9.913 Pattern Recognition for Vision

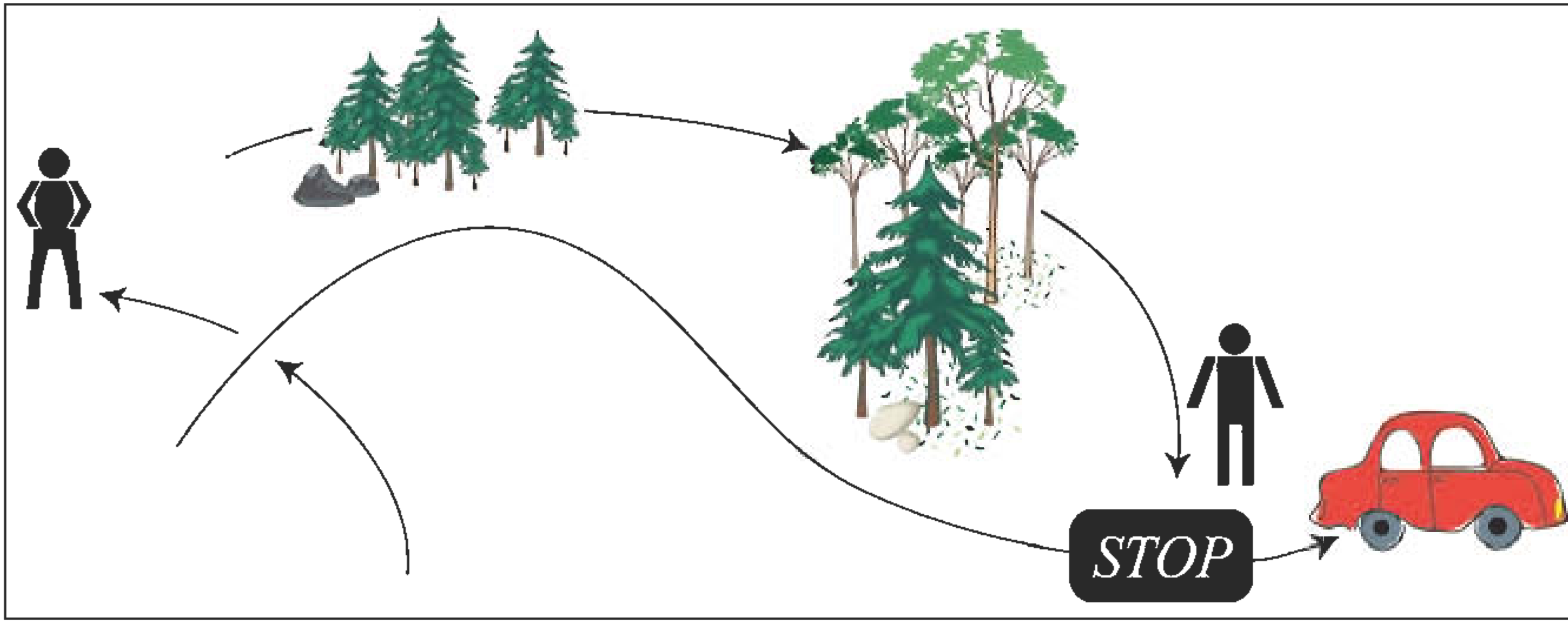
Class XIII, Motion and Gesture

Yuri Ivanov

TOC

- Movement – Activity – Action
- View-based representation
- Sequence comparison
- Hidden Markov Models
- Hierarchical representations

From Tracking to Classification



How do we describe that?

How do we classify that?

Figure by MIT OCW.

From Tracking to Classification

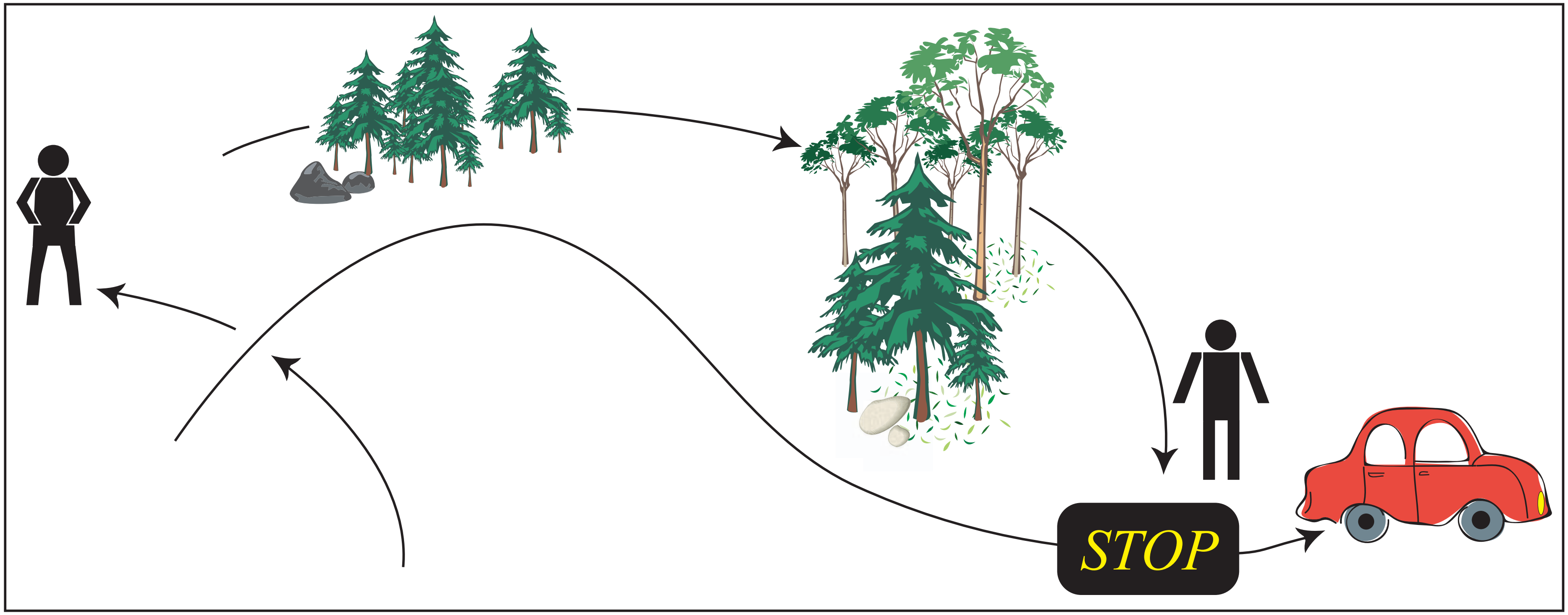


Figure by MIT OCW.

How do we describe that?
How do we classify that?

Sequence Analysis

- We might want to ask:
  - Is <it> doing something meaningful?
  - What exactly?
  - How does it do it?
    - How fast – e.g. conducting
    - How accurately – e.g. dance instruction
    - What style?

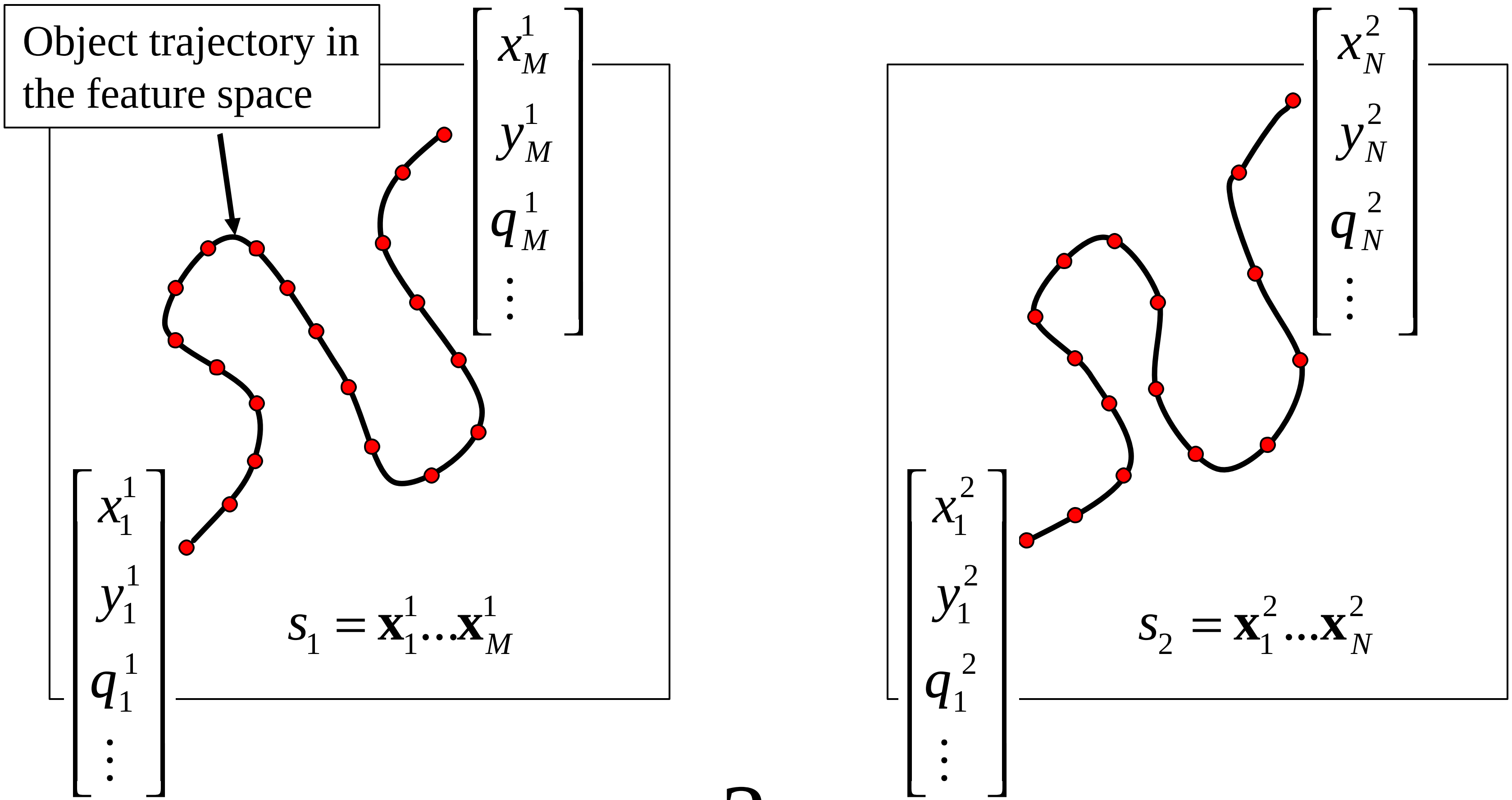
- That leads us to a sequence analysis

Motion Taxonomy

- Movement
  - Primitive motion
  - Self-evidential, is "what it looks like"
- Activity
  - Requires explicit sequence model
- Action
  - Requires contextual information
  - Requires relational information
  - And many other things…

Images removed due to copyright considerations. Please see: Bobick, A. "Movement, Activity, and Action: The Role of Knowledge in Perception of Motion." *Phyl Trans of Royal Statistical Society* 352 (1997).

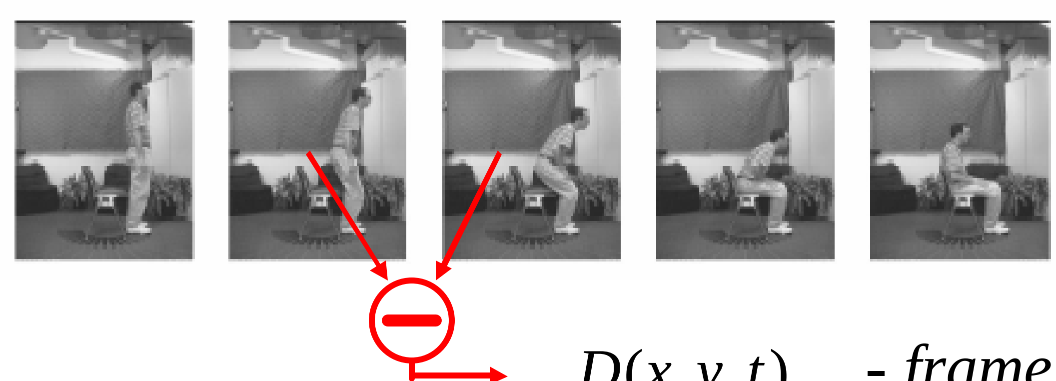
Basic Problem



$$S_1 \overset{?}{<>} S_2$$

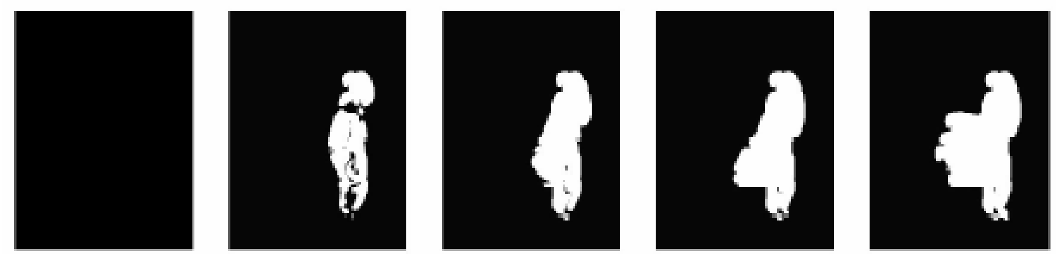
Motion Energy Image

First idea –implicit representation of time



$D(x,y,t)$ - *frame differencing*

Sum the differences over the last τ frames:

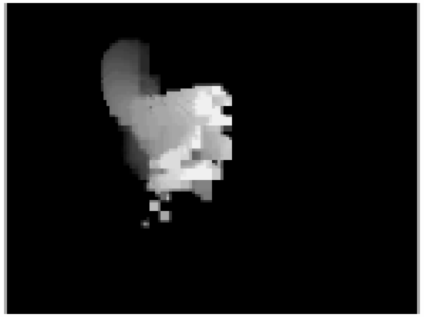


$$E_t(x,y,t) = \bigcup_{i=0}^{t-1} D(x,y,t-i)$$ - *WHERE motion happened*

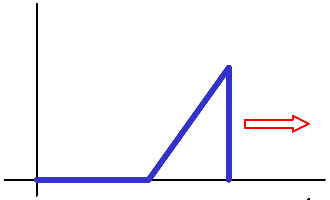
Photographs and figures from: Bobick, A., and J. Davis. "The Representation and Recognition of Action Using Temporal Templates." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 23, no. 3 (2002). Courtesy of IEEE, A. Bobick, and J. Davis. Copyright 2002 IEEE. Used with Permission.

Motion History Image

Step two: include temporal information

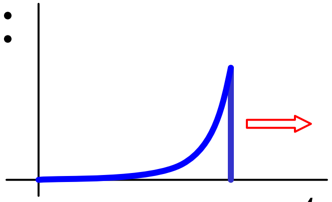
$$H_t(x,y,t) = \begin{cases} t & \text{if } D(x,y,t) = 1 \\ \max(0, H_t(x,y,t-1)-1) & \text{otherwise} \end{cases}$$



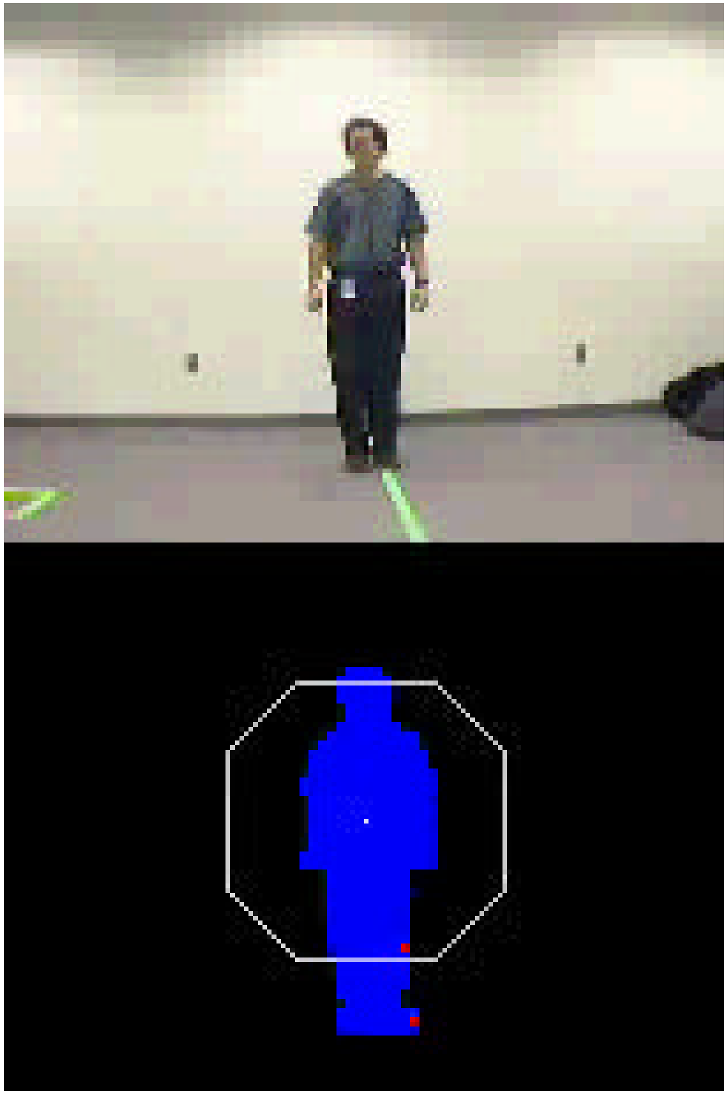
- *HOW motion happened*

Aside - you can compute a similar measure recursively:

$$H_t(x,y,t) = H_t(x,y,t-1) + a\,(D(x,y,t) - H_t(x,y,t-1))$$



Illustration



OpenCV – Intel Open source Computer Vision Library

Classification

Feature vector:

x = [7 *Hu moments* for MEI + 7 *Hu moments* for MHI]

RTS invariant shape descriptors (see the end of notes)

With the usual Gaussian assumption on distribution of x:

$$m_w = E[x_w]; \quad \Sigma_w = E[(x_w - m_w)^2]$$

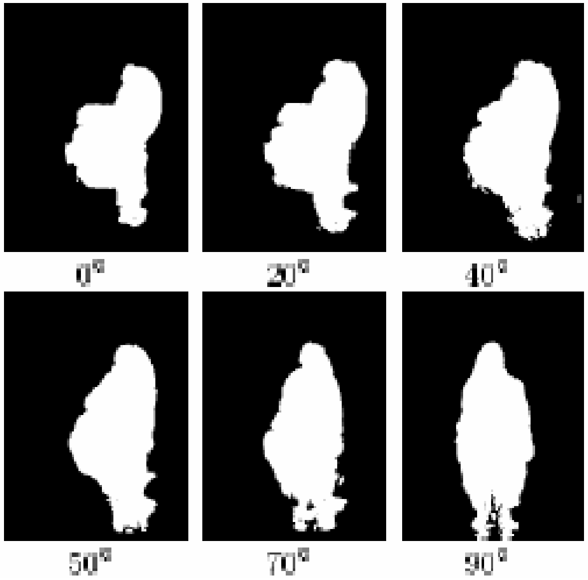
Then the class, ω :

$$w = \operatorname{argmin}\left[(x - m_w)^T \Sigma_w^{-1} (x - m_w) \right]$$

Multi-View Recognition

The model is replicated for a discrete number of views:

For $q = \{0^\circ \ldots 90^\circ\}$



$$V(w_i) = \min_q \left[\left(x - m_{w_i}^q \right)^T \Sigma_{w_i}^{-1,q} \left(x - m_{w_i}^q \right) \right]$$

Closest member of each class

$$w = \operatorname{argmin} \left[V(w_i) \right]$$

Nearest class

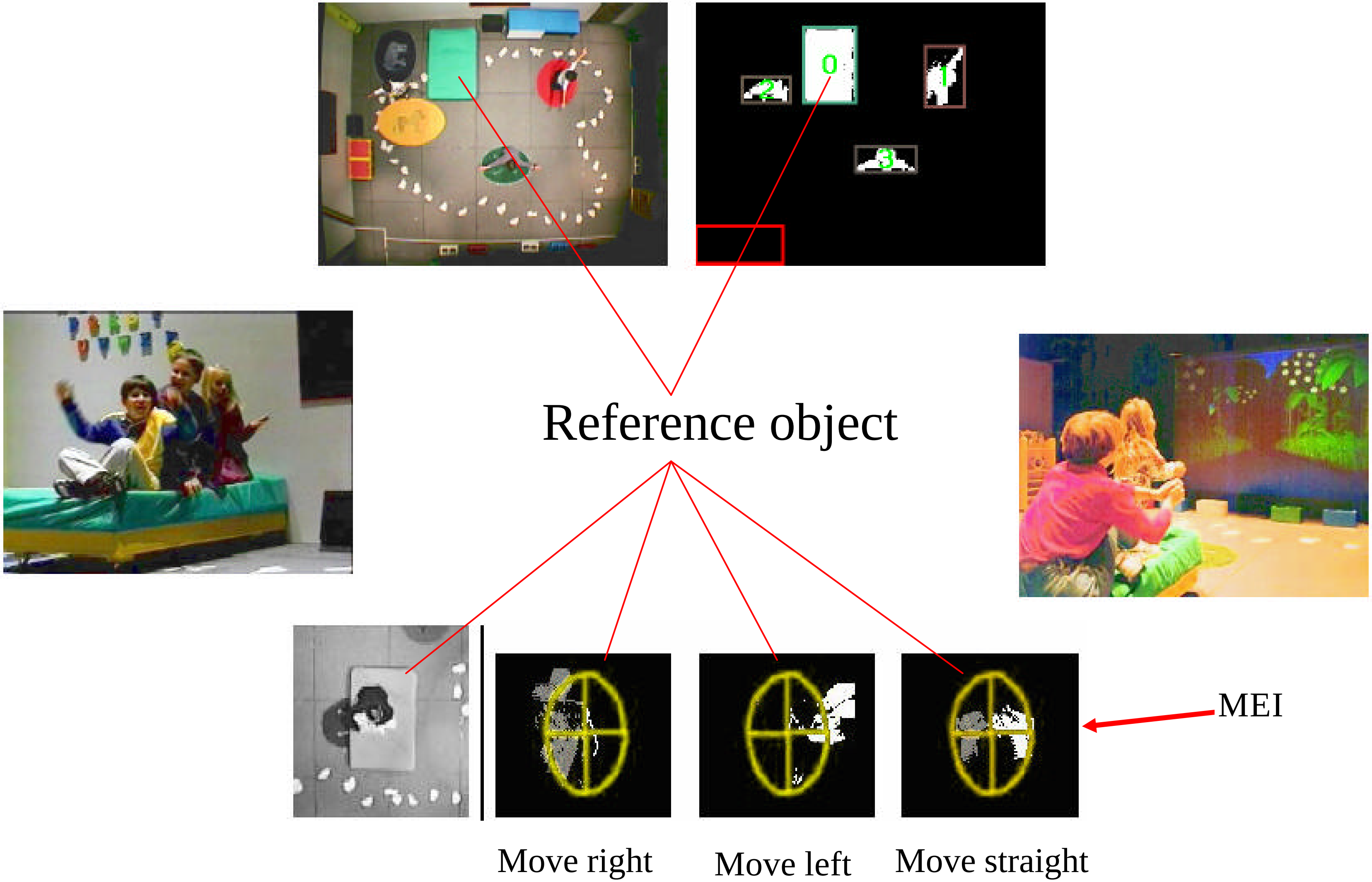
Example Application

KidsRoom
- Interactive story
- Autonomous system
- Narration is controlled
- Input from cameras and mike

Image removed due to copyright considerations. See: http://whitechapel.media.mit.edu/vismod/demos/kidsroom/kidsroom.html

- Visual events:
  - position
  - motion energy
  - motion direction
  - gross body motion

Motion Energy

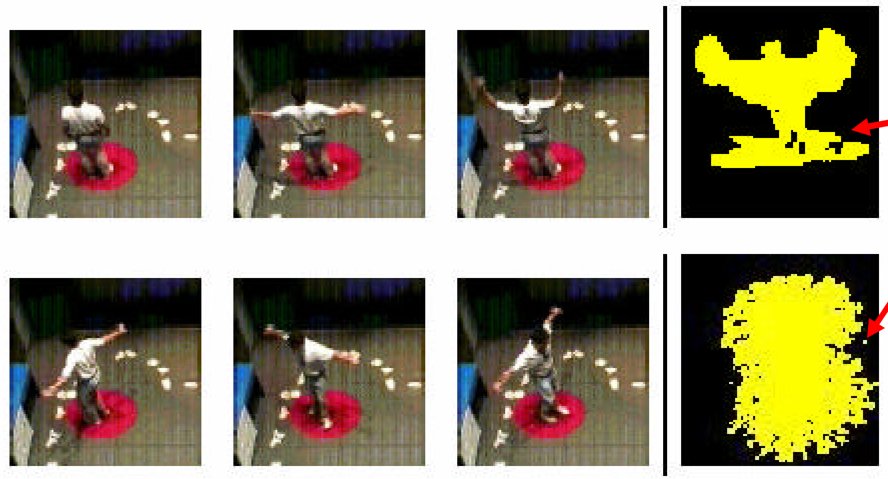


Vismod Tech Report # 398

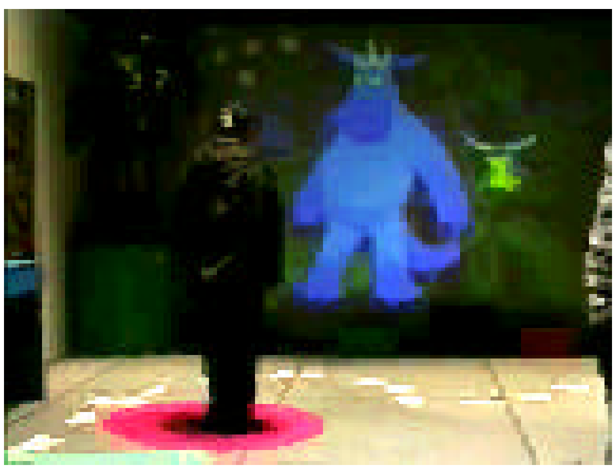
Movement Classification

"Flap"

"Spin"



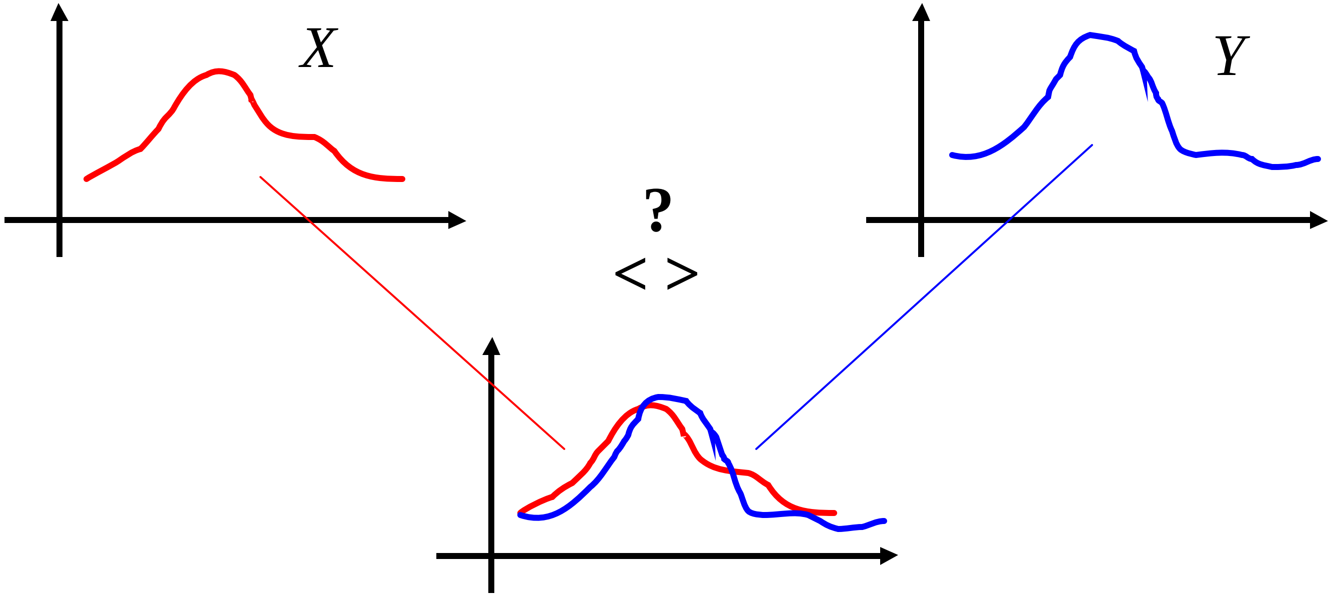
MEI/MHI



Last game sequence

Temporal Alignment

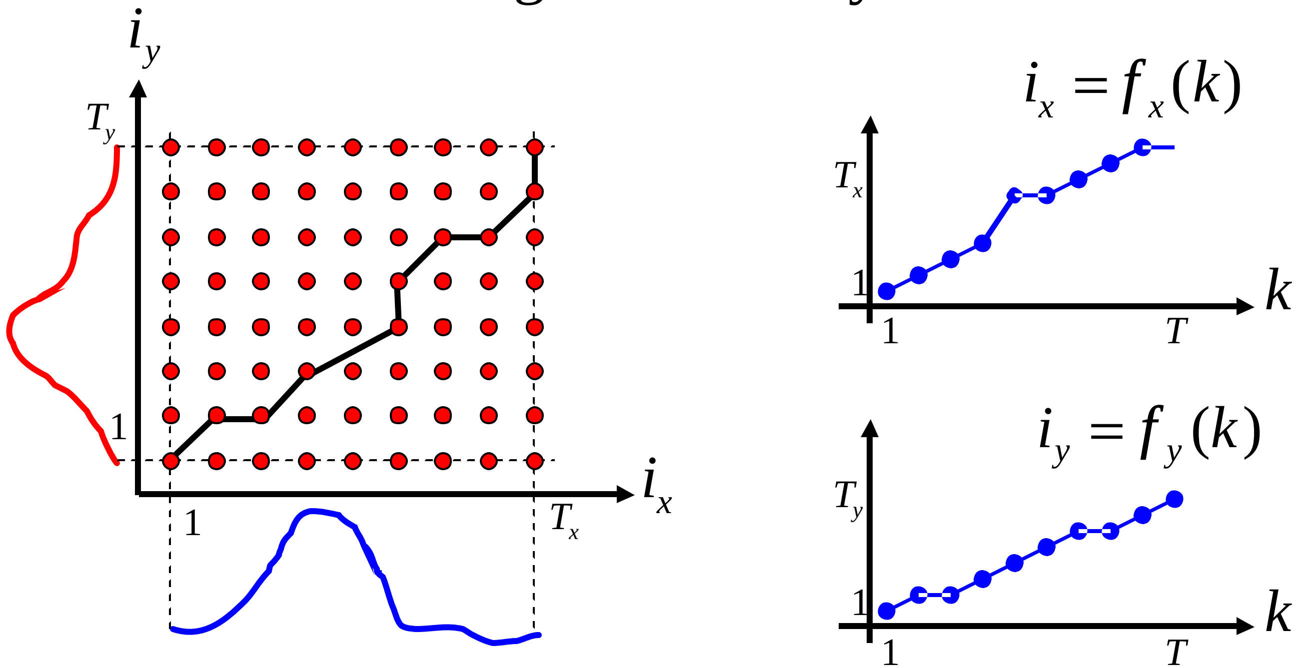
Another idea – temporal alignment



If sequences are aligned to a common time axis, then we can treat them as vectors

Temporal Alignment

Find re-indexing sequences i_x and i_y that align X and Y to a common time axis k while minimizing dissimilarity.

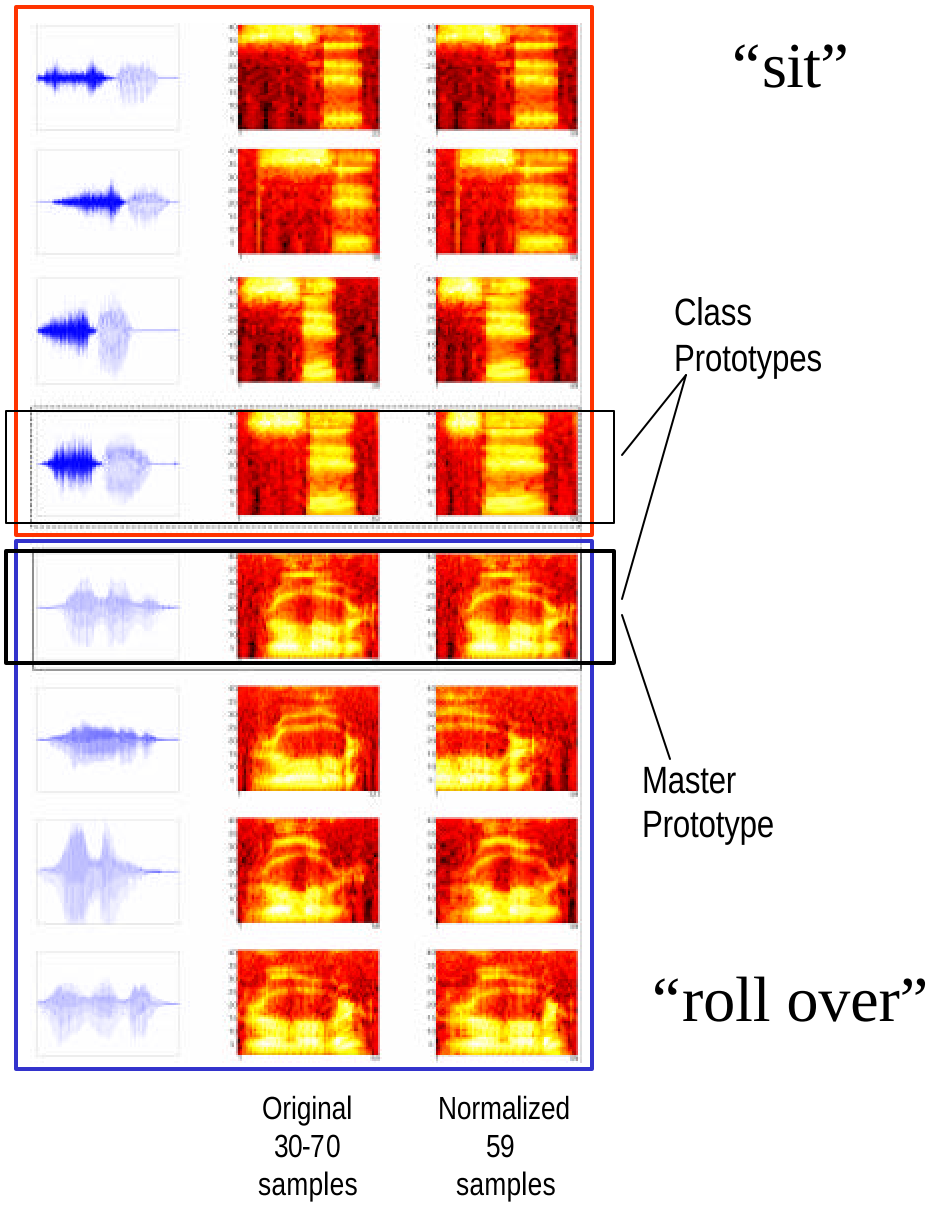


One solution – Dynamic Time Warp algorithm

$$E = \frac{1}{M_f} \sum_{n=1}^{T} \left\{ m(n) \left(s_x\left[i_x(n)\right] - s_y\left[i_y(n)\right] \right)^2 \right\}$$

Global normalization Local weighting

Example: Utterance Classification



Time normalization:

1. Find the least distortion prototype in each class
2. Pick the longest one
3. Warp all data to it
4. Train classifier

Example: Utterance Classification

Alternative - pair-wise alignment:

$$\text{SVM:} \quad f(x) = \sum_{i=1}^{N} a_i y_i K(\mathbf{x}, \mathbf{x}_i) + b$$

1. Compute the symmetric DTW between all pairs

$$d_{ij} = \frac{D(s_i, s_j) + D(s_j, s_i)}{2}$$

2. Compute an RBF Kernel

$$K(s_i, s_j) = \exp(-g\, d_{ij})$$

Danger: *K* might not be a proper kernel matrix – need to regularize

Example: Utterance Classification

Japanese Vowel Set (UCI Machine Learning Repository):

- Speaker identification task
- 9 speakers
- saying the same Japanese vowel
- features 12 cepstral coefficients
- each utterance – 7-30 samples
- 340 training examples
- 240 testing examples

	Accuracy
KNN	94.60%
MCC	94.10%
HMM	96.20%
SVM	98.20%
DynSVM	98.20%

Hidden Markov Model (HM&M)

Yet another idea:

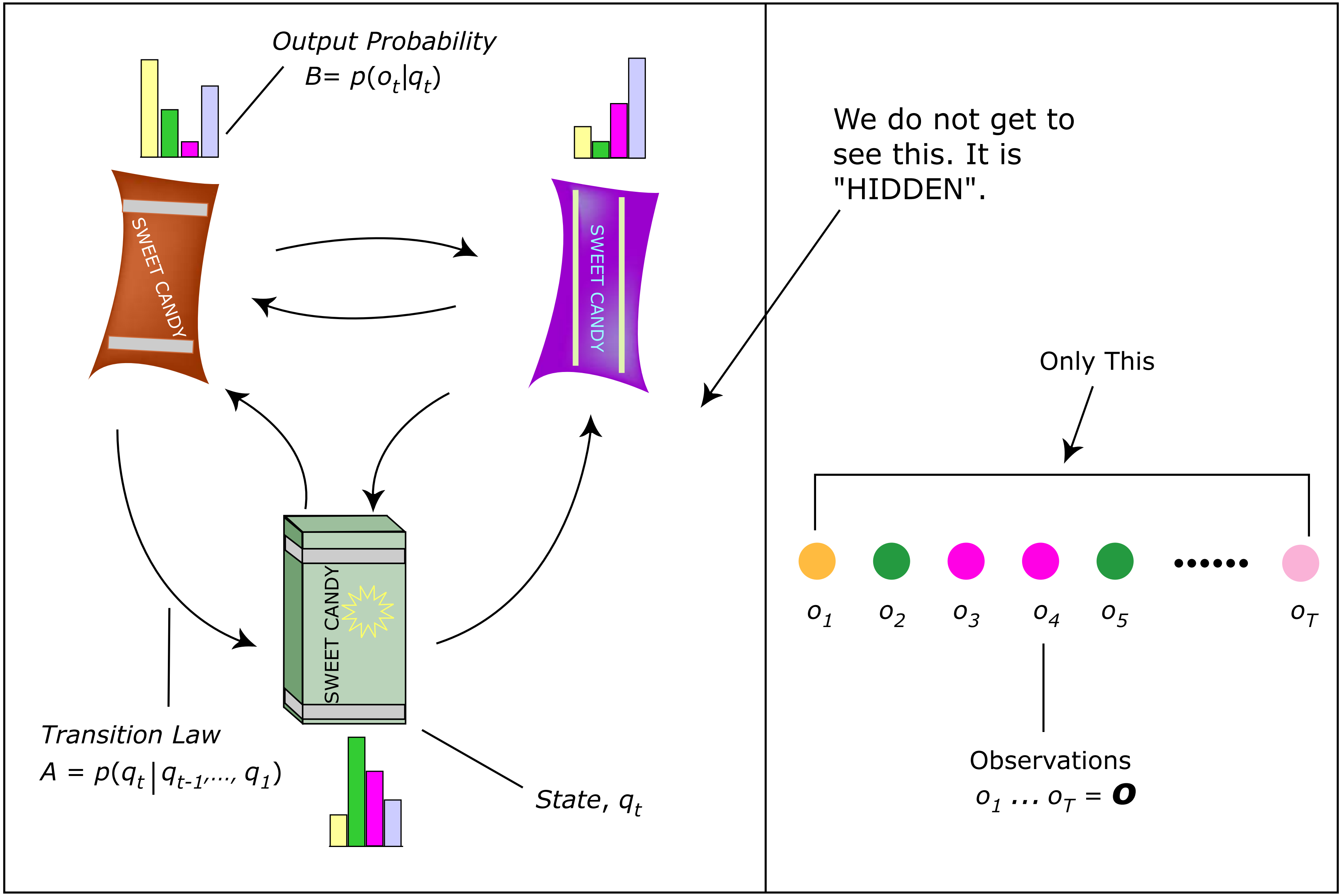


Figure by MIT OCW.

Hidden Markov Model (HM&M)

Yet another idea:

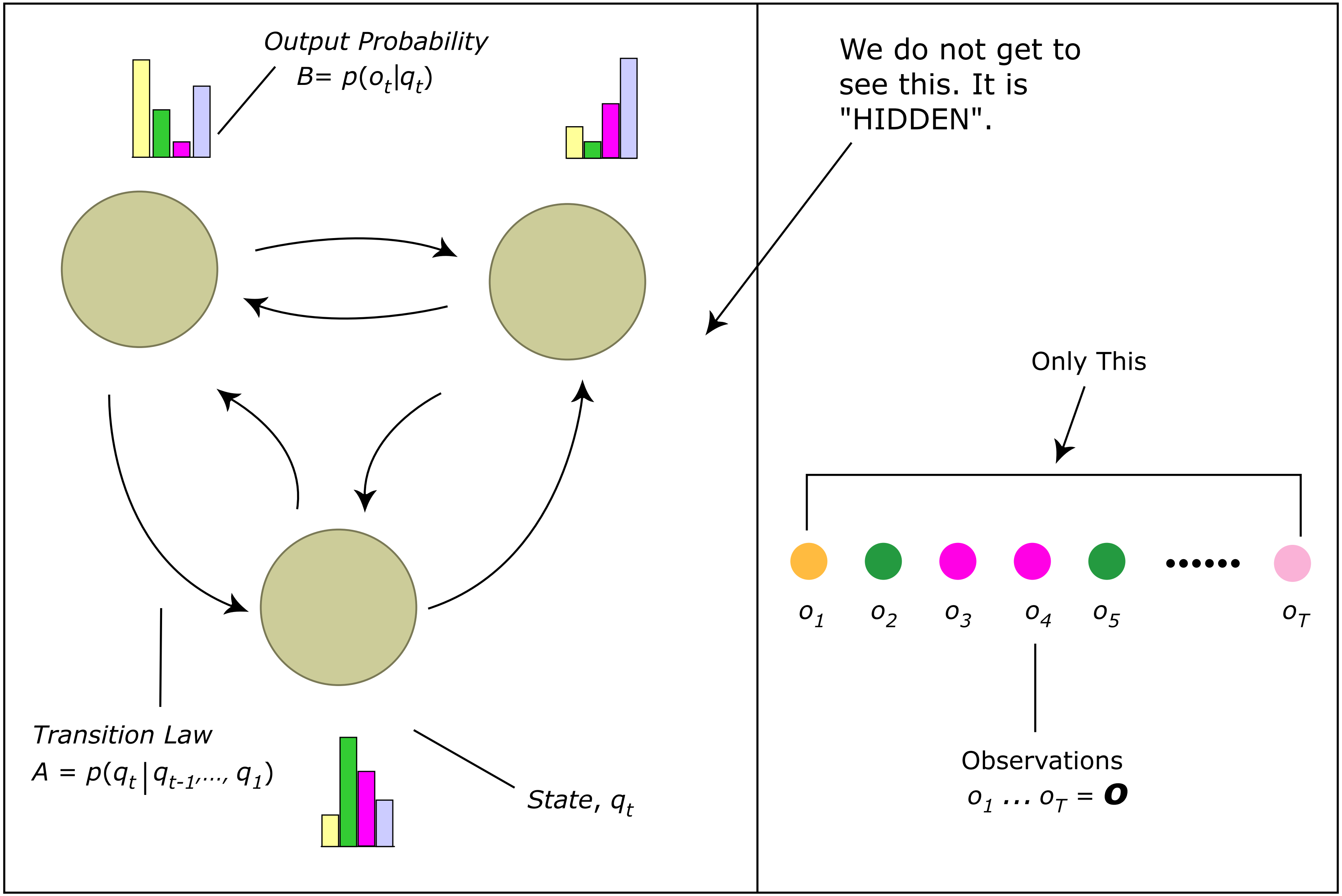


Figure by MIT OCW

Another view – "Graphical Model":

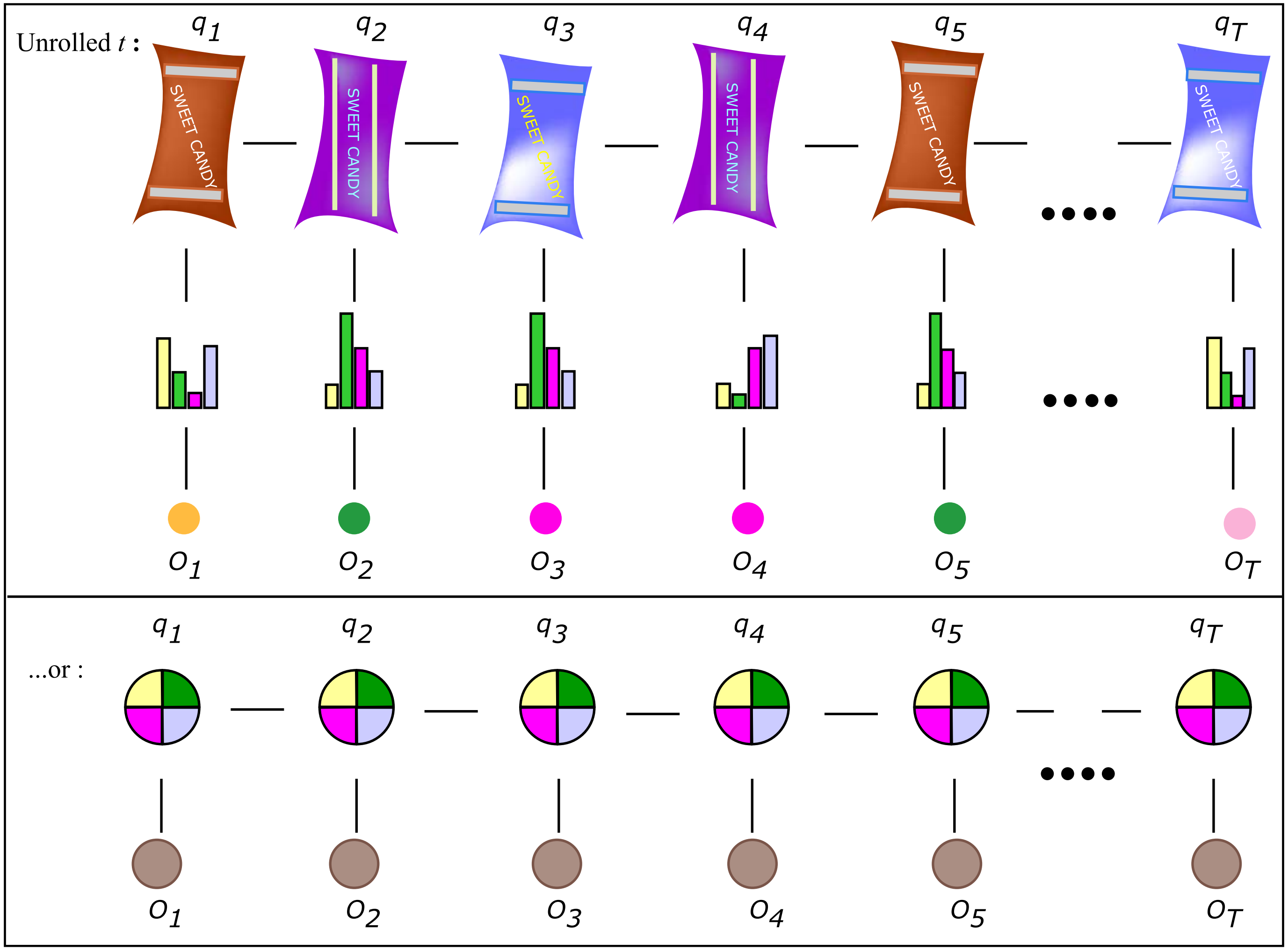


Figure by MIT OCW.

Components of an HMM

$$l = \{p , A, B\}$$

1) π - probability of starting from a particular state

$$\pi_i = p(q_1=i)$$

$$\sum_{i=1}^{N} p_i = 1$$

2) A - probability of moving to a state, given the history

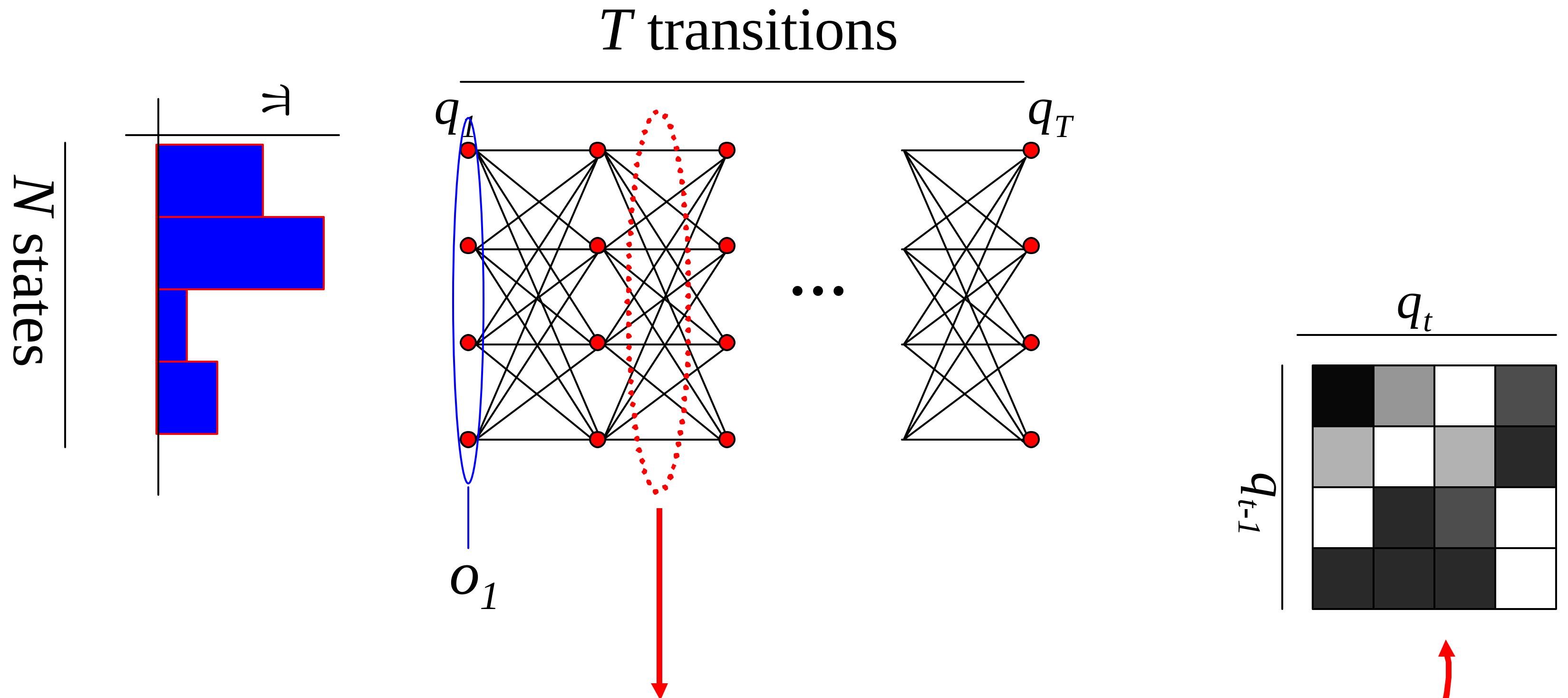
$$a_{ij}=p(q_t=i|\, q_{t-1}=j)$$ – Markov assumption

$$\sum_{j=1}^{N} a_{ij} = 1$$

3) B - probability of outputting a particular observation from a given state: $b_i(o)=p(o_t|q_t=i)$

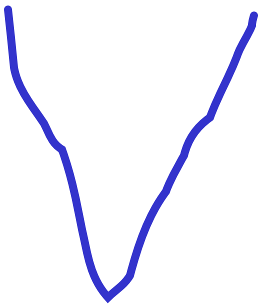
$$\int b_i(x)dx = 1$$

HMM in Pictures



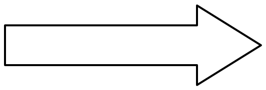
- $A = p(q_t | q_{t-1},\ldots, q_1) = p(q_t | q_{t-1})$ – Markov assumption
- $p(q_t | q_{t-1})$ is independent of t – stationary => a matrix

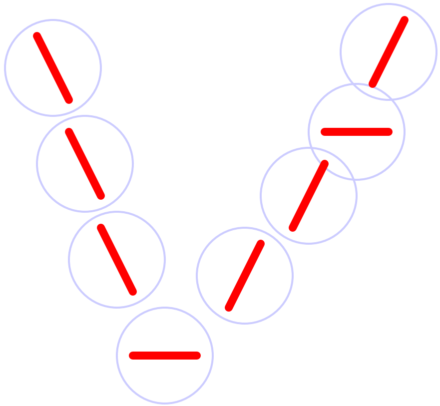
HMM Example



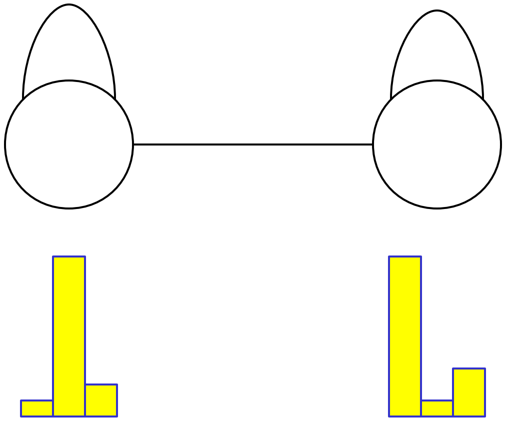
Input

Alphabet

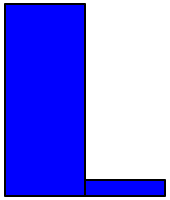




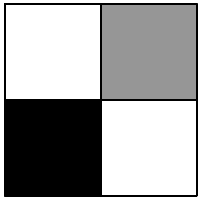
How HMM sees it



Output distributions



Initial state



Transition matrix

Three Tasks of HMM

1. Given a sequence of observations find a probability of it given the model, $p(O|\lambda)$
2. Given a sequence of observations recover a sequence of states, $P(q|O,\lambda)$
3. Given a sequence, estimate parameters of the model

Problem I – Probability Calculation

Take I – brute force:

Given: $\mathbf{O} = (o_1,\ldots,o_T)$

Calculate: $P(\mathbf{O} \mid l)$

Marginalize:

$$P(\mathbf{O} \mid l) = \sum_{\forall \mathbf{q}} P(\mathbf{O},\mathbf{q} \mid l) = \sum_{\forall \mathbf{q}} \boxed{P(\mathbf{O} \mid \mathbf{q}, l)}\boxed{P(\mathbf{q} \mid l)}$$

$$\boxed{P(\mathbf{O} \mid \mathbf{q}, l)} = b_{q_1}(o_1) b_{q_2}(o_2) \ldots b_{q_T}(o_3) \qquad \boxed{P(\mathbf{q} \mid l)} = p_{q_1} a_{q_1 q_2} a_{q_2 q_3} \ldots a_{q_{T-1} q_T}$$

$$\boxed{P(\mathbf{O} \mid \mathbf{q}, l)}\boxed{P(\mathbf{q} \mid l)} = p_{q_1} b_{q_1}(o_1) a_{q_1 q_2} b_{q_2}(o_2) a_{q_2 q_3} b_{q_3}(o_3) \ldots a_{q_{T-1} q_T} b_{q_T}(o_T)$$

N states, *T* transitions => $|\mathbf{q}| = \mathrm{N}^\mathrm{T}$!!!!

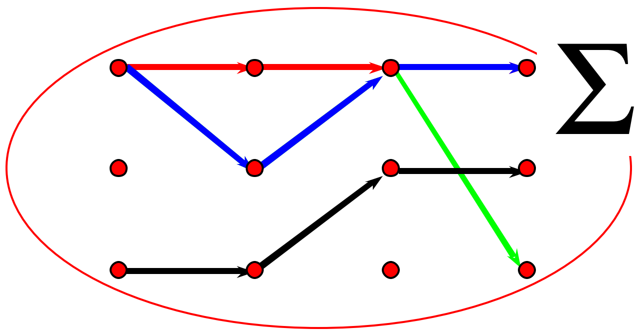
N=5, T=100 => $2TN^T = 2 * 100 * 5^{100} \sim 10^{72}$ computations

$65536 * 10^{72}$ particles in the universe

Try Again

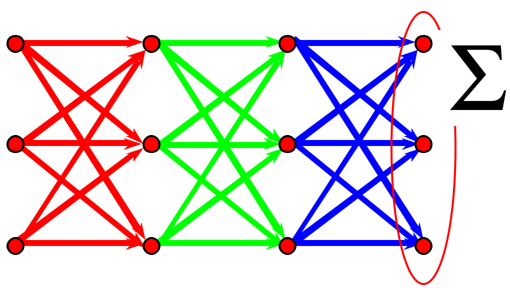
$$P(\mathbf{O} \mid l) = \sum_{\forall \mathbf{q}} P(\mathbf{O}, \mathbf{q} \mid l)$$

$$= \sum_{\mathbf{q}=\mathbf{q}_1}^{\mathbf{q}_{10^{72}}} p_{q(1)} b_{q(1)}(o_1) a_{q(1)q(2)} b_{q(2)}(o_2) a_{q(2)q(3)} b_{q(3)}(o_3) \ldots a_{q(T-1)q(T)} b_{q(T)}(o_T)$$



$$\approx 2TN^T$$

$$= \sum_m \boxed{\sum_l \cdots \boxed{\sum_j \boxed{\sum_i \boxed{p_i b_i(o_1)} a_{ij} b_j(o_2)} a_{jk} b_k}(o_3) \ldots a_{lm} b_m(o_T)}$$



$$\approx N^2 T$$

Problem I – Probability Calculation

Take II – forward procedure:

Define a "forward variable", α

$$\alpha_t(i) = P(o_1 o_2 \ldots o_t, q_t = i \mid \lambda)$$

- probability of seeing the string up to *t* and ending up in state *i*

1. Initialize

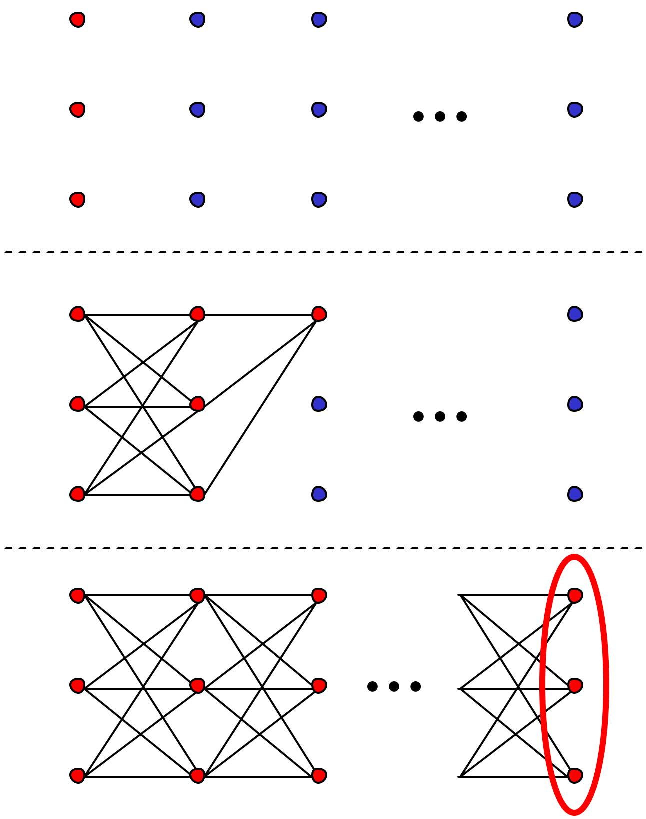
$$\alpha_1(i) = \pi_i b_i(o_1)$$

2. Induce

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(o_{t+1})$$

3. Terminate

$$P(\mathbf{O} \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i)$$



While We Are At It…

Define a "backward variable", β

$$b_t(i) = P(o_{t+1}o_{t+2}...o_T \mid q_t = i, l)$$

- probability of seeing the rest of the string after *t* and after visiting state *i* at *t*

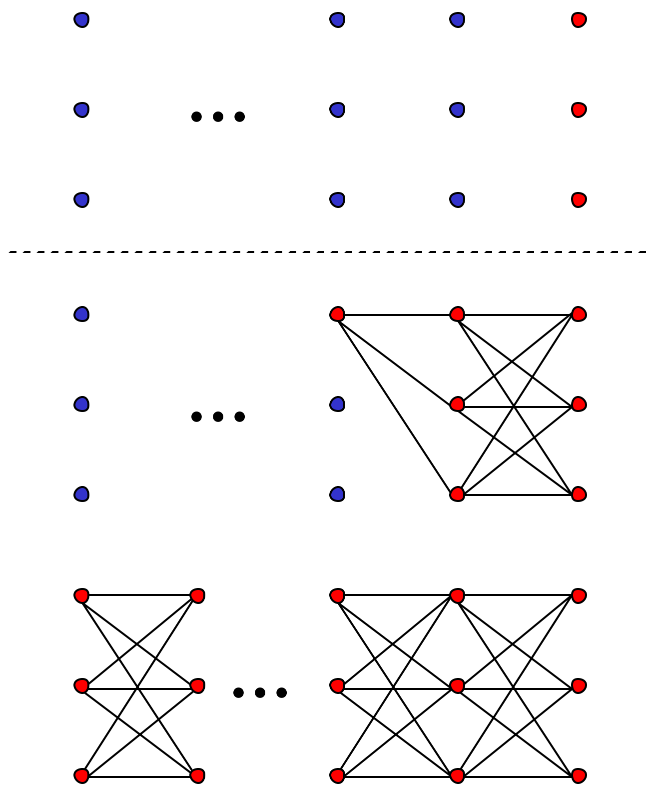
1. Initialize

$$b_T(i) = 1$$

2. Induce

$$b_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) b_{t+1}(j)$$

3. Terminate



Task II – Optimal State Sequence

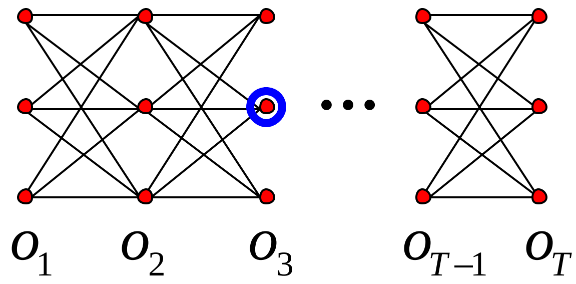
"Optimality" - maximum probability of being in a state *i* at time *t*.

Given: $\mathbf{O} = (o_1, \ldots, o_T)$

Find: $q_t = \underset{q}{\operatorname{argmax}} P(q_t \mid \mathbf{O}, l)$

by Bayes rule

$$P(q_t = i \mid \mathbf{O}) = \frac{P(\mathbf{O}, q_t = i)}{\sum_{i=1}^{N} P(\mathbf{O}, q_t = i)}$$



$$P(\mathbf{O}, q_t) = P(o_1 \ldots o_t, o_{t+1} \ldots o_T, q_t) = P(o_1 \ldots o_t, q_t) P(o_{t+1} \ldots o_T \mid o_1 \ldots o_t, q_t)$$

$$= P(o_1 \ldots o_t, q_t) P(o_{t+1} \ldots o_T \mid q_t) = a_t b_t$$

State Posterior

So,

$$P(q_t = i \mid \mathbf{O}) = \frac{P(\mathbf{O}, q_t = i)}{\sum_{j=1}^{N} P(\mathbf{O}, q_t = j)} = \frac{a_t(i) b_t(i)}{\sum_{j=1}^{N} a_t(j) b_t(j)} = \boxed{g_t(i)}$$

1. Forward pass – compute α matrix $\approx N^2 T$
2. Backward pass – compute β matrix $\approx N^2 T$
3. Multiply element-by element NT
4. Normalize columns $\approx N^2 T$

What's the problem?

Inconsistent paths – some might not even be allowed

But not entirely useless! We will need it later.

Task II – Viterbi Algorithm

"Optimality" – *single* maximum probability path.

Given:	$\mathbf{O} = (o_1, \ldots, o_T)$
Find:	$\underset{\mathbf{q}}{\operatorname{argmax}} P(\mathbf{q} \mid \mathbf{O}, l)$

Define:

$$d_t(i) = \max_{q_1 q_2 \ldots q_{t-1}} P(q_1 \ldots q_{t-1}, q_t = i, o_1 \ldots o_t)$$

Max prob. path so far

By the optimality principle (Bellman, '57):

$$d_{t+1}(j) = \left[\max_i d_t(i) a_{ij} \right] b_j(o_{t+1})$$

Just need to keep track of max probability states along the way

Task II – Viterbi Algorithm (cont.)

1. Initialize

$$d_1(i) = p_i b_i(o_1) \qquad 1 \le i \le N$$

$$y_1(i) = 0$$

Housekeeping variable

2. Recurse

$$d_t(j) = \max_{1 \le i \le N}\left[d_{t-1}(i)a_{ij}\right]b_j(o_t) \qquad \begin{array}{c} 2 \le t \le T \\ 1 \le j \le N \end{array}$$

$$y_t(j) = \operatorname*{argmax}_{1 \le i \le N}\left[d_{t-1}(i)a_{ij}\right] \qquad \begin{array}{c} 2 \le t \le T \\ 1 \le j \le N \end{array}$$

3. Terminate

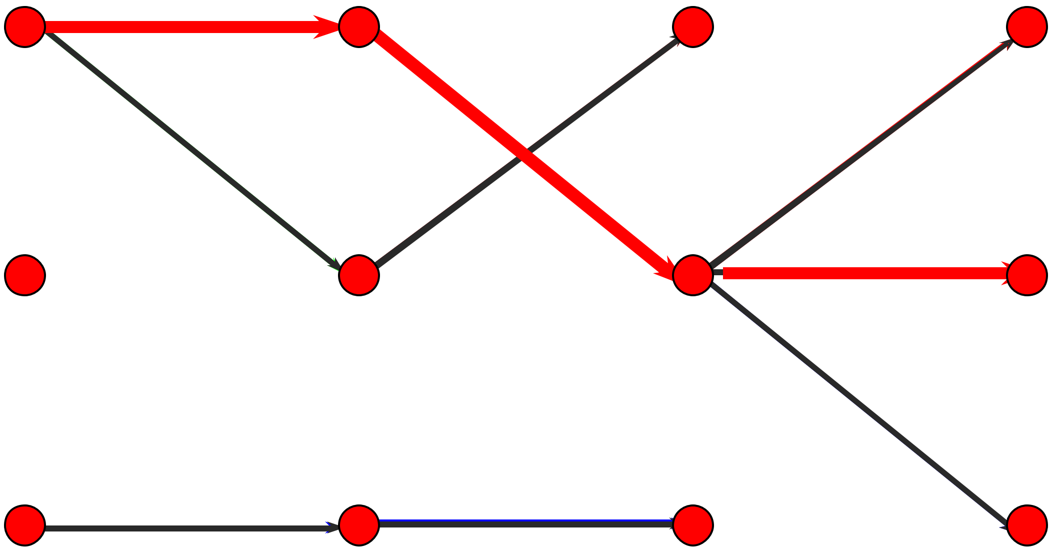
$$P^* = \max_{1 \le i \le N} d_T(i)$$

$$q_T^* = \operatorname*{argmax}_{1 \le i \le N} d_T(i)$$

4. Backtrack

$$q_t^* = y_{t+1}(q_{t+1}^*) \qquad t = (T-1), \ldots, 1$$

Viterbi Illustration



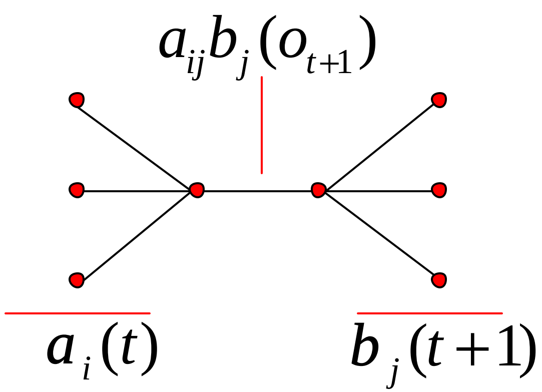
- Similar to the forward procedure
- Typically, you'll do it in log space for speed and underflows:
    - replace all parameters with their logarithms
    - replace all multiplications with additions

Baum-Welch algorithm (EM for HMMs)

Given:	$\mathbf{O} = (o_1, \ldots, o_T)$
Find:	p, A, B

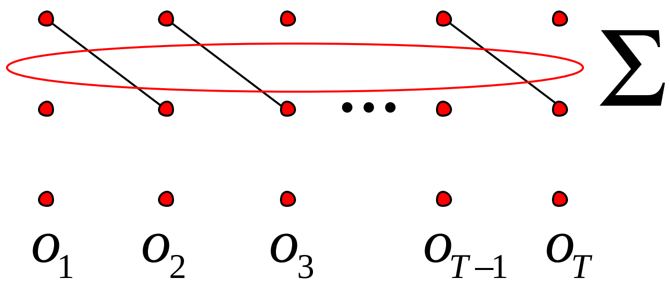
First, introduce another greek letter:

$$x_t(i, j) = P(q_t = i, q_{t+1} = j \mid \mathbf{O}) = \frac{P(q_t = i, q_{t+1} = j, \mathbf{O})}{P(\mathbf{O})}$$



$$= \frac{a_t(i) a_{ij} b_j(o_{t+1}) b_t(j)}{P(\mathbf{O})}$$

Transition Probability



expected # of transitions from i to j

This leads to:

$$\bar{a}_{ij} = \frac{E\left[\#(i \rightarrow j)\right]}{E\left[\#(i \rightarrow .)\right]} = \frac{\sum_{t=1}^{T-1} x_t(i, j)}{\sum_{t=1}^{T-1} g_t(i)}$$

The rest is easy

Priors and Outputs

Prior distribution:

$$\bar{p}_i = E\left[\#(i, t=1)\right] = g_1(i)$$

Output distribution (discrete):

$$\bar{b}_i(k) = \frac{E\left[\#(i, v_k)\right]}{E\left[\#(i)\right]} = \frac{\sum_{\substack{t=1 \\ o_t = v_k}}^{T} g_t(i)}{\sum_{t=1}^{T} g_t(i)}$$

Sum probabilities of being in state i while seeing symbol v_k

Normalize

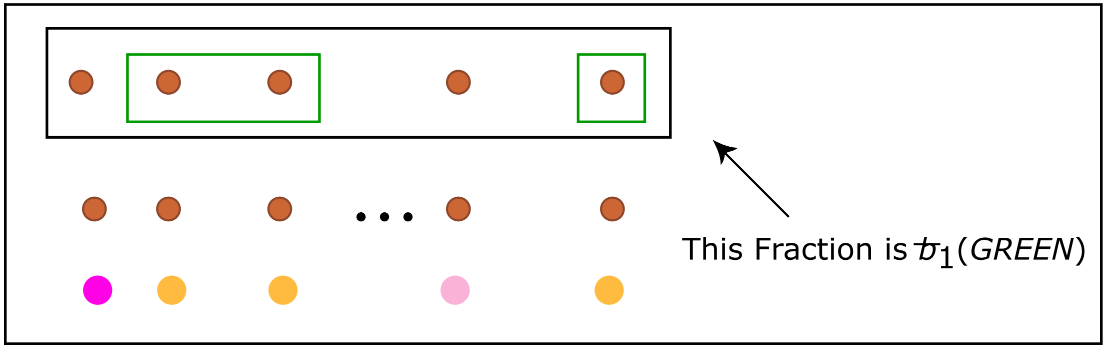


Figure by MIT OCW.

Continuous Output Case

Output distribution (continuous, Gaussian):

$$\bar{b}_i(o) = N(m_i, \Sigma_i)$$

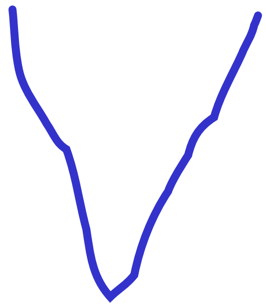
Observation at time t weighted by the probability of being in the state at that time

$$\bar{m}_i = \frac{\sum_{t=1}^{T} g_t(i) \cdot o_t}{\sum_{t=1}^{T} g_t(i)}$$

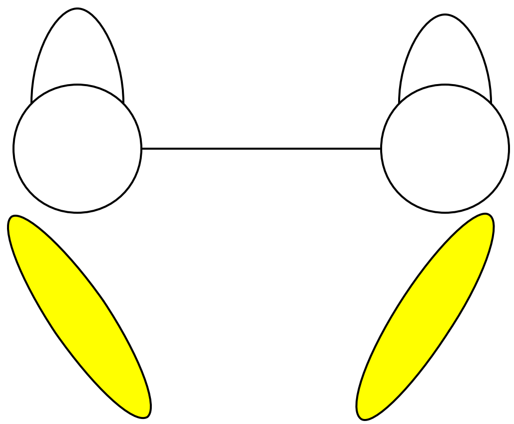
These should look VERY familiar

$$\bar{\Sigma}_i = \frac{\sum_{t=1}^{T} g_t(i) \cdot (o_t - m_i)(o_t - m_i)^T}{\sum_{t=1}^{T} g_t(i)}$$

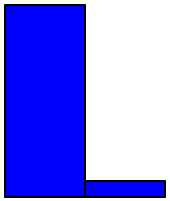
Semi-Continuous HMM Example



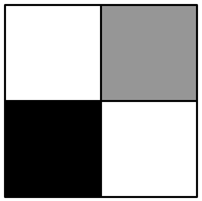
Input



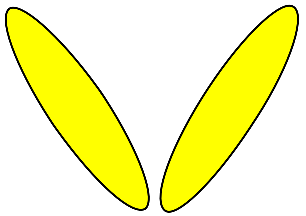
Output distributions



Initial
state

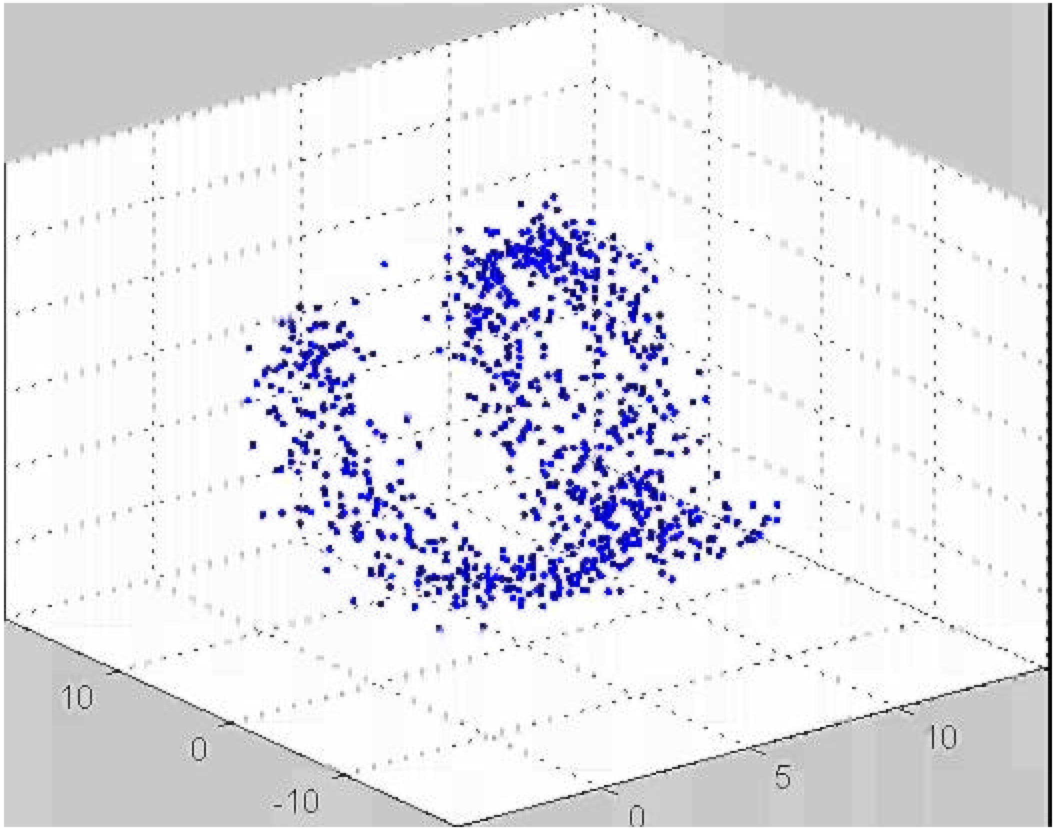


Transition
matrix



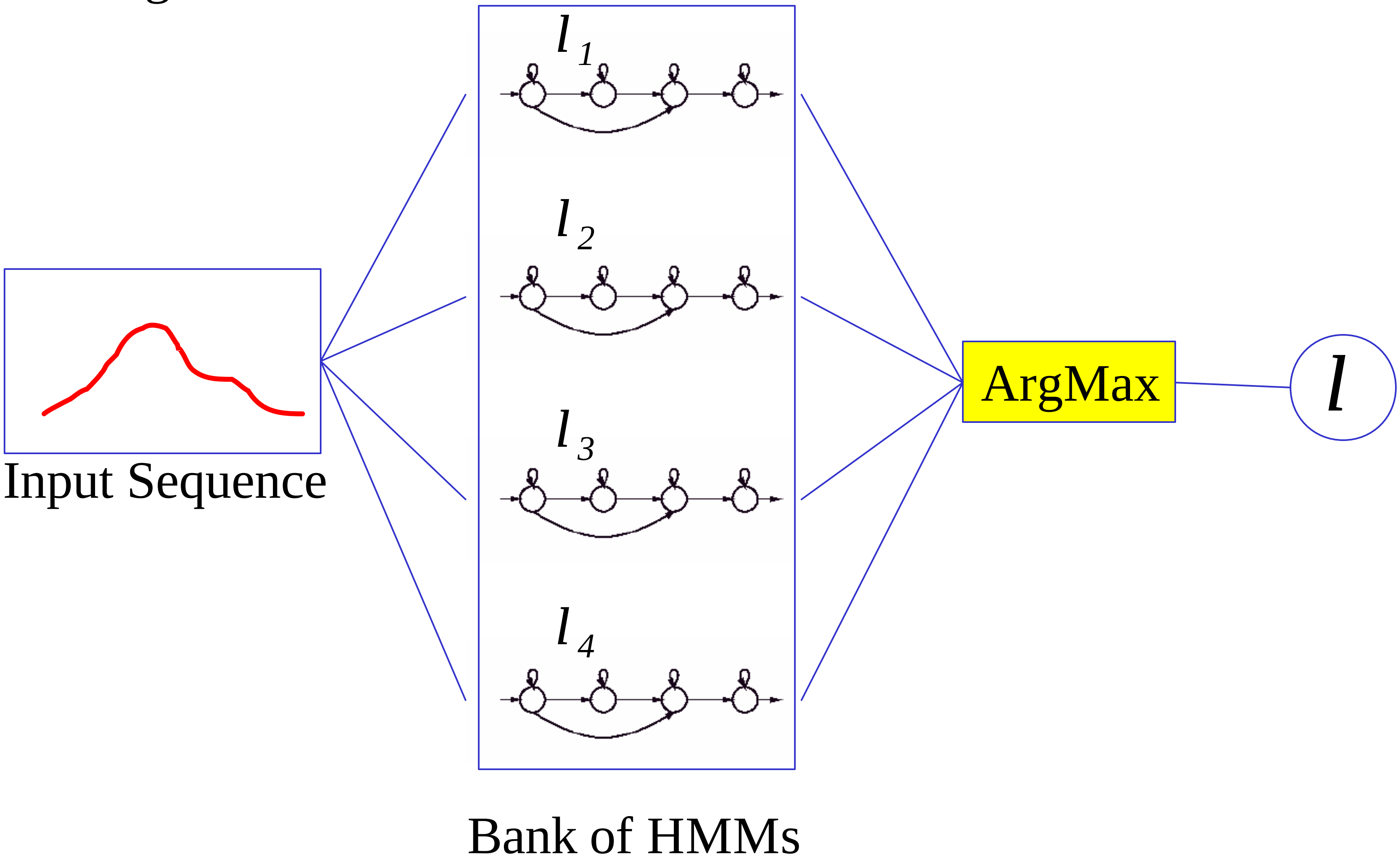
How HMM sees it

Modeling a tracked hand trajectory.



HMM Classifier

Nothing unusual:



Applications – American Sign Language

Task: Recognition of sentences of American Sign Language

40 word lexicon:

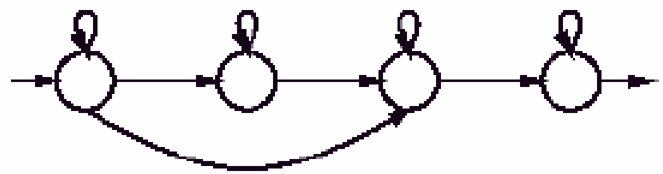
part of speech	*vocabulary*
pronoun	I, you, he, we, you(pl), they
verb	want, like, lose, dontwant, dontlike, love, pack, hit, loan
noun	box, car, book, table, paper, pants, bicycle, bottle, can, wristwatch, umbrella, coat, pencil, shoes, food, magazine, fish, mouse, pill, bowl
adjective	red, brown, black, gray, yellow

- Single camera
- No special markings on hands
- Real-time

Table from: Starner, T., and et. al. "Real-Time American Sign Language Recognition Using Desk and Wearable Computer Based Video." *IEEE Transactions on Pattern Analysis and Machine Intelligence* (1998).

ASL – Features and Model

“Word” model – a 4-state L-R HMM with a single skip transition:



Features
(from skin model):

$$o = \left[\left(x, y, dx, dy, area, q, l_{\max}, l_{\max} / l_{\min} \right)_{right}, (\ldots)_{left} \right]^T$$

System 1: Second person

System 2: First person

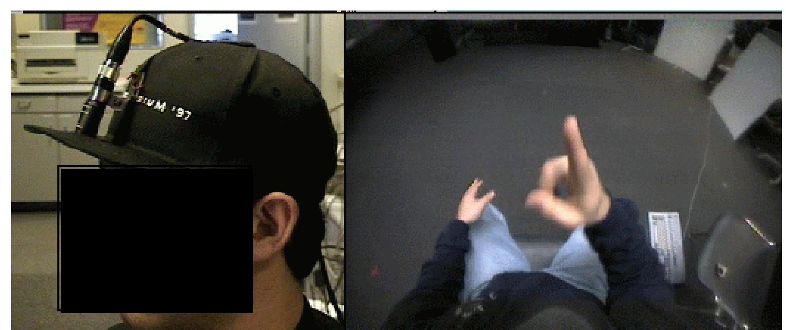


Photo marked with black box
due to copyright consideration.

Nose could be used for initializing the skin model

ASL – In Action

Courtesy of Thad Starner. Used with permission.

Applications – American Sign Language

500 sentences (400 training, 100 testing)

System 1:

experiment	*training set*	*test set*
all features	94.10%	91.90%
relative features	89.60%	87.20%
all features & unrestricted grammar	81.0% (87%) (D=31, S=287, I=137, N=2390)	74.5% (83%) (D=3, S=76, I=41, N=470)

% words recognized correctly

Word accuracy, $1-\frac{D+S+I}{N}$

System 2:

grammar	*training set*	*test set*
part-of-speech	99.30%	97.80%
5-word sentence	98.2% (98.4%) (D = 5, S=36, I=5 N =2500)	97.80%
unrestricted	96.4% (97.8%) (D=24, S=32, I=35, N=2500)	96.8% (98.0%) (D=4, S=6, I=6, N=500)

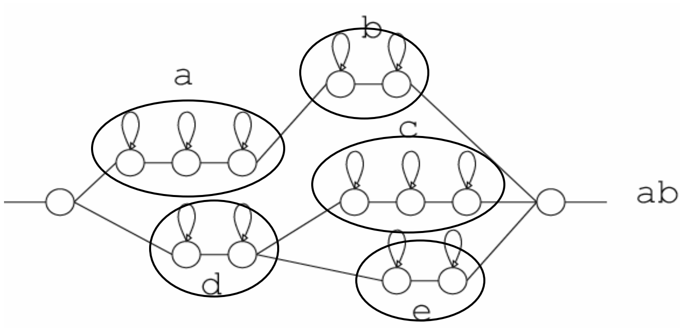
Beyond HMM

Where can we go if HMM is not sufficient?

Ideas:

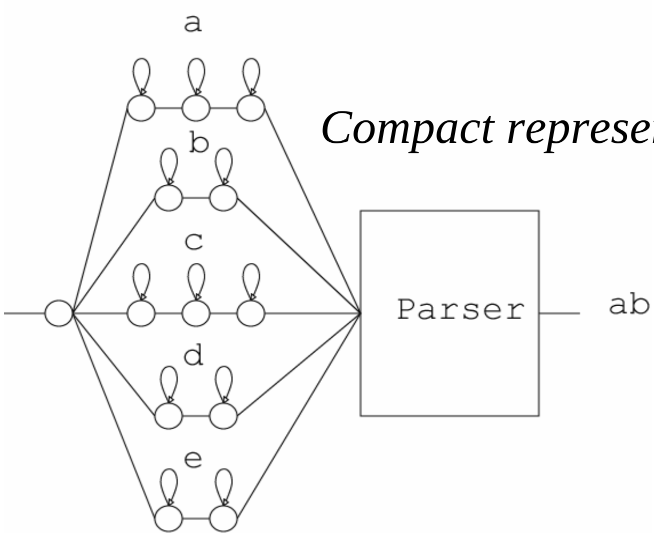
- Hierarchical HMM
- More complex models - SCFG

Explicit representation of structure



Capable of generating only a regular language

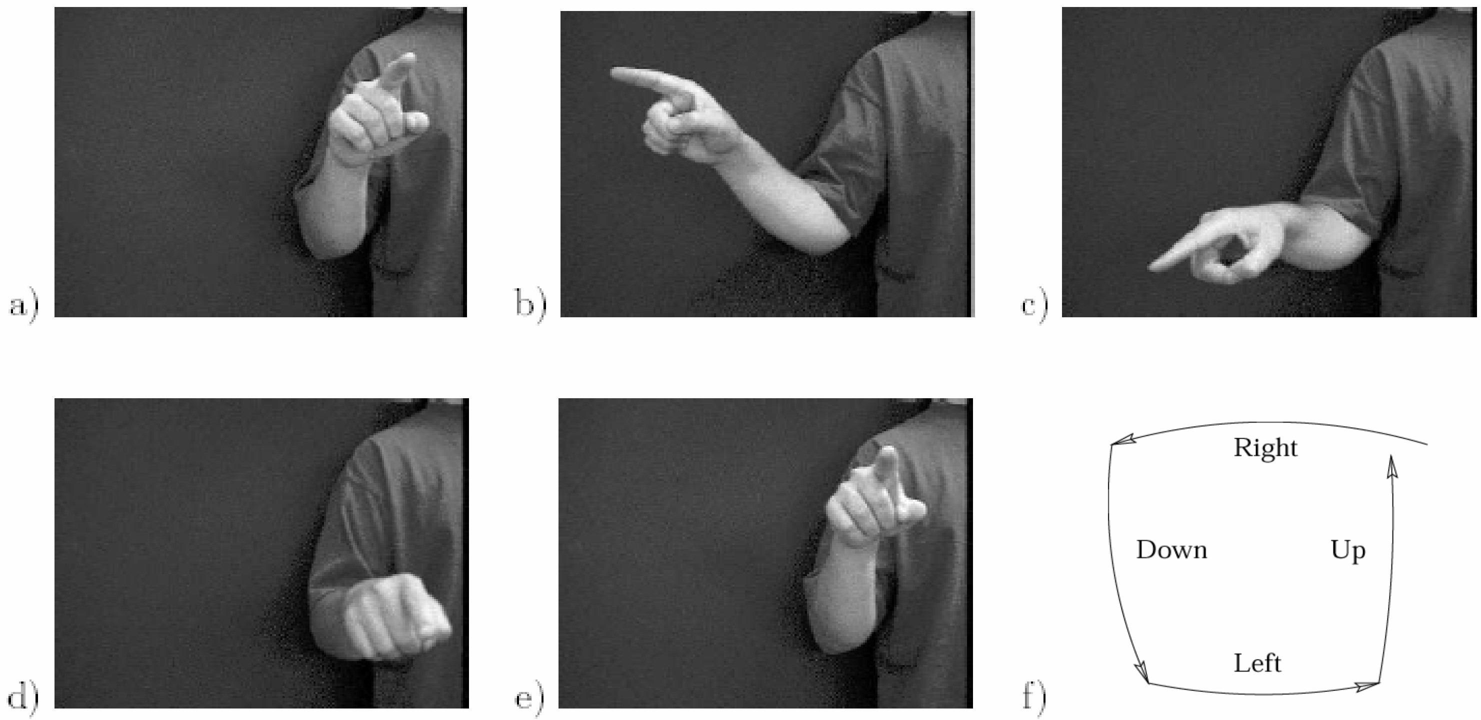
Compact representation



More expressive, may include memory, but harder to deal with

Figures from: Ivanov, Y., and A. Bobick. "Recognition of Visual Activities and Interactions." *IEEE Transactions of Pattern Analysis and Machine Intelligence* (2000). Courtesy of IEEE. Copyright 2000 IEEE. Used with Permission.

Structured Gesture



Problem:

2 directions = 2 models

WHY???

Solution – split the model in two:

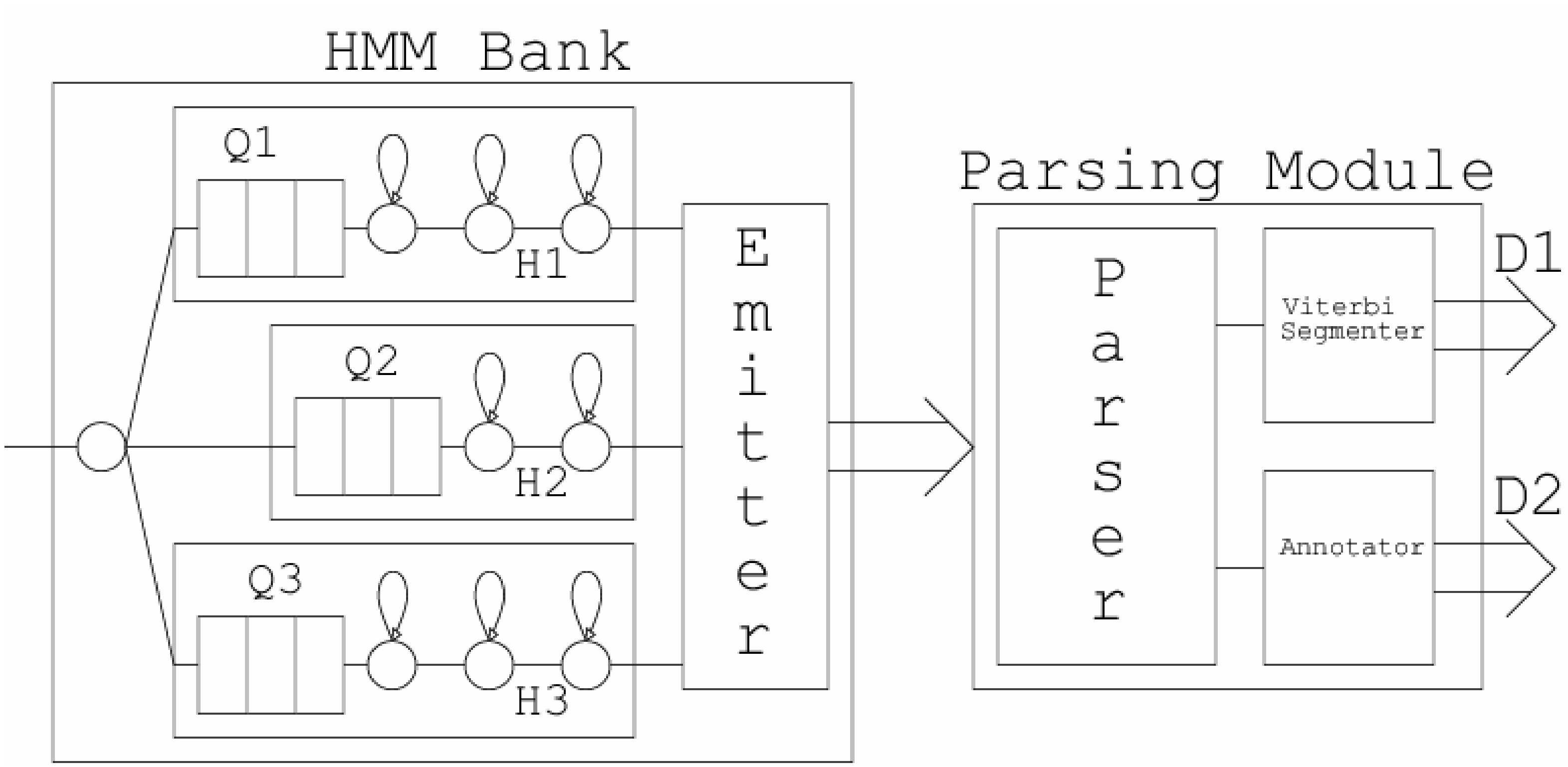
- Components (trajectories)
- Structure (events)

Heterogeneous Representation

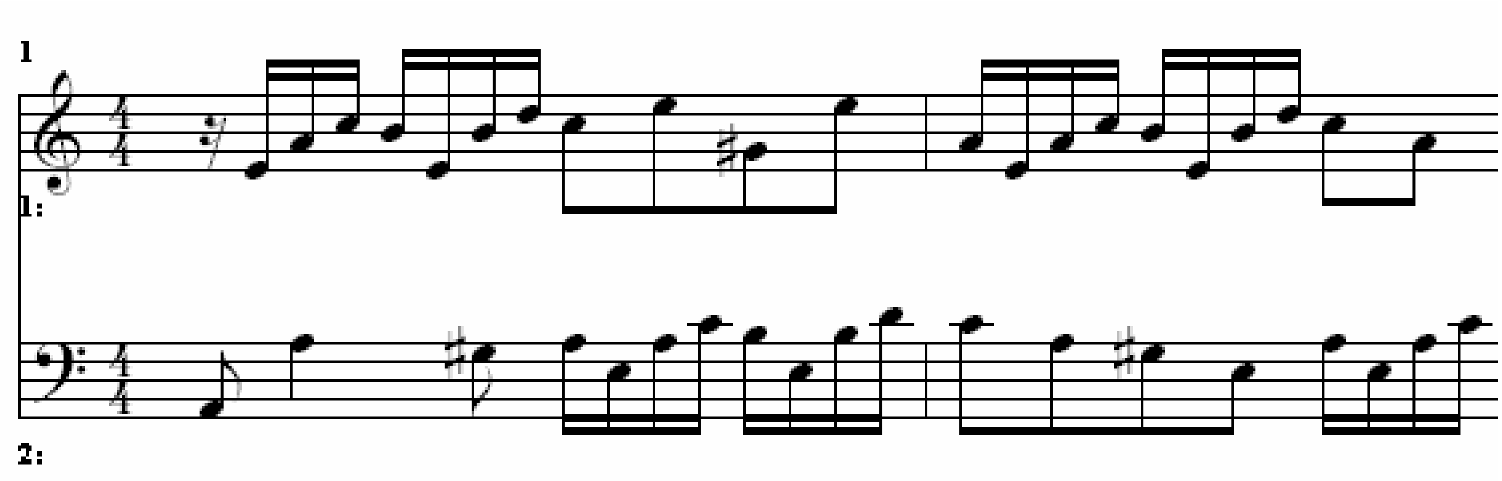
- Many high-level activities are sequences of primitives
  - Pitching, cooking, dancing, stealing a car from a parking lot
- Components
  - Signal level model
  - Variability in performance
  - Hidden state representation (HMM, etc.)
- Structure
  - Event-level model
  - Uncertainty in component detections
  - State is NOT hidden (SRG, SCFG, etc)

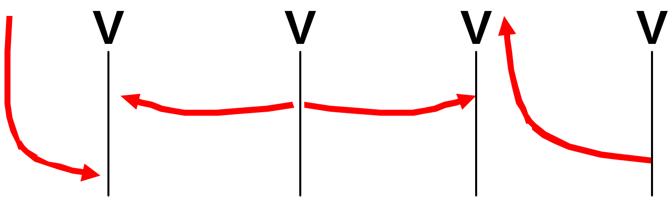
- Right tool for the right task!

Two-tier Recognition Architecture

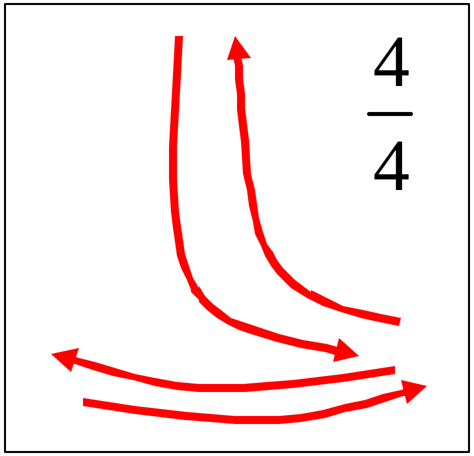


Application: Conducting Music



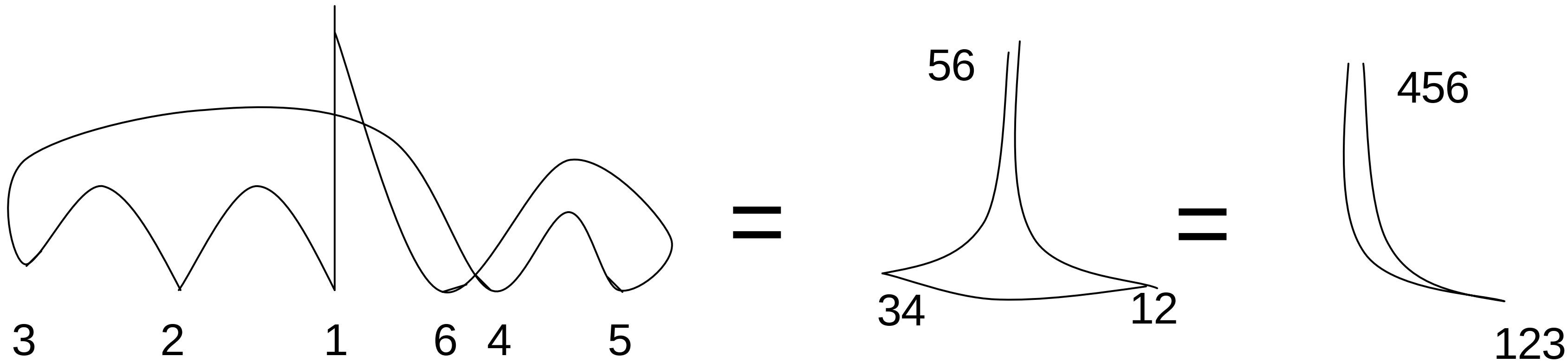


“Dictionary” gestures



Application: Conducting Music

Jean Sibelius, Second Symphony, Opus 43, D Major

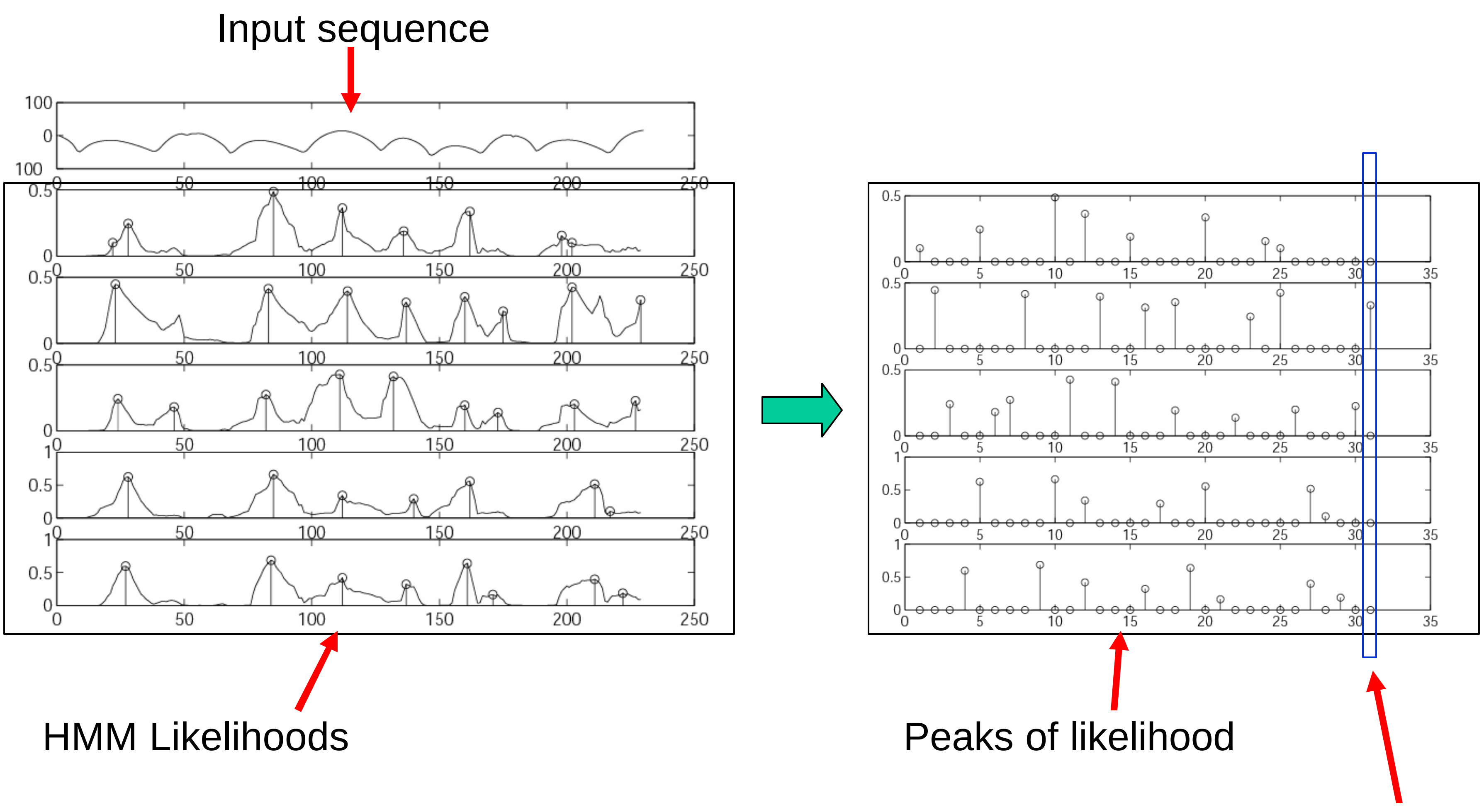


Grammar:

```
Gc :
PIECE   →      BAR PIECE           [0.5]
             | BAR                 [0.5]
BAR     →      TWO                 [0.5]
             | THREE               [0.5]
THREE   →      down3 right3 up3    [1.0]
TWO     →      down2 up2           [1.0]
```

	Correct
Individual	~70%
Component	~85%
Bar	~95%

Component Detection

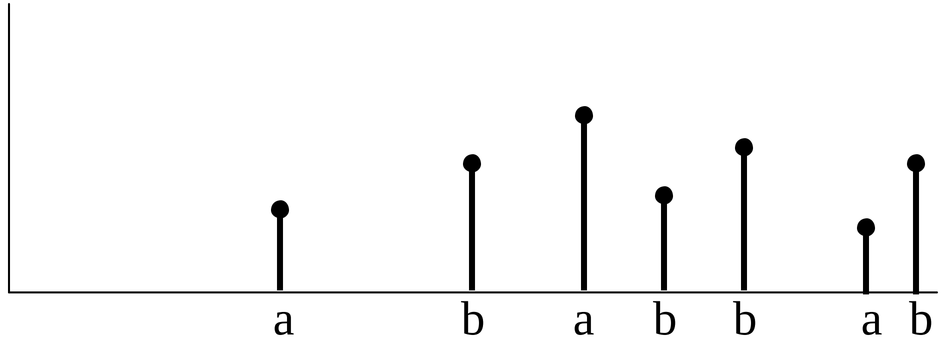


Temporal Consistency

Grammar:

$$A \rightarrow ab \mid abA$$

Input



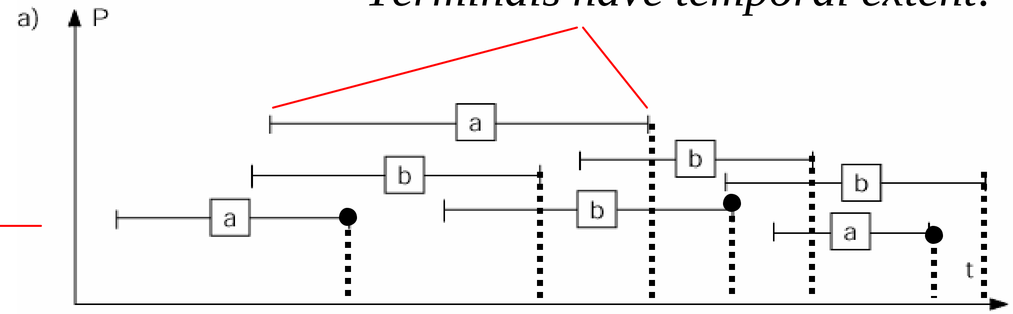
Temporal Consistency

Grammar:

$$A \rightarrow ab \mid abA$$

Input

Terminals have temporal extent!



Figures from: Ivanov, Y., and A. Bobick. "Recognition of Visual Activities and Interactions." *IEEE Transactions of Pattern Analysis and Machine Intelligence* (2000). Courtesy of IEEE. Copyright 2000 IEEE. Used with Permission.

Temporal Consistency

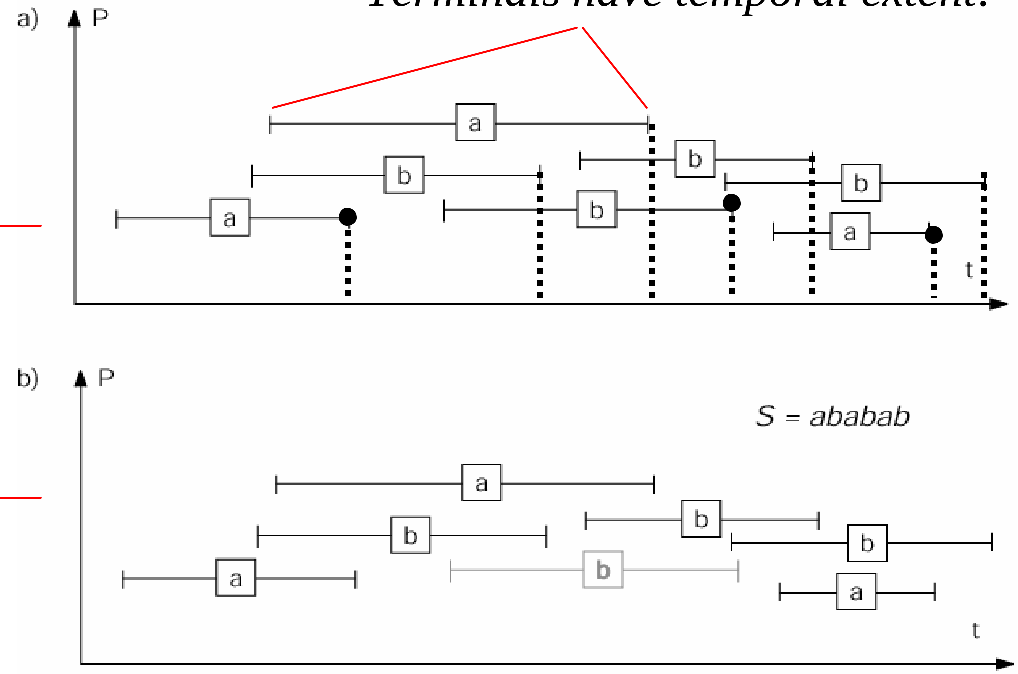
Grammar:

$$A \rightarrow ab \mid abA$$

Terminals have temporal extent!

Input

Inconsistent parse



Figures from: Ivanov, Y., and A. Bobick. "Recognition of Visual Activities and Interactions." *IEEE Transactions of Pattern Analysis and Machine Intelligence* (2000). Courtesy of IEEE. Copyright 2000 IEEE. Used with Permission.

Temporal Consistency

Grammar:

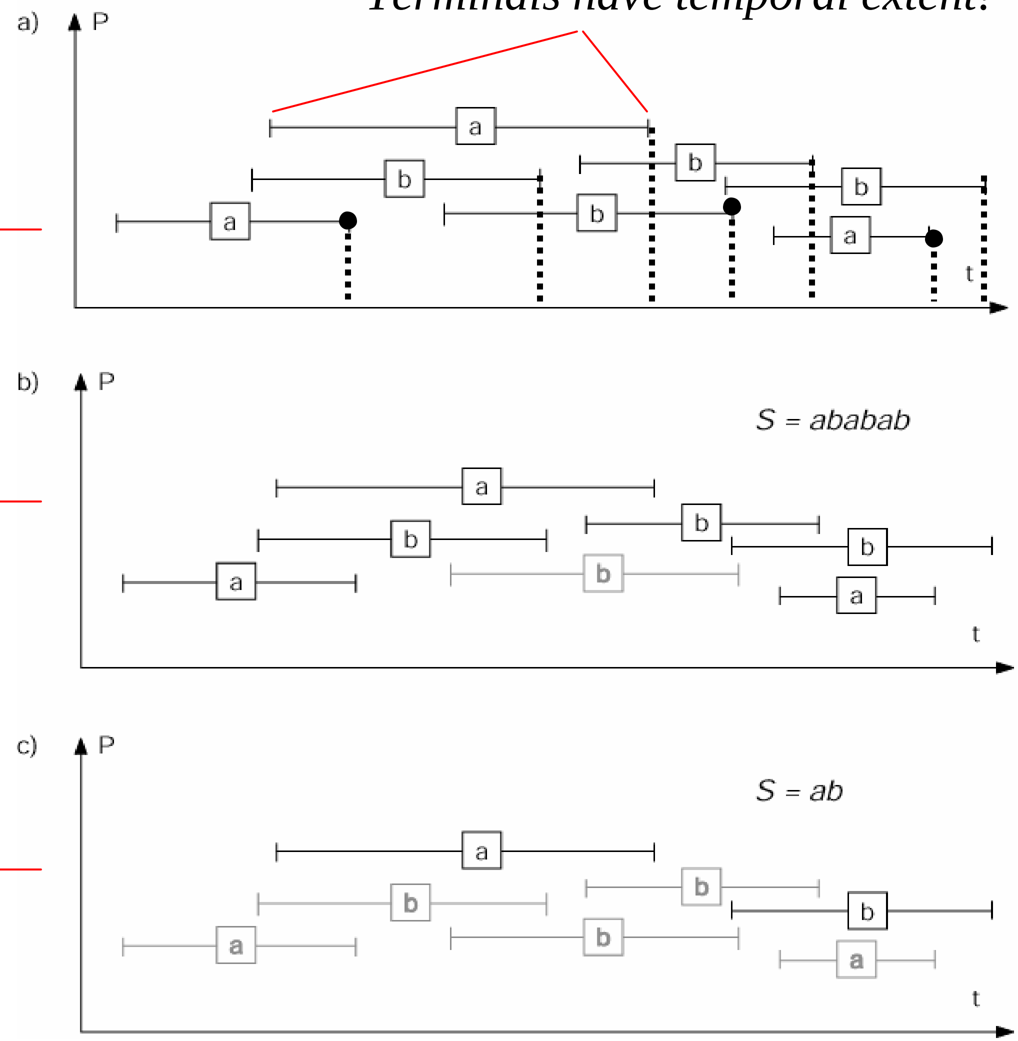
$$A \rightarrow ab \mid abA$$

Input

Inconsistent parse

Consistent parse

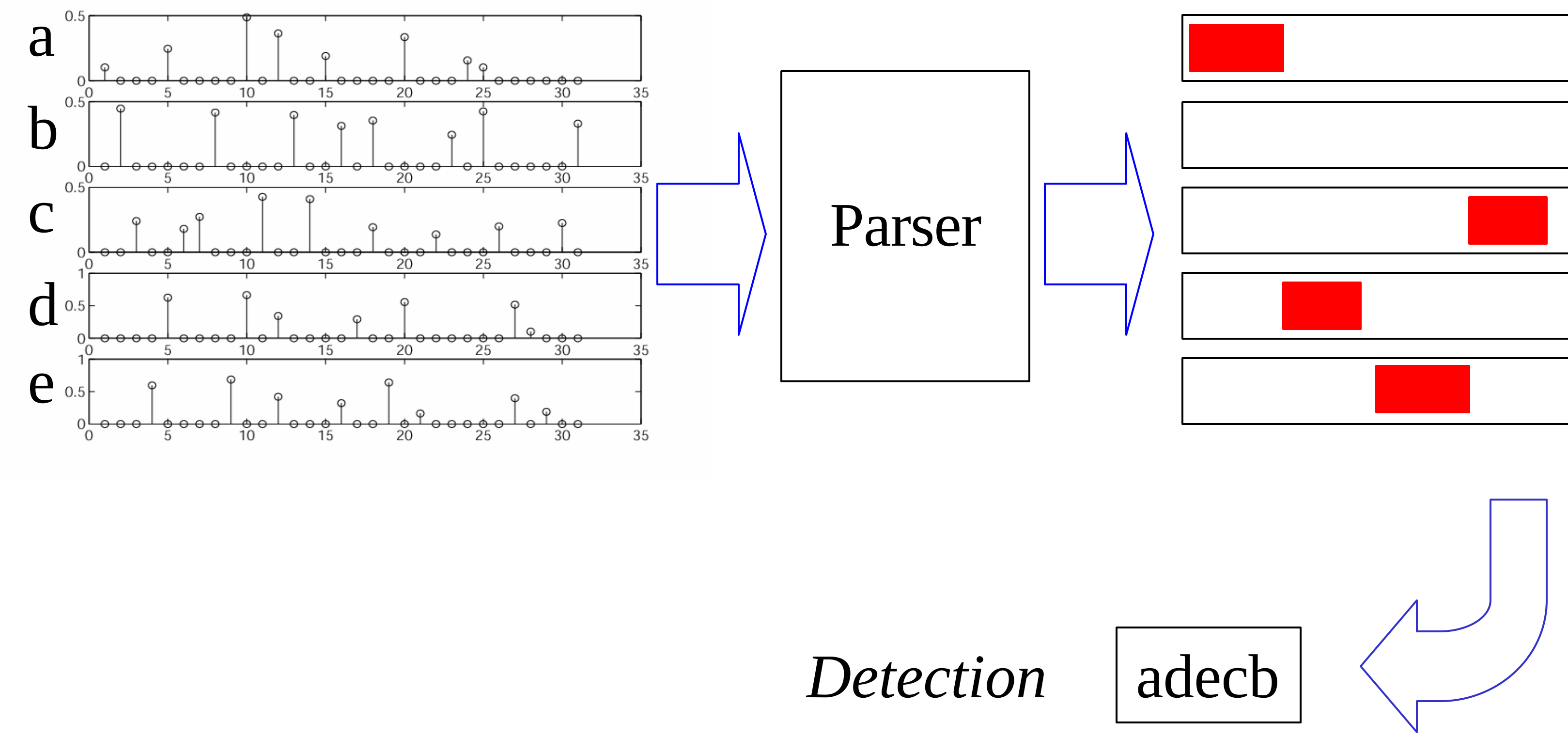
Terminals have temporal extent!



Figures from: Ivanov, Y., and A. Bobick. "Recognition of Visual Activities and Interactions." *IEEE Transactions of Pattern Analysis and Machine Intelligence* (2000). Courtesy of IEEE. Copyright 2000 IEEE. Used with Permission.

Parsing

The idea is that the top level parse will filter out mistakes in low level detections



Stochastic Context-Free Grammar

Example Grammar:

```
TRACK            ->      CAR-TRACK       [0.50]
                       | PERSON-TRACK    [0.50]

CAR-TRACK        ->      CAR-THROUGH     [0.25]
                       | CAR-PICKUP      [0.25]
                       | CAR-OUT         [0.25]
                       | CAR-DROP        [0.25]
CAR-THROUGH . . .

                 . . .

CAR-EXIT         ->      car-exit        [0.70]
                       | SKIP car-exit   [0.30]
```

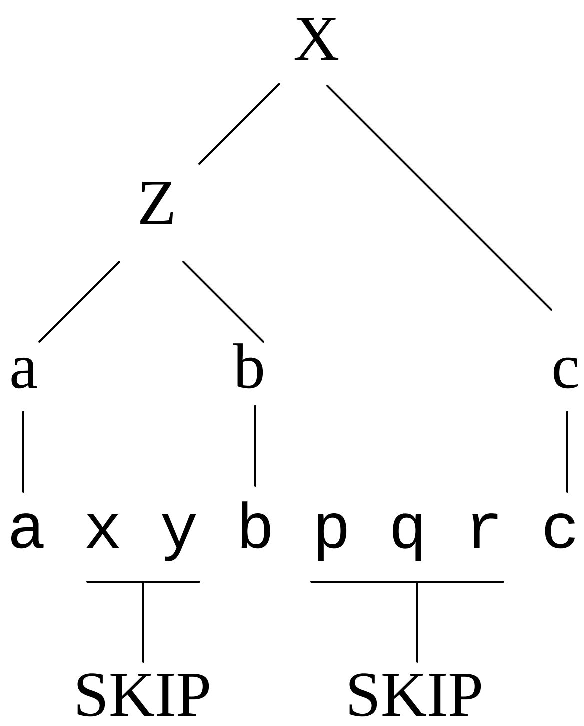
Non-terminals - semantically significant groups of events

Terminals - individual events

SKIP-rule - noise symbol

Rule probabilities

Event Parsing



$$X \rightarrow Zc$$

$$Z \rightarrow ab$$

- production rules (states)

X - target non-terminal (label)

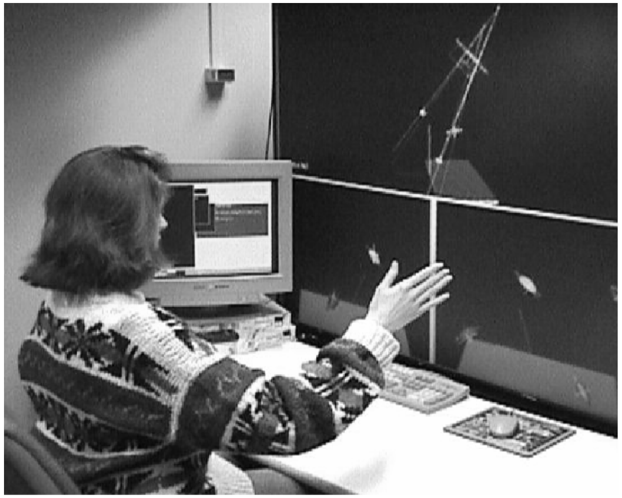
Z - intermediate non-terminal

- input stream (tracking events)

- noise rules

For the production *X*, events *a*, *b* and *c* should be consistent

Application: Musical Conducting

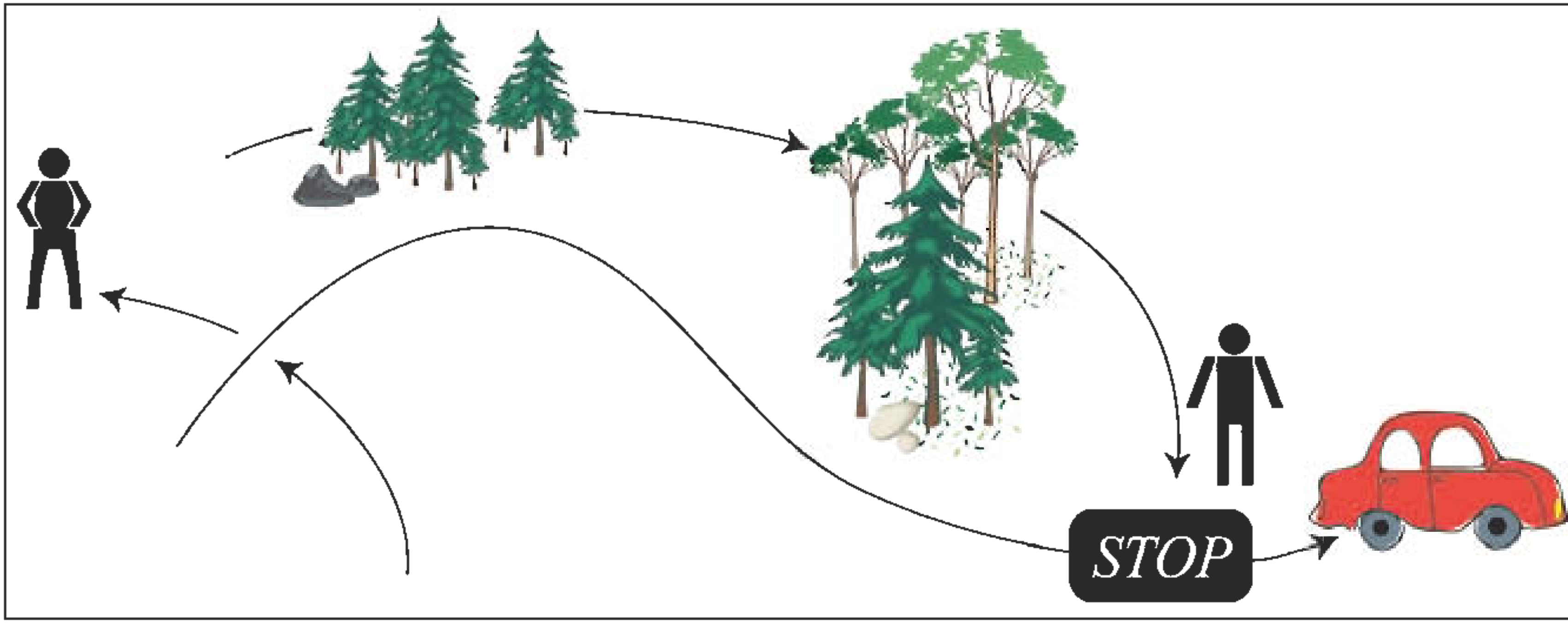


Courtesy of Teresa Marrin-Nakra. Used with permission.

```
Segmentation:
BAR:
    2/4     start/end sample: [0 66]
    Conducted as two quarter beat pattern.
BAR:
    2/4     start/end sample: [66 131]
    Conducted as two quarter beat pattern.
BAR:
    3/4     start/end sample: [131 194]
    Conducted as three quarter beat pattern.
BAR:
    2/4     start/end sample: [194 246]
    Conducted as two quarter beat pattern.
Viterbi probability =  0.00423416
```

	Correct
Individual	~70%
Component	~85%
Bar	~95%

From Tracking to Classification



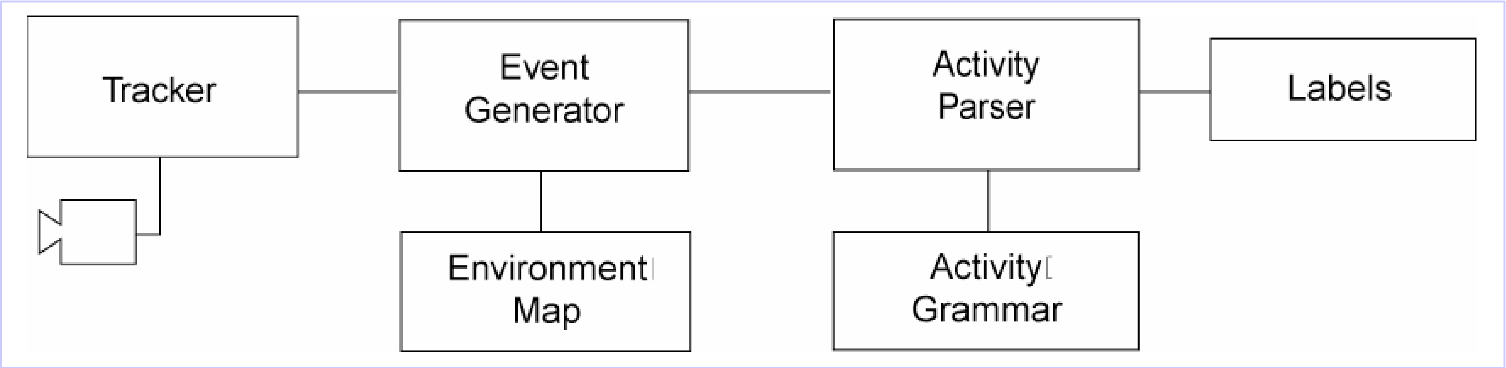
How do we describe that?
How do we classify that?

Figure by MIT OCW.

Application: Surveillance System

- Outdoor environment - occlusions and lighting changes
- Static cameras
- Real-time performance
- Labeling activities and person-vehicle interactions in a parking lot
- Handling simultaneous events

Monitoring System



Photos and figures from: Stauffer, Chris, and Eric Grimson, "Learning Patterns of Activity Using Real-Time Tracking." *IEEE Transactions on Pattern Recognition and Machine Intelligence (TPAMI)* 22, no. 8 (2000): 747-757. Courtesy of IEEE, Chris Stauffer, and Eric Grimson.

- Tracker (Stauffer, Grimson)
  - assigns identity to the moving objects
  - collects the trajectory data into partial tracks
- Event Generator
  - maps partial tracks onto a set of events
- Parser
  - labels sequences of events according to a grammar
  - enforces spatial and temporal constraints

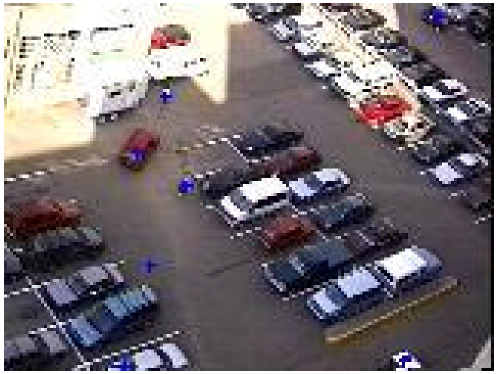
Tracker

- Adaptive to slow lighting changes:
  - Each pixel is modeled by a mixture

$$P(X_t) = \sum_{i=1}^{K} w_{i,t} * h(X_t, m_{i,t}, \Sigma_{i,t})$$

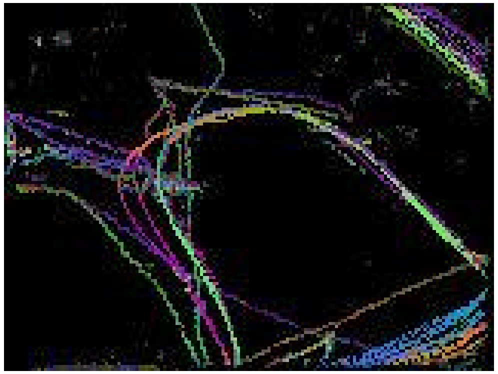
- Foreground regions are found by connected components algorithm
- Object dynamics is modeled in 2D by a set of Kalman filters
- Details - (Stauffer, Grimson CVPR 99)

Tracker



Camera view

Connected components

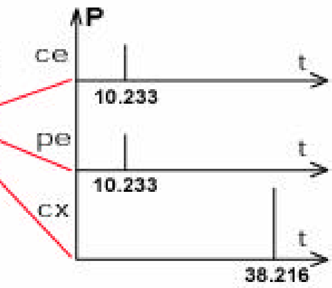


Trajectories over time

An object

Event Generator

Event	Likelihood	x	y	dx	dy	time
car-enter	0.5	0.454	1	-0.01	0.05	10.233
person-enter	0.5	0.454	1	-0.01	0.05	10.233
car-exit	1	1	0.784	0.1	0.1	38.216



Photos and figures from: Stauffer, Chris, and Eric Grimson, "Learning Patterns of Activity Using Real-Time Tracking." *IEEE Transactions on Pattern Recognition and Machine Intelligence (TPAMI)* 22, no. 8 (2000): 747-757. Courtesy of IEEE, Chris Stauffer, and Eric Grimson. Copyright 2000 IEEE. Used with permission.

Map tracks onto events: `car-enter, person-enter, car-found, person-found, car-lost, person-lost, stopped`

- Events along with class likelihoods are posted at the endpoints of each track
  (`car-appear [0.5]`, `car-disappear [1.0]`)
- Action label is assigned to each event in accordance with the environment map
  (`car-enter [0.5]`, `car-exit [1.0]`)
- Each event is complemented if the label probability is < 1
  (`car-enter [0.5], person-enter [0.5]`, `car-exit [1.0]`)

Parking Lot Grammar (Partial)

G_p :

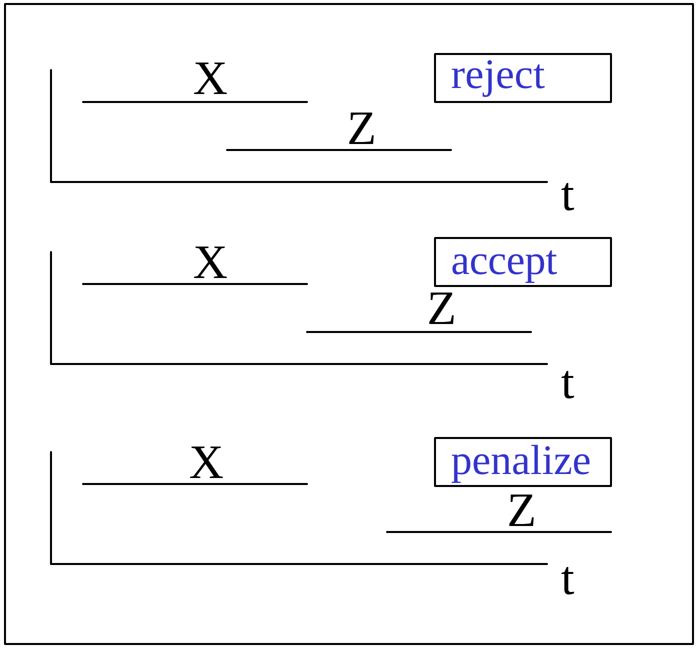
TRACK	$\longrightarrow$		CAR-TRACK	[0.5]
		|	PERSON-TRACK	[0.5]
CAR-TRACK	$\longrightarrow$		CAR-THROUGH	[0.25]
		|	CAR-PICKUP	[0.25]
		|	CAR-OUT	[0.25]
		|	CAR-DROP	[0.25]
CAR-PICKUP	$\longrightarrow$		ENTER-CAR-B CAR-STOP PERSON-LOST B-CAR-EXIT	[1.0]
ENTER-CAR-B	$\longrightarrow$		CAR-ENTER	[0.5]
		|	CAR-ENTER CAR-HIDDEN	[0.5]
CAR-HIDDEN	$\longrightarrow$		CAR-LOST CAR-FOUND	[0.5]
		|	CAR-LOST CAR-FOUND CAR-HIDDEN	[0.5]
B-CAR-EXIT	$\longrightarrow$		CAR-EXIT	[0.5]
		|	CAR-HIDDEN CAR-EXIT	[0.5]
CAR-EXIT	$\longrightarrow$		car-exit	[0.7]
		|	SKIP car-exit	[0.3]
CAR-LOST	$\longrightarrow$		car-lost	[0.7]
		|	SKIP car-lost	[0.3]
CAR-STOP	$\longrightarrow$		car-stop	[0.7]
		|	SKIP car-stop	[0.3]
PERSON-LOST	$\longrightarrow$		person-lost	[0.7]
		|	SKIP person-lost	[0.3]

Photos and figures from: Stauffer, Chris, and Eric Grimson, "Learning Patterns of Activity Using Real-Time Tracking." *IEEE Transactions on Pattern Recognition and Machine Intelligence (TPAMI)* 22, no. 8 (2000): 747-757. Courtesy of IEEE, Chris Stauffer, and Eric Grimson. Copyright 2000 IEEE. Used with Permission.

Consistency

- Temporal
  - Events should happen in particular order
  - Temporally close events are more likely to be related
  - Tracks overlapping in time are **definitely not** related to the same object
- Spatial
  - Spatially close events are more likely to be related
- Other
  - Objects don’t change identity within a track

Spatio-Temporal Consistency



$$\mathbf{r} = (x, y), \qquad d\mathbf{r} = (dx, dy)$$

Predict new position:

$$\mathbf{r}_p = \mathbf{r}_1 + d\mathbf{r}_1 (t_2 - t_1)$$

Penalize:

$$f(\mathbf{r}_p, \mathbf{r}_2) = \begin{cases} 0, & \text{if } (t_2 - t_1) < 0 \\ \exp\left(\dfrac{(\mathbf{r}_2 - \mathbf{r}_p)^T (\mathbf{r}_2 - \mathbf{r}_p)}{\theta} \right) \end{cases}$$

Input Data

Event Generator

DROP-OFF

DRIVE-IN

Event	UID	Avg. Size	Class	P	x	y	t	frame
ENTER	724	0.122553	0	0.5	0.450094	0.938069	917907137.8	1906
ENTER	665	0.046437	1	0.5	0.6107	0.94674	917907122.5	1799
PERSON-LEAVE	665	0.045869	1	0.997846	0.648089	0.98855	917907142.7	1938
STOPPED	724		0	0.995784	0.348569	0.345513	917907146.5	1964
ENTER	780	0.034293	1	0.5	0.74188	0.980292	917907151.3	1998
ENTER	790	0.069093	0	0.5	0.814565	0.032611	917907153.4	2012
FOUND	787	0.033573	1	0.5	0.297585	0.357887	917907153.1	2010
CAR-LEAVE	790	0.061263	0	0.997285	0.975971	0.211984	917907155.3	2025
PERSON-LEAVE	780	0.038616	1	0.999923	0.974494	0.865237	917907158.6	2047
PERSON-LEAVE	787	0.032045	1	0.999997	0.296519	0.183704	917907158.7	2048
ENTER	813	0.034776	1	0.5	0.012821	0.348379	917907160.9	2063
ENTER	816	0.093513	0	0.5	0.960425	0.793899	917907161.9	2070
CAR-LEAVE	724	0.097374	0	0.993211	0.972272	0.693728	917907165.2	2091
CAR-LEAVE	816	0.089424	0	0.99023	0.693699	0.990798	917907165.2	2091

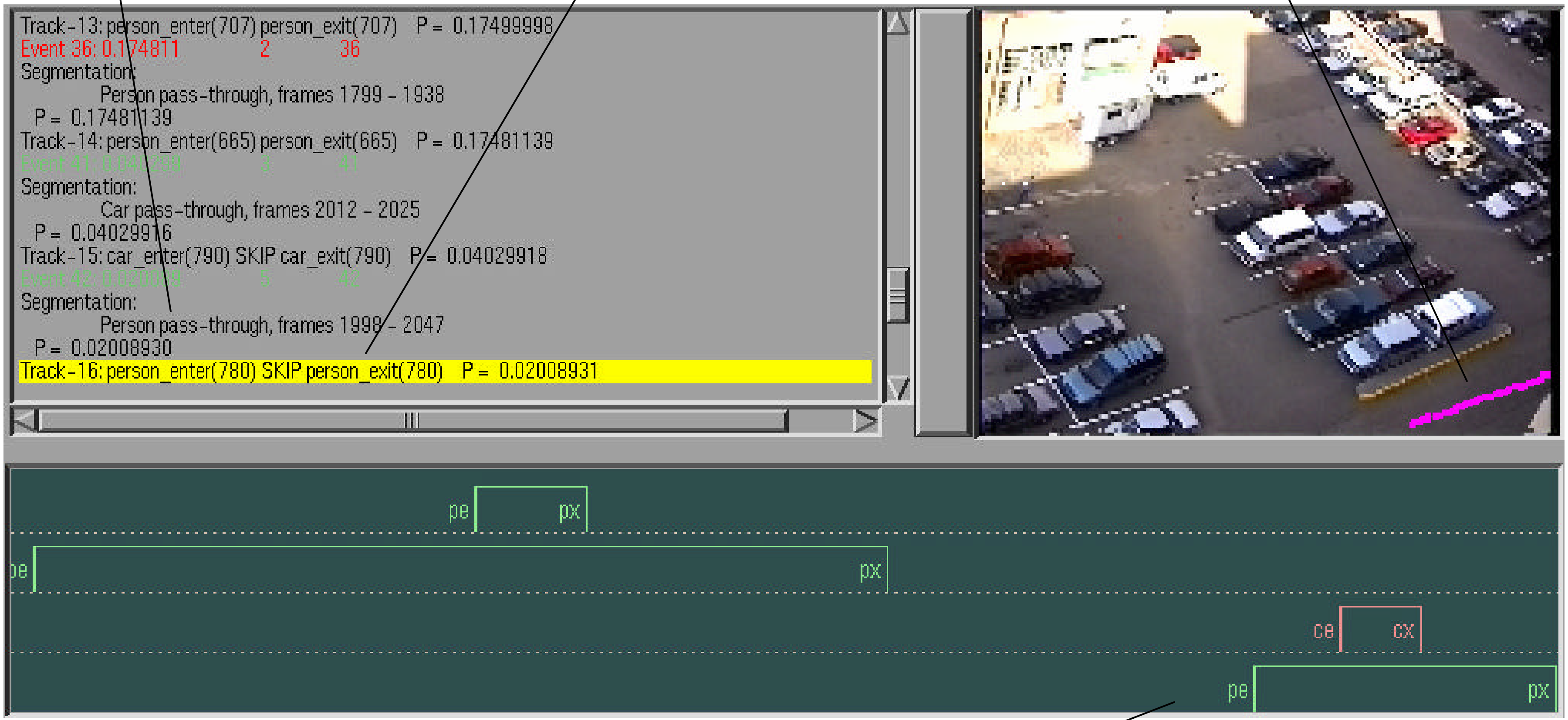
Interleaved events in the input stream

Parse 1: Person-Pass-Through

Action label

Component labels

Object track



Temporal extent

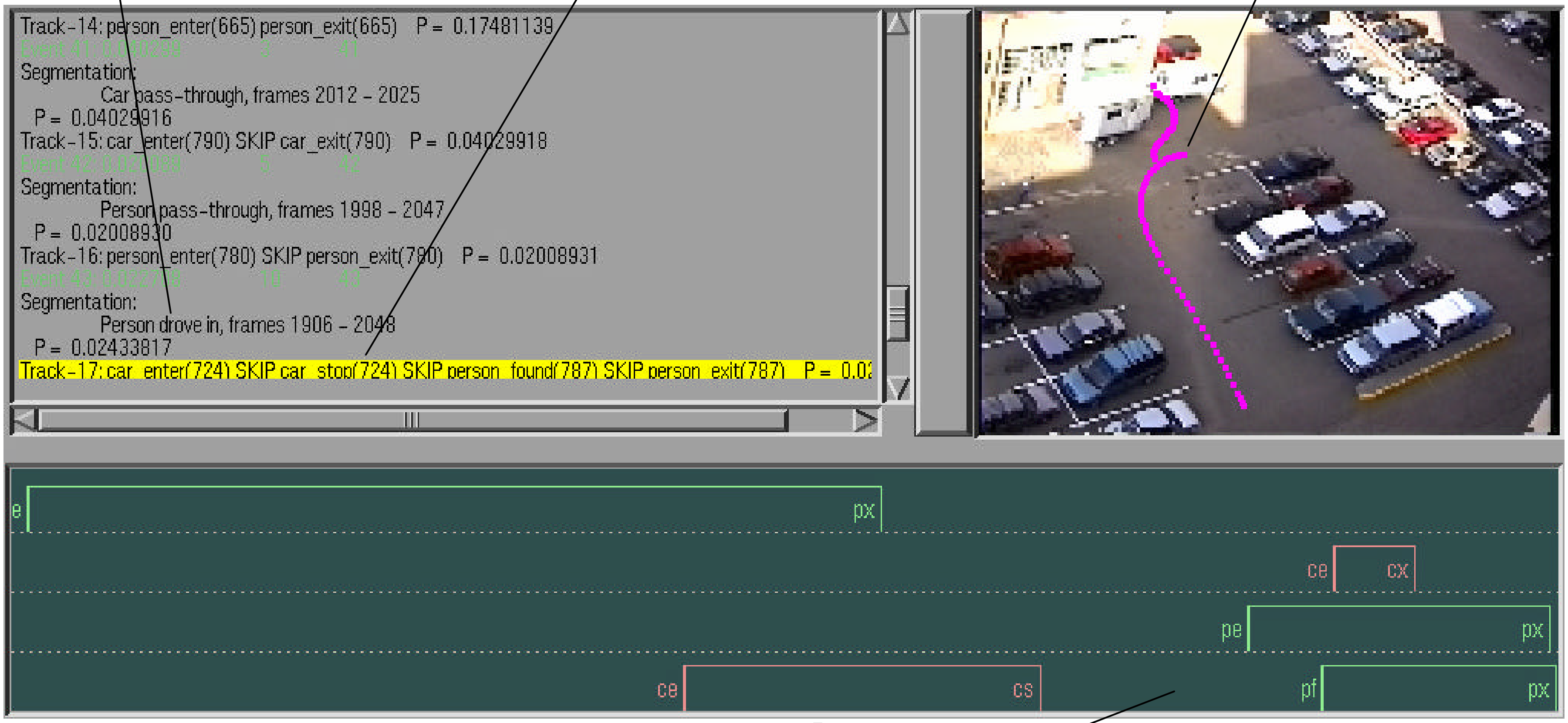
Figures from Ivanov, Yuri, Chris Stauffer, Aaron Bobick, W. E. L. Grimson. "Video Surveillance of Interactions." *IEEE Workshop on Visual Surveillance (ICCV 2001)* (1999). Courtesy of IEEE, Yuri Ivanov, Chris Stauffer, Aaron Bobick, and W. E. L. Grimson.

Parse 2: Drive-In

Action label

Component labels

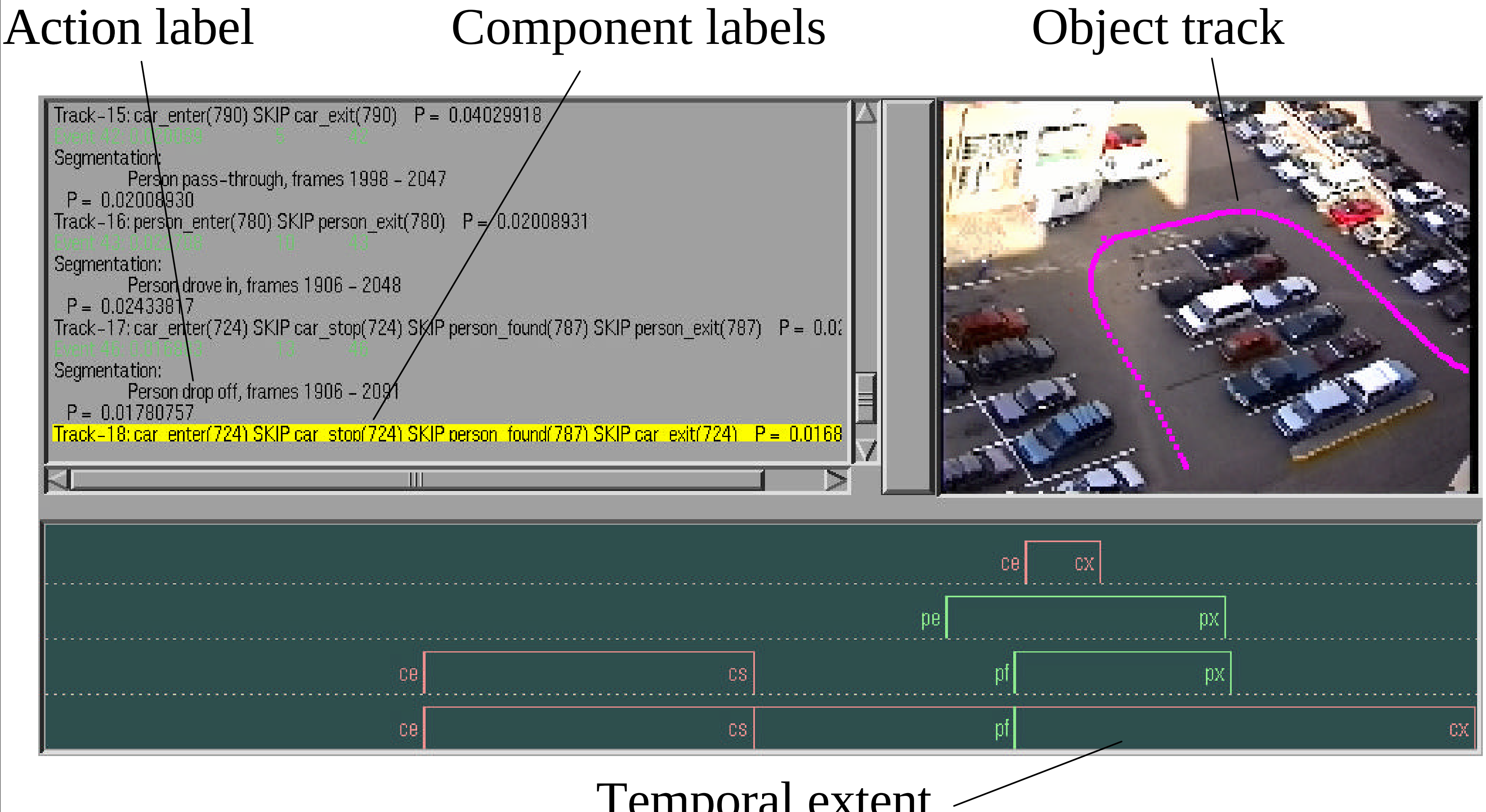
Object tracks



Temporal extent

Figures from Ivanov, Yuri, Chris Stauffer, Aaron Bobick, W. E. L. Grimson. "Video Surveillance of Interactions." *IEEE Workshop on Visual Surveillance (ICCV 2001)* (1999). Courtesy of IEEE, Yuri Ivanov, Chris Stauffer, Aaron Bobick, and W. E. L. Grimson. Copyright 1999 IEEE. Used with Permission.

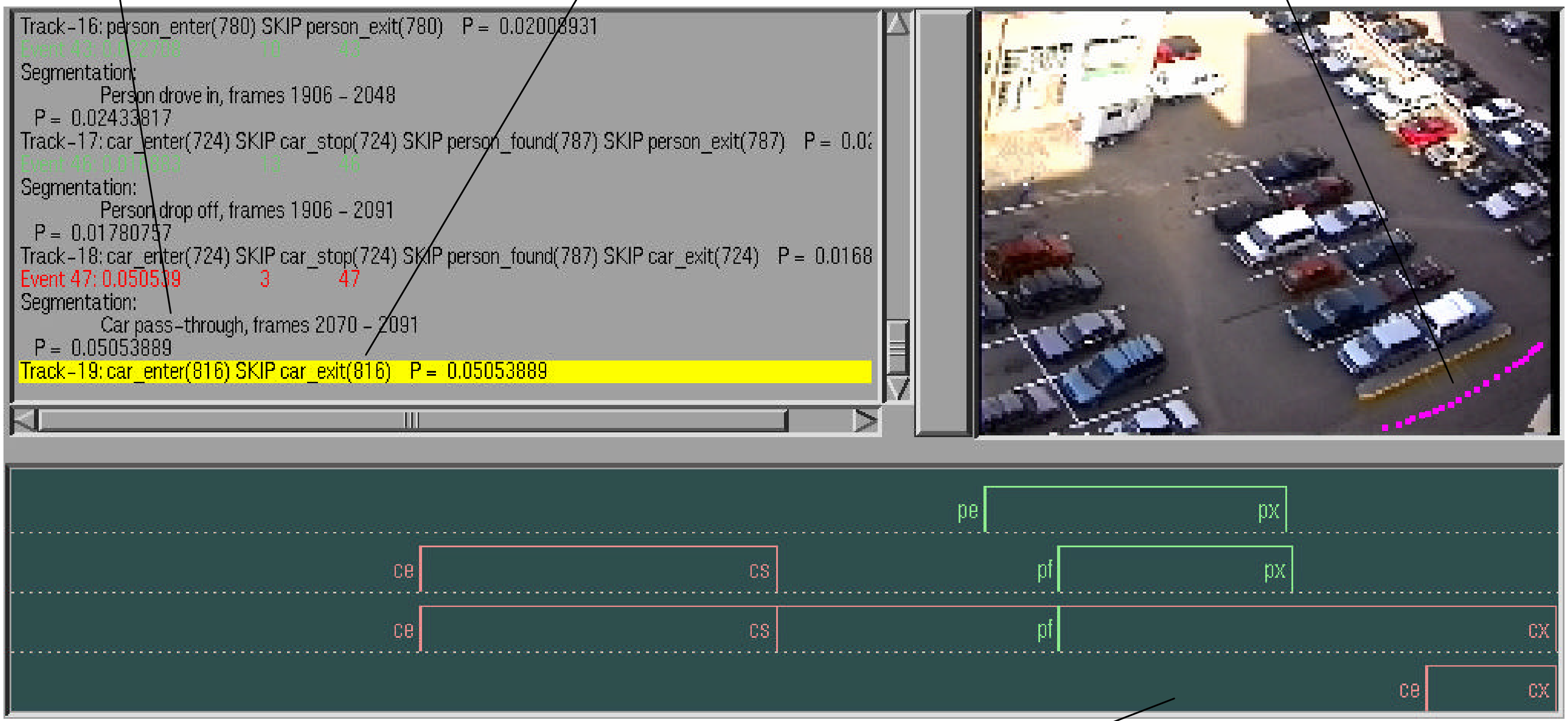
Parse 3: Drop-off



Figures from Ivanov, Yuri, Chris Stauffer, Aaron Bobick, W. E. L. Grimson. "Video Surveillance of Interactions." *IEEE Workshop on Visual Surveillance (ICCV 2001)* (1999). Courtesy of IEEE, Yuri Ivanov, Chris Stauffer, Aaron Bobick, and W. E. L. Grimson.

Parse 4: Car-Pass-Through

Action label   Component labels   Object track



Temporal extent

Figures from Ivanov, Yuri, Chris Stauffer, Aaron Bobick, W. E. L. Grimson. "Video Surveillance of Interactions." *IEEE Workshop on Visual Surveillance (ICCV 2001)* (1999). Courtesy of IEEE, Yuri Ivanov, Chris Stauffer, Aaron Bobick, and W. E. L. Grimson.

Summary

- Real-time system
- First of a kind end-to end system
- Extended robust parsing algorithm
- Events are staged in real environment with other cars and people
- ~10-15 events per minute
- Staged events - 100% detected
- Accidental events - ~80% detected

Automatic Surveillance System

- Outdoor environment - occlusions and lighting changes
- Static cameras
- Real-time performance
- Labeling activities and person-vehicle interactions in a parking lot
- Handling simultaneous events

Appendix: Hu Moments

Image Moments

The two-dimensional $(p+q)$th order moments of a density distribution function $\rho(x,y)$ (e.g., image intensity) are defined in terms of Riemann integrals as:

$$m_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^p y^q \rho(x,y)\,dx\,dy, \qquad (1)$$

for $p, q = 0, 1, 2, \cdots$.

The central moments μ_{pq} are defined as:

$$\mu_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} (x-\bar{x})^p (y-\bar{y})^q \rho(x,y)\,d(x-\bar{x})\,d(y-\bar{y}), \qquad (2)$$

where

$$\bar{x} = m_{10}/m_{00},$$
$$\bar{y} = m_{01}/m_{00}.$$

It is well-known that under the translation of coordinates, the central moments do not change, and are therefore invariants under translation. It is quite easy to express the central moments μ_{pq} in terms of the ordinary moments m_{pq}. For the first four orders, we have

$$\begin{aligned}
\mu_{00} &= m_{00} \equiv \mu \\
\mu_{10} &= 0 \\
\mu_{01} &= 0 \\
\mu_{20} &= m_{20} - \mu\bar{x}^2 \\
\mu_{11} &= m_{11} - \mu\bar{x}\bar{y} \\
\mu_{02} &= m_{02} - \mu\bar{y}^2 \\
\mu_{30} &= m_{30} - 3m_{20}\bar{x} + 2\mu\bar{x}^3 \\
\mu_{21} &= m_{21} - m_{20}\bar{y} - 2m_{11}\bar{x} + 2\mu\bar{x}^2\bar{y} \\
\mu_{12} &= m_{12} - m_{02}\bar{x} - 2m_{11}\bar{y} + 2\mu\bar{x}\bar{y}^2 \\
\mu_{03} &= m_{03} - 3m_{02}\bar{y} + 2\mu\bar{y}^3.
\end{aligned}$$

To achieve invariance with respect to orientation and scale, we first normalize for scale defining η_{pq}:

$$\eta_{pq} = \frac{\mu_{pq}}{(\mu_{00})^{\gamma}},$$

where $\gamma = (p+q)/2 + 1$ and $p+q \geq 2$. The first seven orientation invariant Hu moments are defined as:

$$\begin{aligned}
\nu_1 &= \eta_{20} + \eta_{02} \\
\nu_2 &= (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \\
\nu_3 &= (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \\
\nu_4 &= (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2 \\
\nu_5 &= (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\
&\quad + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03}) \\
&\quad \cdot [3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\
\nu_6 &= (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\
&\quad + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03}) \\
\nu_7 &= (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\
&\quad - (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2].
\end{aligned}$$

These moments can be used for pattern identification independent of position, size, and orientation.