Using 3D Scene Structure to Improve Tracking

UNIVERSITY OF SOUTHERN CALIFORNIA

Jan Prokaj and Gérard Medioni

http://iris.usc.edu/people/prokaj/

{prokaj|medioni}@usc.edu

Problem

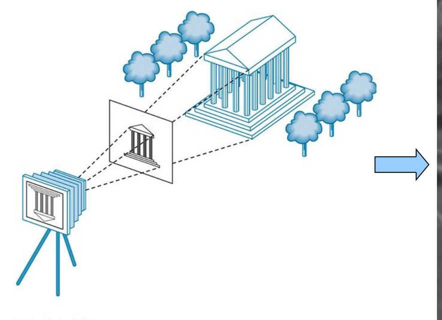
- Aerial imagery collected over urban areas contains many occlusions for vehicles
- Current tracking algorithms cannot track vehicles through such occlusions due to weak object appearance and complexity of motion prediction over long period

Strategy

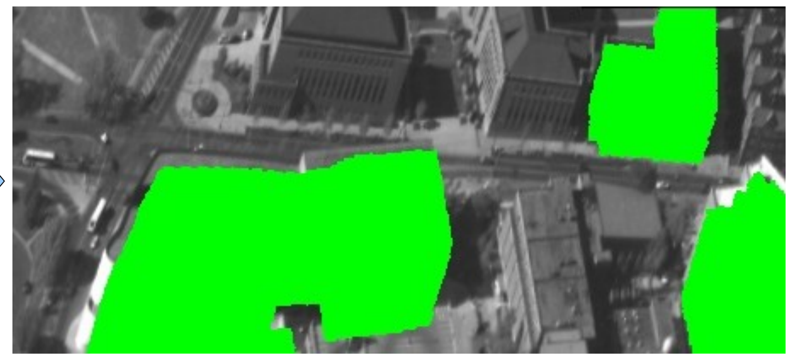
- Use known 3D scene structure to estimate a *dynamic occlusion map*: binary map, which indicates what regions of the image are occluders of moving objects; dynamic due to camera motion
- Use this map to link broken tracks

Estimating dynamic occlusion maps

Geo-referenced 3D models



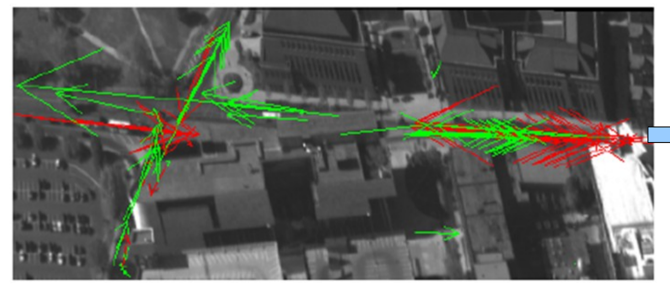
Camera pose in geo-coordinates



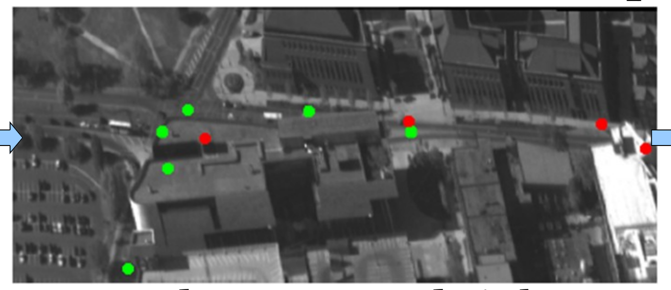
Occlusion map is the projection

Sources and Sinks

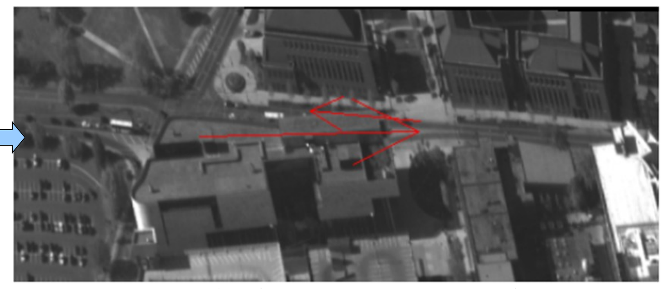
- Occlusions cause tracks to prematurely end (generating a traffic sink) and reinitiate (generating a traffic source)
- Estimate these locations by clustering those track terminations and initiations near an occlusion region
- Use Hungarian algorithm to find source-sink correspondence



Example of occlusion events

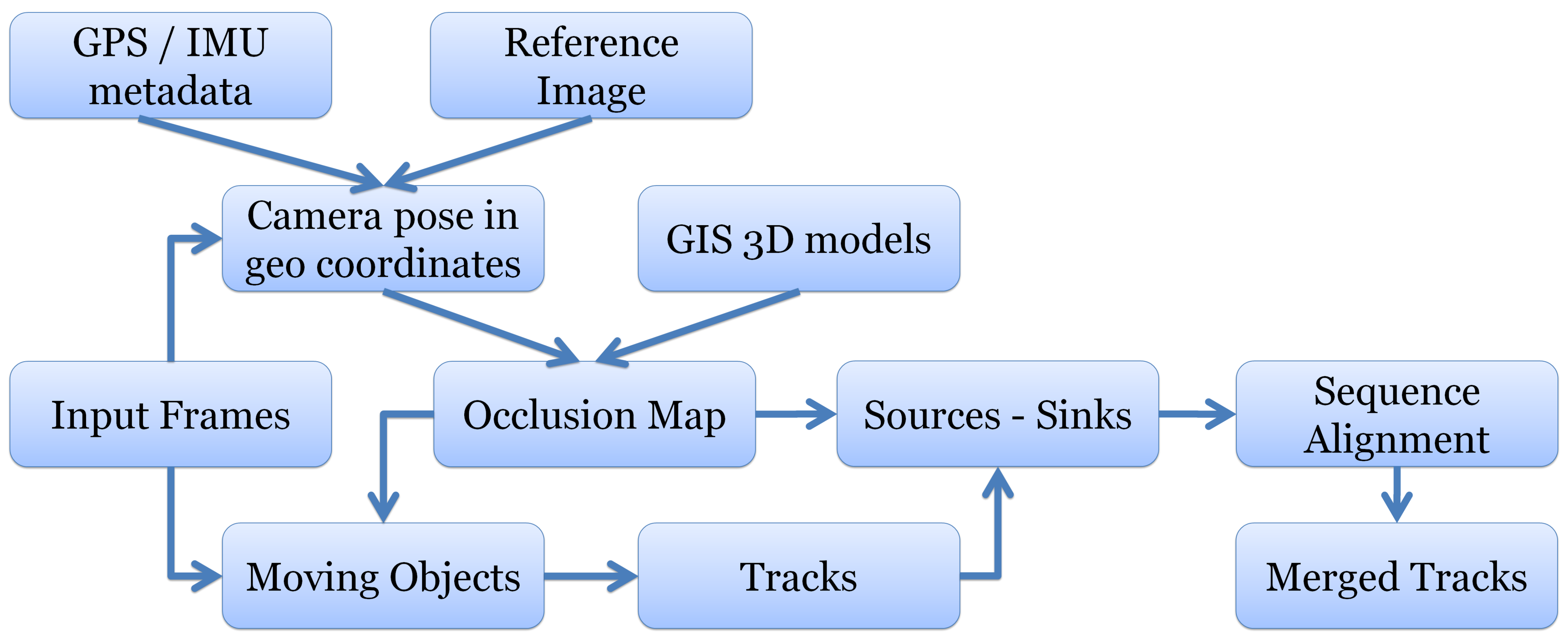


Detected sources and sinks (clusters)



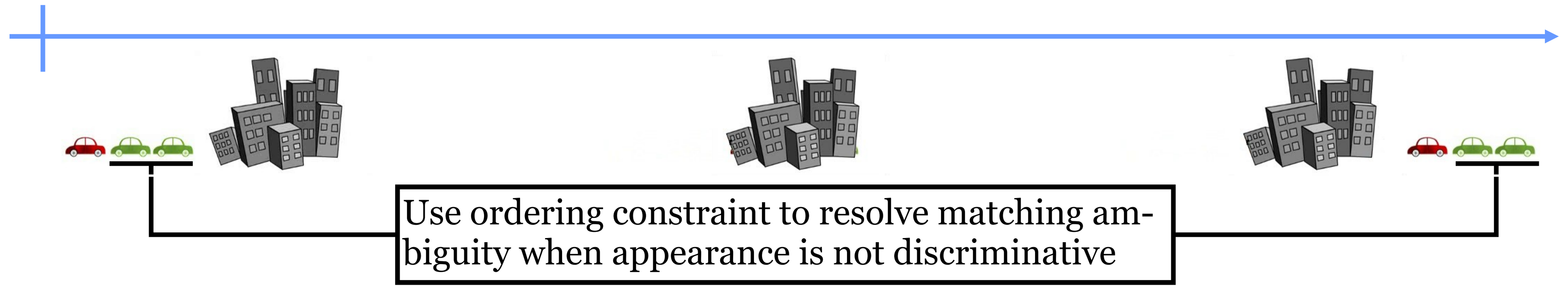
Source—sink correspondence

Approach at a glance



Ordering Constraint

- Want to match a set of tracks at each source with tracks at corresponding sink
- *The sequence of vehicles becoming occluded should be approximately the same as the sequence of vehicles becoming visible*



Sequence Alignment

- Match two *sequences* of tracks, not two sets of tracks
- Dynamic programming solution

$$f(i,j) = \min[\xi_{x_i y_j} + f(i-1,j-1), \delta + f(i-1,j), \delta + f(i,j-1)]$$

Cost of matching track i in sequence x with track j in sequence y

Cost of not matching a track in a sequence (to handle missing/extra tracks)

- Matching cost based on feasibility, motion, appearance, and vehicle size

$$\xi_{x_i y_j} = \tau(x_i, y_j) * \gamma(x_i, y_j) * (\alpha(x_i, y_j) + \beta(x_i, y_j))$$

$$\tau(x_i, y_j) = \begin{cases} 1 & \text{link } y_j \text{ with } x_i \text{ is feasible} \\ \infty & \text{otherwise} \end{cases}$$

$$\gamma(x_i, y_j) = \min(|t'_j - t_j|, G)/G$$

$$\alpha(x_i, y_j) = \min(D(h(x_i), h(y_j)), A)/A$$

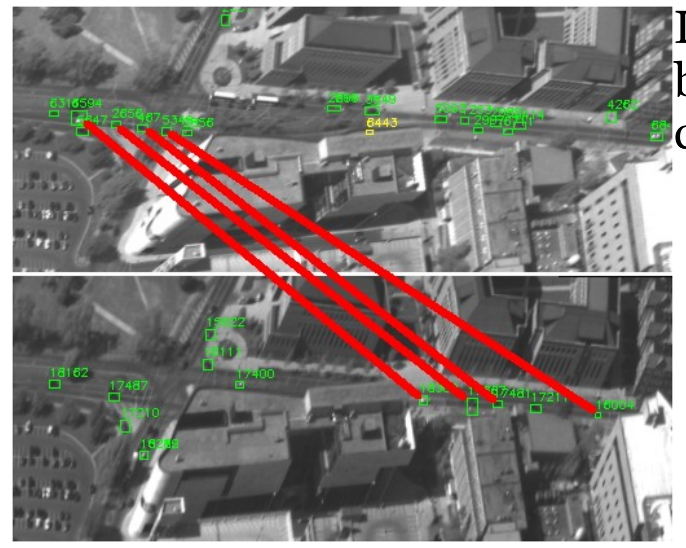
$$D(p,q) = 0.5 * (D_{KL}(p||q) + D_{KL}(q||p))$$

$$\beta(x_i, y_j) = \min(|s(x_i) - s(y_j)|, B)/B$$

Results

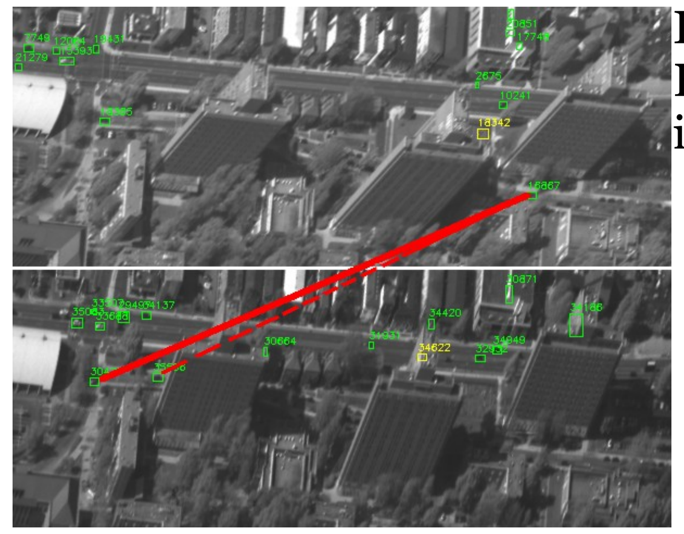
	Proposed Algorithm	Hungarian Algorithm
Correctly Linked	58%	42%
Incorrectly Linked	3.4%	5.8%

- Examples of linked tracks, as determined by the proposed algorithm. The top images shows track(s) just before termination and the bottom image shows new track(s) being initiated

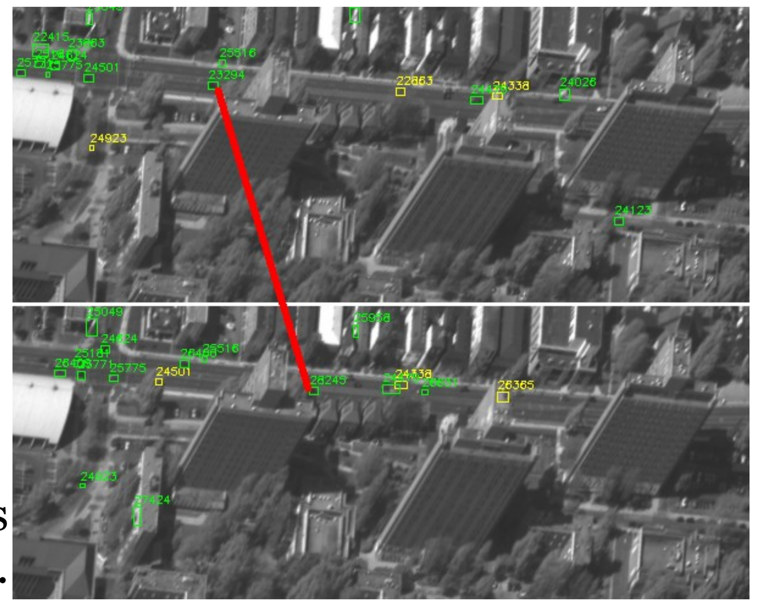


In this sequence, the vehicles stopped behind a building. The proposed algorithm was still able to correctly link most tracks broken by the occlusion.

Tracks are correctly linked even in sparse environments.



In this sequence, both the proposed algorithm and Hungarian algorithm failed to link the correct tracks, indicated by the dashed line.



Short occlusions, as shown by this example, are still handled correctly.

Conclusions and Future Work

- The proposed algorithm outperforms the Hungarian algorithm by correctly linking more tracks while minimizing incorrect links
- Extension to multiple sources/sinks may be possible with known road network

Acknowledgements

This work was supported in part by grant DE-FG52-08NA28775 from the U.S. Department of Energy and by funding from Lawrence Livermore National Lab.