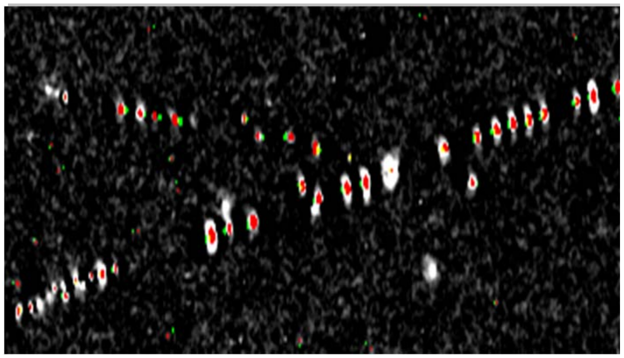
SAND2015-0223PE

Exceptional service in the national interest

Sandia National Laboratories



Large-Scale Tracking LDRD

Dave Melgaard, Ross Hansen, Joshua Love, Ray Byrne

U.S. DEPARTMENT OF ENERGY

NNSA National Nuclear Security Administration

Outline

- Project Goals
- Tracking Algorithms
  - PMHT – Ray/Dave
  - RANSAC - Joshua
  - Proximity Tracker – Dave
  - Tracklet Inference from Factor Graphs - Ross
- Summary

Project Goals

- One year internal R&D effort, 10/2013-9/2014
- Project goals:
  - Identify tracking algorithms that scale well to large numbers of targets (e.g. 100's – 1000's of simultaneous tracks)
  - Identify high-performance computing architectures for large-scale tracking
  - Quantify the impact of target phenomenology and sensor characteristics on vehicle detection and tracking in an urban environment

Evaluation Metrics

- **Multi Object Tracking Accuracy (MOTA)**: $1 - \frac{\sum_t FP(t)+FN(t)+IDS(t)}{\sum_t N_{truth}(t)}$
- **Mostly Tracked (MT):** Percentage of targets that are tracked for more than 80% of its detections regardless of identity switches
- **Mostly Lost (MT):** Percentage of targets that are not tracked for more than 20% of its detections regardless of identity switches
- **Mostly Singly Tracked (MST)**: Similar to MT, accounting for identity switches (ie, 80% of detections are followed by a single track)
- **Mostly Singly Lost (MSL):** Similar to ML, accounting for identity switches
- **False Positives**: The number of tracked observations that were not true detections
- **False Negatives:** The number of true detections that were not associated with a track
- **Identity Switches**: The number of times a tracker switches between two ground-truth targets

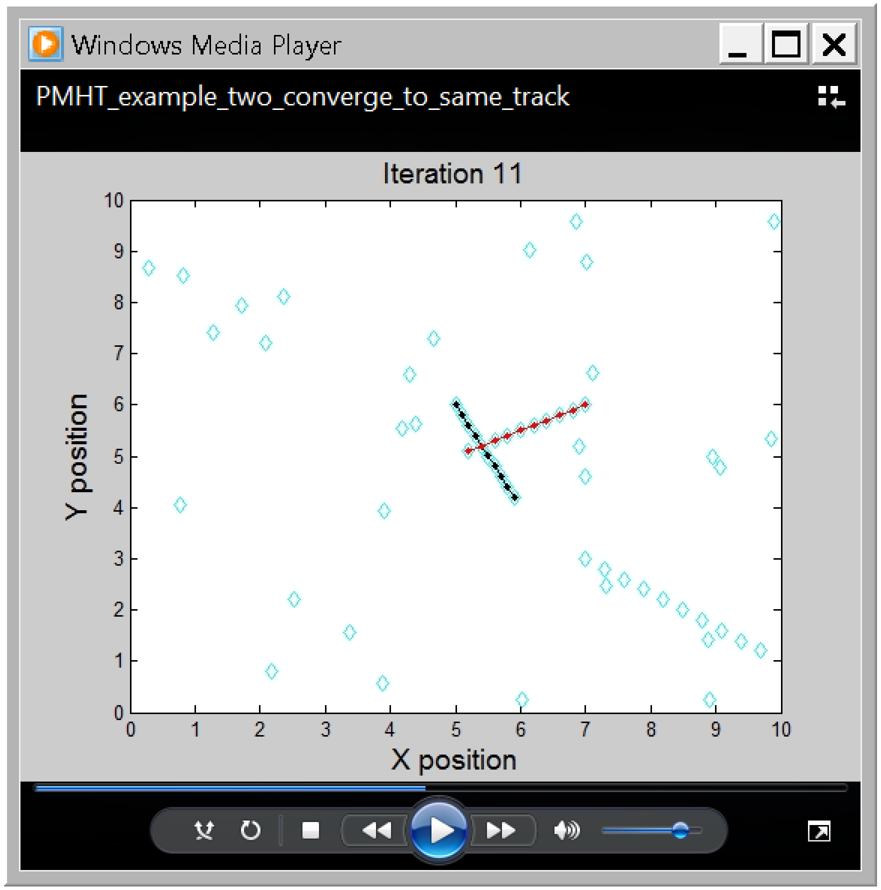
Data Sets

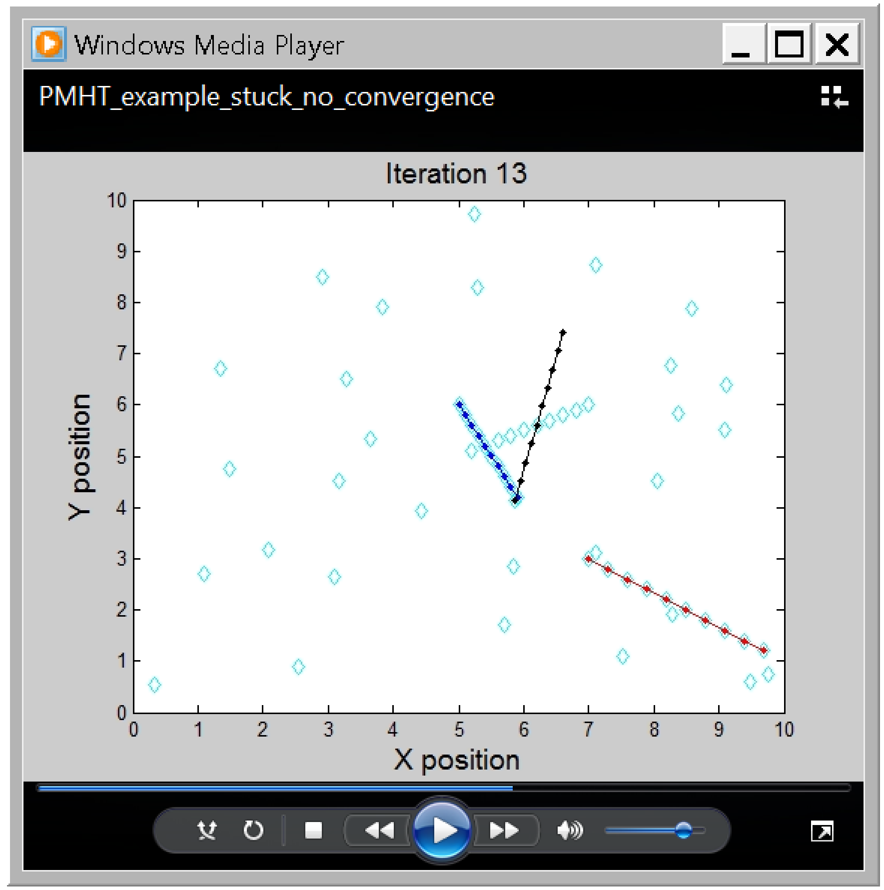
- Algorithms were evaluated with the following data sets:
  - Video from Sandia Peak (limited truth data)
  - SUMO vehicle simulator (Socorro, NM data set with ~780 vehicles, 10Hz data, 6000 samples)
    - Simulated intensity (based on heading)
    - Simulated vehicle color (based on distribution of vehicle colors)
  - AFRL UAV data (truth data available)

Tracking Algorithms - PMHT

- Probabilistic Multi-Hypothesis Tracker (PMHT) was the initial approach
  - Probabilistic detection to track association
  - Scales well to large numbers of targets
  - Batch-processing algorithm
- MATLAB simulations of the algorithm identified the following concerns:
  - Sensitivity to track initial conditions (e.g. impacts convergence)
  - Poor convergence
    - Lack of convergence, e.g. settles on false track
    - Missed convergence, e.g. multiple tracks converge to the same track, others missed altogether

Tracking Algorithms - PMHT





Example: PMHT missed track

Example: PMHT poor convergence

Tracking Algorithms - PMHT

- There are some potential modifications to “fix” the PMHT
  - Remove a track once it has converged
  - Preprocessing to identify better initial conditions
  - “PMHT: Problems and Some Solutions” by Peter Willet, Yanhua Ruan, and Roy Streit go through a list of problems and potential solutions. Also make the statement: “The probabilistic multihypothesis tracker (PMHT) is a target tracking algorithm of considerable theoretical elegance. In practice, its performance turns out to be at best similar to that of the probabilistic data association filter (PDAF); and since the implementation of the PDAF is less intense numerically the PMHT has been having a hard time finding acceptance.”
- Based on our experience, and the comments of Peter Willet, Yanhua Ruan, and Roy Streit, we concluded that PMHT is not a viable solution. (Roy Streit invented the PMHT)

RANSAC: Random Sample Consensus

- RANSAC video image

1. Input: a set of measurements
2. Randomly sample n measurements
3. Fit the n measurements to the model's free parameters
4. Calculate how many measurements are inliers of the model
5. If an insufficient number of inliers, repeat at 2, if a sufficient number of inliers, terminate.

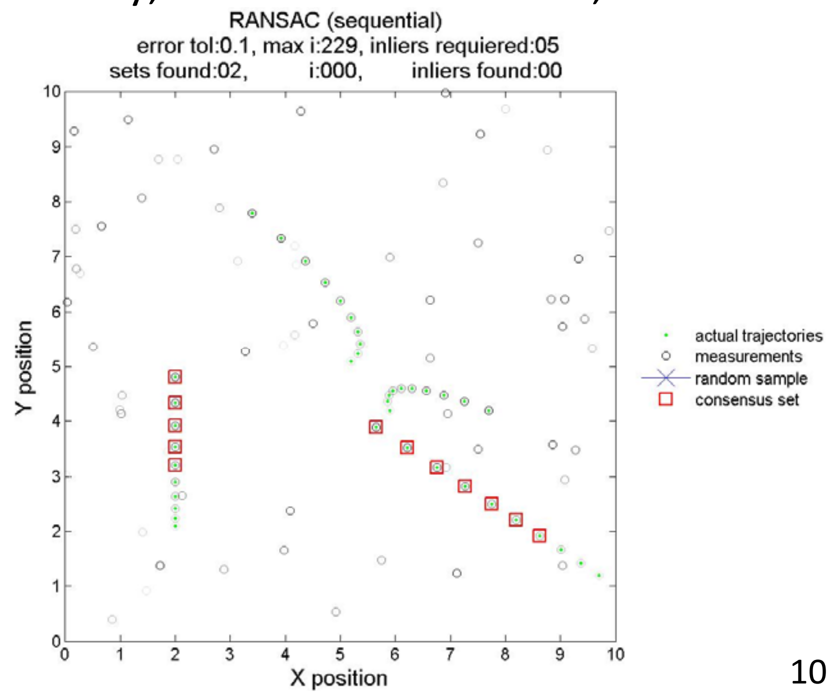
- Sequential RANSAC video image

6. Remove all inliers in the consensus set, repeat from 1.

- Sequential RANSAC with measurement noise & missed detections video image

RANSAC: Random Sample Consensus

- RANSAC parameters
  - Error tolerance: chosen based on measurement errors
  - Max iterations: chosen based on number of measurements
  - Inliers required: chosen based on gross errors (false positives)
- RANSAC also requires a model to be specified
  - Examples: constant position, constant velocity, constant acceleration, …
- A constant velocity model cannot generally describe a constant acceleration target video



RANSAC: Random Sample Consensus

- Some large scale tracking applications are not constant position / velocity / acceleration / jerk / …
  - e.g. cars driving in a city
  - More advanced dynamic models (e.g. a Dubin's vehicle) require the input to be known
  - The input (e.g. desired turn rate, desired velocity) cannot be directly measured
  - Human drivers decide when to stop/start/accelerate/decelerate
  - Small segments of a car's trajectories can be approximated as constant, then potentially connected at a higher level.

- This issue exists for parallel RANSAC implementations as well (multiRANSAC & Recursive RANSAC)

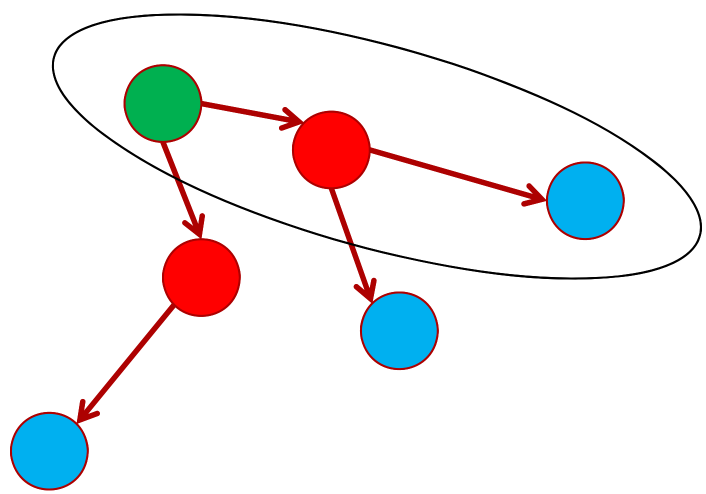
Proximity Tracker

Sandia National Laboratories

- Concern:
  - Vehicles follow a nonlinear motion model
  - Tracking methods typically employ a linear model for tracking
- Approach:
  - Form pairs of detections based on nearest neighbor
  - Merge points to form tracks based on tolerances for velocity
  - Combine tracks
  - Eliminate obviously poor links
    - Span of 5 frames with max velocity 3.5

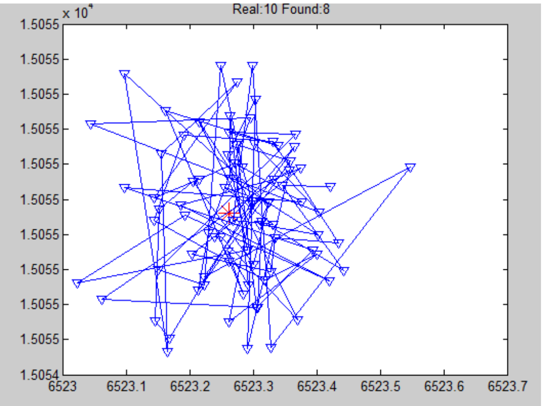
Pairing Details

- Match pairs of detections based on proximity
  - Within the expected maximum velocity (3.5)
  - Within a range of frames (5)
  - Find the closest two, because of the possibility of false alarms or other close vehicles



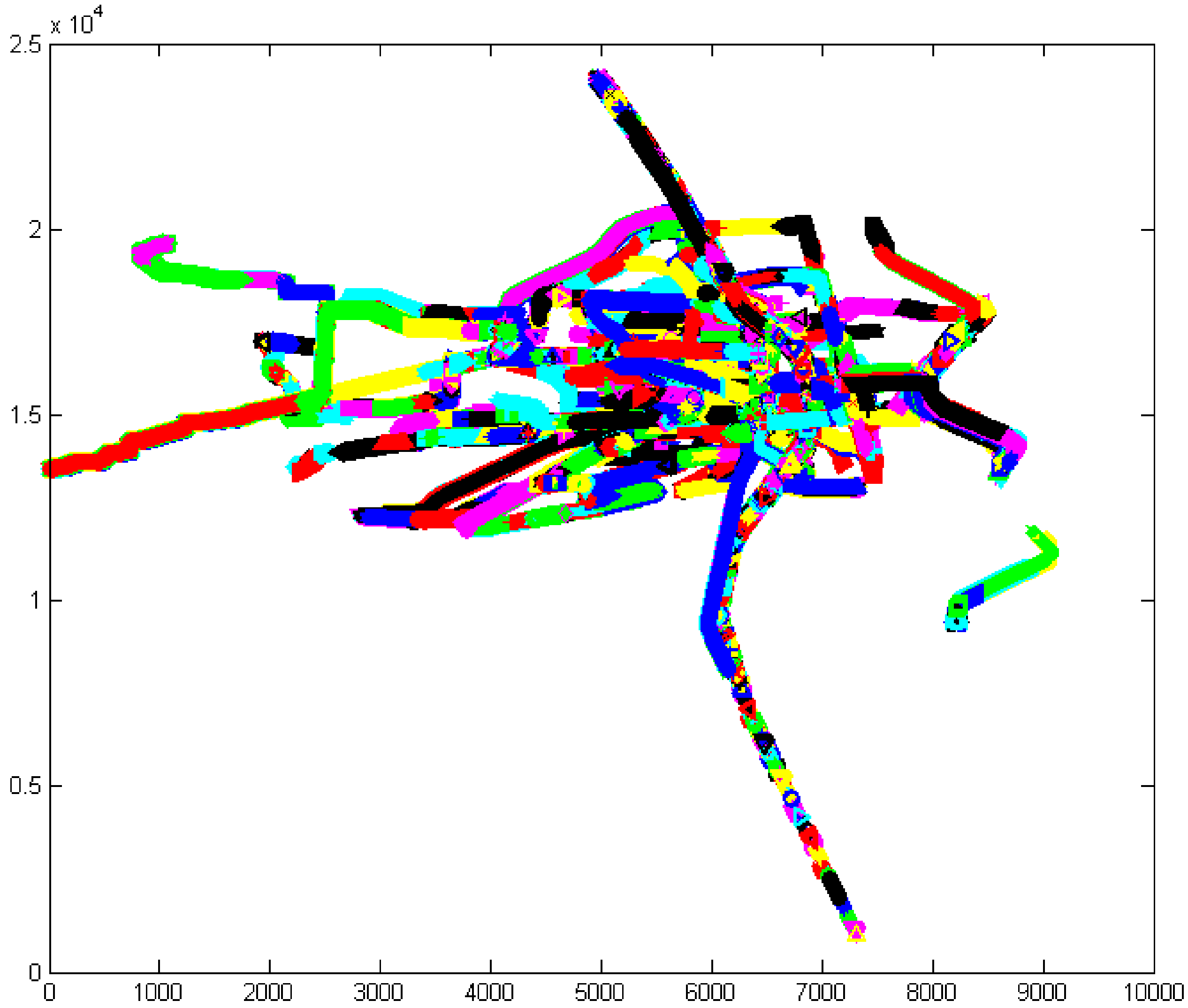
Merge Details

- Maintain active tracks
  - Initial pairs are the initial tracks if they are linked to a future frame (3 points)
  - Subsequent pairs are merged with tracks or start new ones
  - Tracks are terminated if no detections after specified number of frames (5)
- Tracklets are merged based on velocity (2.5) and direction (1.6) (tolerances)
  - For higher velocities direction is useful
  - For low velocity direction is not very useful
    - Below lower limit not used (0.6)
    - Between limits use scaled weighting
    - Above upper limit (0.8) use full tolerance

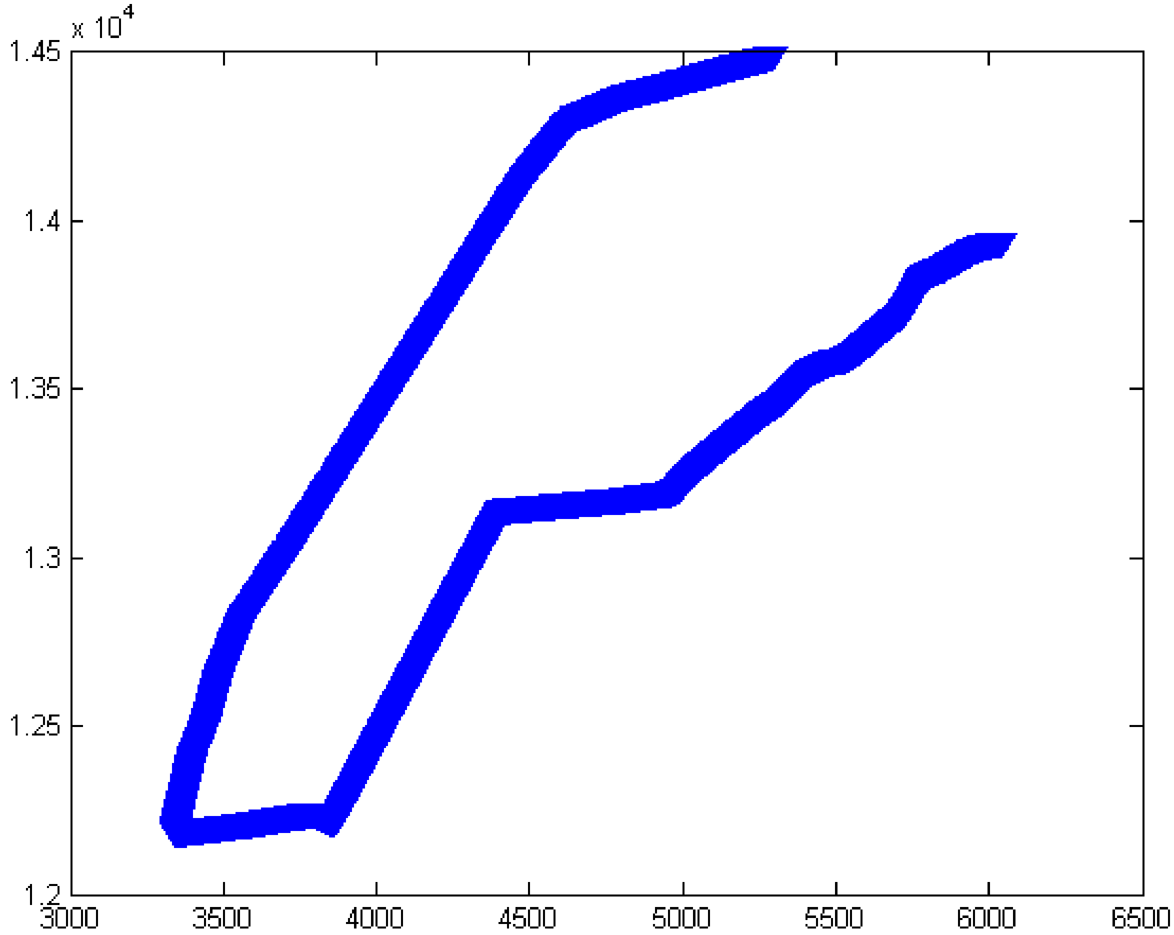


- Need to resolve tracks pointing to the same detection

Socorro Tracks



Good Track Example

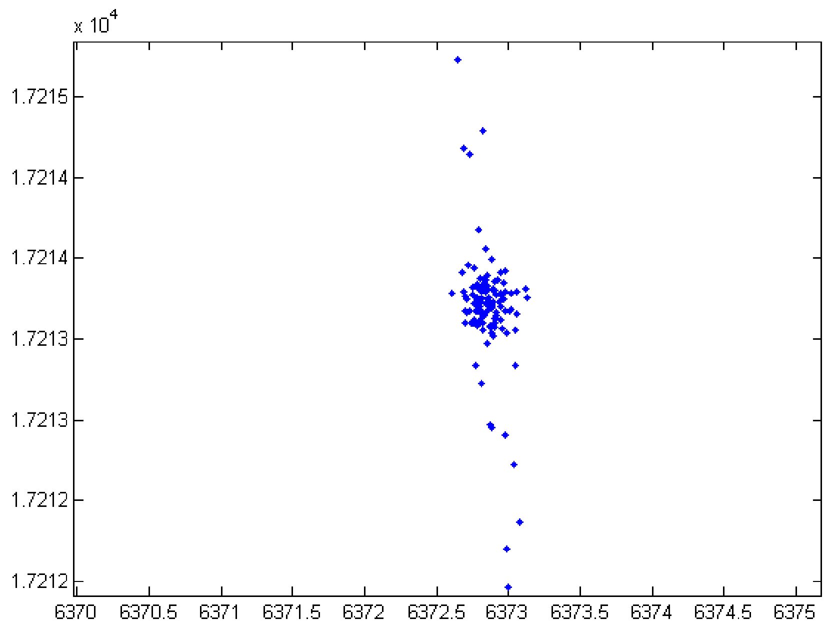


Benchmark Results

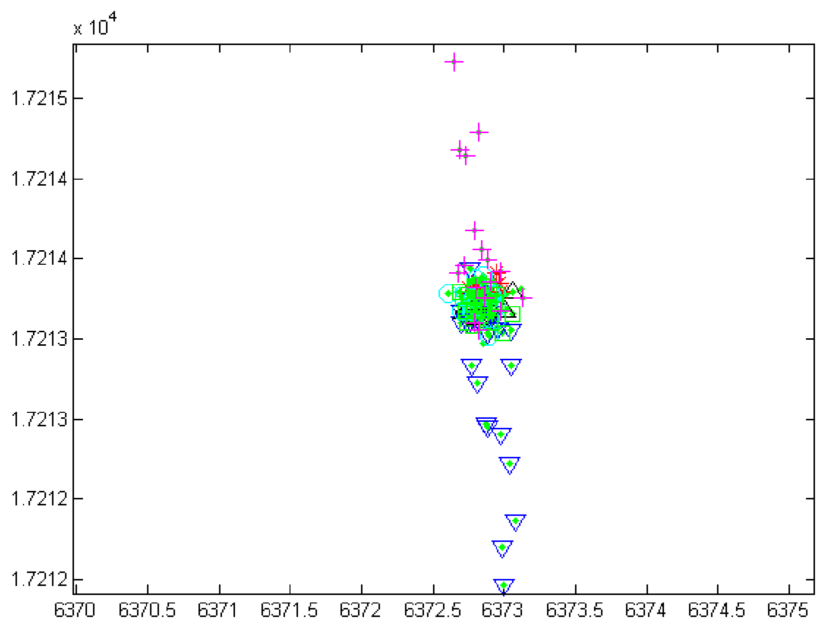
- mota = 0.7308
- mt = 0.9117
- ml = 0.0128
- mst = 0.2394
- msl = 0.1575
- gt = 781
- total_fp=0
- total_fn=96598
- total_ids=304338 (High value)

Tracking Issues: Stopped Vehicles

Truth

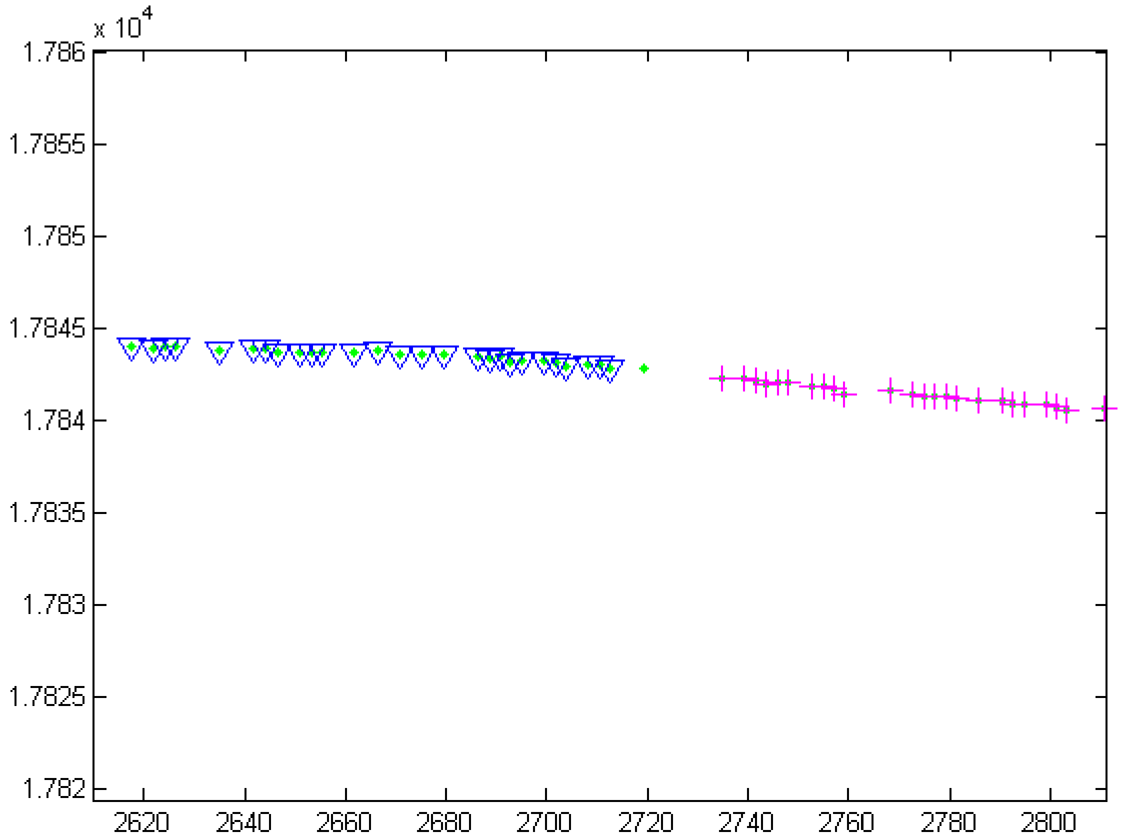


Tracks



Algorithm had breaks when vehicles stopped and restarted

Long breaks caused breaks



Comments

- Algorithm was able to track most of the vehicles
- There are issues with this data set that caused tracks to be broken

Tracklet Influence from Factor Graphs

- Tracklet-based method with a sliding window
- Use a factor graph to model appearance and motion dynamics
  - MAP inference on the factor graph yields tracklets
- Combine tracklets over sliding windows to form persistent tracks

J. Prokaj, M. Duchaineau, G. Medioni, “Inferring Tracklets for Multi-Object Tracking”, *Workshop of Aerial Video Processing Joint w/ CVPR*, 2011

High Level Algorithm over Sliding Window

Construct
Networks

Infer
Tracklets

High Level Algorithm over Sliding Window

Construct Networks

Infer Tracklets

High Level Algorithm over Sliding Window

Construct Networks

1. Gather detections in window of length T
2. Form a Bayesian network rooted at each detection in the first frame of the window
3. Find MPE for all networks (get MAP estimate for each detection)

Infer Tracklets

High Level Algorithm over Sliding Window

Construct Networks

1. Gather detections in window of length T
2. Form a Bayesian network rooted at each detection in the first frame of the window
3. Find MPE for all networks (get MAP estimate for each detection)

Infer Tracklets

High Level Algorithm over Sliding Window

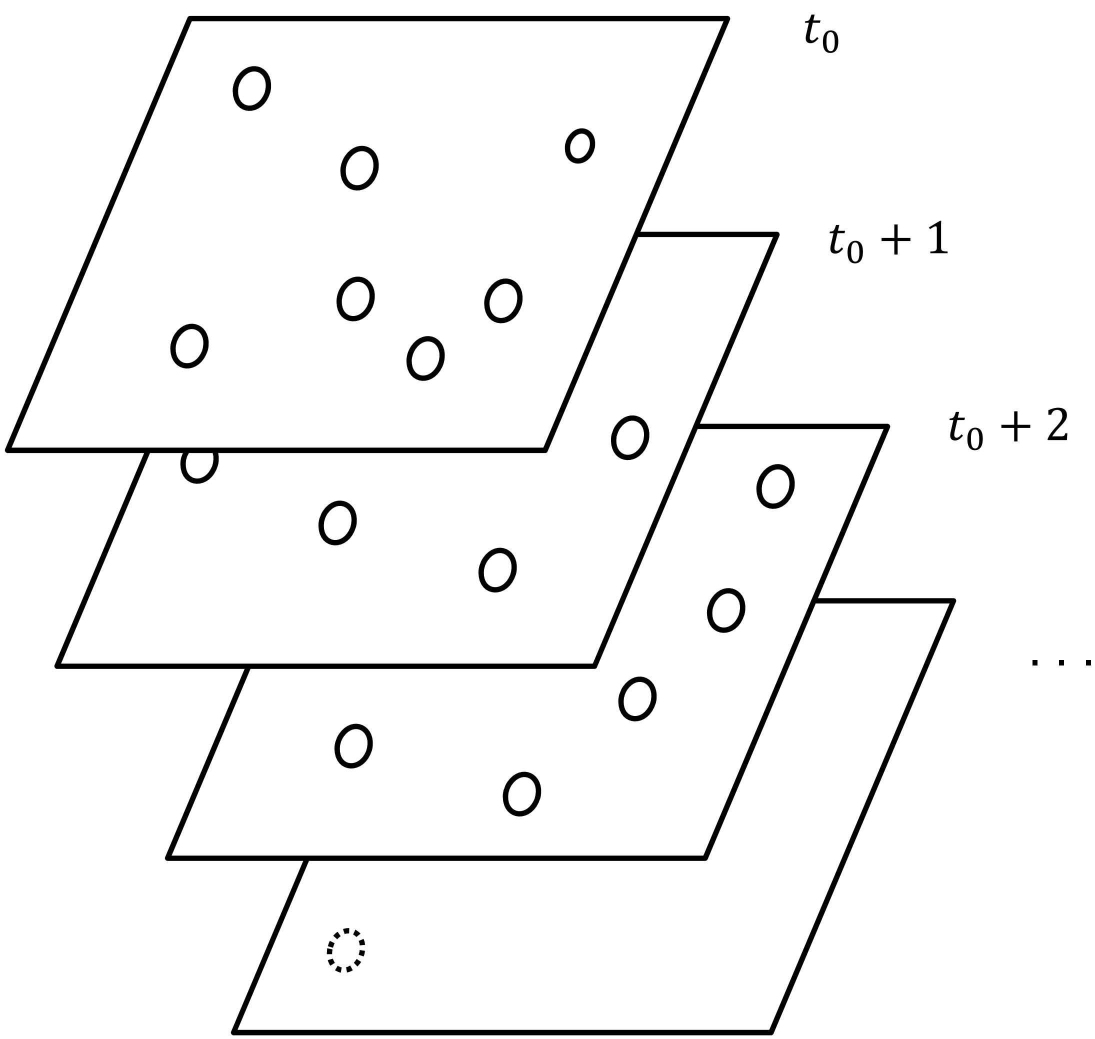
Construct Networks

1. Gather detections in window of length T
2. Form a Bayesian network rooted at each detection in the first frame of the window
3. Find MPE for all networks (get MAP estimate for each detection)

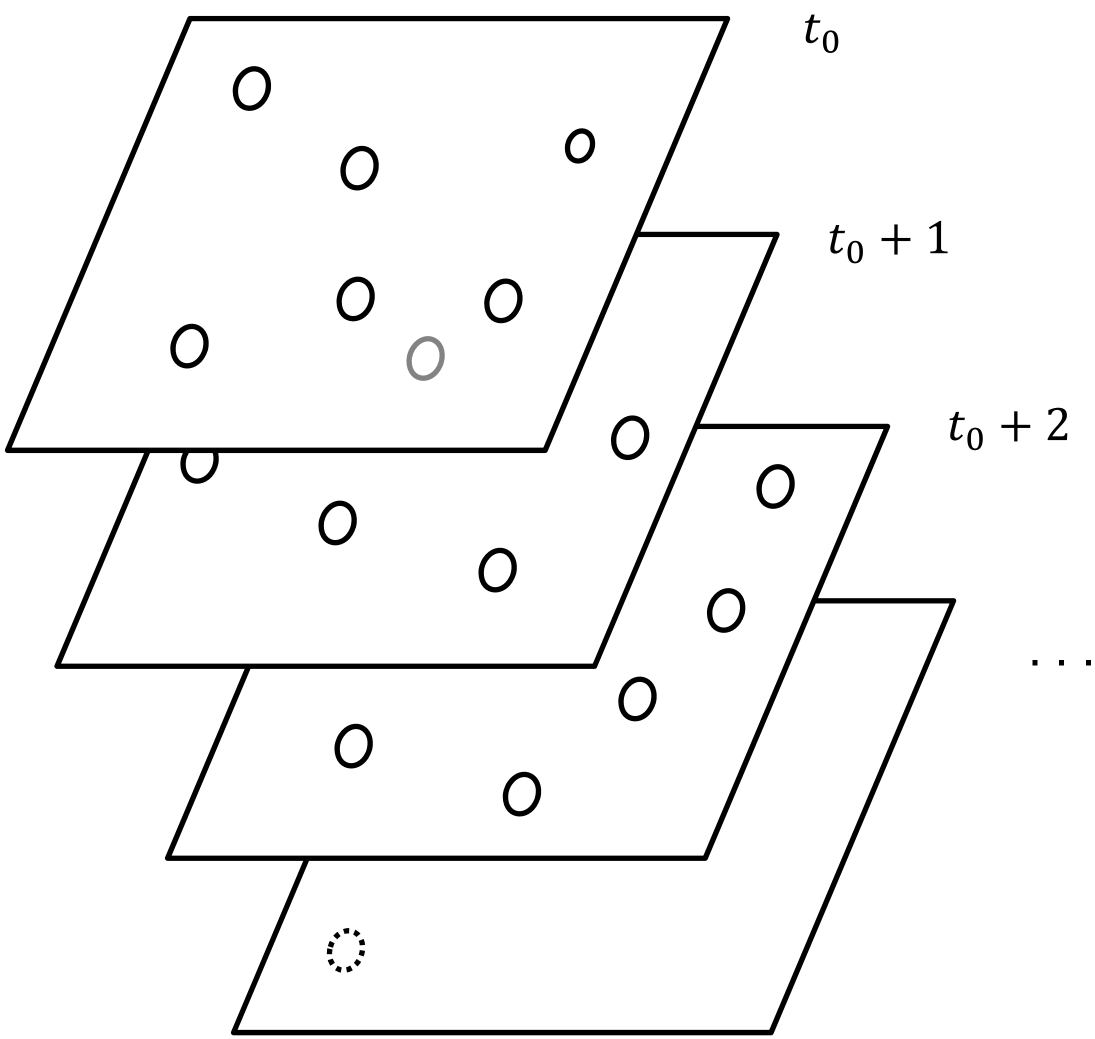
Infer Tracklets

4. Discover tracklets from MPE of networks
5. Combine and prune tracklets within window
6. Combine tracklets from current window with previous windows

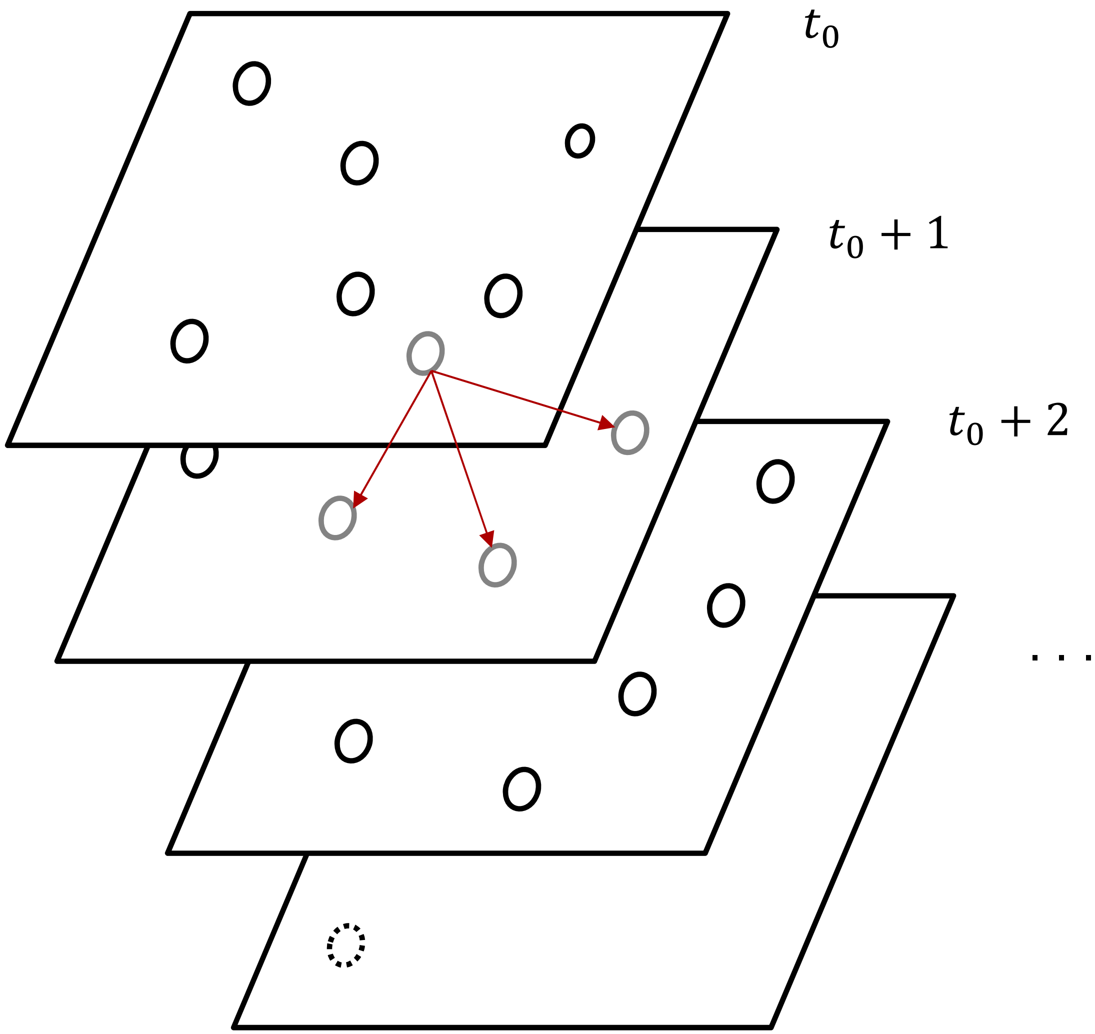
Construct a Bayes Network rooted at each detection in the first frame of window



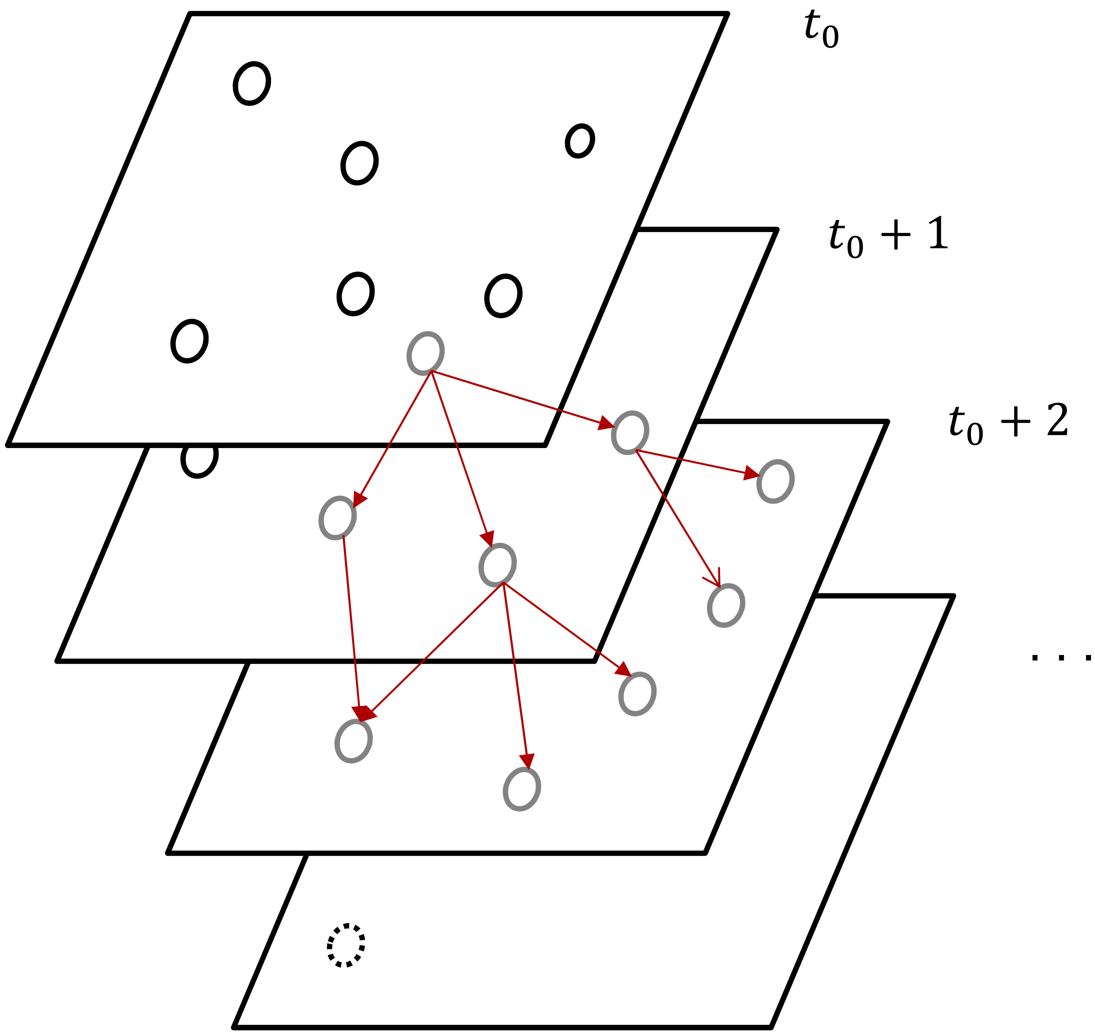
Construct a Bayes Network rooted at each detection in the first frame of window



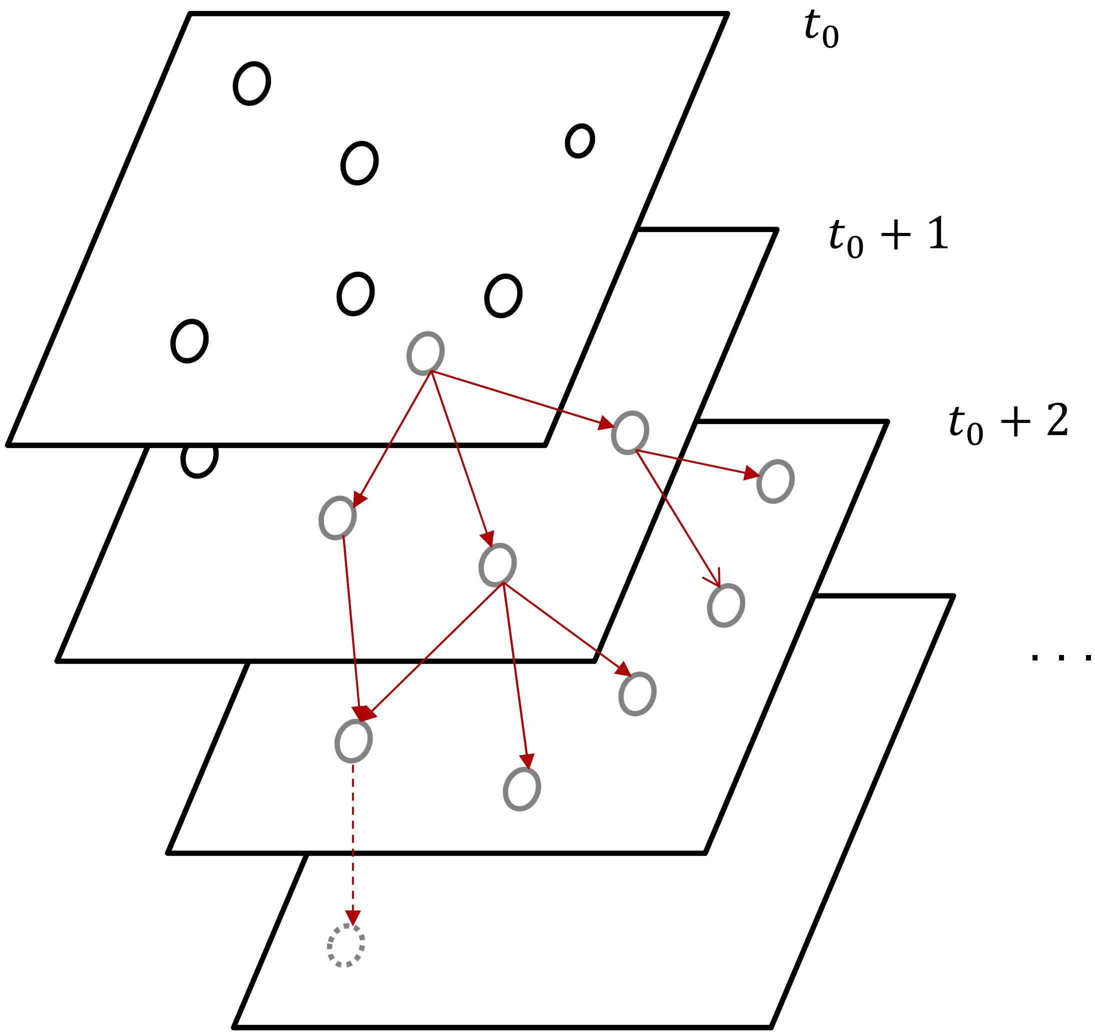
Construct a Bayes Network rooted at each detection in the first frame of window



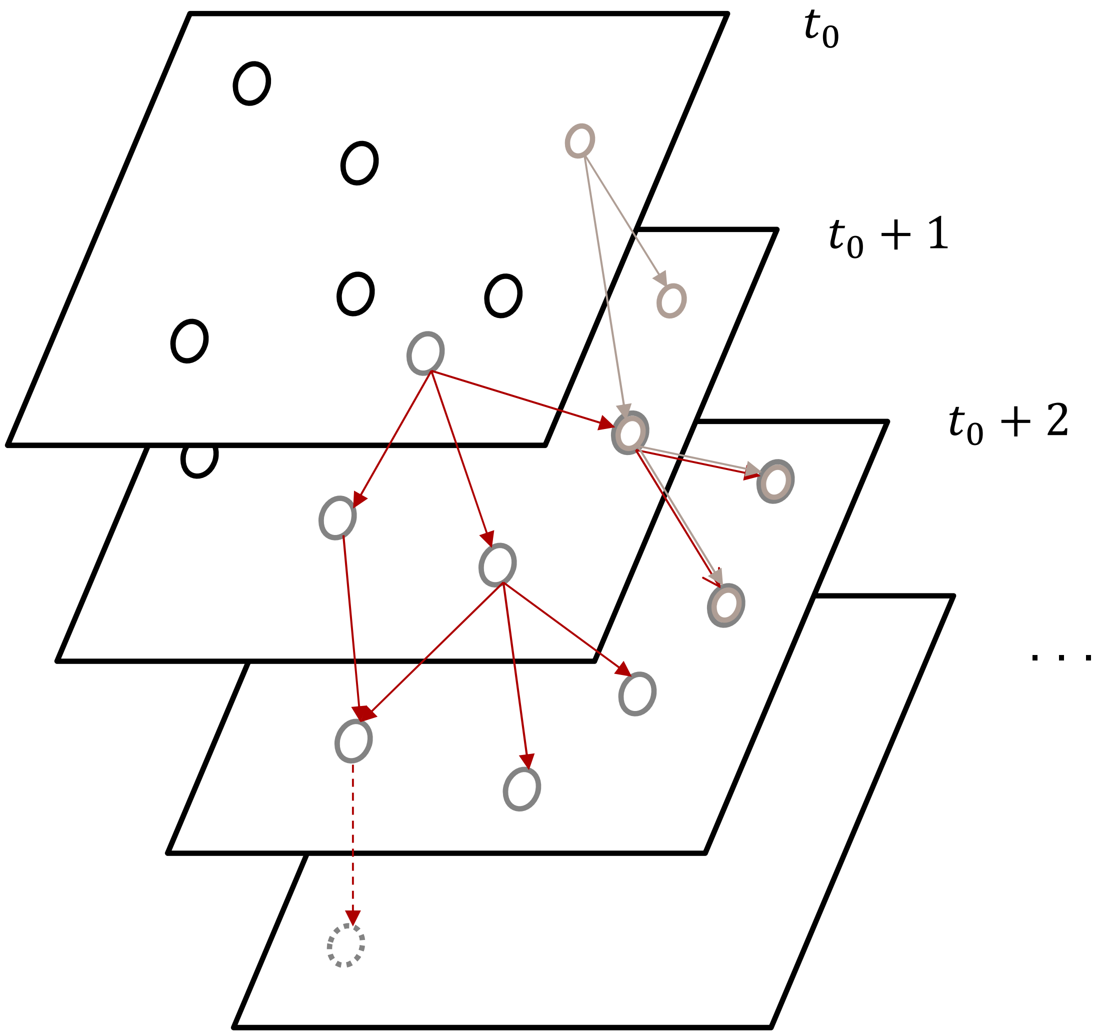
Construct a Bayes Network rooted at each detection in the first frame of window



Construct a Bayes Network rooted at each detection in the first frame of window



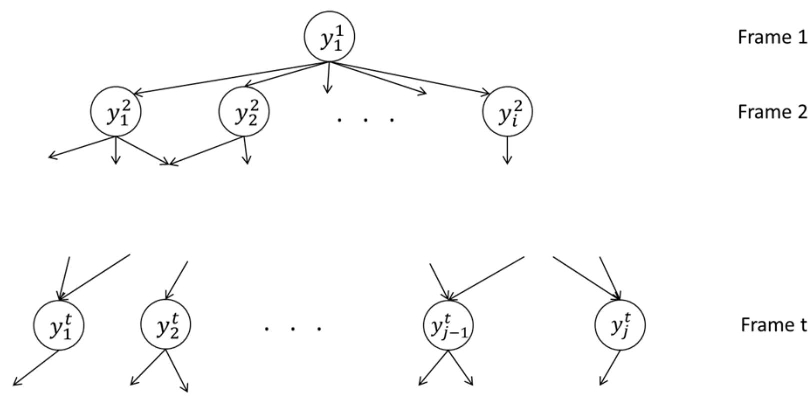
Construct a Bayes Network rooted at each detection in the first frame of window



Bayesian Network Formulation

- Bayesian network of binary variables, one for each detection
  - True = detection is valid (associated with root of network)
  - False = detection is invalid (not associated with root of network)
  - Edge probability tables based on appearance and motion dynamics

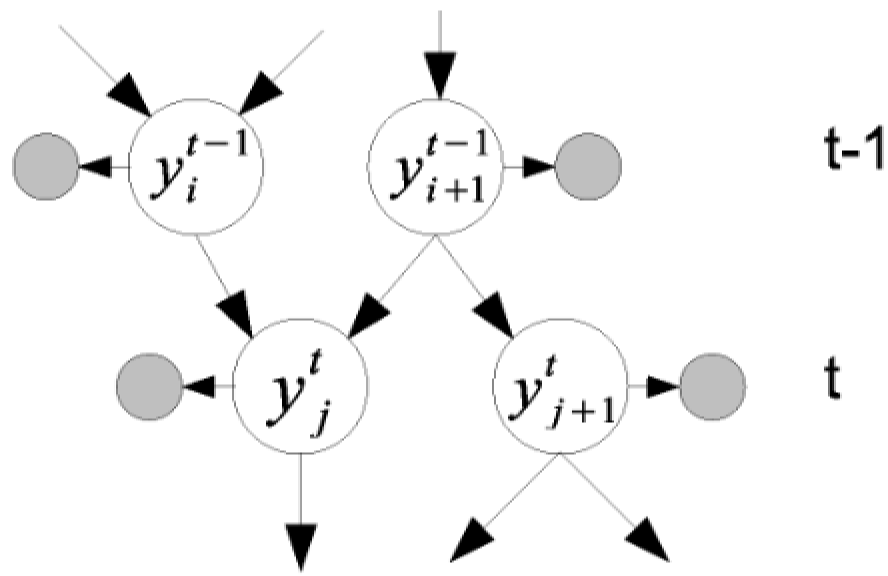
MPE (Most Probable Explanation) estimate finds the most likely state of all variables



Bayesian Network Edge Probabilities

Transition CPT

	$y_i^t = 0$	$y_i^t = 1$
$y_j^{t-1} = 0$	0.5	0.5
$y_j^{t-1} = 1$	$1 - a(\mathbf{o}_i^t, \mathbf{o}_j^{t-1})m(\mathbf{o}_i^t)$	$a(\mathbf{o}_i^t, \mathbf{o}_j^{t-1})m(\mathbf{o}_i^t)$



Bayesian Network Edge Probabilities

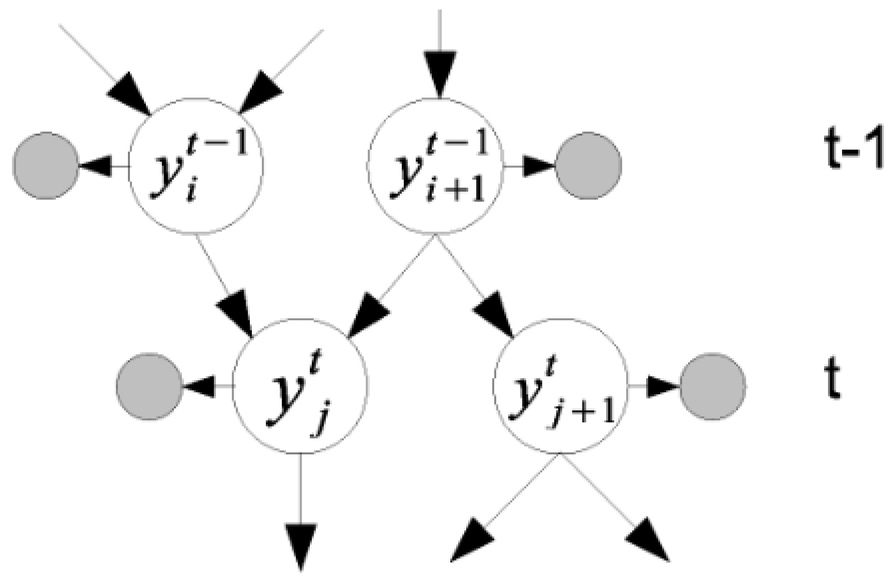
Sandia National Laboratories

Transition CPT

	$y_i^t = 0$	$y_i^t = 1$
$y_j^{t-1} = 0$	0.5	0.5
$y_j^{t-1} = 1$	$1 - a(\mathbf{o}_i^t, \mathbf{o}_j^{t-1})m(\mathbf{o}_i^t)$	$a(\mathbf{o}_i^t, \mathbf{o}_j^{t-1})m(\mathbf{o}_i^t)$

Appearance Similarity

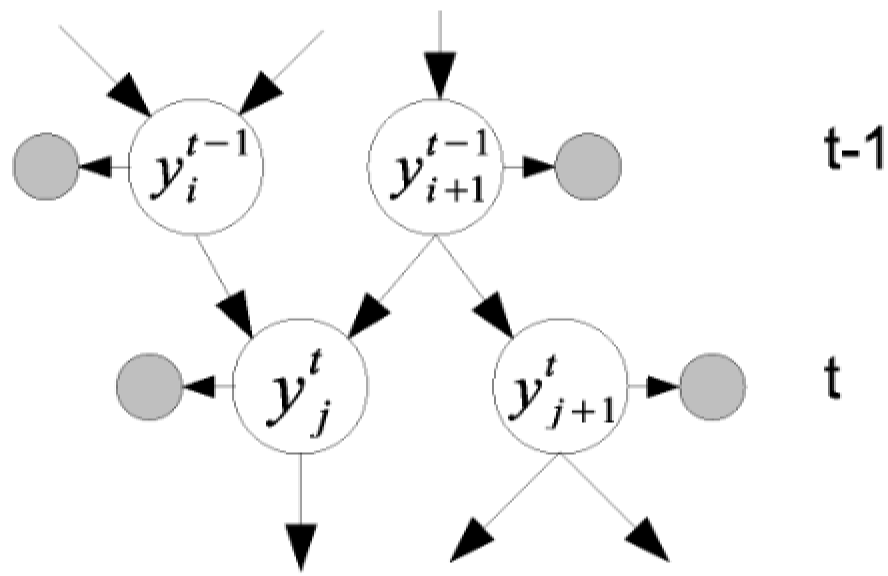
Motion Similarity



Bayesian Network Edge Probabilities

Transition CPT

	$y_i^t = 0$	$y_i^t = 1$
$y_j^{t-1} = 0$	0.5	0.5
$y_j^{t-1} = 1$	$1 - a(\mathbf{o}_i^t, \mathbf{o}_j^{t-1})m(\mathbf{o}_i^t)$	$a(\mathbf{o}_i^t, \mathbf{o}_j^{t-1})m(\mathbf{o}_i^t)$



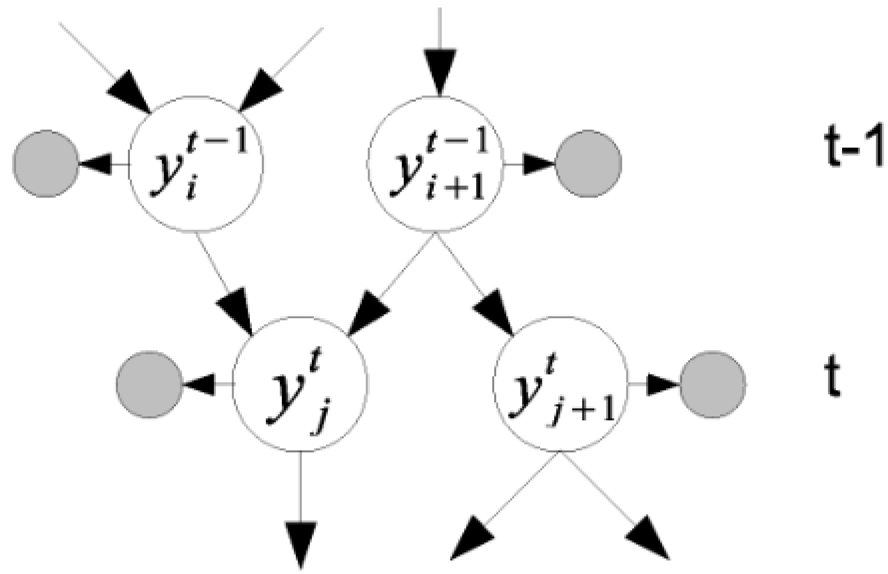
Observation CPT

	$y_i^t = 0$	$y_i^t = 1$
$p(\mathbf{o}_i^t \mid y_i^t)$	$1 - a(\mathbf{o}_i^t, \mathbf{o}^0)$	$a(\mathbf{o}_i^t, \mathbf{o}^0)$

Bayesian Network Edge Probabilities

Transition CPT

	$y_i^t = 0$	$y_i^t = 1$
$y_j^{t-1} = 0$	0.5	0.5
$y_j^{t-1} = 1$	$1 - a(\mathbf{o}_i^t, \mathbf{o}_j^{t-1})m(\mathbf{o}_i^t)$	$a(\mathbf{o}_i^t, \mathbf{o}_j^{t-1})m(\mathbf{o}_i^t)$

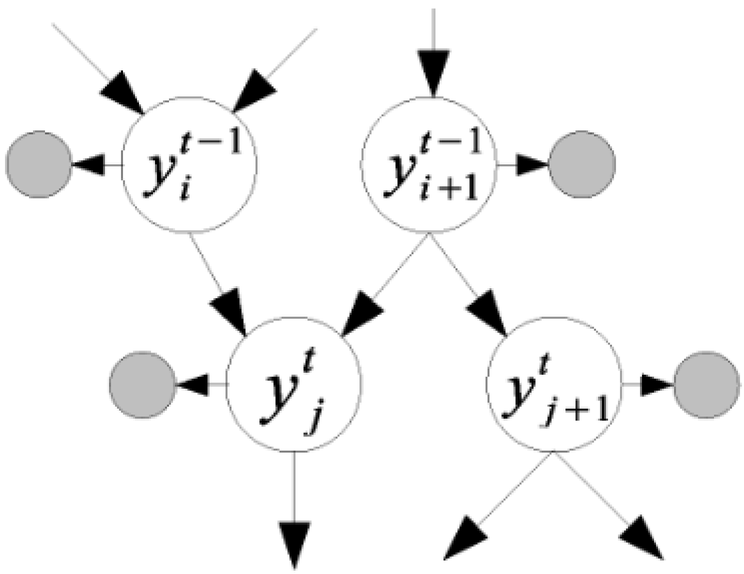


PROBLEM: Size of transition CPT is exponential in number of parents.

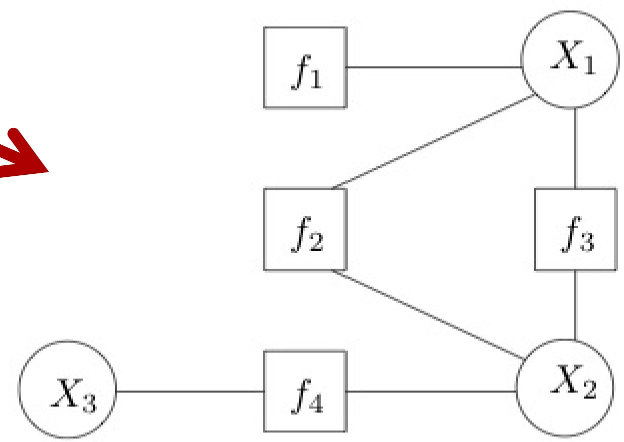
SOLUTION: Make simplifying assumption:

$$p(y_i^t | y_1^{t-1}, y_2^{t-1} \dots y_K^{t-1}) = \prod_{k=1}^{K} p(y_i^t | y_k^{t-1})$$

Bayesian Network → Factor Graph



$$p(y_i^t | y_1^{t-1}, y_2^{t-1} \dots y_K^{t-1}) = \prod_{k=1}^{K} p(y_i^t | y_k^{t-1})$$

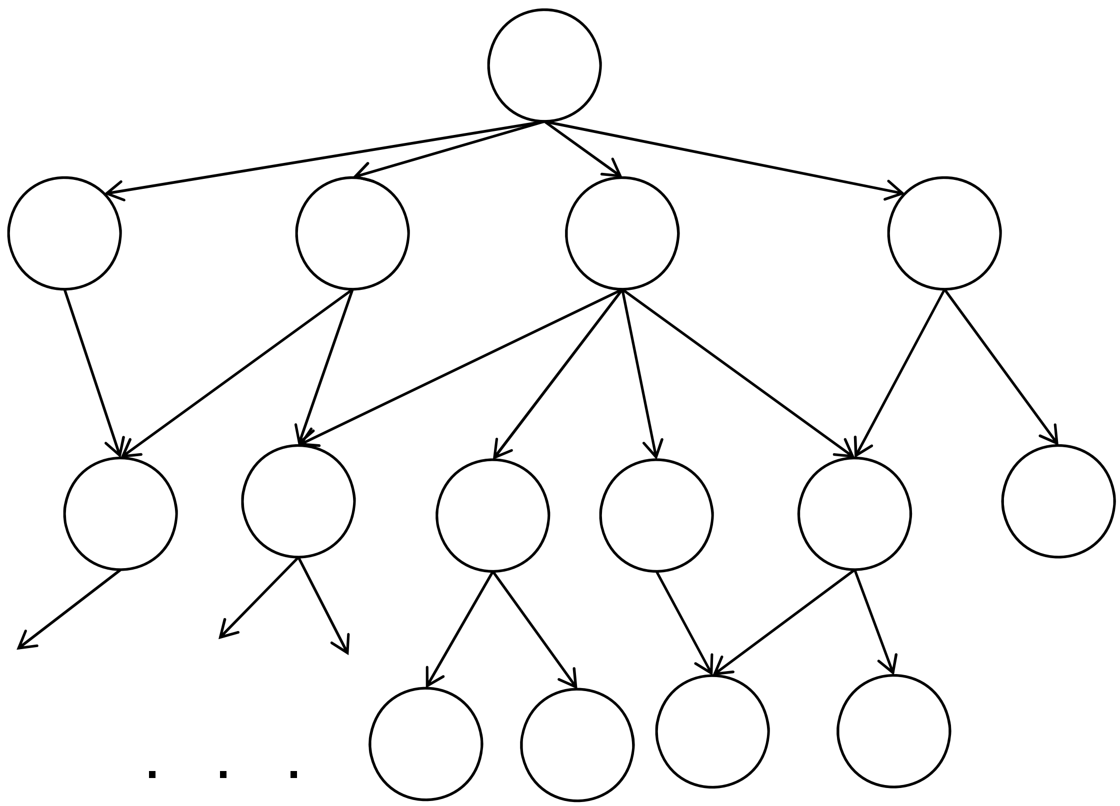


Now the detection network becomes a **Factor Graph**

Can solve MPE with *Max-Product* message passing algorithm

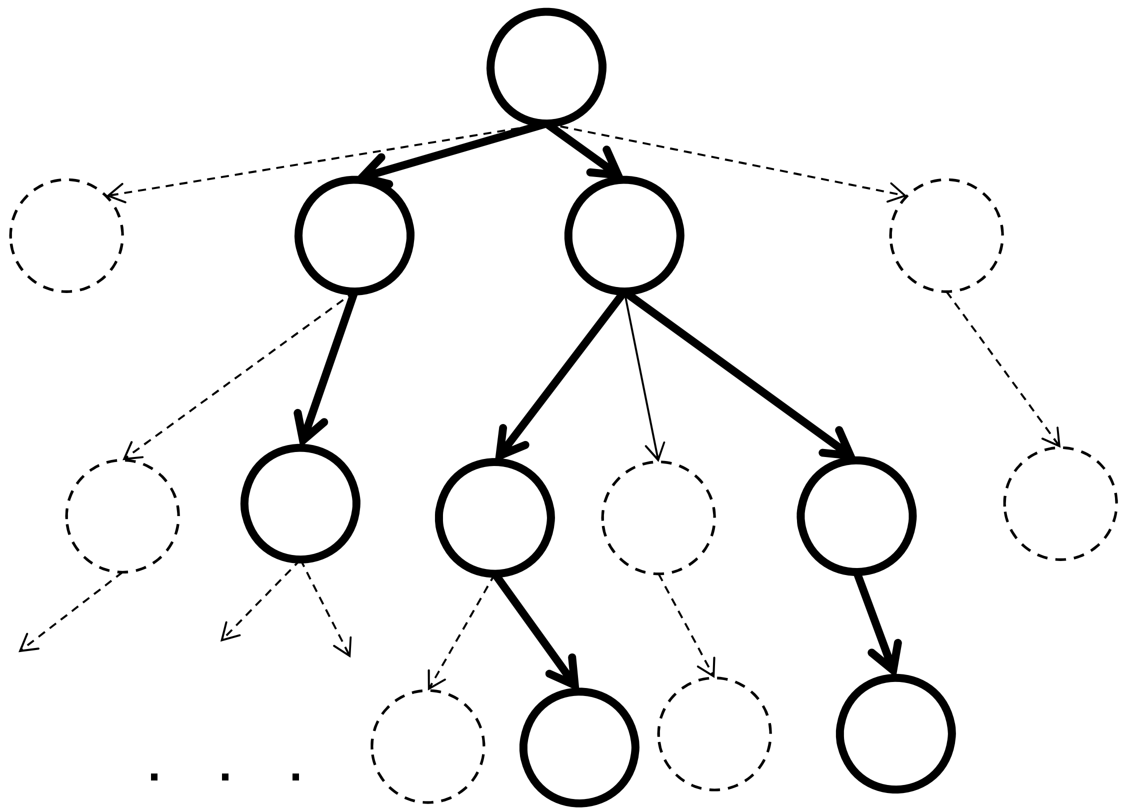
Inferring Tracklets from Factor Graph

Factor Graph



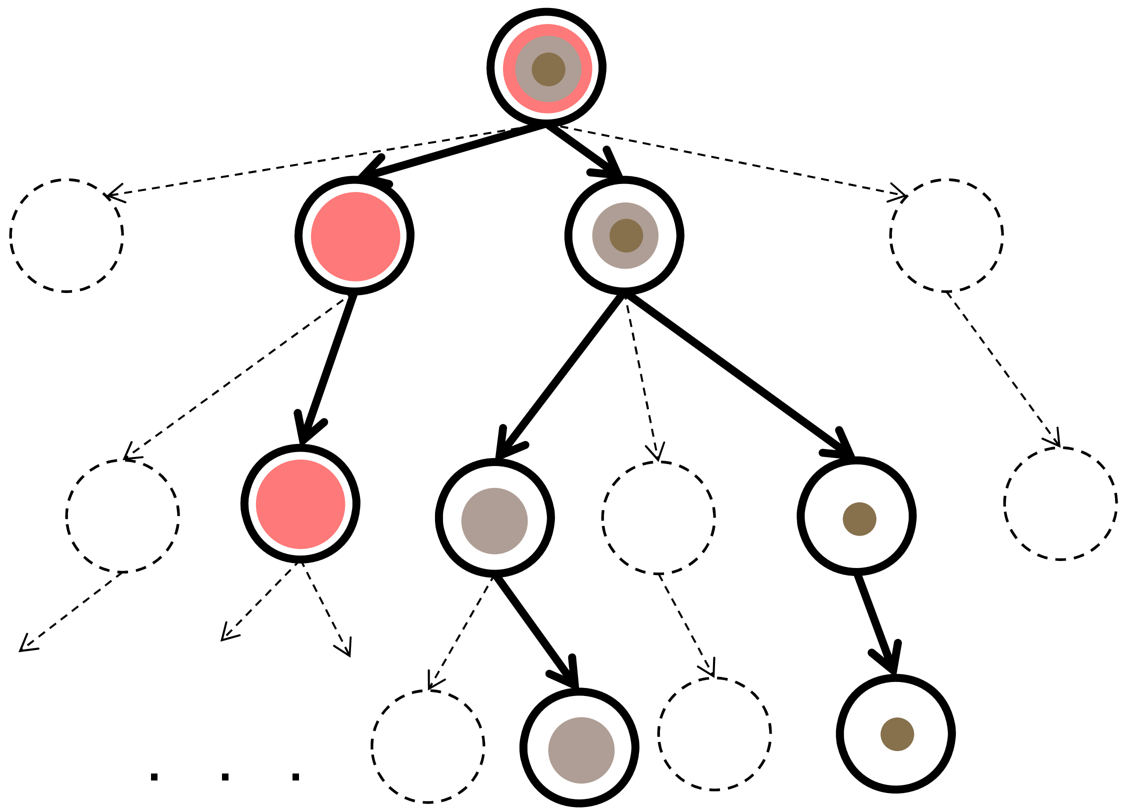
Inferring Tracklets from Factor Graph

MPE Result



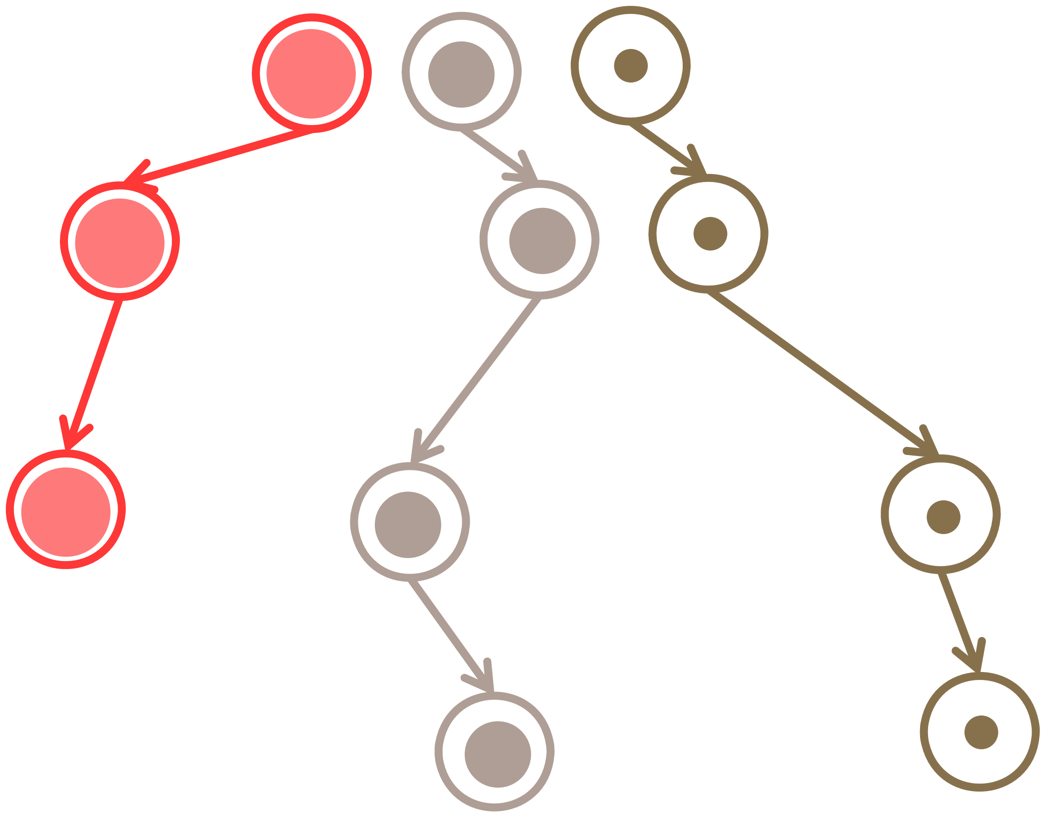
Inferring Tracklets from Factor Graph

Tracklet Discovery



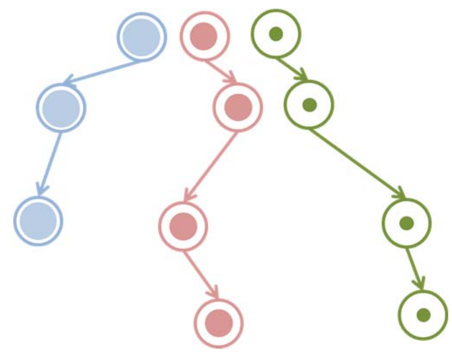
Inferring Tracklets from Factor Graph

Tracklet Result



Prune and Combine Tracklets

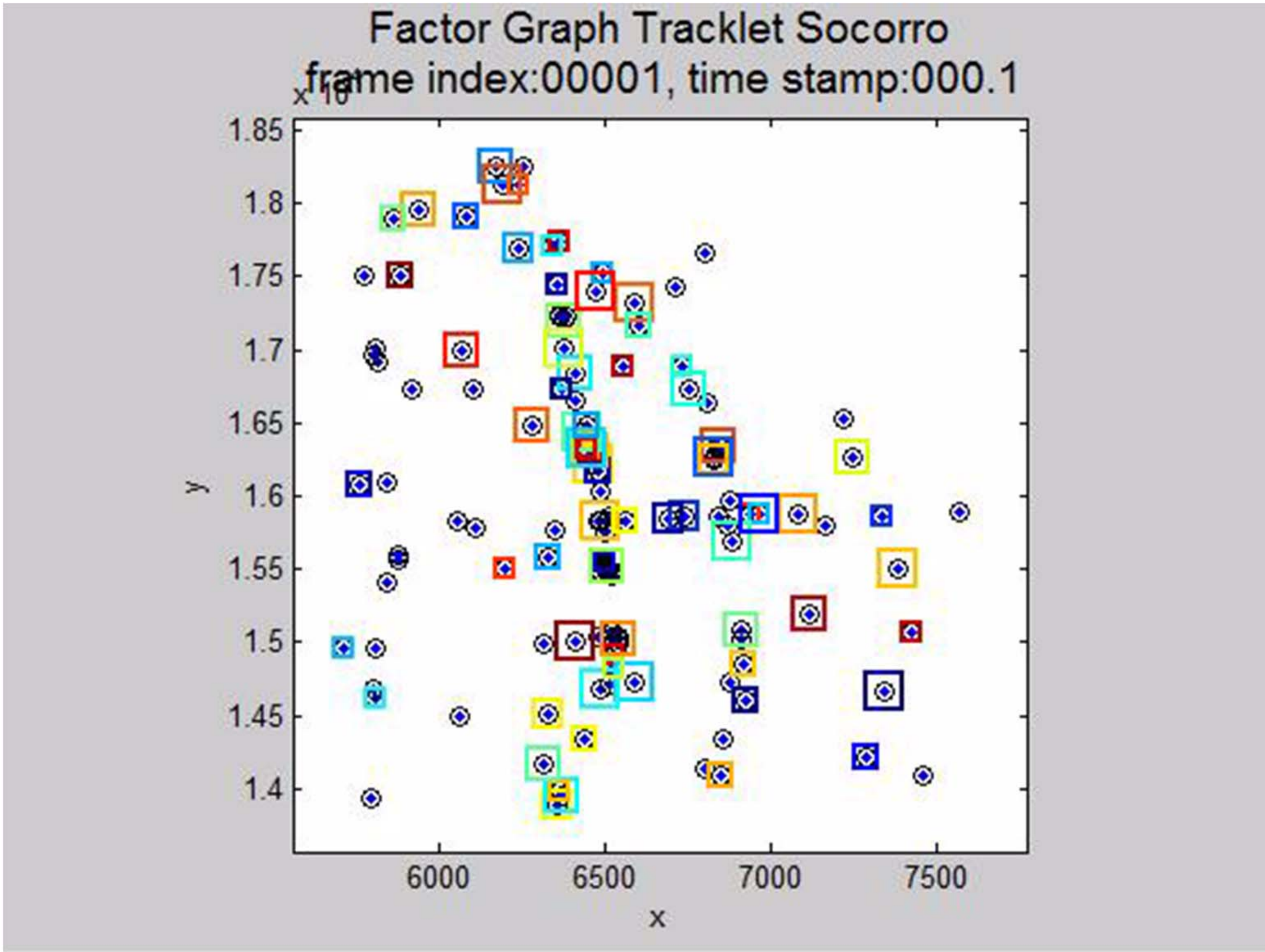
- Delete any tracklets that don't satisfy pruning conditions
  - Minimum length
  - Minimum smoothness
  - Maximum acceleration



- Combine current tracklets with tracklets from previous window

Results

Socorro Small ROI



mota = 0.6275
mt = 0.4191
ml = 0.0083
mst = 0.1120
msl = 0.2739
gt = 241
total_fp = 0
total_fn = 17746
total_ids = 8314

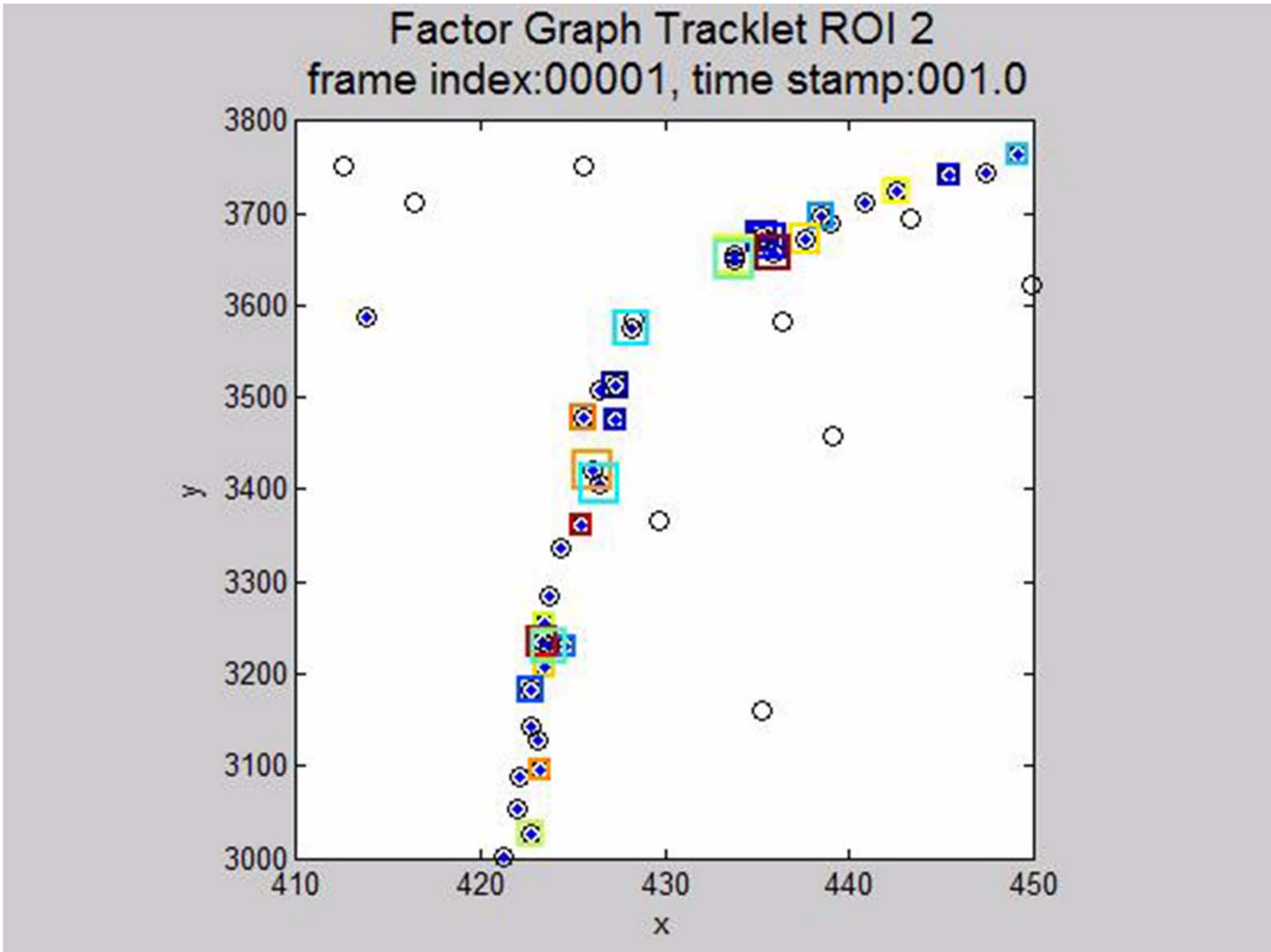
Results

Socorro Full Dataset

mota = 0.6200
mt = 0.3188
ml = 0.0013
mst = 0.0282
msl = 0.7106
gt = 781
total_fp = 0
total_fn = 442151
total_ids = 123769

Results

AFRL WPAFB Highway ROI



mota = 0.7041
mt = 0.6118
ml = 0.0824
mst = 0.5529
msl = 0.1059
gt = 85
total_fp = 23
total_fn = 570
total_ids = 182

Algorithm Pros and Cons

- **Pros**
  - Incorporates both appearance and motion in same framework
    - Current results only utilize motion
  - Elegantly handles merged detections
  - Very parallelizable
- **Cons**
  - Solving factor graph MPE problem is not straightforward
  - Requires tuning of pruning parameters (length, smoothness, and acceleration thresholds) and motion parameters
- **Future**
  - Incorporate appearance
  - Combination with regression tracker to maintain track through slowdowns and stops (CVPR ‘14)

Summary – Relative Performance

- Relative performance on Socorro data set:

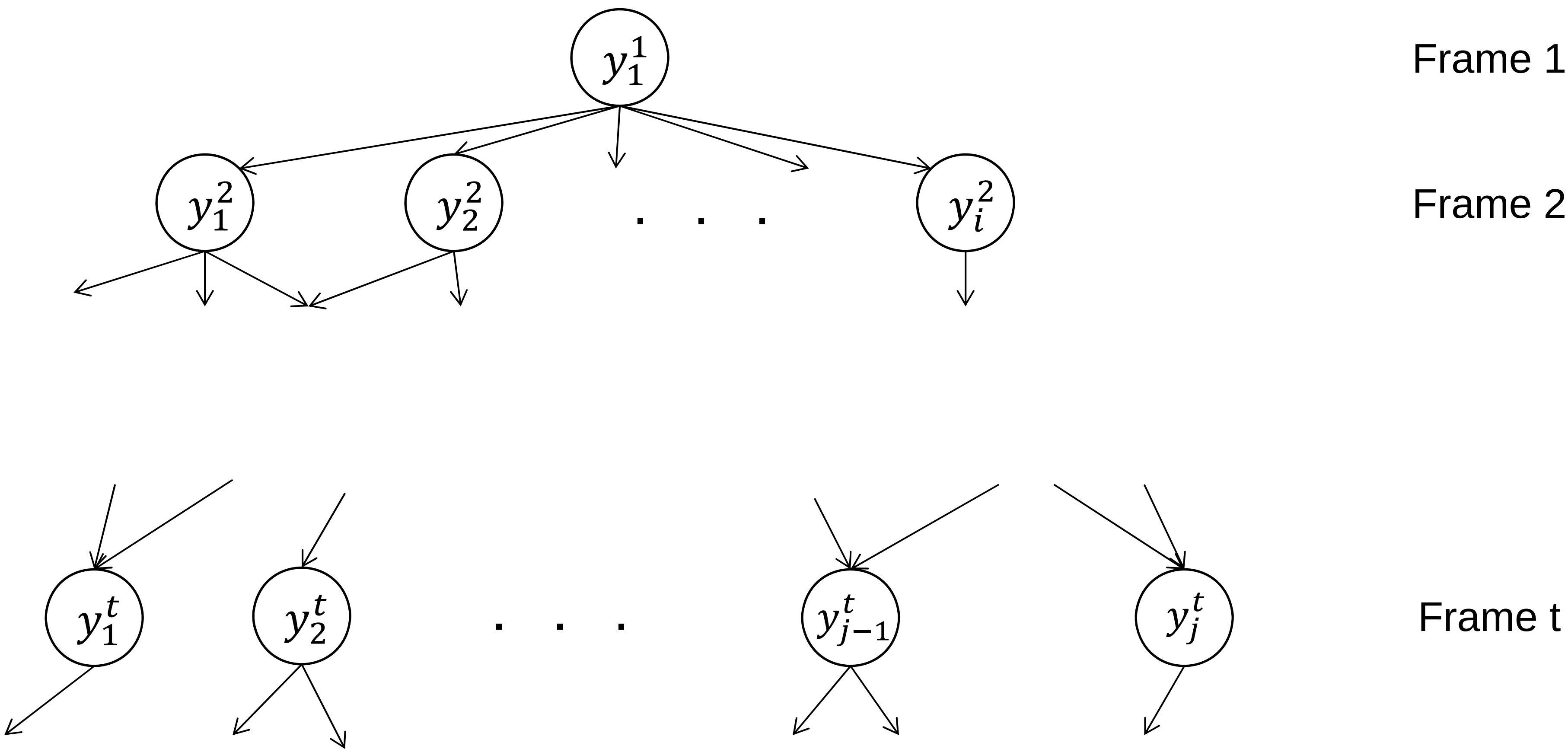
PROXIMITY TRACKER
mota = 0.7308
mt = 0.9117
ml = 0.0128
mst = 0.2394
msl = 0.1575
gt = 781
total_fp=0
total_fn=96598
total_ids=304338 (High value)

TRACKLETS FROM FACTOR GRAPHS
mota = 0.6200
mt = 0.3188
ml = 0.0013
mst = 0.0282
msl = 0.7106
gt = 781
total_fp = 0
total_fn = 442151
total_ids = 123769

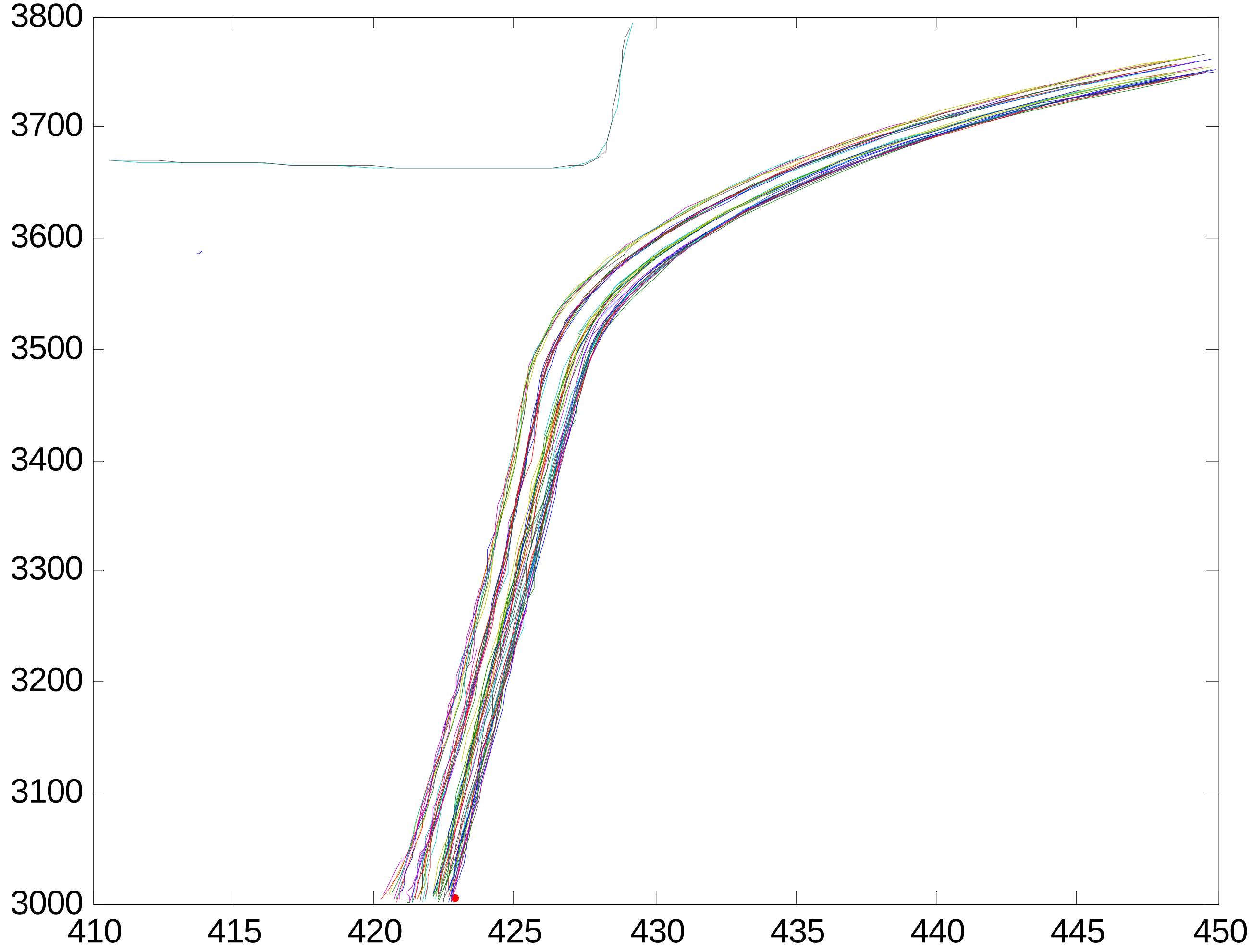
RANSAC
mota = TBD
mt = TBD
ml = TBD
mst = TBD
msl = TBD
gt = TBD
total_fp=TBD
total_fn=TBD
total_ids=TBD

Backup

Construct a Bayes Network rooted at each detection in the first frame of window



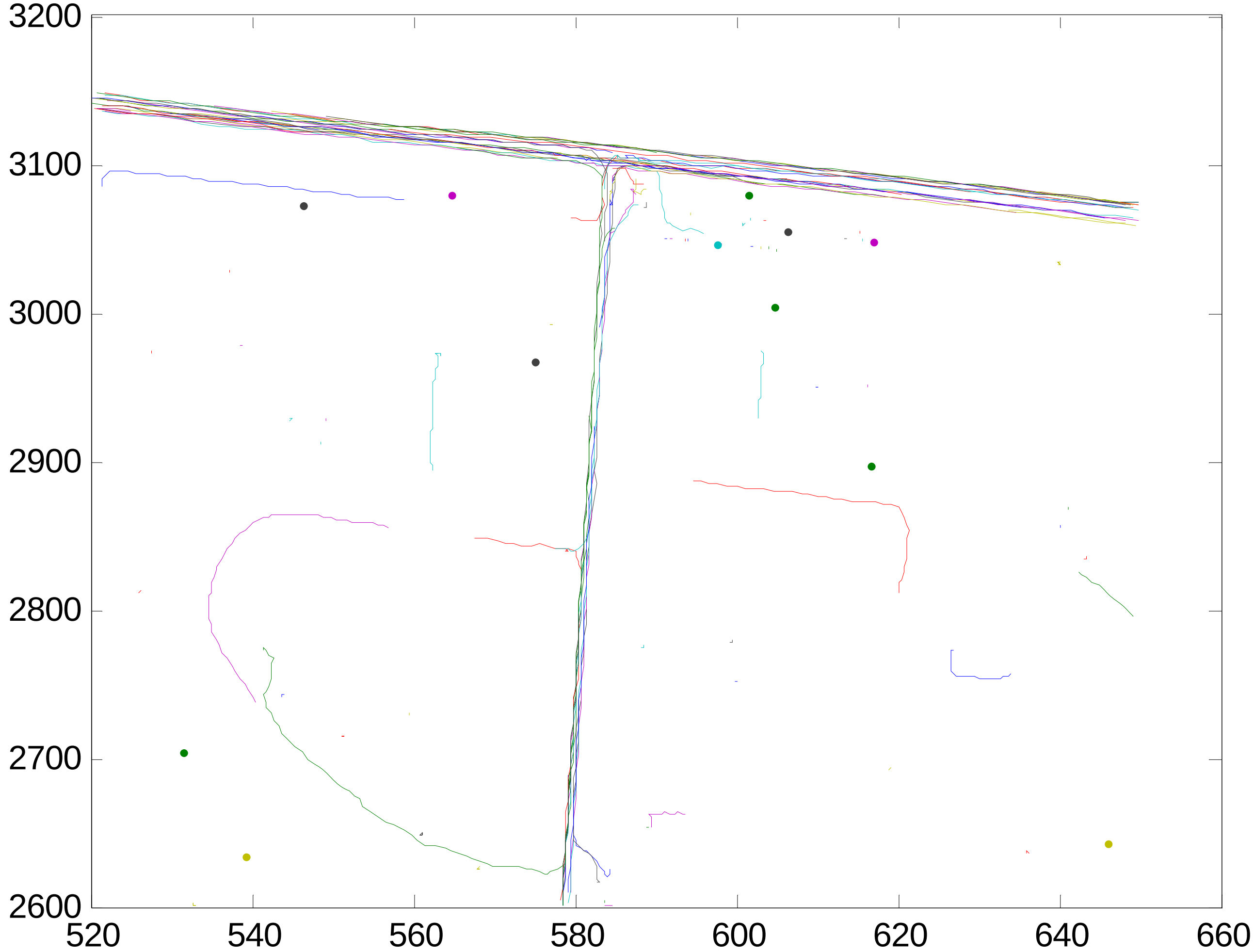
ROI 2 (highway driving with turn)



85 vehicles

ROI 2 (highway driving with turn)

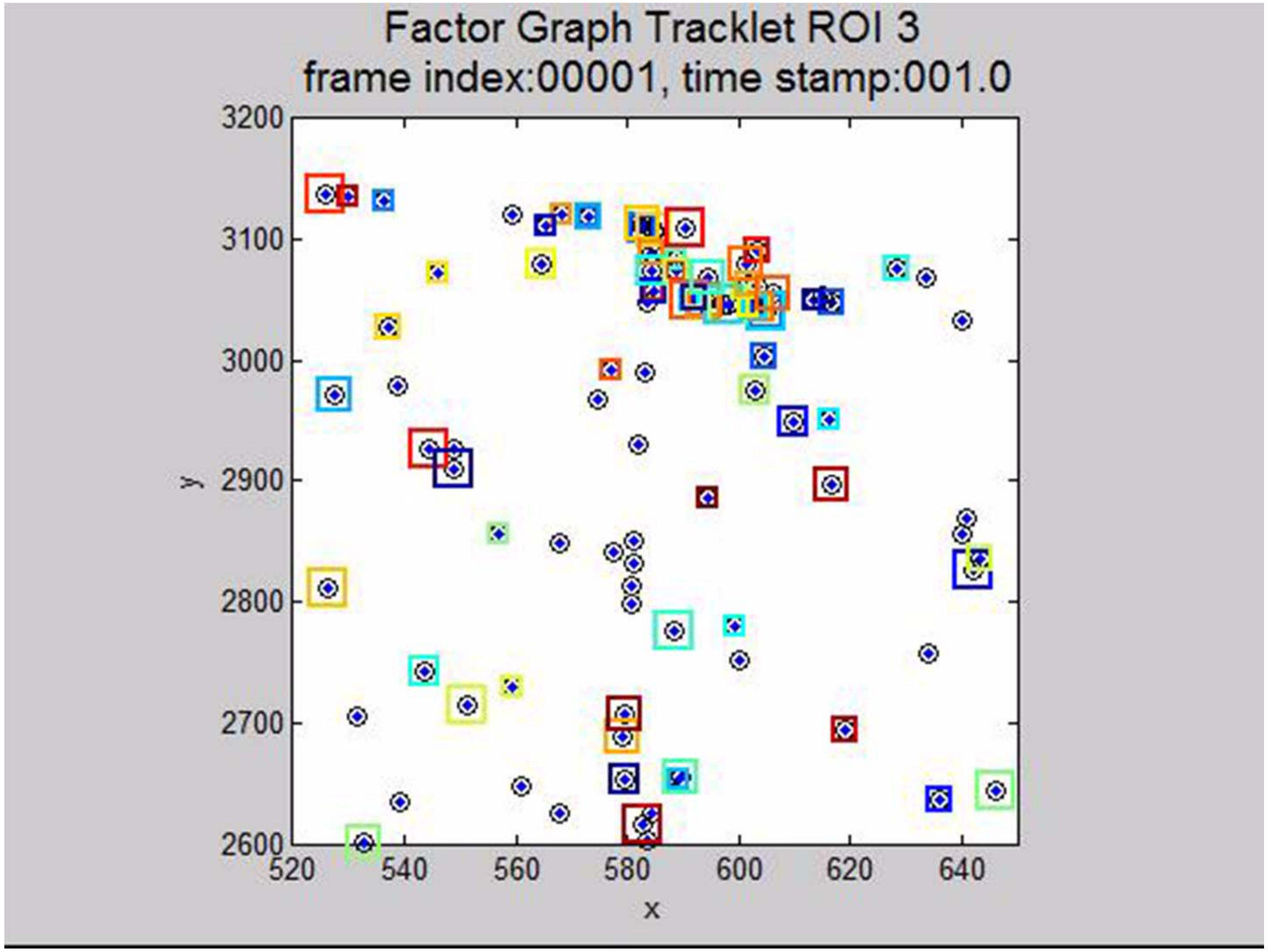
ROI 3 (intersection)



125 Vehicles

ROI 3 (intersection)

ROI 3 Results



mota = 0.5531
mt = 0.3520
ml = 0.1920
mst = 0.2800
msl = 0.3200
gt = 125
total_fp = 0
total_fn = 2692
total_ids = 298

High Level Algorithm over Sliding Window

In each window:

1. Construct Bayesian networks of detections and find MAP estimates

2. Infer tracklets from MAP estimates