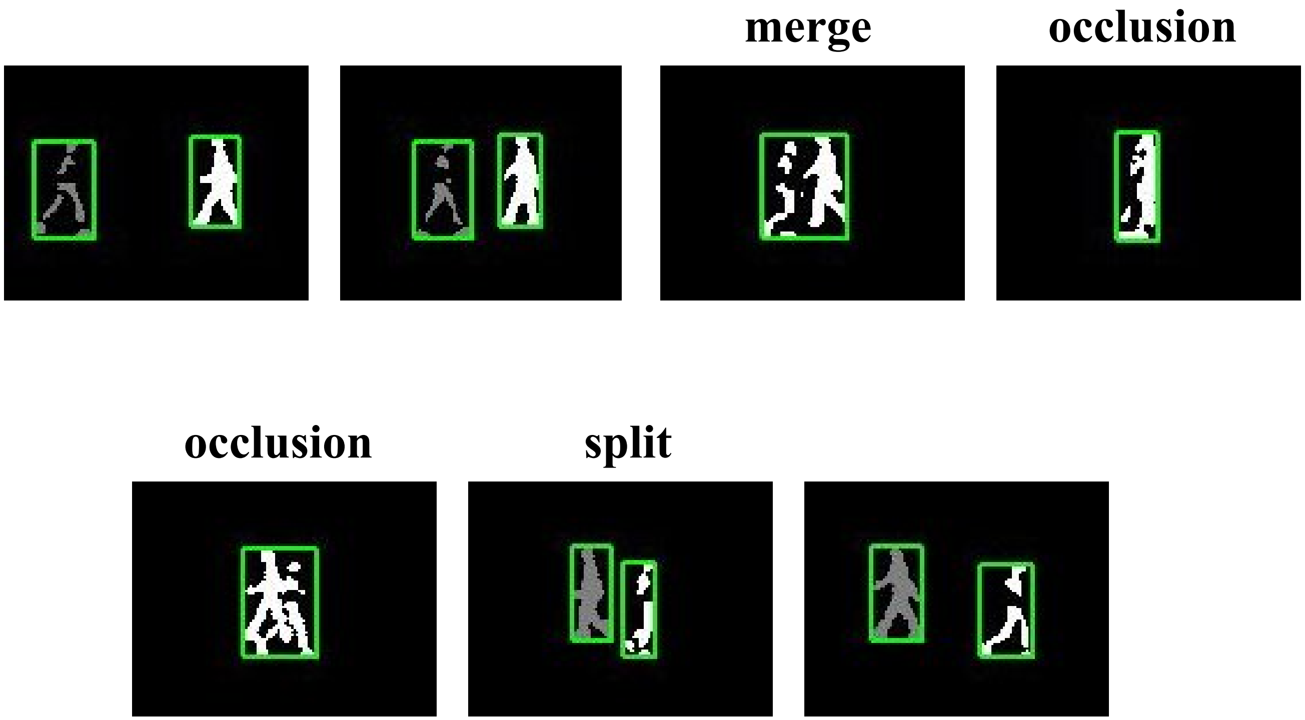
Lecture 28

Intro to Tracking

Some overlap with T&V Section 8.4.2 and Appendix A.8

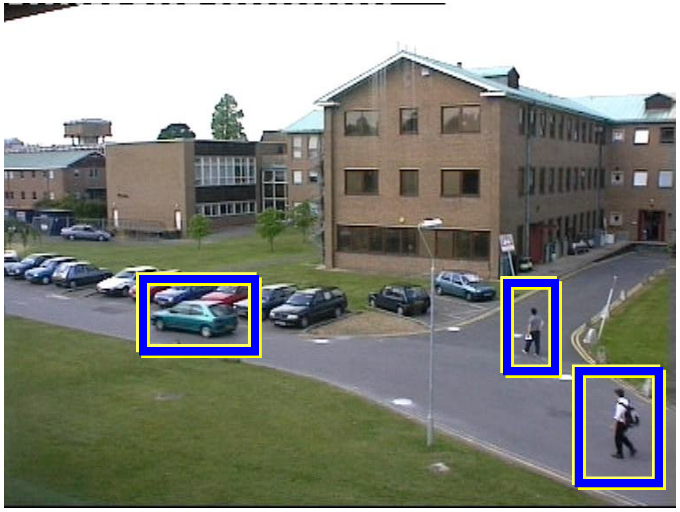
Recall: Blob Merge/Split

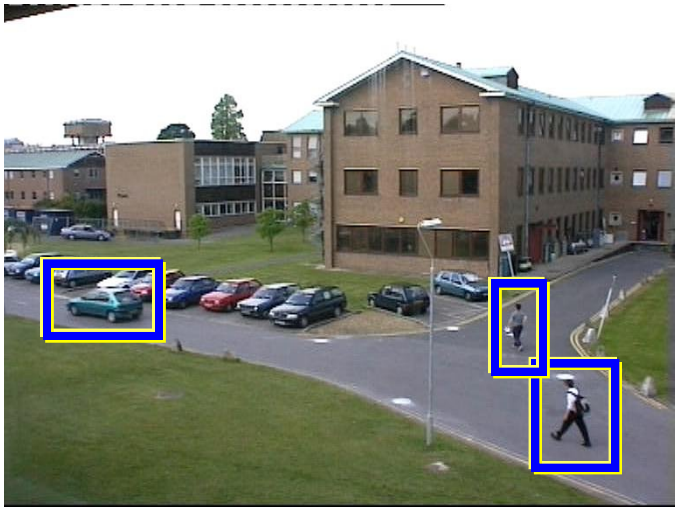


When two objects pass close to each other, they are detected as a single blob. Often, one object will become occluded by the other one. One of the challenging problems is to maintain correct labeling of each object after they split again.

Data Association

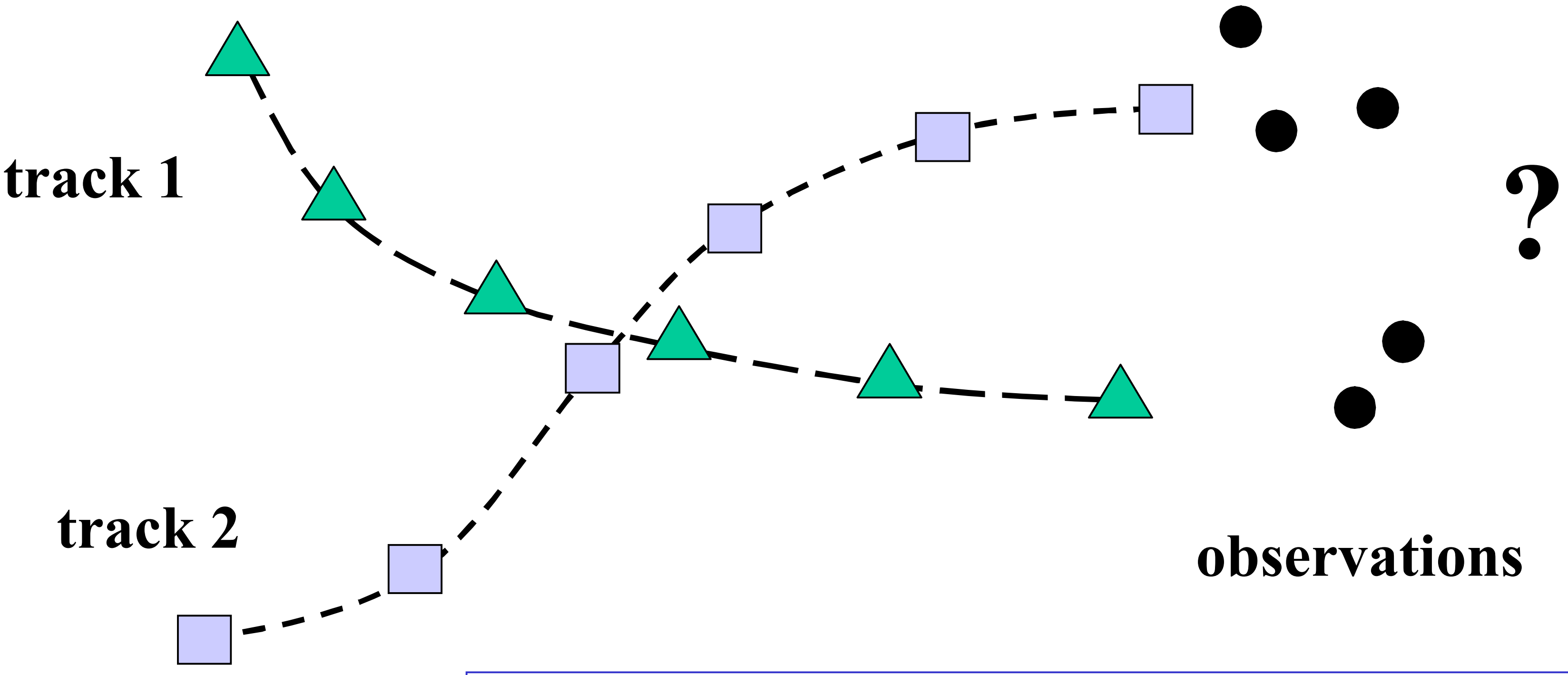
More generally, we seek to match a set of blobs across frames, to maintain continuity of identity and generate trajectories.





Data Association Scenarios

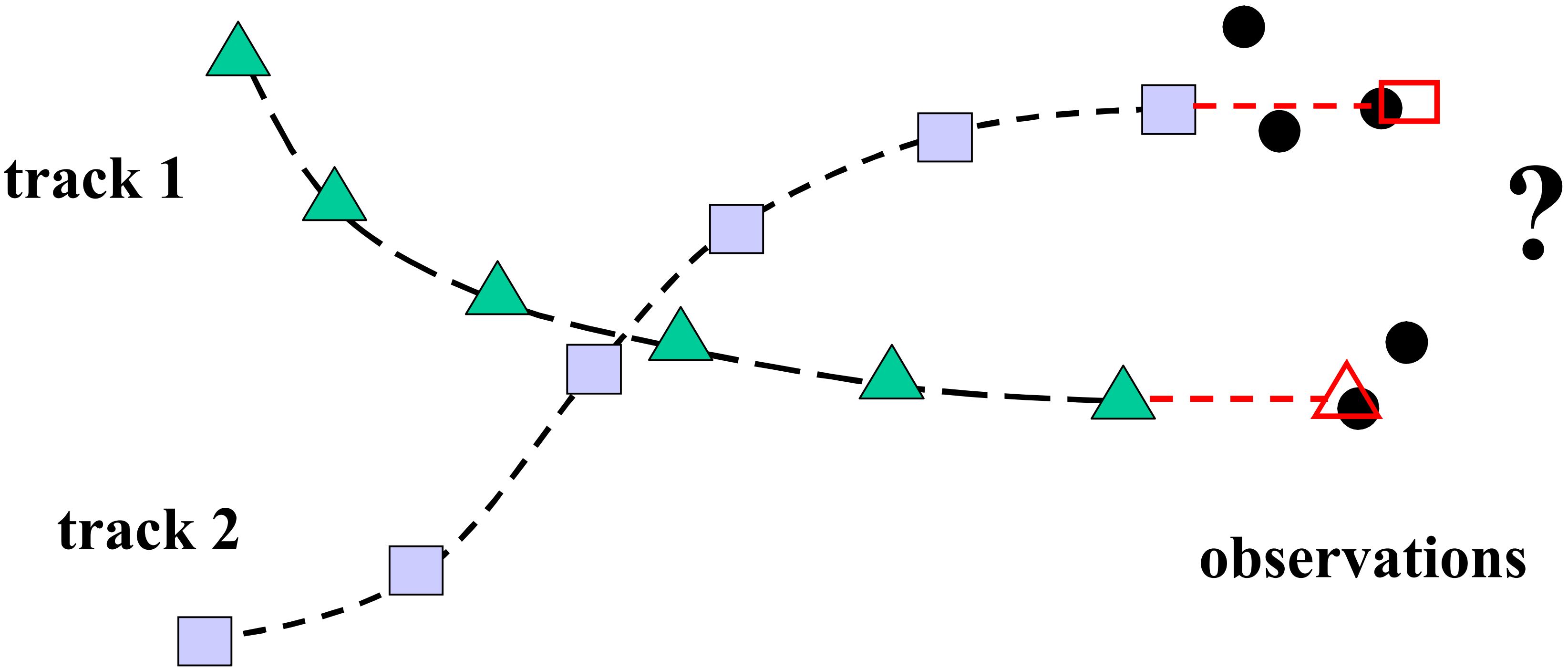
Multi-frame Matching (matching observations in a new frame to a set of tracked trajectories)



How to determine which observations to add to which track?

Tracking Matching

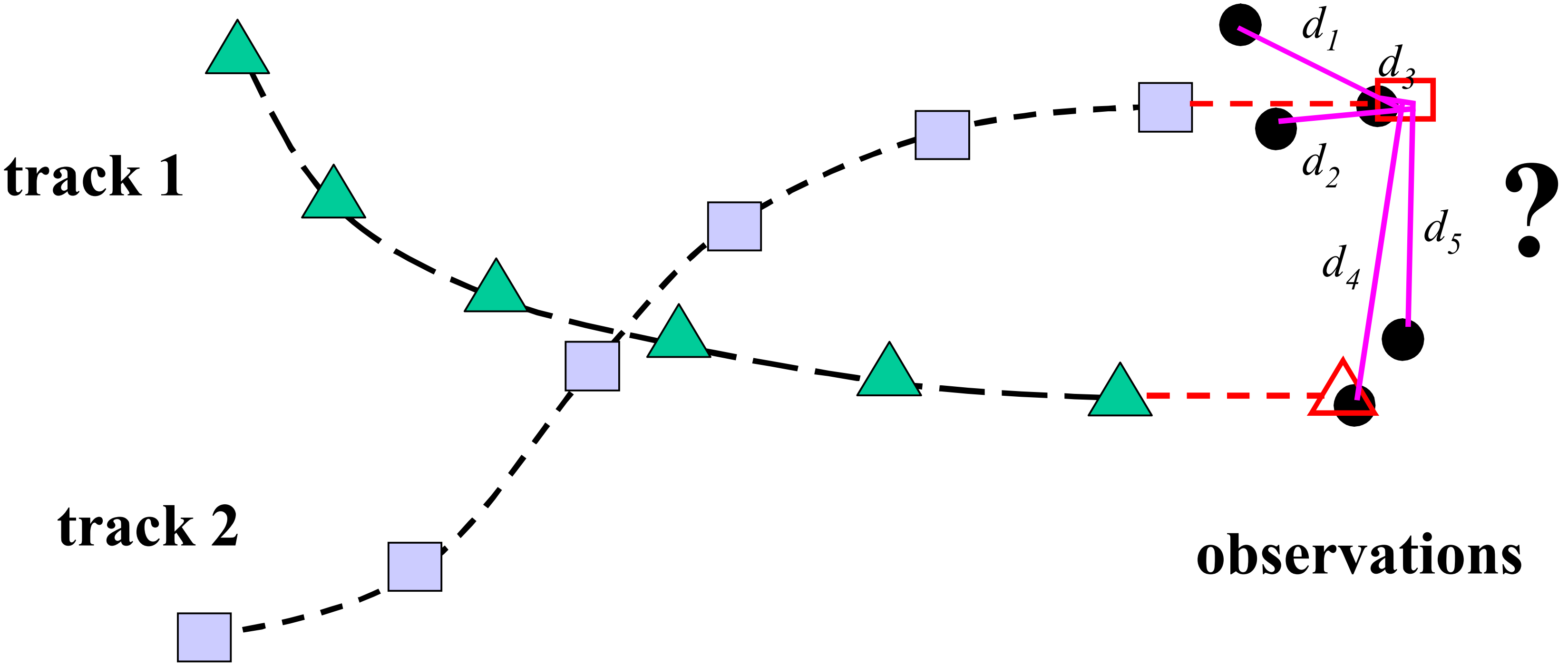
Intuition: predict next position along each track.



How to determine which observations to add to which track?

Tracking Matching

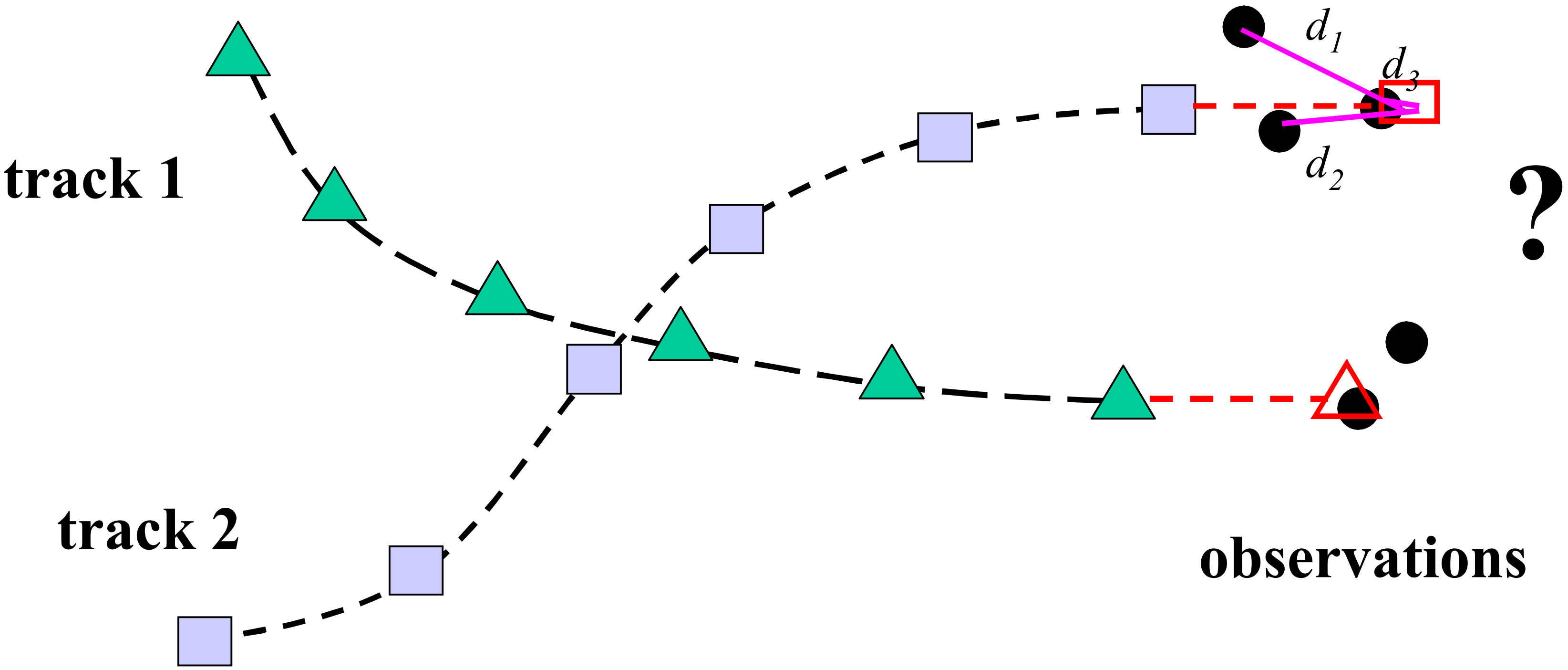
Intuition: predict next position along each track.
Intuition: match should be close to predicted position.



How to determine which observations to add to which track?

Tracking Matching

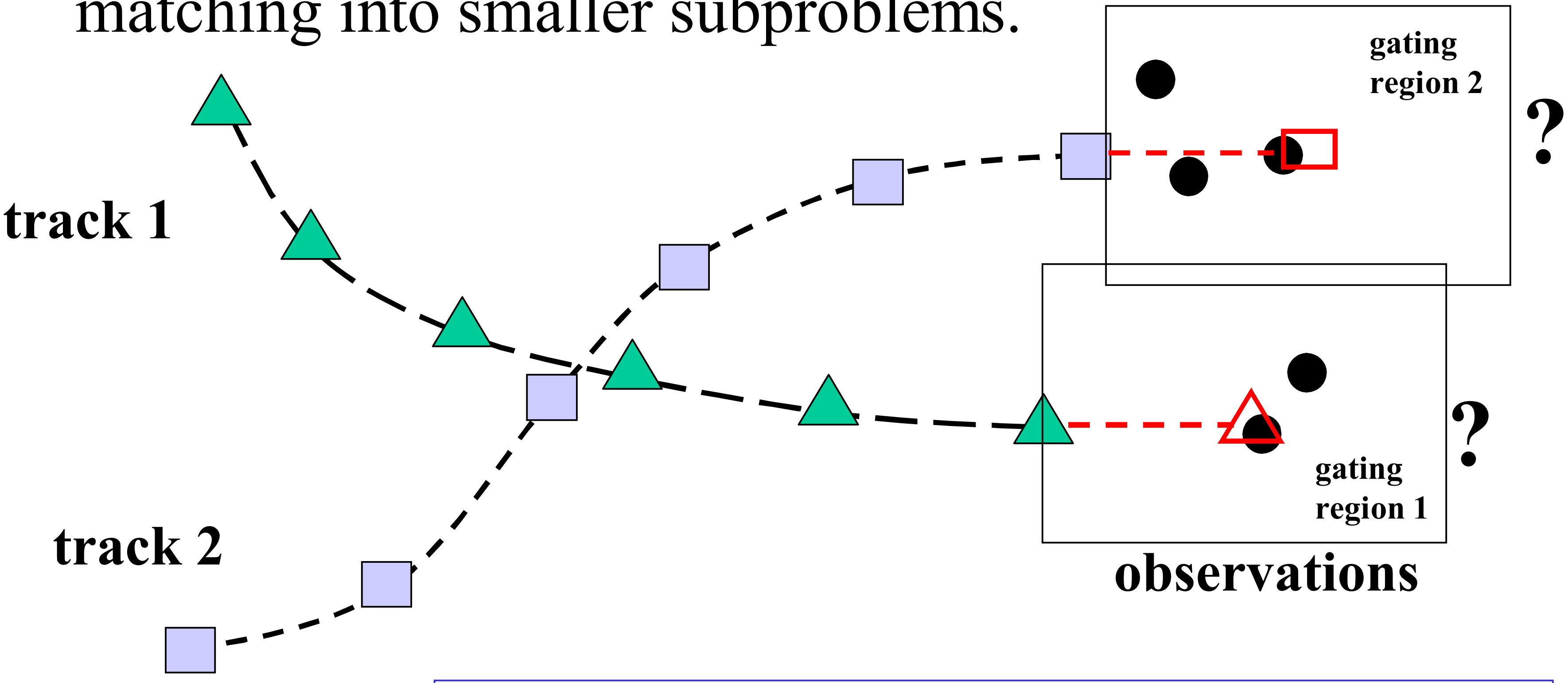
Intuition: predict next position along each track.
Intuition: match should be close to predicted position.
Intuition: some matches are highly unlikely.



How to determine which observations to add to which track?

Gating

A method for pruning matches that are geometrically unlikely from the start. Allows us to decompose matching into smaller subproblems.



How to determine which observations to add to which track?

Filtering Framework

Discrete-time state space filtering

We want to recursively estimate the current state at every time that a measurement is received.

Two step approach:

1) prediction: propagate state pdf forward in time, taking process noise into account (translate, deform, and spread the pdf)

2) update: use Bayes theorem to modify prediction pdf based on current measurement

Prediction

Kalman filtering is a common approach.
System model and measurement model are linear.
Noise is zero-mean Gaussian
Pdfs are all Gaussian

1) System model

$$\mathbf{x}_k = F_k \mathbf{x}_{k-1} + \mathbf{v}_{k-1} \qquad p(v_k) = N(v_k \mid 0, Q_k)$$

2) Measurement model

$$\mathbf{z}_k = H_k \mathbf{x}_k + \mathbf{n}_k \qquad p(n_k) = N(n_k \mid 0, R_k)$$

More detail is found in T&V Section 8.4.2 and Appendix A.8

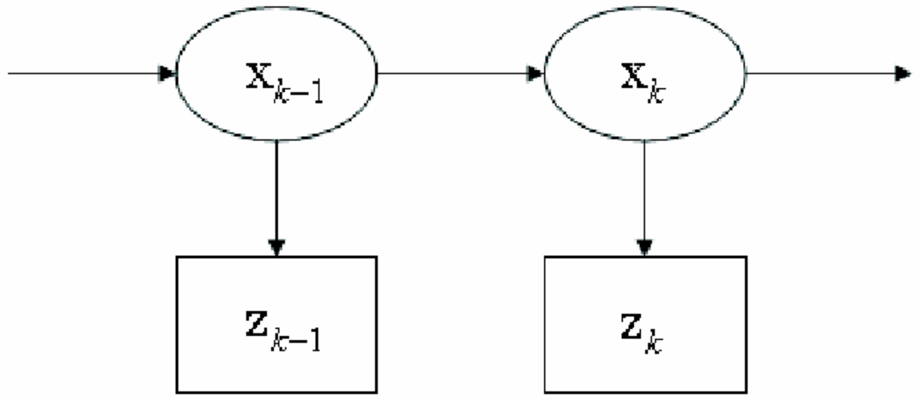
Kalman Filter

All pdfs are then Gaussian. (note: all marginals of a Gaussian are Gaussian)

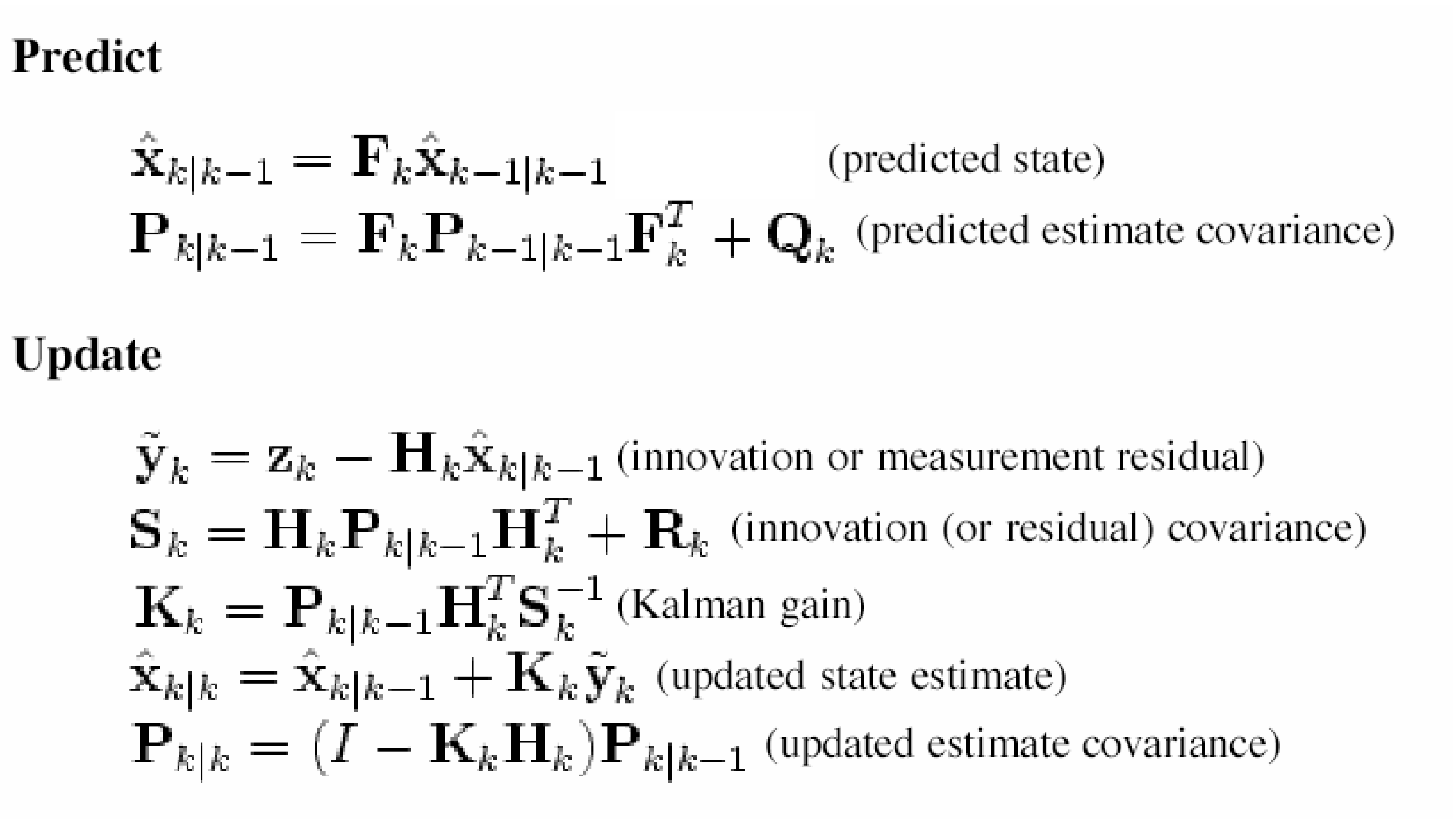
$$p(\mathbf{x}_k|\mathbf{x}_{k-1}) = N(\mathbf{x}_k, \mathbf{F}_k\mathbf{x}_{k-1}, \mathbf{Q}_k)$$

$$p(\mathbf{z}_k|\mathbf{x}_k) = N(\mathbf{z}_k, \mathbf{H}_k\mathbf{x}_k, \mathbf{R}_k)$$

$$p(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1}) = N(\mathbf{x}_{k-1}, \hat{\mathbf{x}}_{k-1}, \mathbf{P}_{k-1})$$



Kalman Filter



Predict

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{F}_k \hat{\mathbf{x}}_{k-1|k-1} \quad \text{(predicted state)}$$

$$\mathbf{P}_{k|k-1} = \mathbf{F}_k \mathbf{P}_{k-1|k-1} \mathbf{F}_k^T + \mathbf{Q}_k \quad \text{(predicted estimate covariance)}$$

Update

$$\tilde{\mathbf{y}}_k = \mathbf{z}_k - \mathbf{H}_k \hat{\mathbf{x}}_{k|k-1} \quad \text{(innovation or measurement residual)}$$

$$\mathbf{S}_k = \mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^T + \mathbf{R}_k \quad \text{(innovation (or residual) covariance)}$$

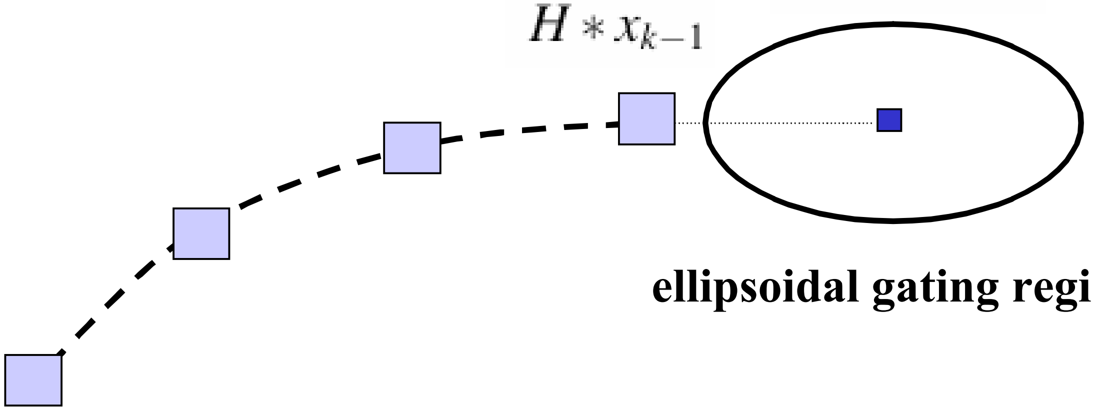
$$\mathbf{K}_k = \mathbf{P}_{k|k-1} \mathbf{H}_k^T \mathbf{S}_k^{-1} \quad \text{(Kalman gain)}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \tilde{\mathbf{y}}_k \quad \text{(updated state estimate)}$$

$$\mathbf{P}_{k|k} = (I - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{k|k-1} \quad \text{(updated estimate covariance)}$$

Example

$$x = \begin{bmatrix} x \\ y \\ u \\ v \end{bmatrix} \quad F = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$



$H * F * x_{k-1}$

$H * x_{k-1}$

$\tilde{y}' \, S^{-1} \tilde{y} \leq G$

ellipsoidal gating region

$\hat{\mathbf{x}}_{k|k-1} = \mathbf{F}_k \hat{\mathbf{x}}_{k-1|k-1}$ (predicted state)

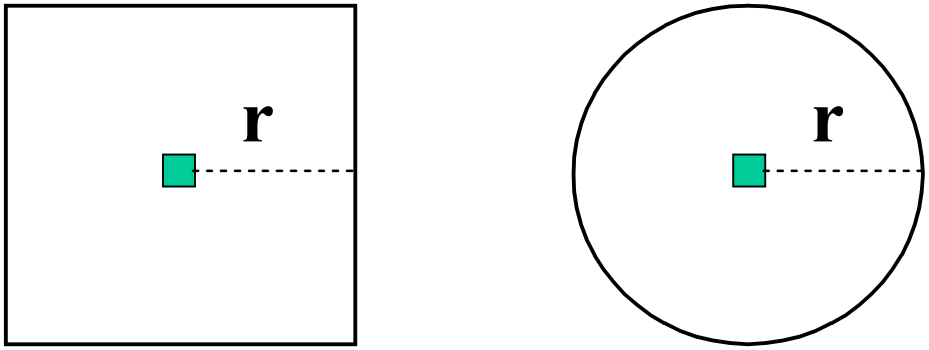
$\mathbf{P}_{k|k-1} = \mathbf{F}_k \mathbf{P}_{k-1|k-1} \mathbf{F}_k^T + \mathbf{Q}_k$ (predicted estimate covariance)

$\tilde{\mathbf{y}}_k = \mathbf{z}_k - \mathbf{H}_k \hat{\mathbf{x}}_{k|k-1}$ (innovation or measurement residual)

$\mathbf{S}_k = \mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^T + \mathbf{R}_k$ (innovation (or residual) covariance)

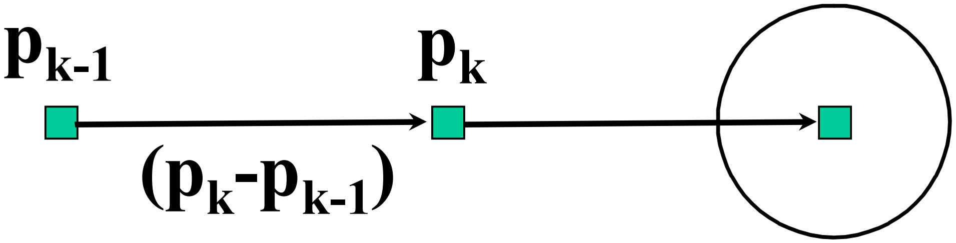
Simpler Prediction/Gating

Constant position + bound on maximum interframe motion



constant position
prediction

Three-frame constant velocity prediction



prediction
$p_k + (p_k-p_{k-1})$

typically, gating
region can be smaller

Aside: Camera Motion

Hypothesis: constant velocity target motion model is adequate provided we first compensate for effects of any background camera motion.

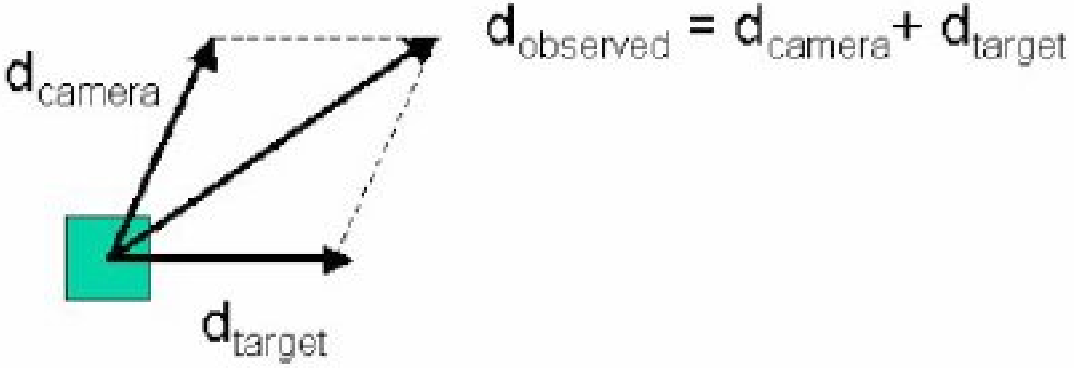


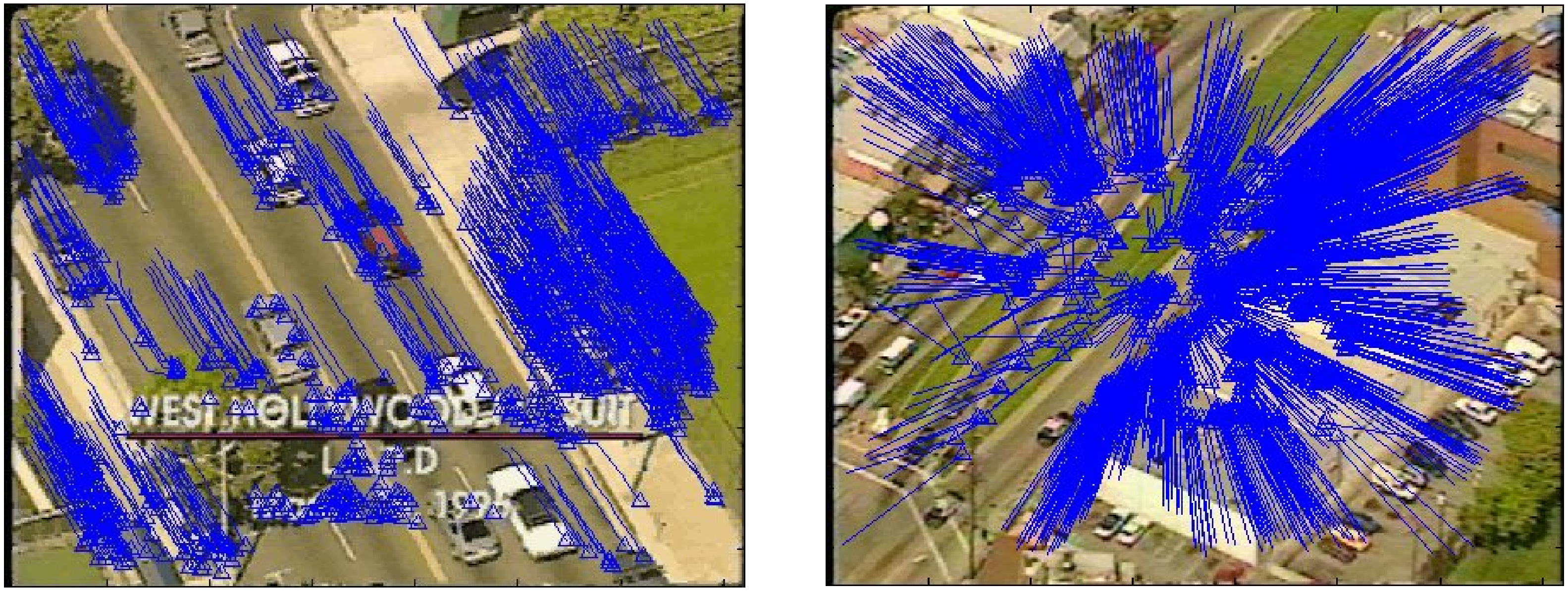
Figure 3: Decomposition of observed displacement of a target between two video frames into terms based only on motion of the camera and motion of the target.

Camera Motion Estimation

Approach:

Estimate sparse optic flow using Lucas-Kanade algorithm (KLT)
Estimate parameteric model (affine) of scene image motion

Note: this offers a low computational cost alternative to image warping and frame differencing approaches.



used for motion prediction, and zoom detection

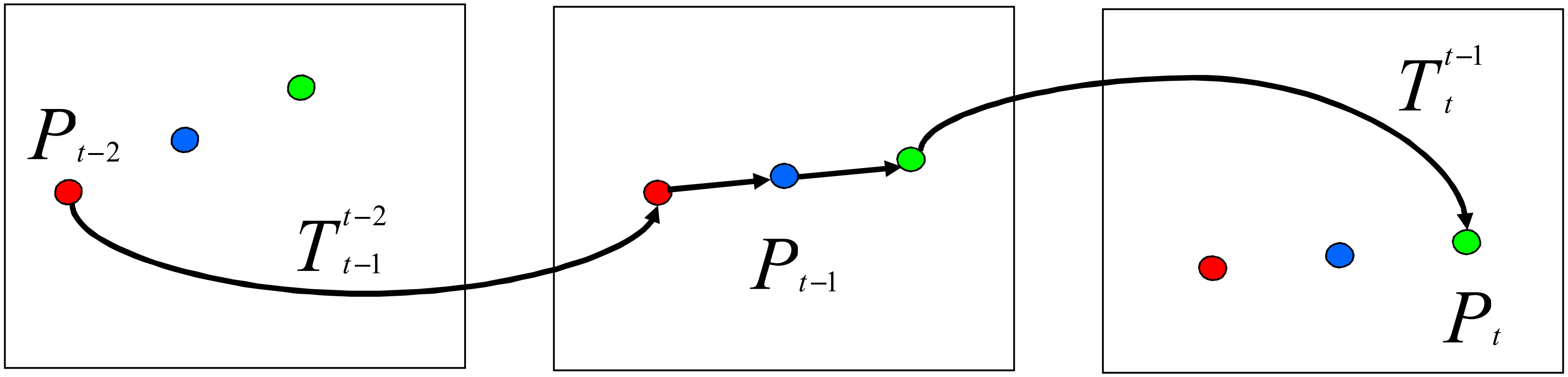
Target Motion Estimation

Approach: Constant velocity estimate, after compensating for camera motion

P_f = target position in frame f

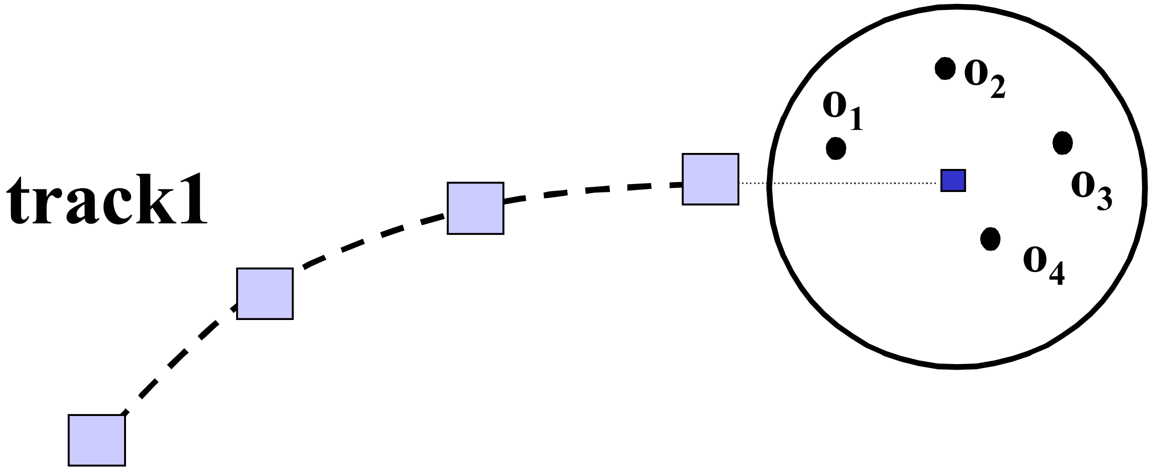
T_g^f = camera motion from frame f to frame g

$$P_t = T_t^{t-1} * [P_{t-1} + (P_{t-1} - (T_{t-1}^{t-2} * P_{t-2}))]$$



Global Nearest Neighbor (GNN)

Evaluate each observation in track gating region.
Choose “best” one to incorporate into track.

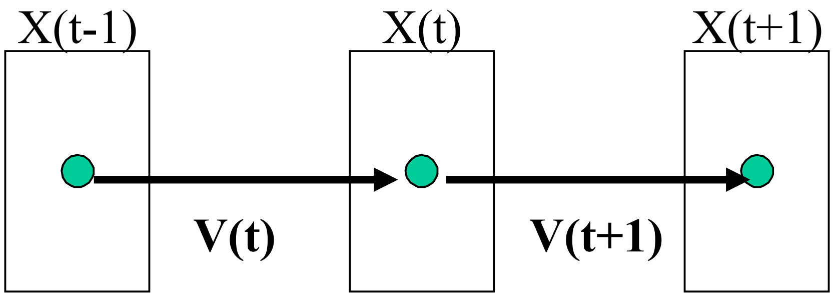


a_{1j} = score for matching observation j to track 1

Could be based on Euclidean or Mahalanobis distance to predicted location (e.g. $\exp\{-d^2\}$). Also could be based on similarity of appearance (e.g. template correlation score)

Data Association

We have been talking as if our objects are points. (which they are if we are tracking corner features or radar blips). But our objects are blobs – they are an image region, and have an area.



constant velocity assumes $V(t) = V(t+1)$

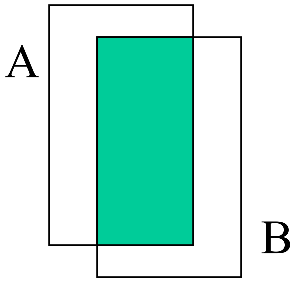
Map the object <u>region</u> forward in time to predict a new <u>region</u>.

Data Association

Determining the correspondence of blobs across frames is based on feature similarity between blobs.

Commonly used features: location , size / shape, velocity, appearance

For example: location, size and shape similarity can be measured based on bounding box overlap:

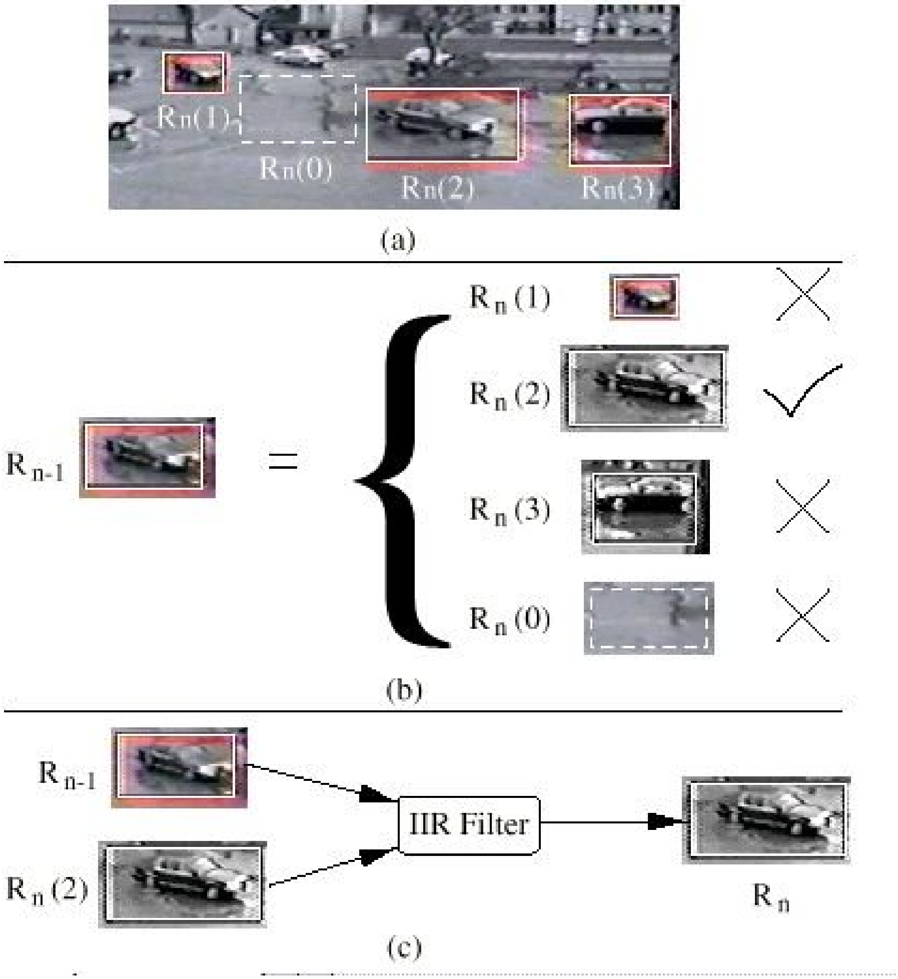


$$\textbf{score} = \frac{\textbf{2 * area(A and B)}}{\textbf{area(A) + area(B)}}$$

A = bounding box at time t
B = bounding box at time t+1

Appearance Information

Correlation of image templates is an obvious choice (between frames)



(a)

(b)

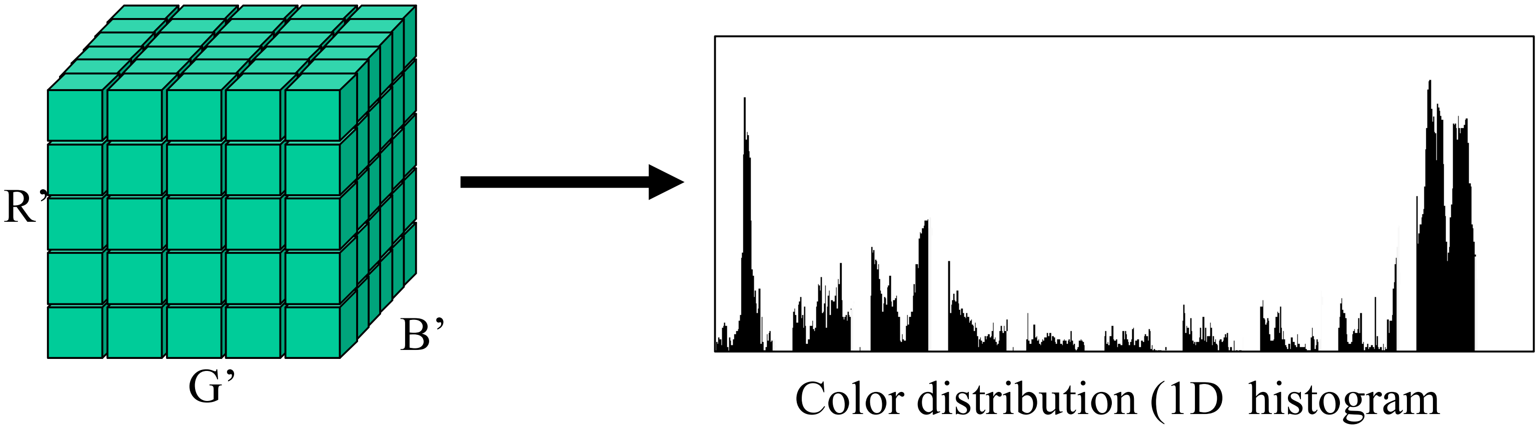
(c)

Extract blobs

Data association via normalized correlation.

Update appearance template of blobs

Appearance via Color Histograms



discretize

Color distribution (1D histogram normalized to have unit weight)

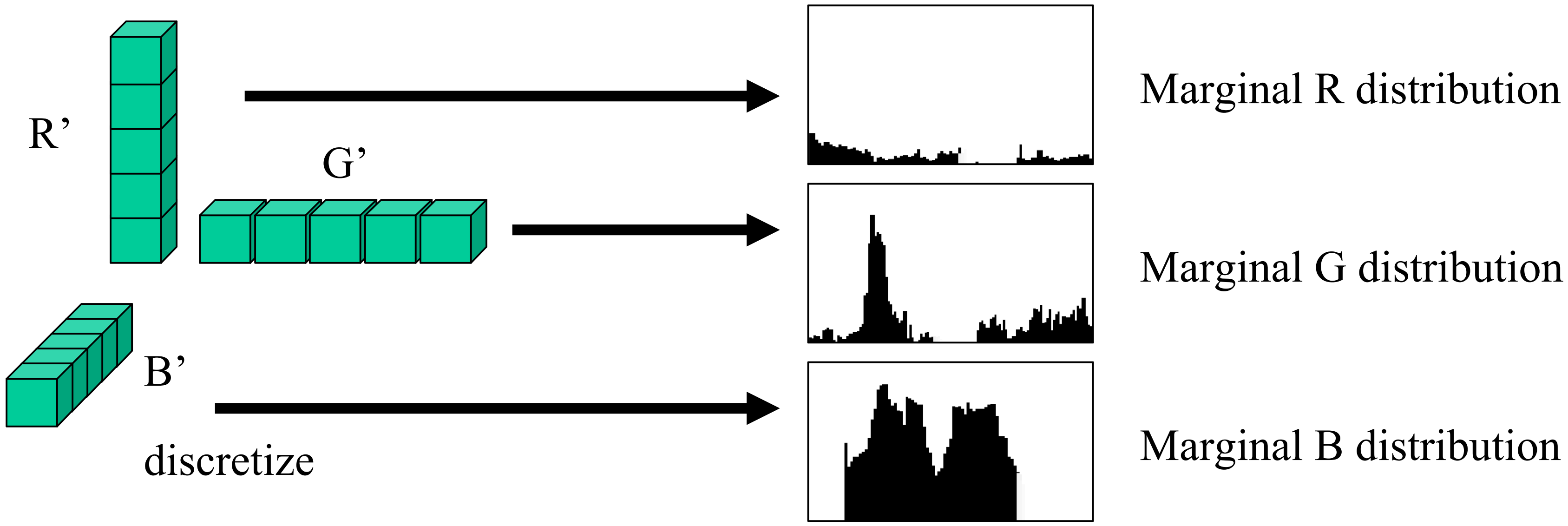
R’ = R << (8 - nbits)
G’ = G << (8 - nbits)
B’ = B << (8 - nbits)

Total histogram size is (2^(8-nbits))^3

example, 4-bit encoding of R,G and B channels yields a histogram of size 16*16*16 = 4096.

Smaller Color Histograms

Histogram information can be much much smaller if we are willing to accept a loss in color resolvability.



R'

G'

B'

discretize

Marginal R distribution

Marginal G distribution

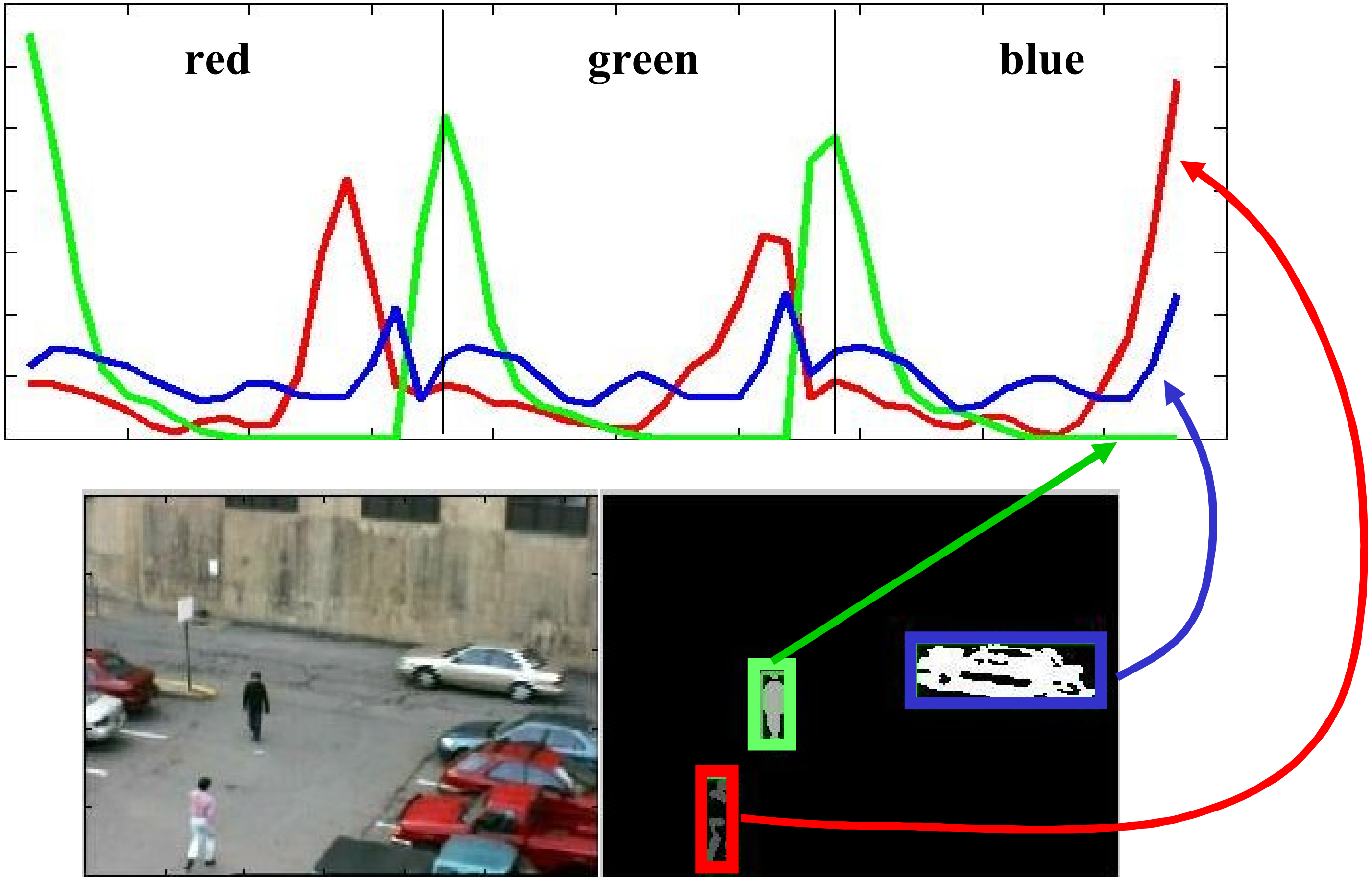
Marginal B distribution

R' = R << (8 - nbits)
G' = G << (8 - nbits)
B' = B << (8 - nbits)

Total histogram size is 3*(2^(8-nbits))

example, 4-bit encoding of R,G and B channels yields a histogram of size 3*16 = 48.

Color Histogram Example



Comparing Color Distributions

Given an n-bucket model histogram $\{m_\mathrm{i} \mid \mathrm{i}=1,\ldots,\mathrm{n}\}$ and data histogram $\{d_\mathrm{i} \mid \mathrm{i}=1,\ldots,\mathrm{n}\}$, we follow Comanesciu, Ramesh and Meer * to use the distance function:

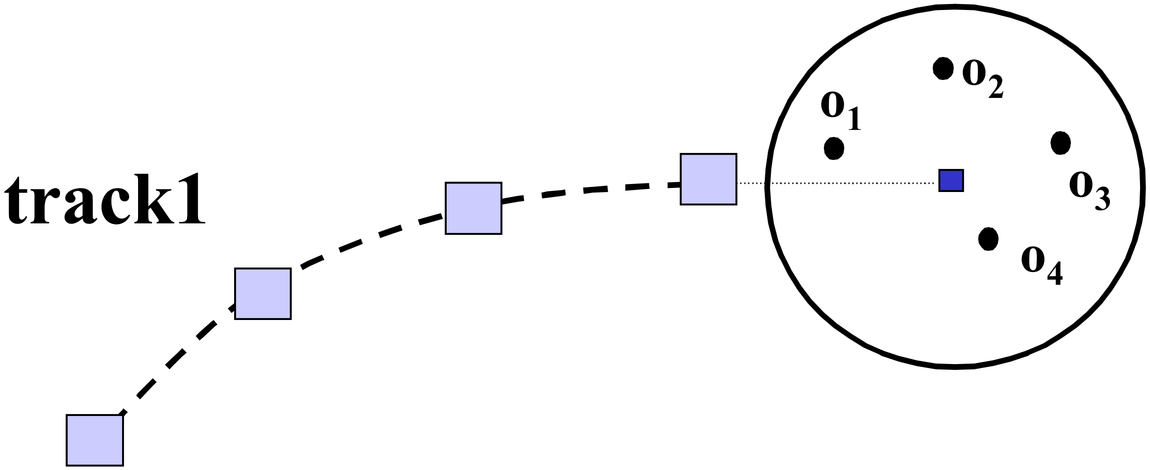
$$\Delta(\mathrm{m,d}) = \sqrt{1 - \sum_{i=1}^{n} \sqrt{m_i \times d_i}}$$

Why?
1) it shares optimality properties with the notion of Bayes error
2) it imposes a metric structure
3) it is relatively invariant to object size (number of pixels)
4) it is valid for arbitrary distributions (not just Gaussian ones)

*Dorin Comanesciu, V. Ramesh and Peter Meer, “Real-time Tracking of Non-Rigid Objects using Mean Shift,” IEEE Conference on Computer Vision and Pattern Recognition, Hilton Head, South Carolina, 2000 (best paper award).

Global Nearest Neighbor (GNN)

Evaluate each observation in track gating region.
Choose "best" one to incorporate into track.

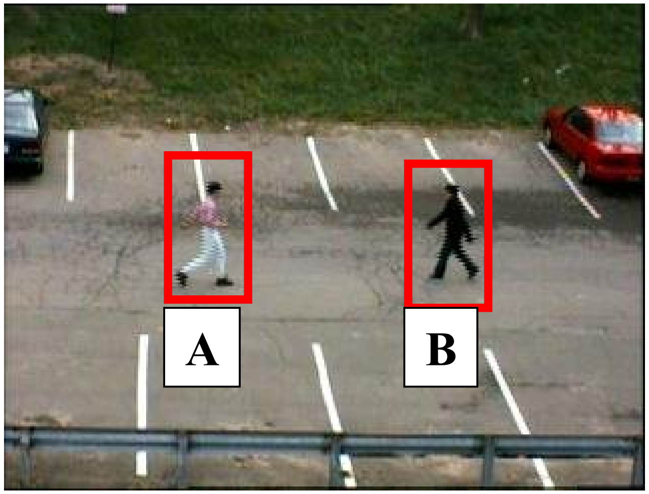


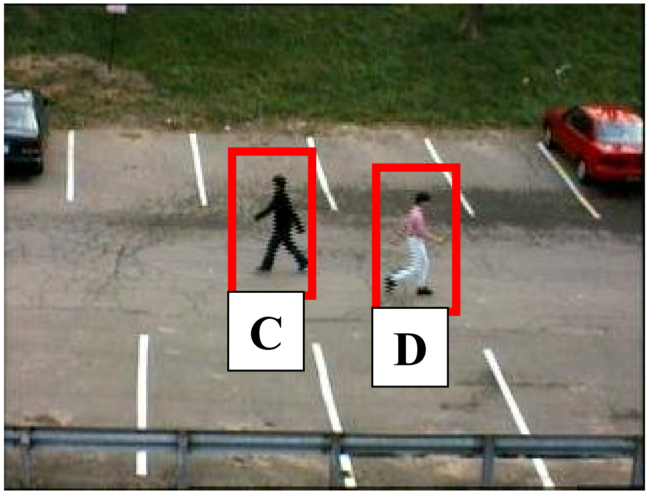
	ai1	
1	3.0	
2	5.0	
3	6.0	
4	9.0	max

a_{i1} = score for matching observation i to track 1

Choose best match $a_{m1} = \max\{a_{11}, a_{21},a_{31},a_{41}\}$

Example of Data Association After Merge and Split





$\Delta(A,C) = 2.03$
$\Delta(A,D) = 0.39$ ●

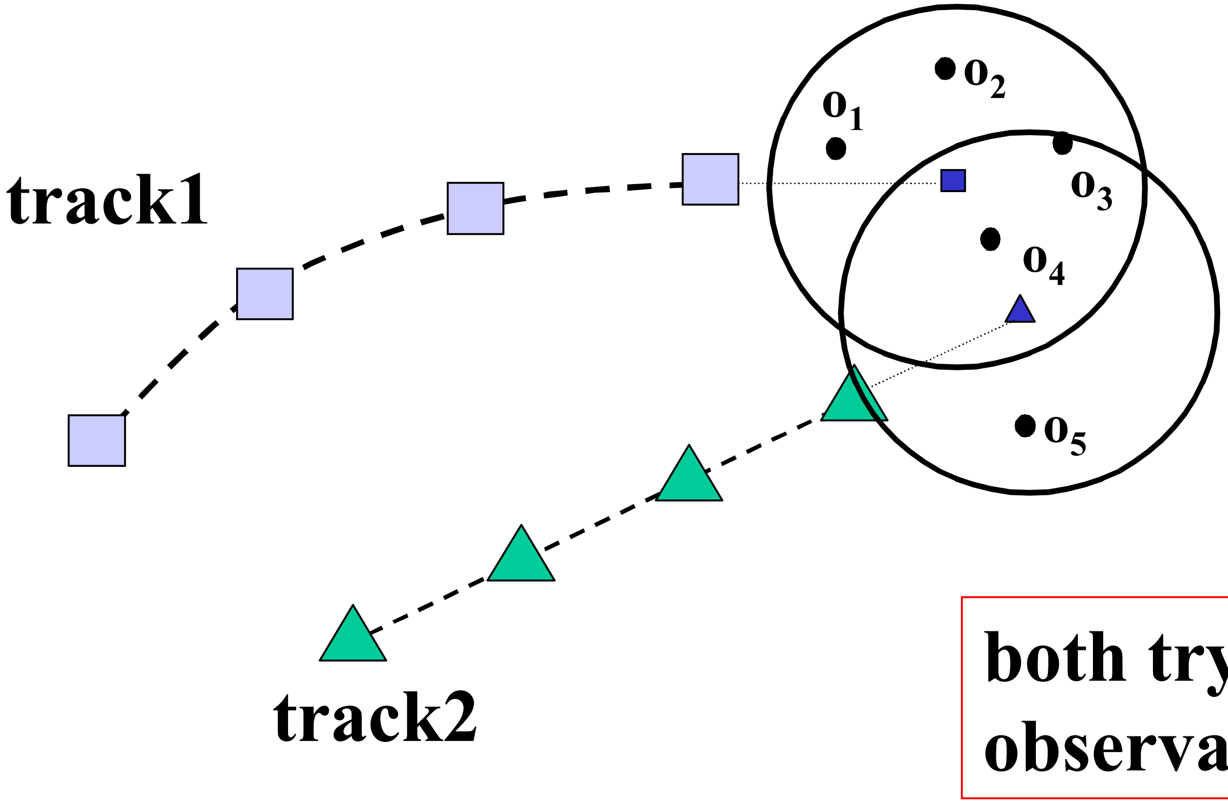
A -> D

$\Delta(B,C) = 0.23$ ●
$\Delta(B,D) = 2.0$

B -> C

Global Nearest Neighbor (GNN)

Problem: if do independently for each track, could end up with contention for the same observations.



	ai1	ai2
1	3.0	
2	5.0	
3	6.0	1.0
4	9.0	8.0
5		3.0

both try to claim observation o_4

Linear Assignment Problem

We have N objects in previous frame and M objects in current frame. We can build a table of match scores m(i,j) for i=1...N and j=1...M. For now, assume M=N.

	1	**2**	**3**	**4**	**5**
1	0.95	0.76	0.62	0.41	0.06
2	0.23	0.46	0.79	0.94	0.35
3	0.61	0.02	0.92	0.92	0.81
4	0.49	0.82	0.74	0.41	0.01
5	0.89	0.44	0.18	0.89	0.14

problem: choose a 1-1 correspondence that maximizes sum of match scores.

Assignment Problem

Mathematical definition. Given an NxN array of benefits $\{X_{ai}\}$, determine an NxN permutation matrix M_{ai} that maximizes the total score:

maximize:
$$E = \sum_{a=1}^{N} \sum_{i=1}^{N} M_{ai} X_{ai}$$

subject to:
$$\left.\begin{aligned} \forall i \ \textstyle\sum_{a=1}^{A} M_{ai} &= 1 \\ \forall a \ \textstyle\sum_{i=1}^{I} M_{ai} &= 1 \\ M_{ai} &\in \{0,1\} \end{aligned}\right\} \text{constraints that say M is a permutation matrix}$$

The permutation matrix ensures that we can only choose one number from each row and from each column.

Example:

5x5 matrix of match scores

0.95	0.76	0.62	0.41	0.06
0.23	0.46	0.79	0.94	0.35
0.61	0.02	0.92	0.92	0.81
0.49	0.82	0.74	0.41	0.01
0.89	0.44	0.18	0.89	0.14

working from left to right, choose one number from each column, making sure you don’t choose a number from a row that already has a number chosen in it.

How many ways can we do this?

$$5 \times 4 \times 3 \times 2 \times 1 = 120 \quad \text{(N factorial)}$$

$$\begin{array}{ccccc} \boxed{0.95} & 0.76 & 0.62 & 0.41 & 0.06 \\ 0.23 & \boxed{0.46} & 0.79 & 0.94 & 0.35 \\ 0.61 & 0.02 & \boxed{0.92} & 0.92 & 0.81 \\ 0.49 & 0.82 & 0.74 & \boxed{0.41} & 0.01 \\ 0.89 & 0.44 & 0.18 & 0.89 & \boxed{0.14} \end{array}$$

score: 2.88

$$\begin{array}{ccccc} 0.95 & 0.76 & 0.62 & \boxed{0.41} & 0.06 \\ \boxed{0.23} & 0.46 & 0.79 & 0.94 & 0.35 \\ 0.61 & 0.02 & \boxed{0.92} & 0.92 & 0.81 \\ 0.49 & \boxed{0.82} & 0.74 & 0.41 & 0.01 \\ 0.89 & 0.44 & 0.18 & 0.89 & \boxed{0.14} \end{array}$$

score: 2.52

$$\begin{array}{ccccc} 0.95 & \boxed{0.76} & 0.62 & 0.41 & 0.06 \\ 0.23 & 0.46 & 0.79 & \boxed{0.94} & 0.35 \\ 0.61 & 0.02 & 0.92 & 0.92 & \boxed{0.81} \\ 0.49 & 0.82 & \boxed{0.74} & 0.41 & 0.01 \\ \boxed{0.89} & 0.44 & 0.18 & 0.89 & 0.14 \end{array}$$

score: 4.14

A Greedy Strategy

```
Choose largest value and mark it
For i = 1 to N-1
   Choose next largest remaining value that isn't in a row/col already marked
End
```

$$\begin{matrix} \boxed{0.95} & 0.76 & 0.62 & 0.41 & 0.06 \\ 0.23 & 0.46 & 0.79 & \boxed{0.94} & 0.35 \\ 0.61 & 0.02 & \boxed{0.92} & 0.92 & 0.81 \\ 0.49 & \boxed{0.82} & 0.74 & 0.41 & 0.01 \\ 0.89 & 0.44 & 0.18 & 0.89 & \boxed{0.14} \end{matrix}$$

score: 3.77

not as good as our current best guess!

$$\begin{matrix} 0.95 & \boxed{0.76} & 0.62 & 0.41 & 0.06 \\ 0.23 & 0.46 & 0.79 & \boxed{0.94} & 0.35 \\ 0.61 & 0.02 & 0.92 & 0.92 & \boxed{0.81} \\ 0.49 & 0.82 & \boxed{0.74} & 0.41 & 0.01 \\ \boxed{0.89} & 0.44 & 0.18 & 0.89 & 0.14 \end{matrix}$$

score: 4.14

Is this the best we can do?

Some (possible) Solution Methods

maximize: $$E = \sum_{a=1}^{N} \sum_{i=1}^{N} M_{ai} X_{ai}$$

subject to: $$\forall i \;\; \sum_{a=1}^{A} M_{ai} = 1$$

$$\forall a \;\; \sum_{i=1}^{I} M_{ai} = 1$$

$$M_{ai} \in \{0, 1\}$$

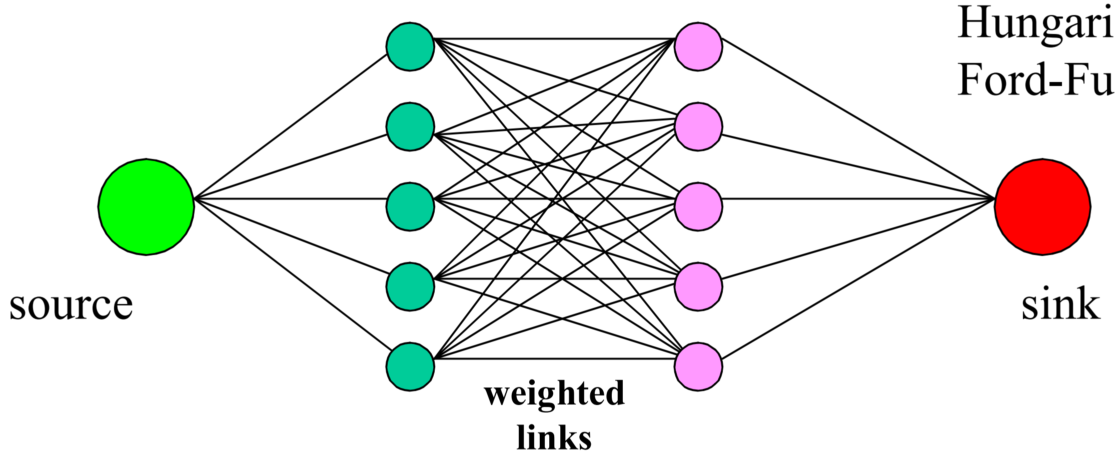
This has the form of a 0-1 integer linear program. Could solve using the simplex method. However, bad (exponential) worst-case complexity (0-1 integer programming is NP-hard)

Some (possible) Solution Methods

maximize: $$E = \sum_{a=1}^{N} \sum_{i=1}^{N} M_{ai} X_{ai}$$

subject to: $$\forall i \ \sum_{a=1}^{A} M_{ai} = 1$$
$$\forall a \ \sum_{i=1}^{I} M_{ai} = 1$$
$$M_{ai} \in \{0, 1\}$$

Can also be viewed as a maximal matching in a weighted bipartite graph, which in turn can be characterized as a max-flow problem.



Possible solution methods:
- Hungarian algorithm
- Ford-Fulkerson algorithm

Review: SoftAssign

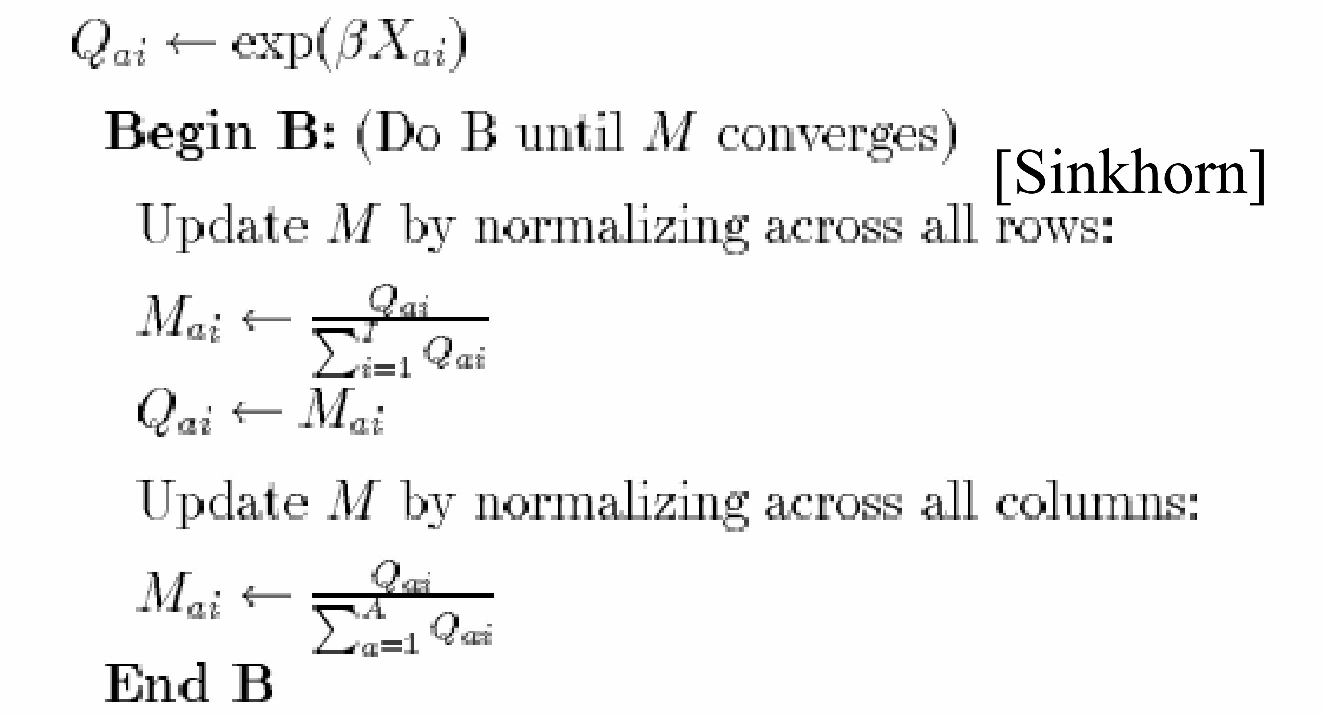
We are going to use an efficient approach called SoftAssign, based on the work of

> J. Kosowsky and A. Yuille. The invisible hand algorithm: Solving the assignment problem with statistical physics. *Neural Networks*, 7:477-490, 1994.

Main points:

- relax 0,1 constraint to be 0 <= Mai <=1
- init with Mai = exp(B*score). This ensures positivity and also spreads out scores as (B approaches infinity)
- perform repeat row and col normalizations to get a doubly stochastic matrix (rows and cols sum to 1)

SoftAssign



```
$Q_{ai} \leftarrow \exp(\beta X_{ai})$
  Begin B: (Do B until $M$ converges)
    Update $M$ by normalizing across all rows:
    $M_{ai} \leftarrow \frac{Q_{ai}}{\sum_{i=1}^{I} Q_{ai}}$
    $Q_{ai} \leftarrow M_{ai}$
    Update $M$ by normalizing across all columns:
    $M_{ai} \leftarrow \frac{Q_{ai}}{\sum_{a=1}^{A} Q_{ai}}$
  End B
```

[Sinkhorn]

In practive, should use an iterative version to avoid numerical issues with large B.

Why it works? Consider SoftMax

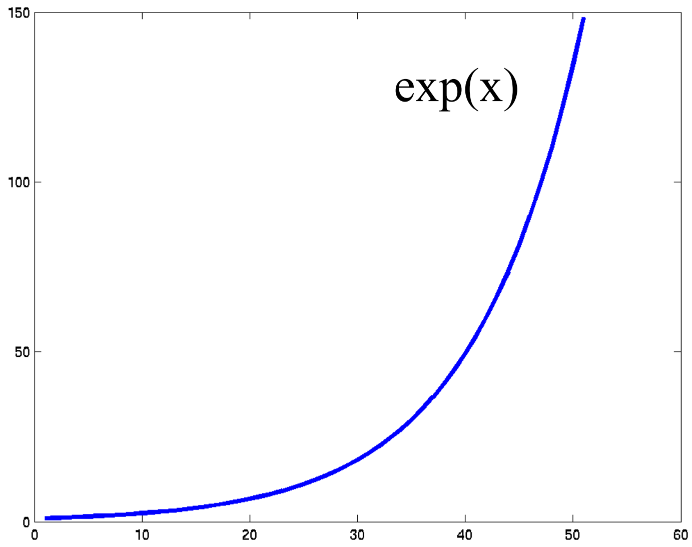
Softmas is a similar algorithm, but just operates on a single vector of numbers.

$$m_j = \frac{\exp(\beta X_j)}{\sum_{i=1}^{I} \exp(\beta X_i)}$$

Notes:

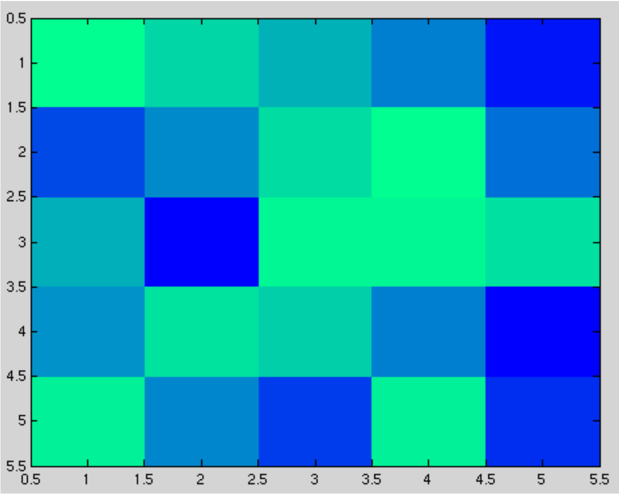
The exp() function serves to ensure that all numbers are positive (even negative number map to positive values through exp)

As B increases, the m_i associated with the max element approaches 1, while all other m_i values approach 0.

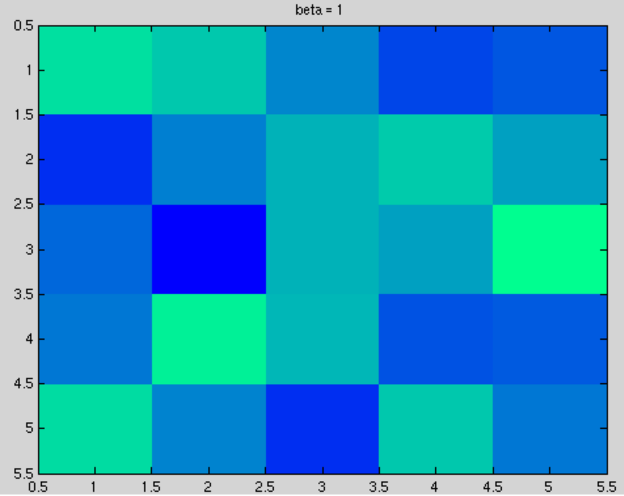


SoftAssign

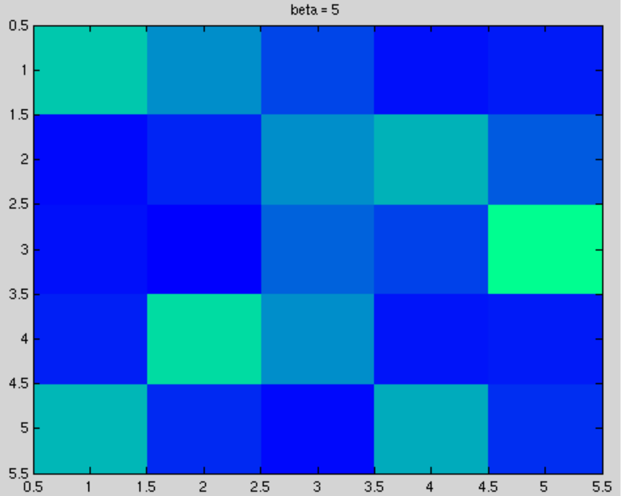
X_{ai} (benefits)



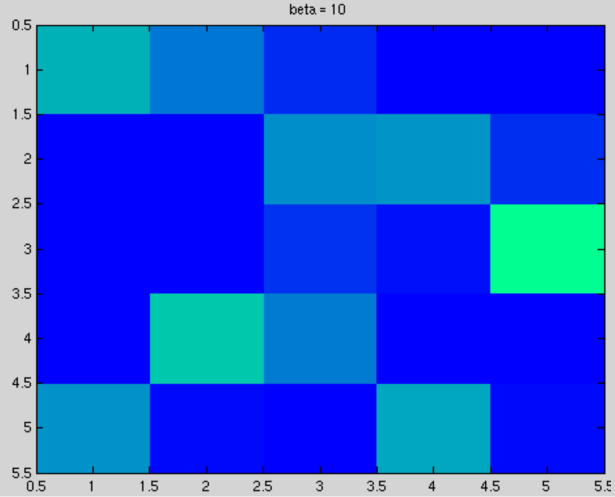
B=1



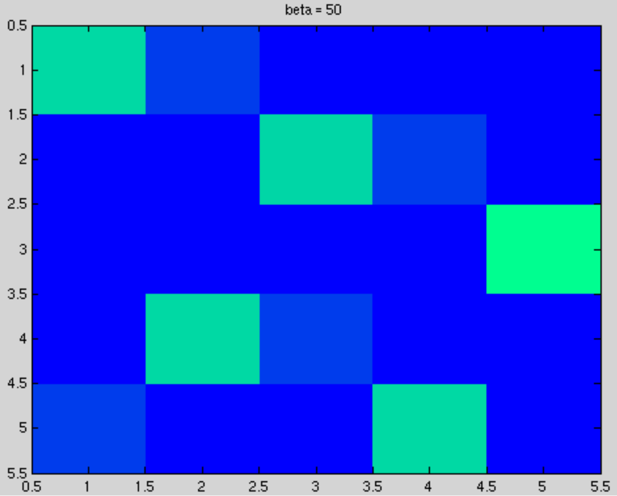
B=5



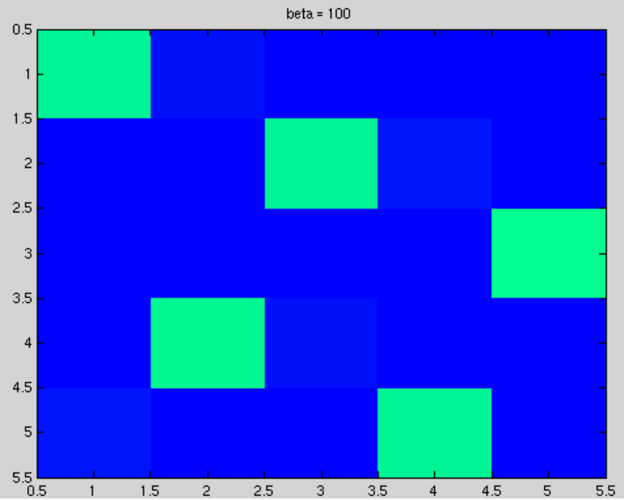
B=10



B=50

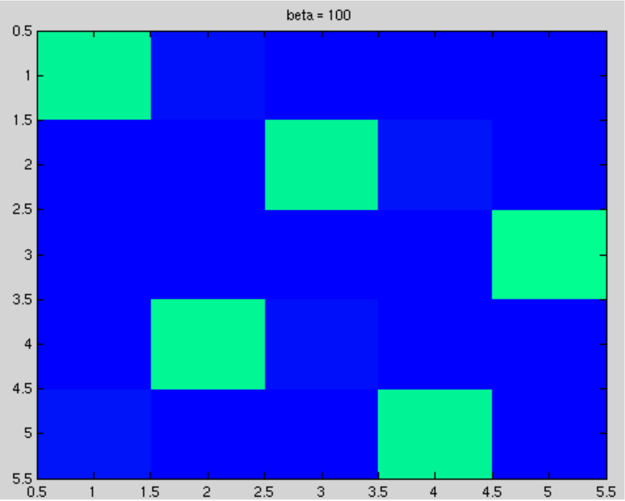


B=100



permutation matrix!

SoftAssign



permutation matrix
computed by SoftAssign

$$\begin{array}{ccccc}
\boxed{0.95} & 0.76 & 0.62 & 0.41 & 0.06 \\
0.23 & 0.46 & \boxed{0.79} & 0.94 & 0.35 \\
0.61 & 0.02 & 0.92 & 0.92 & \boxed{0.81} \\
0.49 & \boxed{0.82} & 0.74 & 0.41 & 0.01 \\
0.89 & 0.44 & 0.18 & \boxed{0.89} & 0.14
\end{array}$$

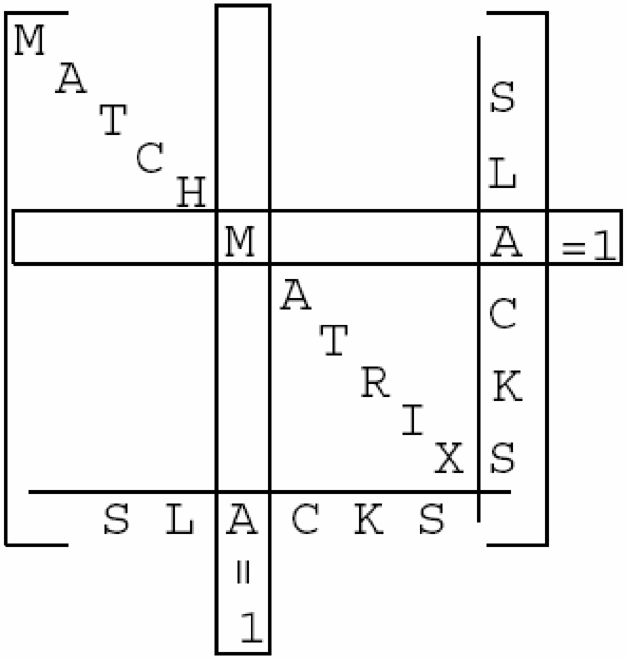
score: 4.26

In this example, we can exhaustively search all 120 assignments.
The global maximum is indeed 4.26

Handling Missing Matches

Typically, there will be a different number of tracks than observations. Some observations may not match any track. Some tracks may not have any observations.

Introduce one row and one column of "slack variables" to absorb any outlier mismatches.


M A T C H
M
A T R I X
S L A C K S
=1
S L A C K S
=
1