EdgeFleet Cricket Ball Tracking Report

1. Introduction

This project implements a computer vision system to detect and track the centroid of a cricket ball in videos recorded from a single, static camera. The objective is to generate per-frame annotations indicating ball visibility and centroid location, along with a processed video overlaying the ball trajectory.

The provided dataset contains **test videos only**, which are explicitly restricted from being used for training machine learning models. Therefore, I have used alternative approaches were explored to meet the task requirements.

2. Problem Understanding

The system is required to:

- Detect the cricket ball centroid in each frame where it is visible.
- Output per-frame annotations containing:
    - frame index
    - x centroid
    - y centroid
    - visibility flag
- Generate a processed video with trajectory overlay.
- Provide reproducible code for inference, tracking, and training.

Constraints:

- Single static camera
- No labeled training dataset
- Test videos must not be used for training

3. Initial Approach and Challenges

An initial attempt was made to design a **fully automatic detector and tracker** using classical computer vision techniques such as:

- Color thresholding
- Motion detection
- Optical flow
- Heuristic-based tracking

However, these approaches were unreliable due to:

- Very small ball size
- Motion blur
- Similar background colors
- Sudden acceleration and direction changes

Training a deep learning model was not feasible due to the absence of a permitted training dataset.

4. Final Approach: Semi-Automatic Tracking

Given the above limitations, a **semi-automatic fallback strategy** was adopted.

Key Idea:

- The user manually selects the ball centroid on key frames where the ball is clearly visible.
- The system interpolates ball positions between selected frames.
- Visibility flags are set accordingly.

Steps:

1. Load video from a fixed camera.
2. Detect frames with motion to identify candidate frames.
3. Display selected frames to the user.
4. User clicks on the ball centroid.
5. Linear interpolation fills missing frames.
6. Trajectory is drawn across frames.
7. Outputs are saved as CSV and processed video.

This approach ensures:

- High annotation accuracy
- Full reproducibility
- Compliance with dataset usage restrictions

---

5. Output Format

CSV Annotation File:

```
frame,x,y,visible
0,512.3,298.1,1
1,518.7,305.4,1
2,-1,-1,0
```

Processed Video:

- Ball centroid marked using a circle
- Trajectory overlay drawn across frames
- MP4 format

---

6. Repository Structure

```
EdgeFleetCricketTracker/
├─ code/
│   ├─ train.py
│   ├─ tracker.py
│   ├─ utils.py
│   ├─ tracking.py     (semi-automatic tracker)
│   └─ inference.py
├─ annotations/
├─ results/
├─ README.md
├─ requirements.txt
└─ report.pdf
```

---

7. Limitations

- Manual clicking is required for key frames
- Fully automatic detection is not implemented
- Performance depends on user accuracy during click.

8. Future Improvements

- Incorporate a trained detector if labeled data becomes available
- Use Kalman filtering for smoother trajectory estimation
- Integrate automatic frame selection using advanced motion cues

9. Conclusion

This project delivers a reliable and reproducible cricket ball tracking system under strict dataset constraints. The semi-automatic approach ensures accurate centroid detection and complete output generation while remaining compliant with the problem requirements.