Ahmedabad University

CSE623 - Group XYZ

Offline Track association problem

Hariohm Bhatt
AU2240085
B.Tech CSE

Meet Rathi
AU2240106
B.Tech CSE

Aditya Agrawal
AU2240153
B.Tech CSE

Harsh Panchal
AU2240160
B.Tech CSE

Jeel Kadivar
AU2140181
B.Tech CSE

Problem Statement

In multi-object tracking, due to occlusion, missed detections and abrupt motion changes in varied environments, results in the fragmented tracklets. This offline tracklet association problem focuses on the merger of these fragmented tracklets to form them into continuous trajectories

Key challenges:

- Identity Switches
- Object occlusion
- Feature consistency (like handling Abrupt motion changes)
- Missed detections

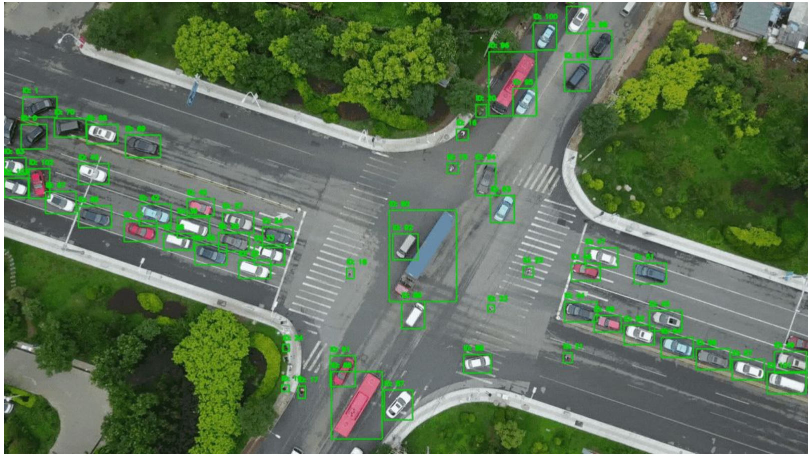
Literature Review

Research Paper	Method Used	Key Findings	Limitations
Detection and Tracking Meet Drones Challenge [1] [2021]	Large-scale VisDrone dataset; Comparative analysis of SOTA	- Introduces VisDrone benchmark for drone-based detection/tracking; - Highlights motion blur, scaling & viewpoint issues; - Reviews SOTA methods	- Primarily a benchmark rather than a solution; - Complexity may lead to overfitting
Traffic Flow Characteristics and Conflict Analysis [2][2024]	YOLO + SORT; KD-Tree + cubic spline	- Reconstructs continuous tracks from fragments; - Uses smoothing and KD-Tree matching; - Cubic spline merges tracklets	- Focused on toll station scenario; - Needs reliable detection for accurate merging
Unifying Short and Long-Term Tracking (SUSHI) [3][2023]	Hierarchical GNN; Multi-level tracklet merging	- Scalable GNN framework for short/long-term tracking; - Merges tracklets progressively; - Learns motion/appearance cues adaptively	- Complex hierarchical GNN; - Dependent on good detection and large graphs

VisDrone Dataset

This dataset is captured in 14 cities of China, under urban and suburban settings under various conditions

- Video Clips: 116 video clips with 179,264 frames
- Static images: 10209 static images
- Annotations: overall 2.5 million bounding boxes are annotated



Dataset Features

- Scale variations
- Occlusion
- Motion Blur
- Viewpoint changes

VisDrone Dataset Annotations

FrameId	1	2	3	4	5	6	7	8	9	10
ObjectId	9	9	9	9	9	9	9	9	9	9
x	916	915	915	914	914	913	913	913	912	912
y	446	447	449	451	452	454	456	457	459	461
width	101	101	101	102	102	103	103	103	104	104
height	150	151	151	152	153	153	154	155	155	156
score	1	1	1	1	1	1	1	1	1	1
class	4	4	4	4	4	4	4	4	4	4
visibility	1	1	1	1	1	1	1	1	1	1
ignored	0	0	0	0	0	0	0	0	0	0

- **FrameId:** Unique identifier for the frames, it tracks the order of the frames
- **Bounding Box:** consists of **x, y, width, height**, which are the coordinates of the object
- **Object Id:** Unique identifier for object across the frames
- **Object Category:** class label for the object (in our case “cars”)
- **Score:** It is the confidence score of detection
- **Visibility:** indicating visibility level, which helps to detect occlusion
- **Ignored:** It is a flag to identify if the object is to be ignored

Methodology

Preprocessing:

- Selected subset of VisDrone sequences & two target objects
- Filter frames & annotations with selected objects
- Create intentionally fragmented tracklets (15–60 frames)
- Group similar frames to form initial short tracklets

Feature Extraction:

Motion Features

- Average Velocity (tracklet-wide mean)
- Bounding Box Area (captures size)
- Aspect Ratio (shape of bounding box)

Appearance Features

- Color Histogram (Flattened Color Histogram (512 values)
- SIFT Descriptors (scale & rotation invariance)Flattened SIFT Descriptor (128 values)

Methodology

Tracklet Merging:

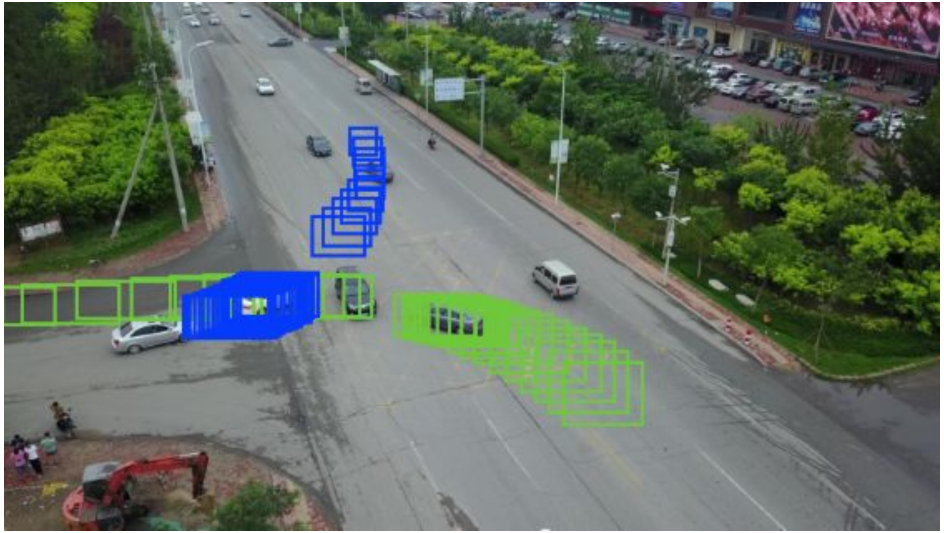
- Feature Matrix Construction:
  - Combine motion + appearance features and standardize them
- KD-Tree Association:
  - Build_kdtree: Recursively builds the KD-Tree by splitting points along alternating axes and selecting the median.
  - Query_kdtree: Finds the nearest neighbor by traversing the tree and backtracking if necessary.
  - Merge_tracklets: Builds the KD-Tree, queries it for each tracklet, and stores the nearest neighbor in a dictionary.
- Saving Results:
  - Output final (combined) tracklet IDs with consistent filenames

Algorithm 1 Merge Tracklets Using KD-Tree

```
function BuildKDTree(points, depth)
    if points = ∅ then return NULL
    end if
    axis ← depth mod number of dimensions
    sorted_points ← sort(points by coordinate at index axis)
    median ← ⌊len(sorted_points)/2⌋ return { point: sorted_points[median], axis: axis, left: BuildKDTree(sorted_points[0:median], depth+1), right: BuildKDTree(sorted_points[median+1:], depth+1) }
end function
function QueryKDTree(node, target, depth, best, bestDist)
    if node = NULL then return (best, bestDist)
    end if
    d ← Distance(target, node.point)
    if best = NULL or d < bestDist then
        best ← node.point
        bestDist ← d
    end if
    axis ← depth mod number of dimensions
    if target[axis] < node.point[axis] then
        nextBranch ← node.left
        otherBranch ← node.right
    else
        nextBranch ← node.right
        otherBranch ← node.left
    end if
    (best, bestDist) ← QueryKDTree(nextBranch, target, depth+1, best, bestDist)
    if |target[axis] − node.point[axis]| < bestDist then
        (best, bestDist) ← QueryKDTree(otherBranch, target, depth+1, best, bestDist)
    end ifreturn (best, bestDist)
end function
function MergeTracklets(feature_matrix, tracklet_ids)
    tree ← BuildKDTree(feature_matrix, 0)
    merged ← {}
    for each id in tracklet_ids do
        (nearest, _) ← QueryKDTree(tree, feature_matrix[id], 0, NULL, ∞)
        merged[id] ← nearest
    end forreturn merged
end function
```

Ahmedabad University

Results



Future work

- **Evaluate Performance on Diverse Objects & Broken Tracklets**
  Further we plan to test across varied object categories (cars, bikes, buses etc.) and fragmented tracklets to identify failure modes in complex scenarios.

- **Implement Cubic Spline Interpolation for Missing Data**
  Propose using cubic spline interpolation to predict and restore missing annotations in discontinuous tracklets, ensuring smoother trajectory recovery.

- **Investigate Feature Robustness & Computational Efficiency**
  Systematically evaluate motion and appearance features to optimize accuracy vs. computational overhead. Also further extract more features based on spatial, temporal and appearance.

- **Work with different datasets**
  We are planning to use other dataset such as VSAI dataset to increase generalisation of objects.

- **Explore Adaptive Tracklet Merging Strategies**
  Test graph-based and probabilistic merging methods as alternatives to KD-Tree for robust association in crowded scenes.

References

[1] P. Zhu, L. Wen, D. Du, X. Bian, H. Fan, Q. Hu, and H. Ling, “Detection and Tracking Meet Drones Challenge,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. X, no. Y, pp. 1–15, 2025.

[2] Q. Ren, J. He, Z. Liu, and M. Xu, “Traffic Flow Characteristics and Traffic Conflict Analysis in the Downstream Area of Expressway Toll Station Based on Vehicle Trajectory Data,” Asian Transport Studies, vol. 10, 100138, 2024.

[3] O. Cetintas, G. Bras´o, and L. Leal-Taix´e, “Unifying Short and Long- Term Tracking with Graph Hierarchies,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. X, no. Y, pp. 1–15, 2025.

[4] X. Zhang, H. Yu, Y. Qin, X. Zhou, and S. Chan, “Video-Based Multi-Cacmera Vehicle Tracking via Appearance-Parsing Spatio-Temporal Trajectory Matching Network,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 34, no. 10, pp. 10077, Oct. 2024.

[5] D. B. Reid, “An Algorithm for Tracking Multiple Targets,” IEEE Transactions on Automatic Control, vol. 24, no. 6, pp. 843–854, 1979.

[6] Y. Bar-Shalom and T. E. Fortmann, Tracking and Data Association, Academic Press, 1988.

[7] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, “Simple Online and Realtime Tracking,” in Proc. IEEE International Conference on Image Processing (ICIP), 2016, pp. 3464–3468.

[8] J. L. Bentley, “Multidimensional Binary Search Trees Used for Associative Searching,” Communications of the ACM, vol. 18, no. 9, pp. 509–517, 1975.

[9] D. G. Lowe, “Distinctive Image Features from Scale-Invariant Keypoints,” International Journal of Computer Vision, vol. 60, no. 2, pp.91–110, 2004.

[10] C. de Boor, A Practical Guide to Splines, Springer-Verlag, 1978.