9.913 Pattern Recognition for Vision

Class XIII, Motion and Gesture
Yuri Ivanov
• Movement – Activity – Action
• View-based representation
• Sequence comparison
• Hidden Markov Models
• Hierarchical representations
How do we describe that?
How do we classify that?
From Tracking to Classification

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

Figure by MIT OCW.

Fall 2004
Pattern Recognition for Vision
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…

Basic Problem

Object trajectory in the feature space

\[ s_1 = x_1^1 \ldots x_M^1 \]

\[ s_2 = x_1^2 \ldots x_N^2 \]

\[ \begin{bmatrix} x_1^1 \\ y_1^1 \\ \theta_1^1 \\ \vdots \end{bmatrix} \]

\[ \begin{bmatrix} x_M^1 \\ y_M^1 \\ \theta_M^1 \\ \vdots \end{bmatrix} \]

\[ \begin{bmatrix} x_1^2 \\ y_1^2 \\ \theta_1^2 \\ \vdots \end{bmatrix} \]

\[ \begin{bmatrix} x_N^2 \\ y_N^2 \\ \theta_N^2 \\ \vdots \end{bmatrix} \]

\[ S_1 \leftrightarrow S_2 \]
Motion Energy Image

First idea – implicit representation of time

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

Sum the differences over the last \( \tau \) frames:

Motion History Image

Step two: include temporal information

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

- HOW motion happened

Aside - you can compute a similar measure recursively:

$$H_{\tau}(x, y, t) = H_{\tau}(x, y, t-1) + \alpha (D(x, y, t) - H_{\tau}(x, y, t-1))$$

Illustration

OpenCV – Intel Open source Computer Vision Library
Classification

Feature vector:
\[ x = [7 \text{ Hu moments for MEI} + 7 \text{ Hu moments for MHI}] \]

RTS invariant shape descriptors (see the end of notes)

With the usual Gaussian assumption on distribution of \( x \):

\[ \mu_\omega = E[x_\omega]; \quad \Sigma_\omega = E[(x_\omega - \mu_\omega)^2] \]

Then the class, \( \omega \):

\[ \omega = \arg\min \left[ (x - \mu_\omega)^T \Sigma_\omega^{-1} (x - \mu_\omega) \right] \]
Multi-View Recognition

The model is replicated for a discrete number of views:

For $\theta = \{0^\circ...90^\circ\}$

$$V(\omega_i) = \min_{\theta} \left[ (x - \mu_{\omega_i}^\theta)^T \Sigma_{\omega_i}^{-1,\theta} (x - \mu_{\omega_i}^\theta) \right]$$

$$\omega = \arg\min[V(\omega_i)]$$

Example Application

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

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

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

Reference object

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

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

Global normalization \hspace{1cm} 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

"sit"

"roll over"
Example: Utterance Classification

Alternative - pair-wise alignment:

SVM: \[ f(x) = \sum_{i=1}^{N} \alpha_i y_i K(x, x_i) + b \]

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

\[ K(s_i, s_j) = \exp(-\gamma d_{ij}) \]

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

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>KNN</td>
<td>94.60%</td>
</tr>
<tr>
<td>MCC</td>
<td>94.10%</td>
</tr>
<tr>
<td>HMM</td>
<td>96.20%</td>
</tr>
<tr>
<td>SVM</td>
<td>98.20%</td>
</tr>
<tr>
<td>DynSVM</td>
<td>98.20%</td>
</tr>
</tbody>
</table>
Hidden Markov Model (HM&M)

Yet another idea:

Output Probability
\[ B = p(o_t | q_t) \]

Transition Law
\[ A = p(q_t | q_{t-1}, \ldots, q_1) \]

State, \( q_t \)

We do not get to see this. It is "HIDDEN".

Only This

Observations
\[ o_1 \ldots o_T = \mathcal{O} \]
Hidden Markov Model (HM&M)

Yet another idea:

Output Probability
\[ B = p(o_t|q_t) \]

Transition Law
\[ A = p(q_t|q_{t-1},...,q_1) \]

State, \( q_t \)

We do not get to see this. It is "HIDDEN".

Only This

Observations
\[ o_1 \ldots o_T = \emptyset \]
HMMs

Another view – “Graphical Model”:

Figure by MIT OCW.
Components of an HMM

$$\lambda = \{ \pi, A, B \}$$

1) $\pi$ - probability of starting from a particular state
   $$\pi_i = p(q_t = i)$$
   $$\sum_{i=1}^{N} \pi_i = 1$$

2) $A$ - probability of moving to a state, given the history
   $$a_{ij} = p(q_t = i \mid q_{t-1} = j) - \text{Markov assumption}$$
   $$\sum_{j=1}^{N} a_{ij} = 1$$

3) $B$ - probability of outputing a particular observation from a given state:
   $$b_i(o) = p(o_t \mid q_t = i)$$
   $$\int b_i(x) \, dx = 1$$
HMM in Pictures

- $A = p(q_t|q_{t-1},...,q_1) = p(q_t|q_{t-1})$ – Markov assumption
- $p(q_t|q_{t-1})$ is independent of $t$ – stationary $\Rightarrow$ 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: \( O = (o_1, \ldots, o_T) \)

Calculate: \( P(O | \lambda) \)

Marginalize:

\[
P(O | \lambda) = \sum_{\forall q} P(O, q | \lambda) = \sum_{\forall q} P(O | q, \lambda) P(q | \lambda)
\]

\[
P(O | q, \lambda) = b_{q_1}(o_1)b_{q_2}(o_2) \cdots b_{q_T}(o_T)
\]

\[
P(q | \lambda) = \pi_{q_1} a_{q_1q_2} a_{q_2q_3} \cdots a_{q_{T-1}q_T}
\]

\[
P(O | q, \lambda) P(q | \lambda) = \pi_{q_1} b_{q_1}(o_1)a_{q_1q_2} b_{q_2}(o_2)a_{q_2q_3} b_{q_3}(o_3) \cdots a_{q_{T-1}q_T} b_{q_T}(o_T)
\]

\( N \) states, \( T \) transitions \( \Rightarrow |q| = N^T \) !!!!

\( N=5, T=100 \Rightarrow 2TN^T = 2 \times 100 \times 5^{100} \sim 10^{72} \) computations

65536 \times 10^{72} \) particles in the universe
\[ P(O \mid \lambda) = \sum_{q} P(O, q \mid \lambda) \]

\[ = \sum_{q=1}^{10^{72}} \pi_{q(1)} b_{q(1)} (o_1) a_{q(1)q(2)} b_{q(2)} (o_2) a_{q(2)q(3)} b_{q(3)} (o_3) \ldots a_{q(T-1)q(T)} b_{q(T)} (o_T) \]

\[ \approx 2TN^T \]

\[ = \sum_{m} \sum_{l} \cdots \sum_{j} \sum_{i} \pi_i b_i (o_1) a_{ij} b_j (o_2) a_{jk} b_k (o_3) \ldots a_{lm} b_m (o_T) \]

\[ \approx N^2T \]
Problem I – Probability Calculation

Take II – forward procedure:

Define a “forward variable”, $\alpha$

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

- 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”, $\beta$

$$\beta_t(i) = P(o_{t+1}o_{t+2}\ldots o_T \mid q_t = i, \lambda)$$ - probability of seeing the rest of the string after $t$ and after visiting state $i$ at $t$

1. Initialize

$$\beta_T(i) = 1$$

2. Induce

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

3. Terminate
Task II – Optimal State Sequence

“Optimality” - maximum probability of being in a state $i$ at time $t$.

Given: $O = (o_1, \ldots, o_T)$
Find: $q_t = \arg\max_q P(q_t \mid O, \lambda)$

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

by Bayes rule

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

\[
= P(o_1 \ldots o_t, q_t)P(o_{t+1} \ldots o_T \mid q_t) = \alpha_t \beta_t
\]
State Posterior

So,

\[
P(q_t = i \mid O) = \frac{P(O, q_t = i)}{\sum_{j=1}^{N} P(O, q_t = j)} = \frac{\alpha_t(i) \beta_t(i)}{\sum_{j=1}^{N} \alpha_t(j) \beta_t(j)} = \gamma_t(i)
\]

1. Forward pass – compute \( \alpha \) matrix \( \approx N^2 T \)
2. Backward pass – compute \( \beta \) 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: \( \mathbf{O} = (o_1, ..., o_T) \)
Find: \( \underset{q}{\text{argmax}} \ P(q | \mathbf{O}, \lambda) \)

Define:
\[
\delta_t(i) = \max_{q_t q_{t-1} ... q_1} P(q_1...q_{t-1}, q_t = i, o_1...o_t)
\]

By the optimality principle (Bellman, ’57):
\[
\delta_{t+1}(j) = \max_i \delta_t(i) a_{ij} b_j(o_{t+1})
\]

Just need to keep track of max probability states along the way
Task II – Viterbi Algorithm (cont.)

1. Initialize
   \[ \delta_1(i) = \pi_i b_i(o_1) \quad 1 \leq i \leq N \]
   \[ \psi_1(i) = 0 \quad \text{Housekeeping variable} \]

2. Recurse
   \[ \delta_t(j) = \max_{1 \leq i \leq N} \left[ \delta_{t-1}(i) a_{ij} \right] b_j(o_t) \quad 2 \leq t \leq T \]
   \[ \psi_t(j) = \arg\max_{1 \leq i \leq N} \left[ \delta_{t-1}(i) a_{ij} \right] \quad 2 \leq t \leq T \]

3. Terminate
   \[ P^* = \max_{1 \leq i \leq N} \delta_T(i) \]
   \[ q_T^* = \arg\max_{1 \leq i \leq N} \delta_T(i) \]

4. Backtrack
   \[ q_t^* = \psi_{t+1}(q_{t+1}^*) \quad t = (T - 1), \ldots, 1 \]
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
Task III – Parameter Estimation

Baum-Welch algorithm (EM for HMMs)

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

Find: \( \pi, A, B \)

First, introduce another greek letter:

\[
\xi_t(i, j) = P(q_t = i, q_{t+1} = j \mid \mathbf{O}) = \frac{P(q_t = i, q_{t+1} = j, \mathbf{O})}{P(\mathbf{O})}
\]

\[
= \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_t(j)}{P(\mathbf{O})}
\]

\[
\alpha_t(i)
\]

\[
\beta_j(t + 1)
\]
Transition Probability

This leads to:

\[
\overline{a}_{ij} = \frac{E[\#(i \rightarrow j)]}{E[\#(i \rightarrow .)]} = \frac{\sum_{t=1}^{T-1} \xi_t(i, j)}{\sum_{t=1}^{T-1} \gamma_t(i)}
\]

The rest is easy
Prior distribution:

\[ \pi_i = E[\#(i, t = 1)] = \gamma_1(i) \]

Output distribution (discrete):

\[ \bar{b}_i(k) = \frac{E[\#(i, v_k)]}{E[\#(i)]} = \frac{\sum_{t=1}^{T} \gamma_t(i)}{\sum_{t=1}^{T} \gamma_t(i)} \]

Sum probabilities of being in state \( i \) while seeing symbol \( v_k \)

Normalize

This Fraction is \( b_1(GREEN) \)

Figure by MIT OCW.
Continuous Output Case

Output distribution (continuous, Gaussian):

\[ \bar{b}_i(o) = N(\mu_i, \Sigma_i) \]

\[ \bar{\mu}_i = \frac{\sum_{t=1}^{T} \gamma_t(i) \cdot o_t}{\sum_{t=1}^{T} \gamma_t(i)} \]

\[ \bar{\Sigma}_i = \frac{\sum_{t=1}^{T} \gamma_t(i) \cdot (o_t - \mu_i)(o_t - \mu_i)^T}{\sum_{t=1}^{T} \gamma_t(i)} \]

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

*These should look VERY familiar*
Semi-Continuous HMM Example

Input

Output distributions

Initial state

Transition matrix

How HMM sees it
Gesture Recognition – Trajectory Model

Modeling a tracked hand trajectory.
HMM Classifier

Nothing unusual:

Input Sequence

Bank of HMMs

ArgMax

λ

λ₁

λ₂

λ₃

λ₄

Fall 2004

Pattern Recognition for Vision
Applications – American Sign Language

Task: Recognition of sentences of American Sign Language

40 word lexicon:

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

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

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

Features (from skin model):

\[ o = \left( x, y, dx, dy, area, \theta, \lambda_{\text{max}}, \lambda_{\text{min}} / \lambda_{\text{min}} \right)_{\text{right}}, (...)_{\text{left}} \]^

System 1: Second person  
System 2: First person

Courtesy of Thad Starner. Used with permission.

Nose could be used for initializing the skin model
### Applications – American Sign Language

#### 500 sentences (400 training, 100 testing)

**System 1:**

<table>
<thead>
<tr>
<th>experiment</th>
<th>training set</th>
<th>test set</th>
</tr>
</thead>
<tbody>
<tr>
<td>all features</td>
<td>94.10%</td>
<td>91.90%</td>
</tr>
<tr>
<td>relative features</td>
<td>89.60%</td>
<td>87.20%</td>
</tr>
<tr>
<td>all features &amp; unrestricted grammar</td>
<td>81.0% (87%)</td>
<td>74.5% (83%)</td>
</tr>
<tr>
<td>(D=31, S=287, I=137, N=2390)</td>
<td>(D=3, S=76, I=41, N=470)</td>
<td></td>
</tr>
</tbody>
</table>

**System 2:**

<table>
<thead>
<tr>
<th>grammar</th>
<th>training set</th>
<th>test set</th>
</tr>
</thead>
<tbody>
<tr>
<td>part-of-speech</td>
<td>99.30%</td>
<td>97.80%</td>
</tr>
<tr>
<td>5-word sentence</td>
<td>98.2% (98.4%)</td>
<td>97.80%</td>
</tr>
<tr>
<td>(D = 5, S=36, I=5 N =2500)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>unrestricted</td>
<td>96.4% (97.8%)</td>
<td>96.8% (98.0%)</td>
</tr>
<tr>
<td>(D=24, S=32, I=35, N=2500)</td>
<td>(D=4, S=6, I=6, N=500)</td>
<td></td>
</tr>
</tbody>
</table>

Word accuracy, \( \frac{D + S + I}{N} \)
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

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

HMM Bank

Q1

H1

Q2

H2

Q3

H3

Emitter

Parsing Module

Parser

Viterbi Segmenter

Annotator

D1

D2
Application: Conducting Music

“Dictionary” gestures
Application: Conducting Music

Jean Sibelius, Second Symphony, Opus 43, D Major

Grammar:

\[ G_c : \]

\[
\begin{align*}
\text{PIECE} & \rightarrow \text{BAR PIECE} [0.5] \\
| & \rightarrow \text{BAR} [0.5] \\
\text{BAR} & \rightarrow \text{TWO} [0.5] \\
| & \rightarrow \text{THREE} [0.5] \\
\text{THREE} & \rightarrow \text{down3 right3 up3} [1.0] \\
\text{TWO} & \rightarrow \text{down2 up2} [1.0]
\end{align*}
\]

Correct

<table>
<thead>
<tr>
<th></th>
<th>~70%</th>
<th>~85%</th>
<th>~95%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Individual</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Component</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bar</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Fall 2004

Pattern Recognition for Vision
Component Detection

- Input sequence
- HMM Likelihoods
- Peaks of likelihood
- Single event
Temporal Consistency

Grammar:

\[ A \rightarrow ab \mid abA \]

Input

---

a b a a b b b a b a b b a b
Temporal Consistency

Grammar:

\[ A \rightarrow ab \mid abA \]

Input

Terminals have temporal extent!

Temporal Consistency

Grammar:

\[ A \rightarrow ab \mid abA \]

Input

Inconsistent parse

Terminals have temporal extent!

\[ S = ababab \]

Temporal Consistency

Grammar:

\[ A \rightarrow ab \mid abA \]

Input

Inconsistent parse

Consistent parse

Terms have temporal extent!

The idea is that the top level parse will filter out mistakes in low level detections.
Stochastic Context-Free Grammar

Example Grammar:

- **Non-terminals** - semantically significant groups of events
- **Terminals** - individual events
- **SKIP-rule** - noise symbol
- **Rule probabilities**

```
<table>
<thead>
<tr>
<th>Non-terminals</th>
<th>Terminals</th>
<th>Rule probabilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRACK</td>
<td>CAR-TRACK</td>
<td>[0.50]</td>
</tr>
<tr>
<td></td>
<td>PERSON-TRACK</td>
<td>[0.50]</td>
</tr>
<tr>
<td>CAR-TRACK</td>
<td>CAR-THROUGH</td>
<td>[0.25]</td>
</tr>
<tr>
<td></td>
<td>CAR-PICKUP</td>
<td>[0.25]</td>
</tr>
<tr>
<td></td>
<td>CAR-OUT</td>
<td>[0.25]</td>
</tr>
<tr>
<td></td>
<td>CAR-DROP</td>
<td>[0.25]</td>
</tr>
<tr>
<td>CAR-THROUGH</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>car-exit</td>
<td>[0.70]</td>
</tr>
<tr>
<td></td>
<td>SKIP car-exit</td>
<td>[0.30]</td>
</tr>
</tbody>
</table>
```

Fall 2004 Pattern Recognition for Vision
Event Parsing

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

- production rules (states)
  
  $X \rightarrow Zc$
  
  $Z \rightarrow ab$

- target non-terminal (label)
  
  $X$

- intermediate non-terminal
  
  $Z$

- input stream (tracking events)
  
  - $a$
  
  - $b$
  
  - $c$
  
  - $p$
  
  - $q$
  
  - $r$

- noise rules
  
  - SKIP
  
  - SKIP
Application: Musical Conducting

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

<table>
<thead>
<tr>
<th>Component</th>
<th>Correct</th>
</tr>
</thead>
<tbody>
<tr>
<td>Individual</td>
<td>~70%</td>
</tr>
<tr>
<td>Component</td>
<td>~85%</td>
</tr>
<tr>
<td>Bar</td>
<td>~95%</td>
</tr>
</tbody>
</table>

Courtesy of Teresa Marrin-Nakra. Used with permission.
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

- **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

\[ P(X_t) = \sum_{i=1}^{K} w_{i,t} \ast \eta(X_t, \mu_{i,t}, \Sigma_{i,t}) \]

- 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

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:

<table>
<thead>
<tr>
<th>Rule</th>
<th>Right-hand Side</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRACK →</td>
<td>CAR-TRACK</td>
<td>[0.5]</td>
</tr>
<tr>
<td></td>
<td>PERSON-TRACK</td>
<td>[0.5]</td>
</tr>
<tr>
<td>CAR-TRACK →</td>
<td>CAR-THROUGH</td>
<td>[0.25]</td>
</tr>
<tr>
<td></td>
<td>CAR-PICKUP</td>
<td>[0.25]</td>
</tr>
<tr>
<td></td>
<td>CAR-OUT</td>
<td>[0.25]</td>
</tr>
<tr>
<td></td>
<td>CAR-DROP</td>
<td>[0.25]</td>
</tr>
<tr>
<td>CAR-PICKUP →</td>
<td>ENTER-CAR-B CAR-STOP PERSON-LOST B-CAR-EXIT</td>
<td>[1.0]</td>
</tr>
<tr>
<td>ENTER-CAR-B →</td>
<td>CAR-ENTER</td>
<td>[0.5]</td>
</tr>
<tr>
<td></td>
<td>CAR-ENTER CAR-HIDDEN</td>
<td>[0.5]</td>
</tr>
<tr>
<td>CAR-HIDDEN →</td>
<td>CAR-LOST CAR-FOUND</td>
<td>[0.5]</td>
</tr>
<tr>
<td></td>
<td>CAR-LOST CAR-FOUND CAR-HIDDEN</td>
<td>[0.5]</td>
</tr>
<tr>
<td>B-CAR-EXIT →</td>
<td>CAR-EXIT</td>
<td>[0.5]</td>
</tr>
<tr>
<td></td>
<td>CAR-HIDDEN CAR-EXIT</td>
<td>[0.5]</td>
</tr>
<tr>
<td>CAR-EXIT →</td>
<td>car-exit</td>
<td>[0.7]</td>
</tr>
<tr>
<td></td>
<td>SKIP car-exit</td>
<td>[0.3]</td>
</tr>
<tr>
<td>CAR-LOST →</td>
<td>car-lost</td>
<td>[0.7]</td>
</tr>
<tr>
<td></td>
<td>SKIP car-lost</td>
<td>[0.3]</td>
</tr>
<tr>
<td>CAR-STOP →</td>
<td>car-stop</td>
<td>[0.7]</td>
</tr>
<tr>
<td></td>
<td>SKIP car-stop</td>
<td>[0.3]</td>
</tr>
<tr>
<td>PERSON-LOST →</td>
<td>person-lost</td>
<td>[0.7]</td>
</tr>
<tr>
<td></td>
<td>SKIP person-lost</td>
<td>[0.3]</td>
</tr>
</tbody>
</table>

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

\[ \mathbf{r} = (x, y), \quad d\mathbf{r} = (dx, dy) \]

Predict new position:
\[ \mathbf{r}_p = \mathbf{r}_1 + d\mathbf{r}_1(t_2 - t_1) \]

Penalize:
\[
 f(\mathbf{r}_p, \mathbf{r}_2) = \begin{cases} 
 0, & \text{if } (t_2 - t_1) < 0 \\
 \exp\left( \frac{(\mathbf{r}_2 - \mathbf{r}_p)^T(\mathbf{r}_2 - \mathbf{r}_p)}{\theta} \right) & \text{otherwise}
\end{cases}
\]
Input Data
Interleaved events in the input stream

Parse 1: Person-Pass-Through

Parse 3: Drop-off

Action label

Component labels

Object track

Temporal extent

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:

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

for \(p, q = 0, 1, 2, \ldots\).

The central moments \(\mu_{pq}\) are defined as:

\[
\mu_{pq} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} (x-x_c)^p (y-y_c)^q \rho(x, y) \, dx \, dy,
\]

where

\[
x = m_{10}/m_{00},
\]

\[
y = m_{01}/m_{00}.
\]

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 \(\mu_{pq}\) in terms of the ordinary moments \(m_{pq}\).

For the first four orders, we have

\[
\mu_{00} = m_{00} \equiv \mu
\]

\[
\mu_{10} = 0
\]

\[
\mu_{01} = 0
\]

\[
\mu_{20} = m_{20} - \mu x^2
\]

\[
\mu_{11} = m_{11} - \mu x y
\]

\[
\mu_{02} = m_{02} - \mu y^2
\]

\[
\mu_{30} = m_{30} - 3m_{21}x + 2\mu x^3
\]

\[
\mu_{21} = m_{21} - 3m_{12}y + 2\mu x^2 y
\]

\[
\mu_{12} = m_{12} - 3m_{03}x - 2m_{11}y + 2\mu x y^2
\]

\[
\mu_{03} = m_{03} - 3m_{21}y + 2\mu y^3.
\]

To achieve invariance with respect to orientation and scale, we first normalize for scale defining \(\eta_{pq}\):

\[
\eta_{pq} = \frac{\mu_{pq}}{(\mu_{00})^{\gamma}}
\]

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

\[
\nu_1 = \eta_{30} + \eta_{21}
\]

\[
\nu_2 = (\eta_{30} - \eta_{21})^2 + 4\eta_{11}^2
\]

\[
\nu_3 = (\eta_{30} - 3\eta_{21})^2 + (3\eta_{21} - \eta_{10})^2
\]

\[
\nu_4 = (\eta_{30} + \eta_{21})^2 + (\eta_{21} + \eta_{10})^2
\]

\[
\nu_5 = (\eta_{30} - 3\eta_{21})(\eta_{30} + \eta_{21})(\eta_{30} + \eta_{10})^2 - 3(\eta_{21} + \eta_{10})^2
\]

\[
+ (3\eta_{21} - \eta_{10})(\eta_{21} + \eta_{10})
\]

\[
+ [3(\eta_{30} + \eta_{10})^2 - (\eta_{21} + \eta_{10})^2]
\]

\[
\nu_6 = (\eta_{30} - \eta_{10})[(\eta_{30} + \eta_{10})^2 - (\eta_{21} + \eta_{10})^2]
\]

\[
+ 4\eta_{11}(\eta_{30} + \eta_{10})(\eta_{21} + \eta_{10})
\]

\[
\nu_7 = (3\eta_{21} - \eta_{30})(\eta_{30} + \eta_{10})(\eta_{30} + \eta_{10})^2 - 3(\eta_{21} + \eta_{10})^2
\]

\[
- (\eta_{30} - 3\eta_{10})(\eta_{21} + \eta_{10})[3(\eta_{30} + \eta_{10})^2 - (\eta_{21} + \eta_{10})^2].
\]

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