Project Report – Object Tracking in Self-Recorded Videos Using CoTracker-v3

Name: Manish Tiwari
Project Title: Tracking an Object in Self-Recorded Videos
Track: Object Tracking

Introduction

Object tracking is a core task in computer vision where an object is identified and its position estimated across video frames. Real-time tracking is critical for applications such as AR/VR, robotics, and activity monitoring. Traditional trackers often struggle with occlusion, motion blur, and cluttered backgrounds.

CoTracker-v3 overcomes these challenges using a transformer-based space-time attention mechanism, enabling robust tracking of multiple points across long video sequences.

Problem Statement

Given self-recorded videos (smartphone or webcam), track a user-specified object (hand, pet, phone, etc.) throughout the video. Challenges include occlusion, abrupt motion, and complex backgrounds. The objective is to extract accurate and continuous trajectories and visualize tracking results.

Objectives

- Identify objects in self-recorded videos and track them using CoTracker-v3.
- Evaluate frame-to-frame **displacement**, trajectory continuity, and stability under occlusion.
- Reserve space to manually insert plots for frames of interest.
- Analyze performance and validate whether desired outputs are achieved.

Methodology

Dataset:

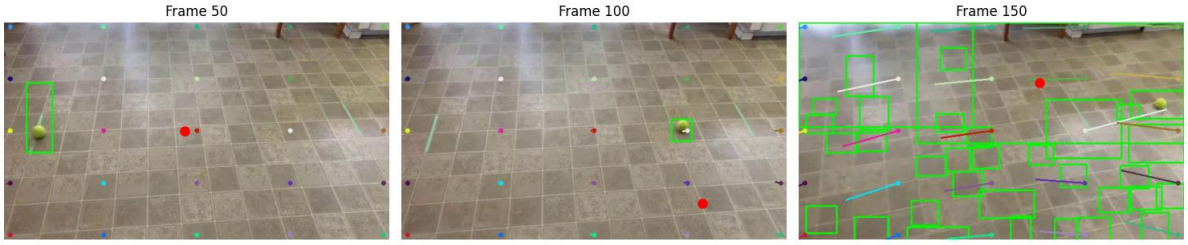
- Self-recorded videos,
- Varied backgrounds, lighting conditions
- Objects: moving items with different frames and background in this case a tennis ball

Model Description:

- Transformer-based **space-time attention**
- Inputs: video frames + initial object cues (points or bounding boxes)
- Outputs: per-frame object trajectories and visibility masks

Pipeline:

1. Preprocess video (resize, frame rate adjustment)
2. Initialize tracker with manually selected points
3. Forward tracking using CoTracker-v3
4. Extract trajectories, bounding boxes, and visibility
5. Visualizations:

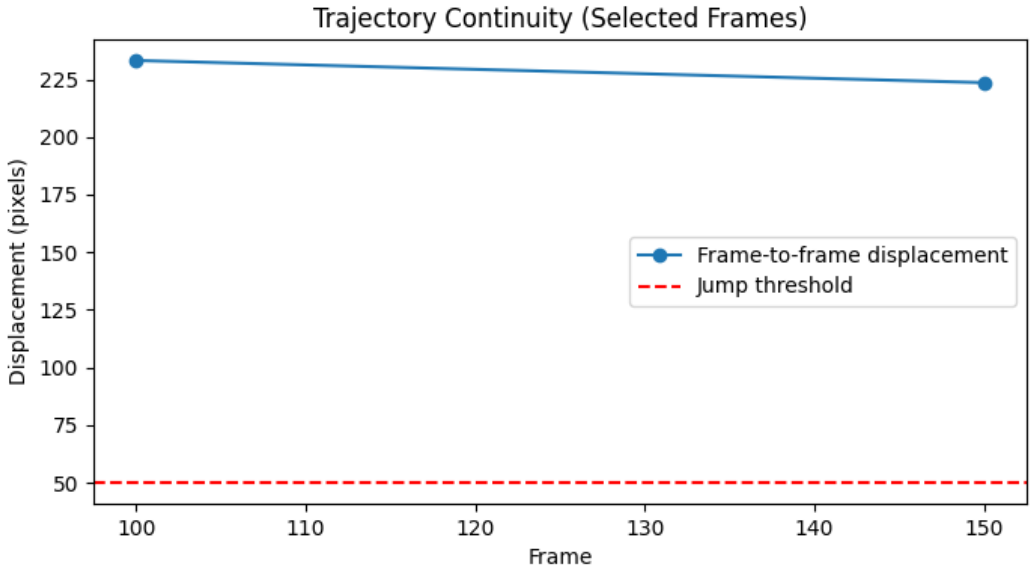


Implementation & Hyperparameters

- **Language & Libraries:** Python 3.10+, PyTorch, NumPy, Matplotlib, ImageIO
- **Model Checkpoint:** scaled_offline.pth
- **Tracking Settings:**
  - grid_size = 20
  - Forward tracking only
  - Object-centric crop: 50×50 pixels
- **Hardware:** GPU-enabled

Outputs:

- Continuous object trajectories
- Bounding box overlays



Results & Observations

- Object successfully tracked from frame 50 onward
- **Frame-to-frame displacement:**
  - Frame 50 → 100: 233.24 pixels
  - Frame 100 → 150: 223.61 pixels
- Qualitative visualization shows smooth trajectories with minimal jumps

References & Citations

- Girdhar, R., et al. "CoTracker: It's Better to Track Together." CVPR 2023. arXiv:2301.03281
- CoTracker GitHub: https://github.com/facebookresearch/cotracker