AthleteRise – AI-Powered Cricket Analytics

Assignment: Real-Time Cover Drive Analysis from Full Video

Objective

Build a Python-based system that processes the **entire** cricket video **in real time** (no screenshots/keyframe exports), performs pose estimation frame-by-frame, and outputs an annotated video with live overlays and a final shot evaluation.

Video to Analyze

YouTube Short: https://youtube.com/shorts/vSX3IRxGnNY

Scope & Requirements (Base)

1) Full Video Processing (Real-Time Flow)

- Download input video and process **all frames** sequentially (OpenCV).
- Normalize FPS/resolution if needed but preserve real-time or near-real-time flow.
- **Output**: a single annotated **.mp4** (or .avi) saved to `/output/`.

2) Pose Estimation (Per Frame)

- Use MediaPipe, OpenPose, or similar.
- Extract keypoints each frame for: head, shoulders, elbows, wrists, hips, knees, ankles.
- Gracefully handle missing joints/occlusions.

3) Biomechanical Metrics (Per Frame or Rolling)

Compute and log:

- **Front elbow angle** (shoulder–elbow–wrist)
- **Spine lean** (hip–shoulder line vs. vertical)
- **Head-over-knee vertical alignment** (projected distance)
- **Front foot direction** (toe/foot angle vs. crease or video x-axis surrogate)

(Bat tracking is not required in base scope.)

4) Live Overlays in the Output Video

- Draw pose skeleton on each frame.
- Display real-time metric readouts (e.g., “Elbow: 115°”).
- Short feedback cues when thresholds are breached:
    - ✅ “Good elbow elevation”
    - ❌ “Head not over front knee”

5) Final Shot Evaluation (End of Video)

- Compute and save a summary score (1–10) for:
    - Footwork
    - Head Position
    - Swing Control
    - Balance
    - Follow-through
- Include 1–2 lines of actionable feedback per category.
- Save summary to `evaluation.json` or `evaluation.txt`.

Deliverables

- `cover_drive_analysis_realtime.py` (main script)
- `/output/`
    - `annotated_video.mp4` (with overlays, full-length)
    - `evaluation.json` (or `.txt`) with category scores & comments
- `requirements.txt` (or `environment.yml`)
- `README.md`
    - Setup & run instructions
    - Notes on assumptions/limitations

Acceptance Criteria (Base)

- Processes the **entire** video and produces one annotated output video.
- Per-frame pose overlays and at least **three** live metrics shown.
- Generates a final multi-category score + feedback file.
- Handles missing detections without crashing.
- Clear instructions; repo is reproducible.

BONUS (Advanced) — Optional but Strongly Valued

(Harder items; implement any that you can. These weigh positively in evaluation.)

1. **Automatic Phase Segmentation**
   Detect phases: **Stance** → **Stride** → **Downswing** → **Impact** → **Follow-through** → **Recovery** using joint velocities/angles or heuristics.

2. **Contact-Moment Auto-Detection**
   Identify likely bat-ball contact via motion peaks, wrist velocity spikes, or ball trajectory heuristics.

3. **Temporal Smoothness & Consistency**
   Compute smoothness metrics (e.g., frame-to-frame angle deltas, variance) and export a small chart (elapsed time vs. elbow angle/spine lean) as an overlay slate at video end or as a PNG in `/output/`.

4. **Real-Time Performance Target**
   Achieve **≥10 FPS** end-to-end on CPU (log average FPS to console). Optimize pipelines (buffering, model choice, reduced resolution, etc.).

5. **Reference Comparison (Benchmarking)**
   Compare the analyzed shot to a provided or public **"ideal" cover drive** (config file with target angle ranges). Report deviations and reflect them in the final score.

6. **Basic Bat Detection/Tracking**
   Approximate bat line via color/shape or a lightweight detector; estimate swing path straightness/angle at impact.

7. **Skill Grade Prediction**
   Map metrics to a grade: **Beginner / Intermediate / Advanced**, with simple thresholds or a rule-based heuristic.

8. **Streamlit Mini-App**
   Simple UI: upload any video → show processed playback → display scores and download link for annotated video & JSON.

9. **Robustness & UX**
    - Fail-safe logging and graceful degradation on missed detections
    - Config file for thresholds/paths
    - Modular function `analyze_video(path: str) -> dict` for future integration

10. **Report Export**
    Generate a brief HTML/PDF report summarizing metrics, scores, and (optional) plots.

Tech Notes & Hints

- Prefer lightweight pose models for speed.
- Normalize coordinates to handle framing differences.
- Calibrate angles to the video frame (define a virtual "crease" axis).