Comparative Study on Object Tracking Algorithms: DeepSORT, ByteTrack, and BoT-SORT

Author: Aditya Kumar Singh

1. Introduction

Object tracking is a fundamental task in computer vision with applications in surveillance, autonomous driving, robotics, and video analytics. Modern tracking-by-detection approaches rely heavily on accurate object detectors combined with robust data association techniques. This report discusses and compares three prominent tracking algorithms: DeepSORT, ByteTrack, and BoT-SORT.

2. Tracking Methods

2.1 DeepSORT

Overview:
DeepSORT extends the original SORT algorithm by incorporating deep appearance features, significantly improving tracking under occlusion and similar-looking objects.

Working Mechanism:

- Detection using an external detector (e.g., YOLO).
- Kalman Filter for motion prediction.
- Appearance features extracted using a CNN for re-identification.
- Data association using a combination of IoU and cosine distance.

Strengths:

- Robust to short-term occlusions.
- Handles identity switches better.

Weaknesses:

- Heavier computational load.
- Needs fine-tuning of the ReID model.

2.2 ByteTrack

Overview:
ByteTrack introduces the idea of associating low-confidence detections, significantly improving tracking performance without using deep features.

Working Mechanism:

- Splits detections into high and low confidence.
- First matches high-confidence detections, then low-confidence ones.
- No deep appearance model used; purely geometric.

Strengths:

- Extremely fast and efficient.
- Performs well in crowded scenes.

Weaknesses:

- No appearance modeling.
- Dependent on detector quality.

2.3 BoT-SORT

Overview:
BoT-SORT merges the strengths of DeepSORT and ByteTrack, adding motion compensation for improved performance, especially with moving cameras.

Working Mechanism:

- Combines appearance features and motion information.
- Incorporates Global Motion Compensation (GMC).
- Prioritized association based on appearance and geometric proximity.

Strengths:

- Excellent performance in dynamic scenes.
- Reduces identity switches.

Weaknesses:

- More complex to tune.
- Slightly higher computational cost.

Method	Appearance Features	Motion Compensation	Speed (FPS)	Accuracy (MOTA)	Best Use Case
DeepSORT	Yes	No	Moderate	Good	Static camera tracking
ByteTrack	No	No	Very High	Very Good	Real-time and crowded environments
BoT-SORT	Yes	Yes	Moderate-High	Excellent	Moving camera, dynamic environments

4. Practical Considerations

- **Application Needs:**
  - For real-time needs — ByteTrack.
  - For occlusion handling — DeepSORT.
  - For moving camera scenes — BoT-SORT.
- **Integration:**
  - Easily combined with detectors like YOLOv8.
- **Deployment:**
  - Docker containers recommended for reproducibility.
  - GPU acceleration improves performance for DeepSORT and BoT-SORT.

5. Conclusion

Each tracker has its own strengths:

- **DeepSORT** excels in identity preservation.
- **ByteTrack** delivers high efficiency with minimal complexity.
- **BoT-SORT** achieves the best overall performance, especially in challenging scenarios with moving cameras.

Selecting the right tracker depends on the specific needs of the application regarding speed, accuracy, and environmental complexity.