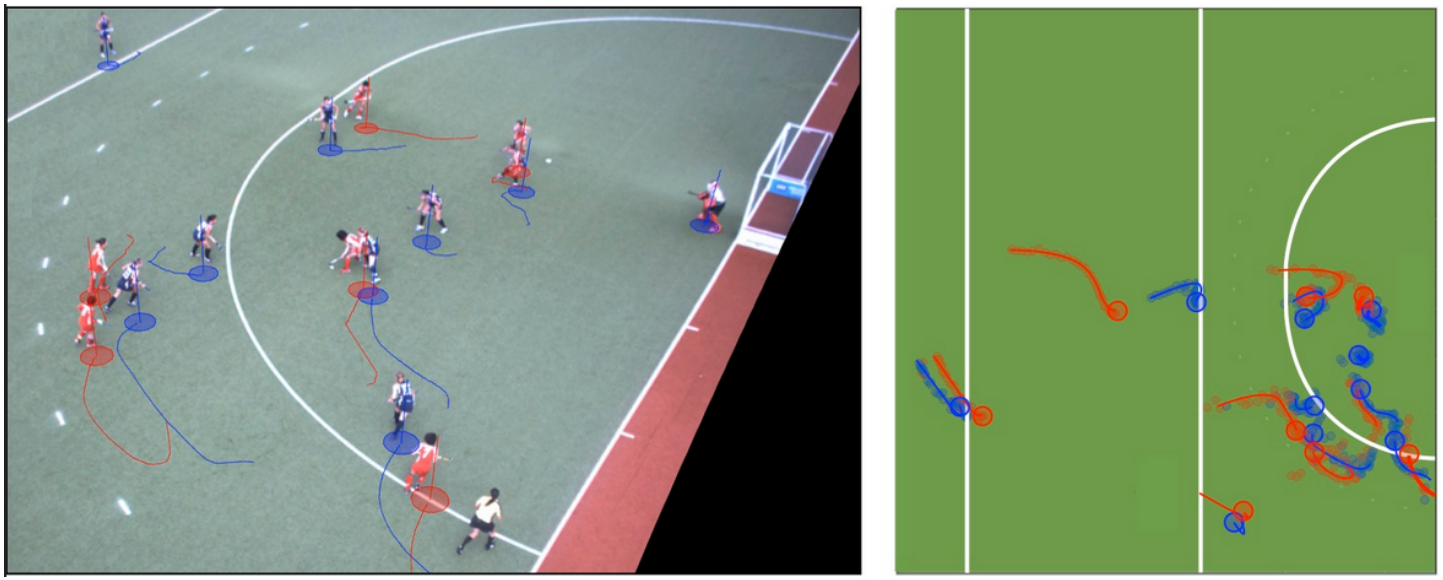
CSC2541 - Sport Analytics

Tracking

Davi Frossard



Credit: *http://bit.ly/2lWbqI6*

February 26, 2017

Summary

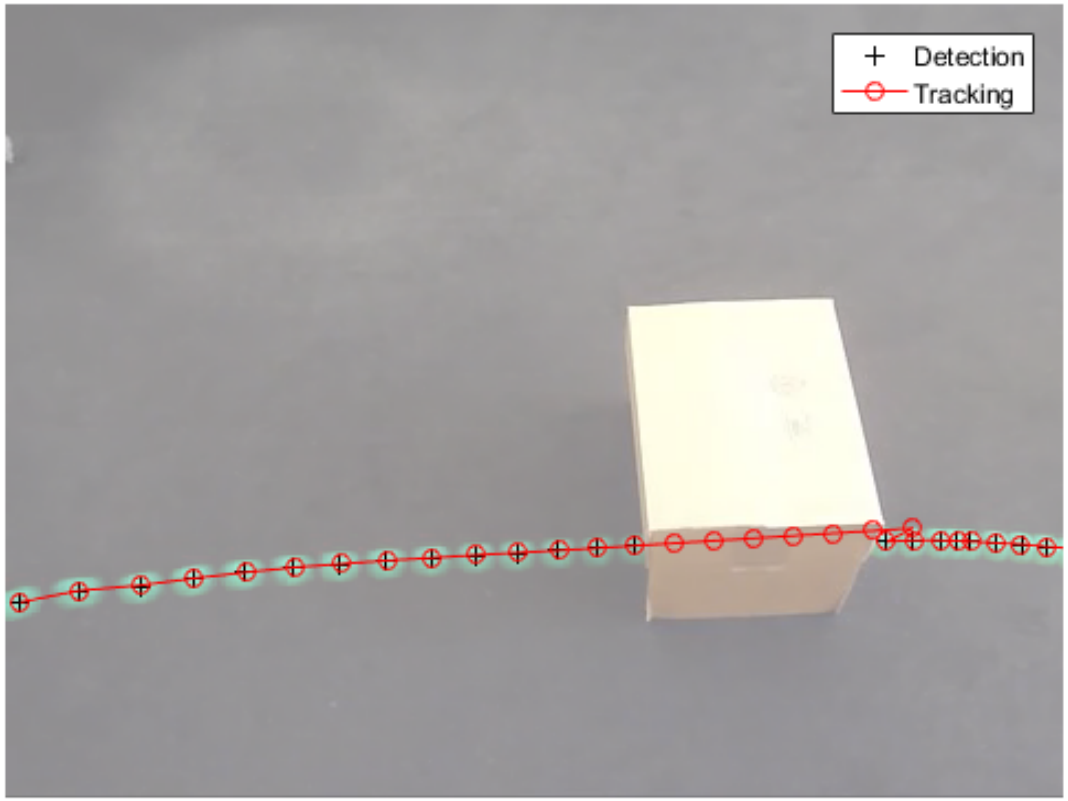
TRACKING

- Tracking is the process of locating an object (or objects) given a sequence of observations.
  - Players during a match of basketball;
  - Cars on a highway;
  - Drone following target.
- Different approaches are used for **Single Object Tracking** and **Multiple Object Tracking**.

SINGLE OBJECT TRACKING

- Simplification of the general problem when we're concerned with just one target.
- Approaches are usually variations of a Hidden Markov Model:
  - Kalman Filter;
  - KLT Feature Tracker;
  - Mean-Shift Algorithm;
  - ...
- Real-time performance and doesn't require much data (if at all) to train.

SINGLE OBJECT TRACKING

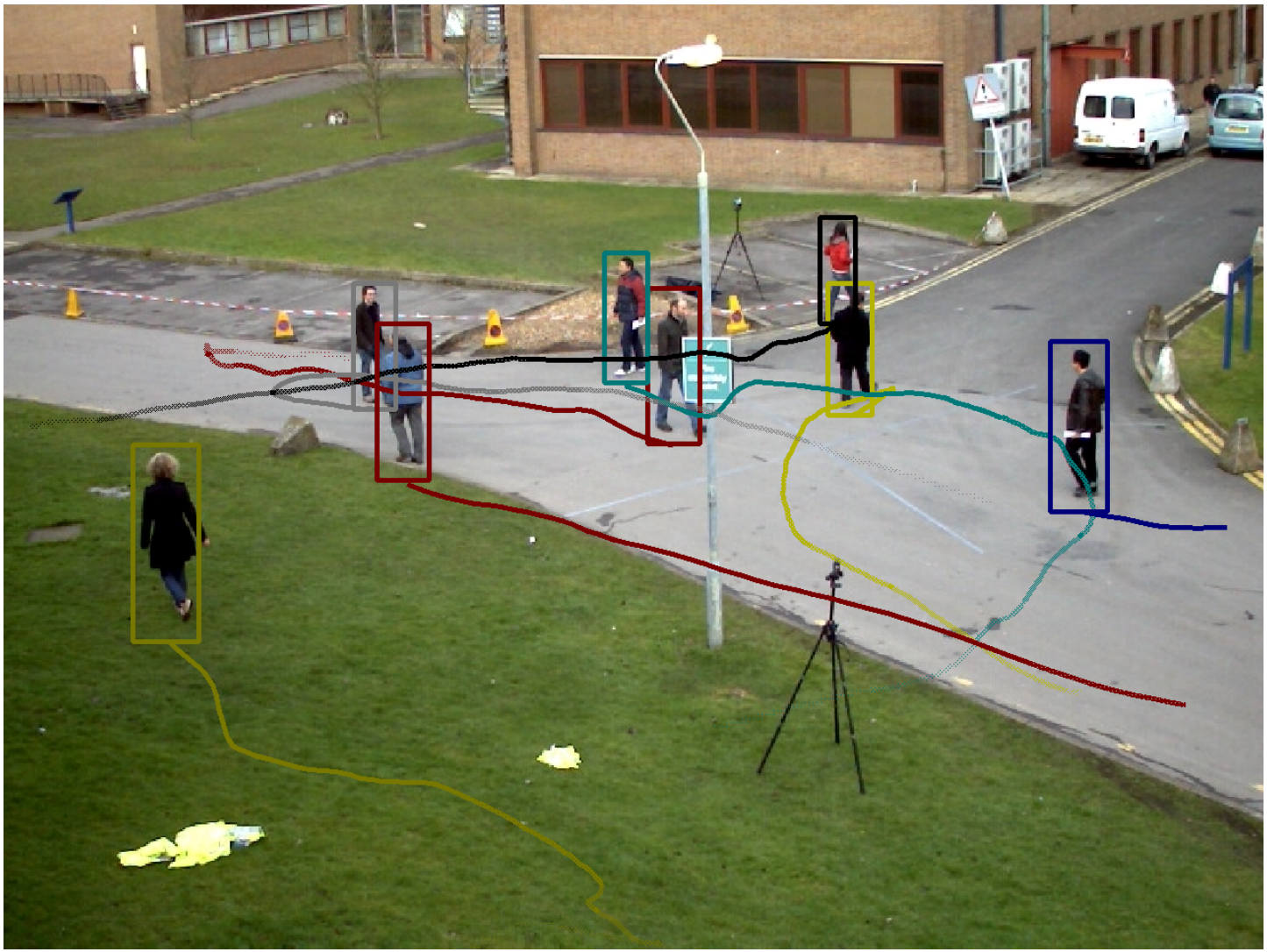


Credit: *http://bit.ly/2mpVrQ7*

MULTIPLE OBJECT TRACKING

- For the general case of tracking multiple objects we must first decide how many of them we have, and then track. So for each observation we have two stages:
  1. Detection (covered last week)
  2. Association
- Additionally, we may want to add a smoothing stage to filter out noise created by the detection phase.

MULTIPLE OBJECT TRACKING



Credit: Breitenstein et al., Online Multi-Person Tracking-by-Detection from a Single, Uncalibrated Camera

CHALLENGES

- Variation in observation conditions:
  - Lighting;
  - Target pose;
- Target truncation;
- Target occlusion;
- Motion dynamics;
  - Mobile observation platform;
  - Fast moving targets;
  - Target interactions;
- ...

ASSOCIATION

- The association problem lies in, given a set of detections $\mathcal{X} = [x_0, x_1, ..., x_n]$, find a set of associations (trajectories) $\mathcal{T} = [T_1, T_2, ..., T_k]$ where $T_z = [x_0^z, x_1^z, ..., x_m^z]$ such that $P(\mathcal{T}|\mathcal{X})$ is maximal.
- Because each trajectory should contain only a single object, we also want to ensure non-overlap between trajectories, or $T_k \cap T_l = \emptyset, \forall k \neq l$.
- We can formulate this as a Maximum a Posteriori (MAP) optimization.

ASSOCIATION AS A MAP

$$\underset{\mathcal{T}}{\text{argmax}}\ P(\mathcal{T}|\mathcal{X}) = \prod_{T_i \in \mathcal{T}} P(T_i) \prod_{x_j \in \mathcal{X}} P(x_j|\mathcal{T})$$

- Here we have a prior over the trajectories $P(T_i)$ and an observation model $P(x_j|\mathcal{T})$.

ASSOCIATION AS A MAP

- Before we further define our objective function, we must first decide what aspects we consider important for tracking:
  - Prefer high scoring detections to minimize false positives;
  - Minimize fragmentation (prefer longer trajectories);
  - Minimize ID switches;

OBSERVATION MODEL

- To minimize false positives we can define our observation model given a detection score β_j as follows:

$$P(x_i|\mathcal{T}) = \begin{cases} \beta_j & \exists T_k \in \mathcal{T}, x_j \in T_k \\ 1 - \beta_j & otherwise \end{cases}$$

- If we introduce a discrete random variable y_j to encode true positives we can rewrite this as:

$$P(x_i|\mathcal{T}) = y_j\beta_j + (1 - y_j)(1 - \beta_j)$$

TRAJECTORY PRIOR

- To incorporate the characteristics we want in our trajectories. We propose a parametrization in terms of 3 additional discrete random variables.
  1. $\mathbf{y}_{new_i}$ encoding whether detection i is the beginning of a new trajectory;
  2. $\mathbf{y}_{end_i}$ encoding whether detection i is the ending of a trajectory;
  3. $\mathbf{y}_{link_{i,j}}$ encoding whether detections i and j are the same object.

TRAJECTORY PRIOR

- We then formulate our prior as:

$$P(T_z) = P(y_{new_{x_0^z}}) \prod_{j=0}^{k-1} P(y_{link_{x_j^z, x_{j+1}^z}}) P(y_{end_{x_k^z}})$$

TRAJECTORY POSTERIOR

- Putting it all together we get:

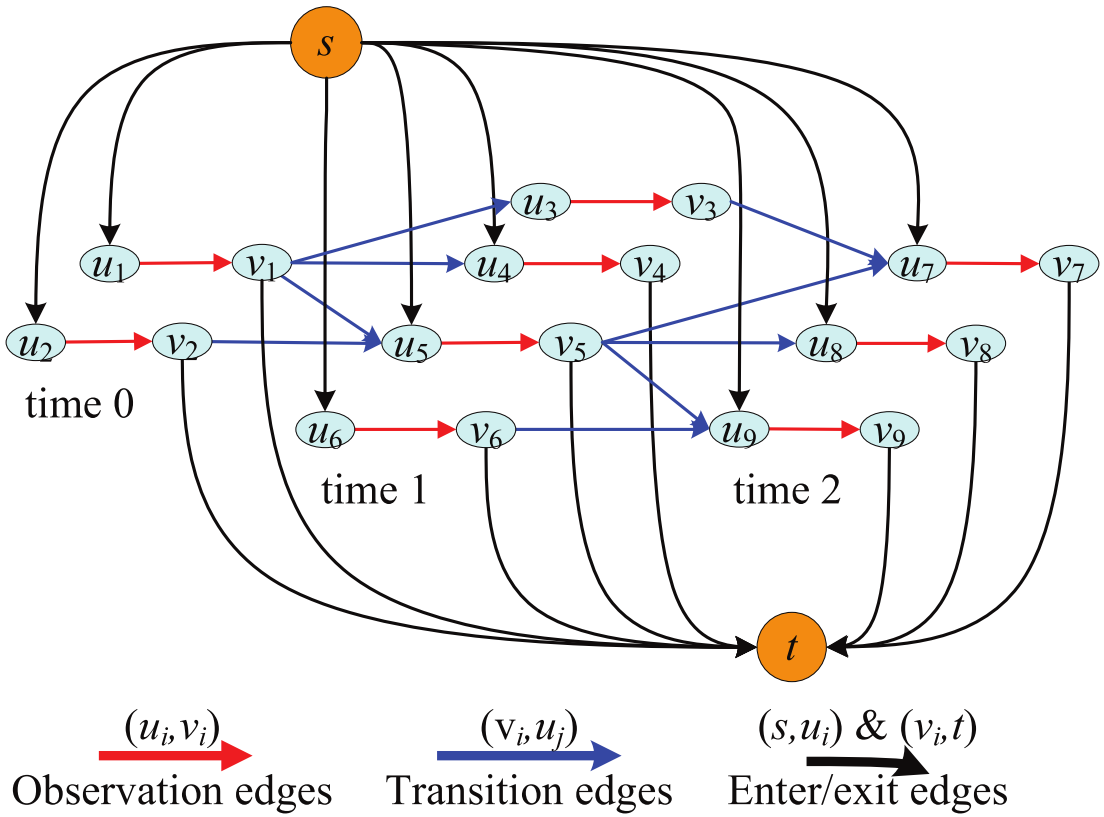
$$P(\mathcal{T}|\mathcal{X}) = \prod_{T_z \in \mathcal{T}} \left(P(y_{new_{x_0^z}}) \prod_{j=0}^{k-1} P(y_{link_{x_j^z, x_{j+1}^z}}) P(y_{end_{x_k^z}}) \right) * \prod_{x_j \in \mathcal{X}} y_j \beta_j + (1 - y_j)(1 - \beta_j)$$

- To ensure our non-overlap constraint is respected we define:

$$y_{new_i} + \sum_j y_{link_{j,i}} = y_{end_i} + \sum_j y_{link_{i,j}} = y_i$$

FLOW MODEL

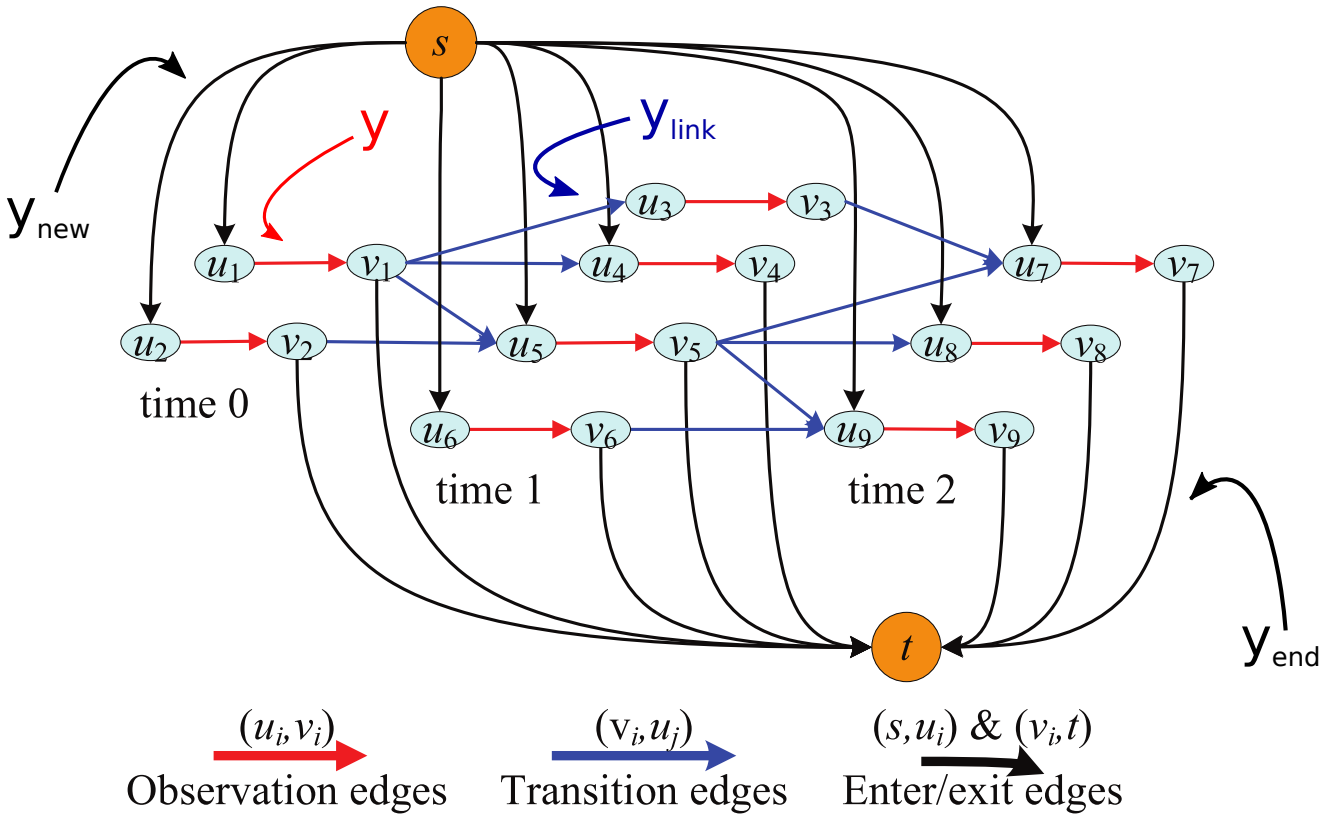
- We can express our posterior as a network flow model:



Credit: Nevatia et al, Global Data Association for Multi-Object Tracking

FLOW MODEL

- We can express our posterior as a network flow model:



Credit: Nevatia et al, Global Data Association for Multi-Object Tracking - Edited

FLOW MODEL

- This way we can solve our MAP as a Max-Flow problem.
  - Each flow path is a trajectory;
  - Flow conservation guarantees no trajectory overlap;
  - Amount of flow from s to t is the number of trajectories.
  - Total cost of flow is the negative log-likelihood of the trajectories' hipothesis
- Global optimality can be guaranteed.

SOLVING WITH MIN-COST FLOW

Input: Observation set $\mathcal{X}$

1 Build flow graph $\mathcal{G}(\mathcal{X})$
2 Start with flow $0, f(\mathcal{G}) = 0$
3 **while** $f(\mathcal{G}) = 0$ *can be augmented* **do**
4 Augment $f(\mathcal{G})$ by 1
5 Compute current cost with Min-Cost Flow
6 **if** *current cost* $<$ *optimal cost* **then**
7 Store current cost as optimal cost
8 **return** *global optimum*

Linear Program

- Our formulation can also be interpreted as an Integer Program: We want to find integer assignments to our random variables such that the negative log-likelihood is minimized.
- Integer Programming is NP-Hard. However we can relax our problem to a Linear Program.
  - Total unimodularity guarantees that we will still find integer solutions.

LINEAR PROGRAM

Maximize:

$$\sum_{x_j \in \mathcal{X}} \left(y_{new_{x_j}} P(y_{new_{x_j}}) + y_{end_{x_j}} P(y_{end_{x_j}}) + y_j \beta_j + (1 - y_j)(1 - \beta_j) \right)$$

$$+ \sum_{(x_j, x_k) \in \mathcal{L}} y_{link_{x_j, x_k}} P(y_{link_{x_j, x_k}})$$

Subject To:

$$y_{new_{x_i}} + \sum_{x_k \in \mathcal{L}(x_i)} y_{link_{x_k, x_i}} = y_{end_{x_i}} + \sum_{x_k \in \mathcal{L}(x_i)} y_{link_{x_i, x_k}} = y_{x_i} \quad \forall x_i \in \mathcal{X}$$

- Where $\mathcal{L}$ denotes the set of detections that can be associated, i.e. are from subsequent frames.

FINDING THE PROBABILITIES

- So we have defined ways to solve the association problem, however we still need to find ways to compute the probabilities of our random variables, ie $P(y)$, $P(y_{link})$, $P(y_{new})$, $P(y_{end})$.
- In the following we propose some ways to address this.

DETECTION SCORE - $P(y)$

- This score can be either the output of the scorer on the detector or a new network trained to classify true and false positives.
- It's important that this score is precise, otherwise you might end up with many fragmentations or false positives.

MATCHING SCORE - $P(y_{link})$

- Many metrics can be used for this, among them we have:
  - Bounding box overlap;
  - Bounding box size;
  - Color histogram similarity;
  - Orientation cosine similarity;
  - Position distance;
  - ...
- As is the case for y_j, a poor estimation of this cost may lead to many ID switches.

MATCHING SCORE - $P(y_{link})$

MATCHING SCORE - $P(y_{link})$

MATCHING SCORE - $P(y_{link})$

NEW/END SCORES - $P(y_{new})$ AND $P(y_{end})$

- The costs for $P(y_{new})$ and $P(y_{end})$ are usually constants estimated via the EM Algorithm.
- Not particularly important to be set on a per-detection basis.

METRICS

- Given a solution to the tracking problem, it's in our interest to have metrics to proper evaluate whether or not the solution satisfies our expectations.
- The most widespread metrics for this purpose are the CLEAR MOT and MT/ML metrics.

MOTA

- **MOTA** - Multiple Object Tracking Accuracy:
  - Accounts for errors in the trajectory configuration: misses, false positives and mismatches.
  - Gives a measure of how well the tracker is able to detect object and keep consistent trajectories, regardless of the precision with which the object detections are estimated.

$$MOTA = \frac{\sum_t (m_t, fp_t, mme_t)}{\sum_t g_t}$$

- Where m_t, fp_t, mme_t and g_t are, respectively, the number of misses, false positives, mismatches and objects present at time t.

MOTP

- **MOTP** - Multiple Object Tracking Precision:
  - Measures the total error in estimated position between object-hypothesis pairs
  - Evaluates the tracker's ability of estimating object positions, regardless of its skill of keeping consistent trajectories.

$$MOTP = \frac{\sum_{i,t} d_t^i}{\sum_t c_t}$$

- Where d_t^i is the distance between object o_i and its corresponding hypothesis at time t and c_t is the number of matches at time t.

MT / ML

- **MT** - Mostly Tracked - evaluates the percentage of trajectories that cover a ground-truth trajectory for more than 80% of its length.
- **ML** - Mostly Lost - percentage of trajectories that cover a GT trajectory for less than 20% of its length.
- **PT** - Partially Tracked - $1.0 - MT - ML$.

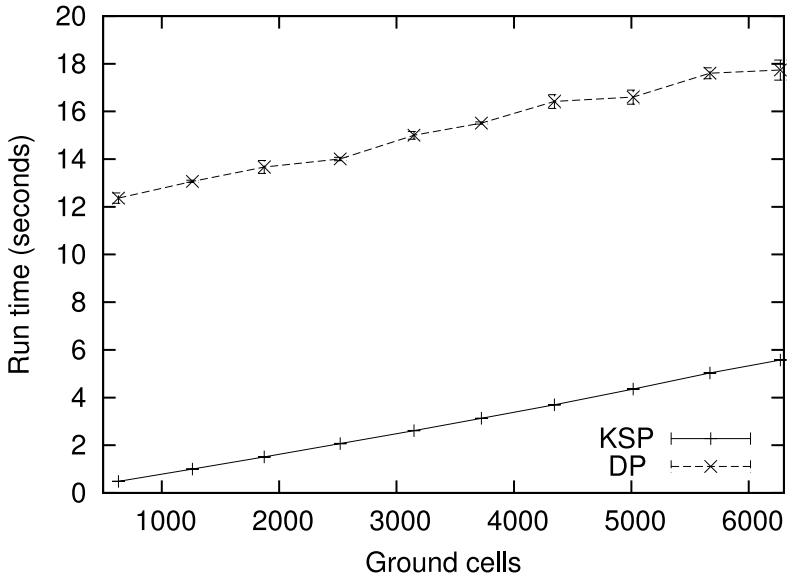
ONLINE TRACKING

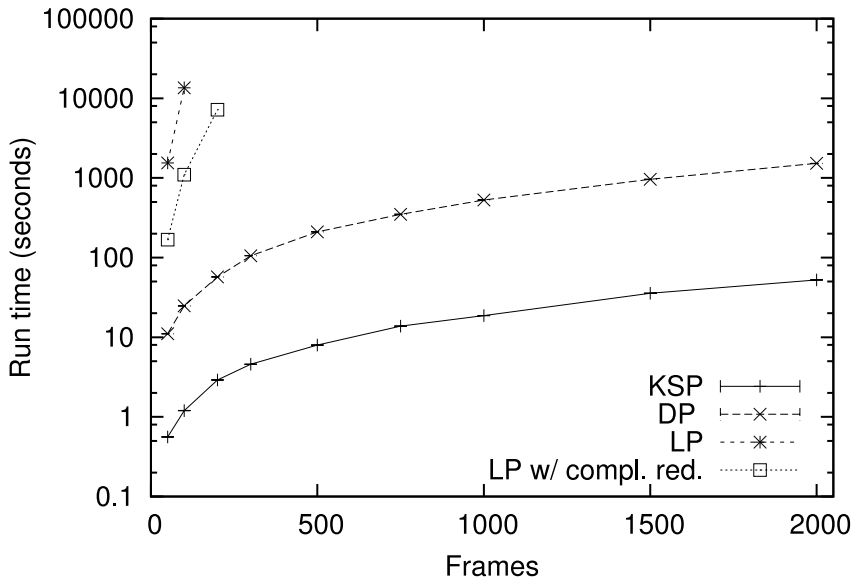
- So far, our optimization scheme follows an offline setting: Given an entire sequence of observations we find the RV assignments with optimal cost.
- This approach doesn't scale too well since as the length of the sequence grows, so does the complexity of the LP.
- Furthermore, for real time applications (robotics, autonomous driving, broadcasting, etc) we need the tracking output with minimal delay.

Online Tracking

- The min-cost flow approach has a complexity of $O(kn^2m \log n)$, where k is the number of objects, m the number of edges and n the number of nodes in the graph.
  - Even if we used a sliding window approach, this complexity would still make the successive re-calculations slow.
- Because our tracking graph is a DAG and we want node-disjoint solutions, we can formulate our problem using k shortest node-disjoint paths.
  - This approach has a complexity of $O(k(m + n \log n))$, making it feasible for online applications.

ONLINE TRACKING





Credit: Berclaz et al, Multiple Object Tracking using K-Shortest Paths Optimization

TRACKING WITH KSP

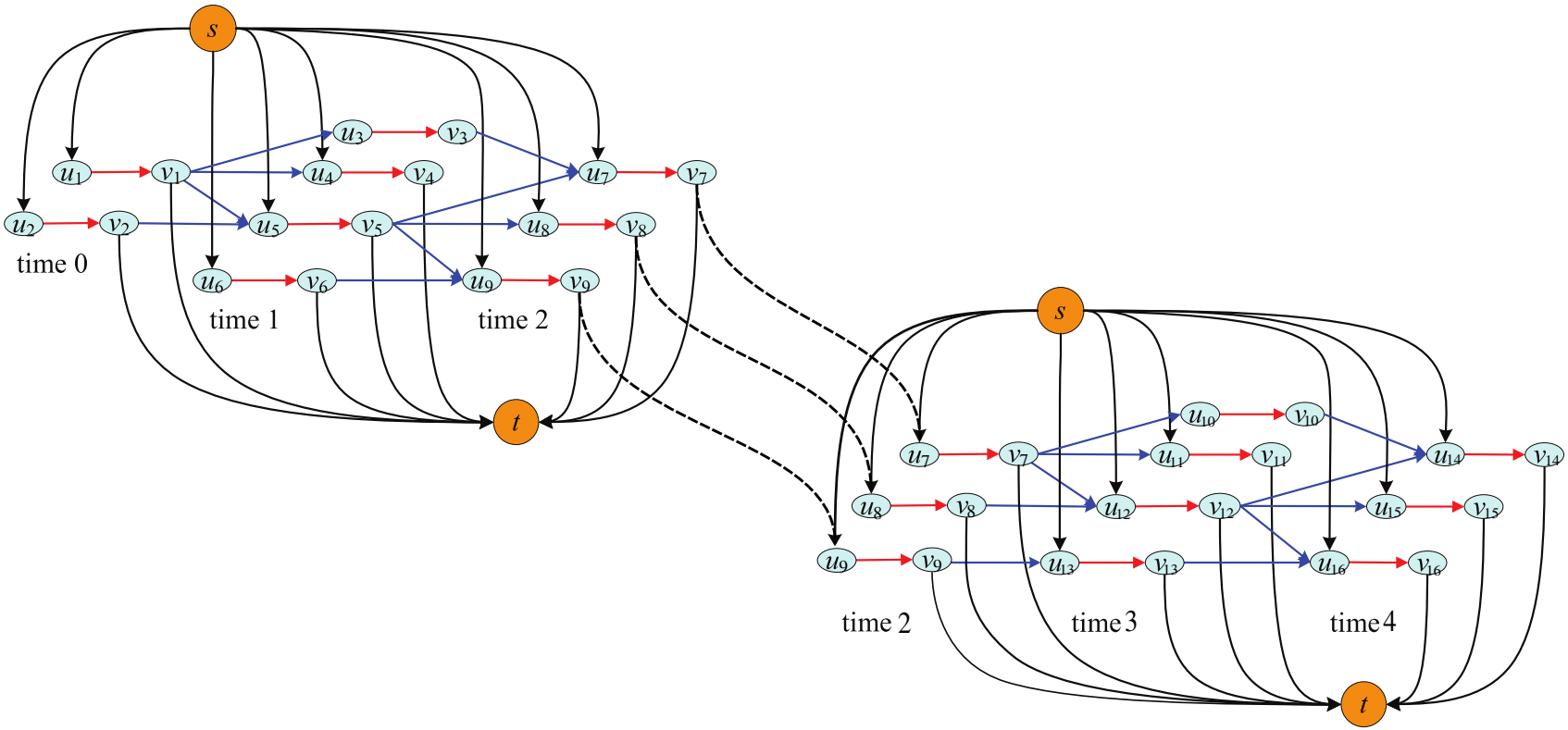
Input: Observation set $\mathcal{X}$

1 Build flow graph $\mathcal{G}(\mathcal{X})$
2 Start with $k = 0$
3 **while** *k can be augmented* **do**
4 Augment k by 1
5 Compute current cost with *KSP*
6 **if** *current cost* $<$ *optimal cost* **then**
7 Store current cost as optimal cost
8 **return** *global optimum*

Online Tracking with KSP

- To transform our offline setting into online, we split the sequence into smaller batches that we can efficiently process.
- To enforce consistency between batches, we introduce an overlap between each sequence, and make it such that the inward flow of each node is equal to the outward flow in the previous batch.

ONLINE TRACKING WITH KSP



Credit: Nevatia et al, Global Data Association for Multi-Object Tracking - Edited

ONLINE TRACKING WITH KSP

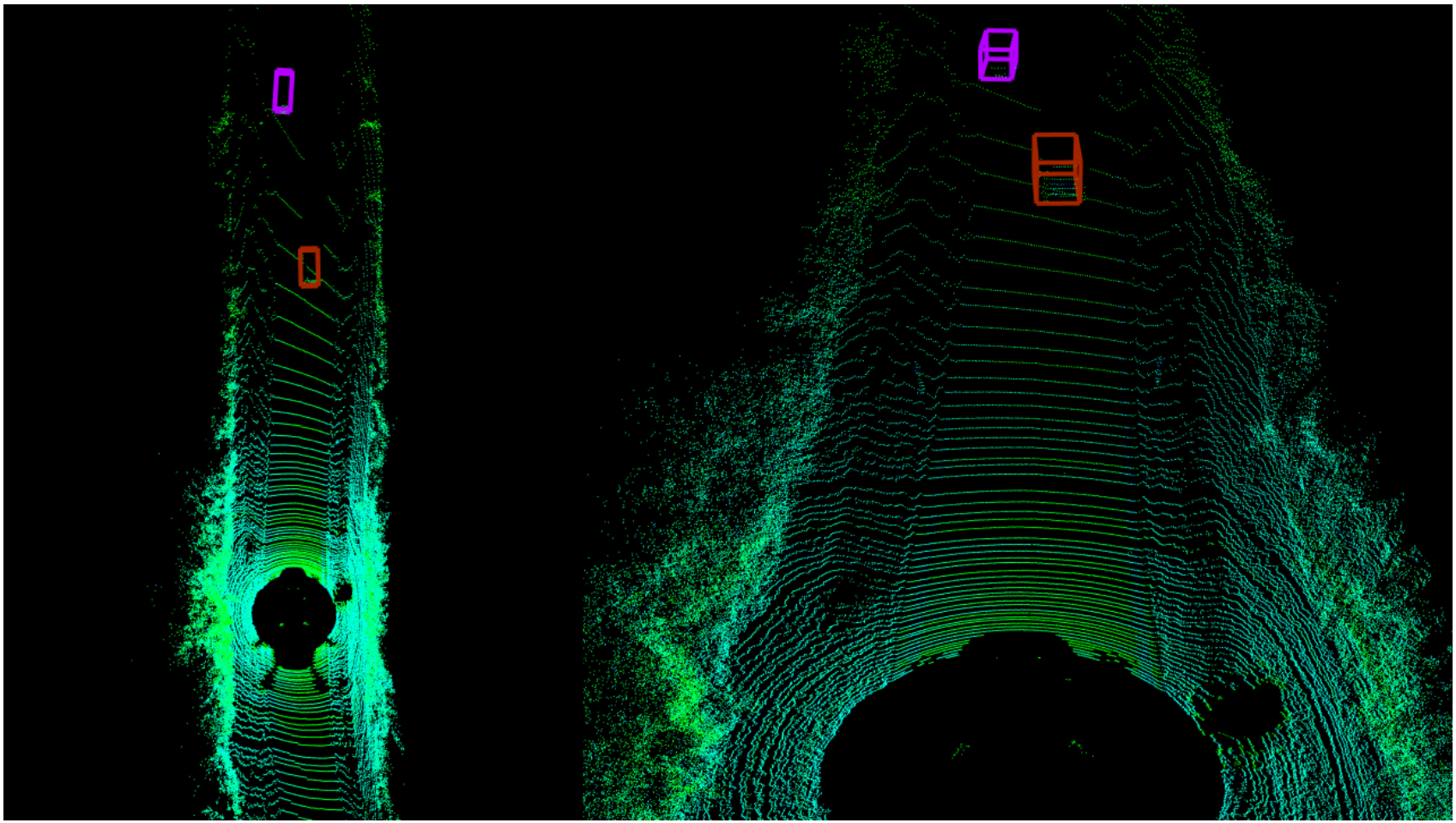
- For implementation details and additional information into bounded computation and memory we refer you to:

  http://www.cs.toronto.edu/boundTracking/

CONTINUOUS OPTIMIZATION

- So far we've only dealt with discrete assignments, ie we either associate two detections or don't.
- This doesn't take into account higher order motion features (speed, acceleration, etc), which will generally produce jagged trajectories.

CONTINUOUS OPTIMIZATION



CONTINUOUS OPTIMIZATION

- However, we can do better by using prior knowledge of the domain:
  - Cars don't change shape and have predictable motion dynamics according to the road;
  - The trajectory of a projectile (ball) can be reasonably estimated given speed and acceleration;
  - People usually move in the direction they are facing.
  - ...

CONTINUOUS OPTIMIZATION

1. We can encode that information as an additional cost in the objective function and minimize using gradient descent. [1]
2. We can use a filtering technique (Bayes, Kalman, Particle, ...) over each individual trajectory. [2]
3. We can use Recurrent Neural Networks to estimate smooth trajectories. [3]

[1] Milan et al., Multi-Target Tracking by Discrete-Continuous Energy Minimization

[2] Thrun et al., Probabilistic Robotics

[3] Milan et al., Online Multi-Target Tracking Using Recurrent Neural Networks

CONTINUOUS OPTIMIZATION

1. **We can encode that information as an additional cost in the objective function and minimize using gradient descent.** [1]
2. We can use a filtering technique (Bayes, Kalman, Particle, ...) over each individual trajectory. [2]
3. We can use Recurrent Neural Networks to estimate smooth trajectories. [3]

[1] Milan et al., Multi-Target Tracking by Discrete-Continuous Energy Minimization

[2] Thrun et al., Probabilistic Robotics

[3] Milan et al., Online Multi-Target Tracking Using Recurrent Neural Networks

AUGMENTED OBJECTIVE FUNCTION

$$O(\mathcal{T}, \mathcal{Y}|\mathcal{X}) = \sum_{x_j \in \mathcal{X}} \left(y_{new_{x_j}} P(y_{new_{x_j}}|\mathcal{T}) + y_{end_{x_j}} P(y_{end_{x_j}}|\mathcal{T}) + y_{x_j} P(y_{x_j}|\mathcal{T}) \right)$$
$$+ \sum_{(x_j,x_k) \in \mathcal{L}} y_{link_{x_j,x_k}} P(y_{link_{x_j,x_k}}|\mathcal{T})$$
$$+ \sum_{\mathbf{T_j} \in \mathcal{T}} \mathbf{P}(\mathbf{T_j}|\mathcal{Y})$$

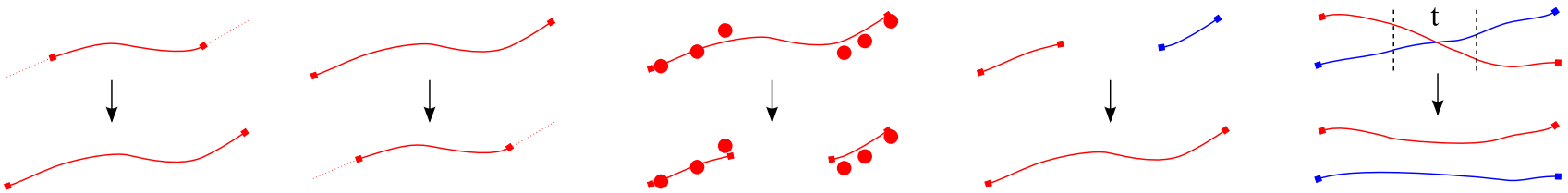
- Notice that we add conditionals to our probabilities since we'll want to use the continuous information in our discrete assignments and vice-versa.

AUGMENTED OBJECTIVE FUNCTION

- With the $P(T_j|\mathcal{Y})$ term we can model higher order functions of our trajectories.
  - Impose smooth shapes to trajectories, such as splines;
  - Constrain their dynamic states to feasible ones;
  - Filter out noise produced by low scoring detections;
  - ...

AUGMENTED OBJECTIVE FUNCTION

- Furthermore, we can exploit motion models to improve our initial trajectory hypothesis



Credit: Milan et al., Multi-Target Tracking by Discrete-Continuous Energy Minimization

OPTIMIZATION SCHEME

Input: Observation set $\mathcal{X}$

1 **repeat**
2 | Minimize $O(\mathcal{T}, \mathcal{Y}|\mathcal{X})$ wrt. $\mathcal{Y}$ via Max-Flow/LP/KSP
3 | Minimize $O(\mathcal{T}, \mathcal{Y}|\mathcal{X})$ wrt. $\mathcal{T}$ via Gradient Descent
4 **until** *convergence;*
5 **return** *global optimum*

Some Numbers - KITTI Dataset

- Matching error rate:

	Error Rate		
Metric	Mean	Positive	Negative
Cosine Similarity	29.16%	19.02%	39.30%
Color Correlation	15.31%	20.21%	10.41%
BBox Size	6.31%	5.27%	7.35%
BBox Position	5.13%	5.86%	4.40%
BBox Overlap	2.46%	4.68%	0.23%

SOME NUMBERS - KITTI DATASET

- Hungarian X Linear Program:

METHOD	MOTA	MOTP	MT	ML	IDS	FRAG	FP
Hungarian	58.69%	84.26%	**70.87%**	**6.80%**	68	196	2880
LP	**71.12%**	**85.24%**	55.66%	12.94%	**19**	**99**	**628**

SOME NUMBERS - KITTI DATASET

- Real X Ideal detections:

METHOD	MOTA	MOTP	MT	ML	IDS	FRAG	FP
Real	71.12%	85.24%	55.66%	12.94%	19	99	628
Ideal	**79.44%**	**89.27%**	**72.17%**	**6.15%**	**7**	**37**	**462**