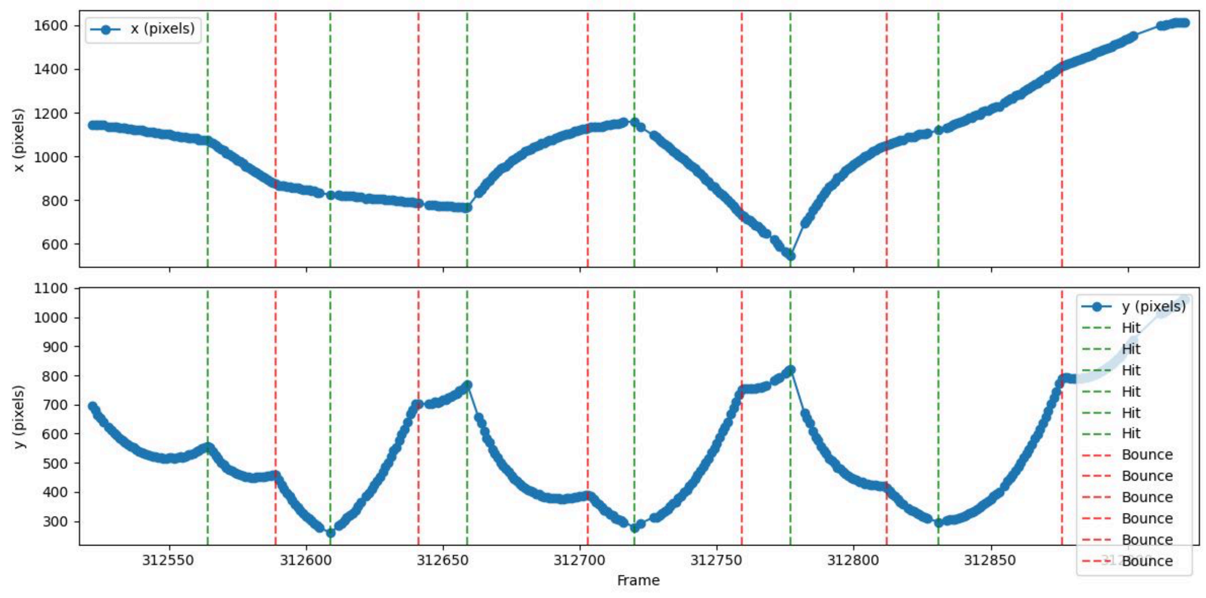
- A folder containing **329 points**, each with a JSON file containing the ball's detected positions.

- Format of the Ball-Tracking Data
    - Each JSON file contains entries indexed by the frame number (eg: 56100)

```
{"56100":{
      "x": 894,
      "y": 395,
      "visible": True,
            "action": "air }...}
```

- 56100: Frame Number in the video
- x : Horizontal pixel position on a 1920 wide-frame
- y : Vertical pixel position on a 1080 wide-frame
- visible : True if the ball is detected in this frame
- action : Ground Truth Label : “air” , “hit” , “bounce”

<u>Example of the ball data plotted</u>



3. Reconstruction Notebook

You also have a notebook containing a function that, given `ball_data_json` and the video, reconstructs a short clip of the point with the ball trajectory overlayed.

You do not need to use this part of the code, but it is a great way for you to track your code output, and vizualize the rebounds in the game

Your Goal

Build **two separate hit & bounce detection pipelines**, starting from the (x,y) time series (see example above)

The final output for both methods should be a structure identical to the input JSON, extended with a new key:

```
"pred_action": "hit" / "bounce" / "air"
```

Example:

```
  "56100": {
      "x": 894,
      "y": 395,
      "visible": true,
      "action": "air",
      "pred_action": "bounce"}
```

Method 1 — UNSUPERVISED

In this method, you must detect *hits* and *bounces* **without using labels**.

You must rely on **physics-based analysis** of the ball trajectory:

- Vertical acceleration patterns
- Sudden changes in velocity
- Ball-height dynamics
- Characteristic signatures of rebounds and racket contacts

Key idea:
➡️ *Hits and bounces can be inferred by analyzing discontinuities, slope changes, and acceleration spikes in the x/y curves.*

Your task:

`unsupervised_hit_bounce_detection(ball_data_i.json)` → **enriched JSON**

Method 2 — SUPERVISED

In this method, you may use the `"action"` label already provided to train a supervised model, the goal is to produce a model that learns the temporal dynamics of ball hits and bounces.

You may use:

- Traditional ML (Random Forest, SVM, Gradient Boosting…
- Time-series models
- Simple neural networks
- Sequence models
- Feature engineering (velocities, accelerations, direction changes…)

Your task:

`supervized_hit_bounce_detection(ball_data_i.json)` → **enriched JSON**

Project Output and Deliverables

The goal is to create **two functions** that will be delivered in a **GitHub project**.

The GitHub repository must contain:

- **README**: describing the content of the repository
- **main.py**: containing the two methods for **Hit** and **Bounce** detection
- **requirements.txt**: listing the packages required for the method
- **The trained supervised model** for detection
- **All necessary files** to run the solution

Evaluation

Your methods will be tested on a **hidden test set**.
Performance depends on how accurately your models classify *hit*, *bounce*, and *air* frames.

Creativity, clarity, and robustness of your approach will be valued.

Best of luck, and we look forward to seeing your ideas!