Templates and Background Subtraction

Prof. D. Stricker
Doz. G. Bleser
Surveillance
Video: Example of multiple people tracking

http://www.youtube.com/watch?v=InqV34BcheM&feature=player_embedded
As for the area “object category classification”:

**Representation choice of the object**

- Window-based
- Part-based
Advantages and Disadvantages

Part-Based
- May be better able to deal with moving body parts
- May be able to handle occlusion, overlaps
- Requires more complex reasoning

Global approaches
- Typically simple, i.e. we train a discriminative classifier on top of the feature descriptions
- Work well for small resolutions
- Typically does detection via classification, i.e. uses a binary classifier
Detection via classification: Main idea

Basic component: a binary classifier
Detection via classification: Main idea

If object may be in a cluttered scene, slide a window around looking for it.
Window-based models: Building an object model

Simple holistic descriptions of image content

- grayscale / color histogram
- vector of pixel intensities

Kristen Grauman
Window-based models: Building an object model

- Pixel-based representations sensitive to small shifts

- Color or grayscale-based appearance description can be sensitive to illumination and intra-class appearance variation
Window-based models: Building an object model

• Consider edges, contours, and (oriented) intensity gradients
Window-based models: Building an object model

- Consider edges, contours, and (oriented) intensity gradients

- Summarize local distribution of gradients with histogram
  - Locally orderless: offers invariance to small shifts and rotations
  - Contrast-normalization: try to correct for variable illumination
Window-based models: Building an object model

Given the representation, train a binary classifier

![Diagram showing Car/non-car Classifier with Yes, car. and No, not a car. as outputs.](image)
Discriminative classifier construction

Nearest neighbor
Shakhnarovich, Viola, Darrell 2003
Berg, Berg, Malik 2005...

Neural networks
LeCun, Bottou, Bengio, Haffner 1998
Rowley, Baluja, Kanade 1998...

Support Vector Machines
Guyon, Vapnik
Heisele, Serre, Poggio, 2001, ...

Boosting
Viola, Jones 2001,
Torralba et al. 2004,
Opelt et al. 2006, ...

Conditional Random Fields
McCallum, Freitag, Pereira 2000; Kumar, Hebert 2003 ...

Slide adapted from Antonio Torralba
Influential Works in Detection

• Sung-Poggio (1994, 1998) : ~1450 citations
  – Basic idea of statistical template detection (I think), bootstrapping to get “face-like” negative examples, multiple whole-face prototypes (in 1994)
  – “Parts” at fixed position, non-maxima suppression, simple cascade, rotation, pretty good accuracy, fast
  – Careful feature engineering, excellent results, cascade
  – Haar-like features, Adaboost as feature selection, hyper-cascade, very fast, easy to implement
  – Careful feature engineering, excellent results, HOG feature, online code
• Felzenszwalb-Huttenlocher (2000): ~800
  – Efficient way to solve part-based detectors
• Felzenszwalb-McAllester-Ramanan (2008)? ~350
  – Excellent template/parts-based blend
Basic framework

• Learn an object model from training samples or online
  • Choose a representation
  • Learn or fit parameters of model / classifier

• Generate candidates in new image

• Score the candidates
Window-based models: Generating and scoring candidates

Car/non-car Classifier

Kristen Grauman
Window-based object detection: recap

Training:
1. Obtain training data
2. Define features
3. Define classifier

Given new image:
1. Slide window
2. Score by classifier

Kristen Grauman
Gradient Histograms
2d Global Detector
Dalal and Triggs, CVPR 2005

- 3-D Histogram of Oriented Gradients (HOG) as descriptors
- Linear SVM for runtime efficiency
- Tolerates different poses, clothing, lighting and background
- Currently works for fully visible upright persons
Gradient Histograms

• Have become extremely popular and successful in the vision community

• Avoid hard decisions compared to edge based features

• Examples:
  • SIFT (Scale-Invariant Image Transform)
  • GLOH (Gradient Location and Orientation Histogram)
  • HOG (Histogram of Oriented Gradients)
Computing gradients: recall

One sided: \[ f'(x) = \lim_{h \to 0} \frac{f(x + h) - f(x)}{h} \]

Two sided: \[ f'(x) = \lim_{h \to 0} \frac{f(x + h) - f(x - h)}{2h} \]

Filter masks in x-direction
- One sided: \begin{bmatrix} -1 & 1 \end{bmatrix}
- Two sided: \begin{bmatrix} -1 & 0 & 1 \end{bmatrix}

Gradient:
- Magnitude: \[ s = \sqrt{s_x^2 + s_y^2} \]
- Orientation: \[ \theta = \arctan \left( \frac{s_y}{s_x} \right) \]
Histograms

Gradient histograms measure the orientations and strengths of image gradients within an image region.
Example: SIFT descriptor (recall)

The most popular gradient-based descriptor
Typically used in combination with an interest point detector

- Region rescaled to a grid of 16x16 pixels
- 4x4 regions = 16 histograms (concatenated)
- Histograms: 8 orientation bins, gradients weighted by gradient magnitude
- Final descriptor has 128 dimensions and is normalized to compensate for illumination differences
Histories of Oriented Gradients

- Gradient-based feature descriptor developed for people detection
  - Authors: Dalal & Triggs (INRIA Grenoble, F)

- Global descriptor for the complete body

- Very high-dimensional
  - Typically ~4000 dimensions
HOG

Very promising results on challenging data sets

Phases
1. Learning Phase
2. Detection Phase
Detector: Learning Phase

1. Learning

- Create normalised train image data set
- Encode images into feature spaces
- Learn binary classifier
- Object/Non-object decision

- Set of cropped images containing pedestrians in normal environment
- Global descriptor rather than local features
- Using linear SVM
Detector: Detection Phase

2. Detection

- Scan image at all scales and locations
- Run classifier to obtain object/non-object decisions
- Fuse multiple detections in 3-D position & scale space
- Object detections with bounding boxes

Sliding window over each scale
Simple SVM prediction
Descriptor

1. Compute gradients on an image region of 64x128 pixels

2. Compute histograms on ‘cells’ of typically 8x8 pixels (i.e. 8x16 cells)

3. Normalize histograms within overlapping blocks of cells (typically 2x2 cells, i.e. 7x15 blocks)

4. Concatenate histograms
Gradients

- Convolution with \([-1 \ 0 \ 1]\) filters
- No smoothing
- Compute gradient magnitude+direction
- Per pixel: color channel with greatest magnitude -> final gradient
Gradient: alternatives

-1 0 1  
centered

-1 1  
uncentered

1 -8 0 8 -1  
cubic-corrected

0 1  

-1 0  
diagonal

-1 0 1  

-2 0 2  

-1 0 1  
Sobel
Cell histograms

- 9 bins for gradient orientations (0-180 degrees)
- Filled with magnitudes
- Interpolated trilinearly:
  - Bilinearly into spatial cells
  - Linearly into orientation bins
Linear and Bilinear interpolation for subsampling

Linear:

\[ y = y_0 + (x - x_0) \frac{y_1 - y_0}{x_1 - x_0} \]

Bilinear:

\[ f(R_1) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21}) \quad \text{where} \quad R_1 = (x, y_1), \]

\[ f(R_2) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22}) \quad \text{where} \quad R_2 = (x, y_2). \]

\[ f(P) \approx \frac{y_2 - y}{y_2 - y_1} f(R_1) + \frac{y - y_1}{y_2 - y_1} f(R_2). \]
Histogram interpolation example

θ = 85 degrees
Distance to bin centers
  • Bin 70 -> 15 degrees
  • Bin 90 -> 5 degrees
Ratios: 5/20 = 1/4, 15/20 = 3/4

Distance to bin centers
  • Left: 2, Right: 6
  • Top: 2, Bottom: 6
Ratio Left-Right: 6/8, 2/8
Ratio Top-Bottom: 6/8, 2/8
Ratios:
  • 6/8 * 6/8 = 36/64 = 9/16
  • 6/8 * 2/8 = 12/64 = 3/16
  • 2/8 * 6/8 = 12/64 = 3/16
  • 2/8 * 2/8 = 4/64 = 1/16
Blocks

- Overlapping blocks of 2x2 cells

- Cell histograms are concatenated and then normalized
  - Note that each cell several occurrences with different normalization in final descriptor

- Normalization
  - Different norms possible
  - We add a normalization epsilon to avoid division by zero
Blocks

- Gradient magnitudes are weighted according to a Gaussian spatial window

- Distant gradients contribute less to the histogram
Final Descriptor

Concatenation of Blocks

Visualization:

Dr. Edgar Seemann
Engineering

• Developing a feature descriptor requires a lot of engineering
  • Testing of parameters (e.g. size of cells, blocks, number of cells in a block, size of overlap)
  • Normalization schemes (e.g. L1, L2-Norms etc., gamma correction, pixel intensity normalization)

• An extensive evaluation of different choices was performed, when the descriptor was proposed

• It’s not only the idea, but also the engineering effort
Demo
How to simplify that?

If the camera is static, yes, we can....
Background subtraction

Simple techniques can do ok with static camera
…But hard to do perfectly

Widely used:

• Traffic monitoring (counting vehicles, detecting & tracking vehicles, pedestrians),
• Human action recognition (run, walk, jump, squat),
• Human-computer interaction
• Object tracking
Blue/Green Screen
An example: The Ultimatte system
Background subtraction

- Goal: Separation of **Foreground** (object) and **Background** (everything else = noise)

- Result could be a
  - Binary image, containing foreground only
  - Probability image, containing the likelihood of each pixel being foreground

- Useful for
  - further processing, such as using silhouettes, etc.
  - Less processing, less room for errors

- Approaches
  - Motion-based
  - Color-based
  - Some approaches can learn!
Background Subtraction

A largely unsolved problem…

One video frame  Estimated background  Difference Image  Thresholded Foreground on blue
Simple Approach

1. Estimate the background for time $t$.
2. Subtract the estimated background from the input frame.
3. Apply a threshold, $Th$, to the absolute difference to get the foreground mask.
Why is it Hard?

1. Illumination Changes
   - Gradual (evening to night)
   - Sudden (overhead clouds)

2. Changes in the background geometry
   - Parked cars (should become part of the background)

3. Camera related issues
   - Camera oscillations (shaking)
   - Grainy noise

4. Changes in background objects
   - Tree branches
   - Sea waves
Foreground-Background Segmentation using Motion and Color

• Motion-based
  – Model-free
  – No learning
  – Image differencing

• Color-based
  • Background subtraction
    – Background used as a model
    – No learning

  • Advanced background subtraction
    – Background is learned

  • Very advanced background subtraction (Gaussian Mixture Models)
    – Background is learned
Motion-based: Image Differencing
Frame Differencing

- Background is estimated to be the previous frame. Background subtraction equation then becomes:

\[ B(x, y, t) = I(x, y, t - 1) \]

\[ \implies |I(x, y, t) - I(x, y, t - 1)| > Th \]

- Depending on the object structure, speed, frame rate and global threshold, this approach may or may not be useful (usually not).

Slide credit: Birgi Tamersoy
Frame Differencing

\[ Th = 25 \]  \hspace{2cm}  \[ Th = 50 \]

\[ Th = 100 \]  \hspace{2cm}  \[ Th = 200 \]

Slide credit: Birgi Tamersoy
5. Deleting Noise

Singular pixels are likely to appear:
  • Pixel-noise!!

Apply Median filter:
  • Depending on filter size, bigger spots can be erased

Alternative: Morphologic

1. Save image in last frame
2. Capture camera image
3. Subtract image
4. Threshold
5. **Delete noise**
Recall: Median filtering

- A **median filter** operates over a window by selecting the median intensity in the window.
Background Subtraction
Background Subtraction

Foreground is moving, background is stable

Algorithm

1. Capture image containing background
2. Capture camera image
3. Subtract image (= difference = motion)
4. Threshold
5. Delete noise
Advanced Background Subtraction
Advanced Background Subtraction

• What if we have small motion in the background?
  • Bushed, leaves, etc. and noise in the camera/lighting

• **Learn(!) the background**
  • Initialization over n frame
  • Update after subtraction: \( B_{i+1} = \alpha I_i + (1 - \alpha) B_i \)

• Capture \( N \) images and calculate the average background image (no object present)

1. Calculate average background image
2. Capture camera image
3. Subtract image (= motion)
4. Threshold
5. Delete noise
Mean Filter

- In this case the background is the mean of the previous $n$ frames:

$$B(x, y, t) = \frac{1}{n} \sum_{i=0}^{n-1} I(x, y, t - i)$$

$$|I(x, y, t) - \frac{1}{n} \sum_{i=0}^{n-1} I(x, y, t - i)| > Th$$

- For $n = 10$:

Estimated Background

Foreground Mask

Slide credit: Birgi Tamesoy
Median Filter

- Assuming that the background is more likely to appear in a scene, we can use the median of the previous $n$ frames as the background model:

$$B(x, y, t) = \text{median}\{I(x, y, t - i)\}$$

$$\Downarrow$$

$$|I(x, y, t) - \text{median}\{I(x, y, t - i)\}| > Th \text{ where } i \in \{0, \ldots, n - 1\}.$$ 

- For $n = 10$:
  
  Estimated Background
  
  Foreground Mask

Slide credit: Birgi Tamersoy
Average/Median Image
Background Subtraction

Alyosha Efros, CMU
Pros and cons

- **Advantages:**
  - Extremely easy to implement and use!
  - All pretty fast.
  - Corresponding background models need not be constant, they change over time.

- **Disadvantages:**
  - Accuracy of frame differencing depends on object speed and frame rate
  - Median background model: relatively high memory requirements.
  - Setting global threshold Th…

Slide credit: Birgi Tamer soy
Very Advanced Background
Subtraction
Very Advanced Background Subtraction

1) Use Neighborhood relation!!
   • Compare pixel with its neighbors!!
   • Weight them!!

2) Learn the background and its variations!!
   • E.g. Gaussian models (mean, var) for each pixel!!!
   • E.g. a Histogram for each Pixel
   • The more images you train on the better!!
   • Idea:
     – Some pixel may vary more than other pixels
   • Algorithm:
     – Consider each pixel \((x,y)\) in the input image and check, how much it varies with respect to the mean and variance of the learned Gaussian models

1. Calculate mean and variance for each pixel
2. Capture camera image
3. Subtract image (= motion)
4. Weight the distances (new)
5. Threshold according to variance
6. Delete noise
Weight the Distances, Correlation between Pixel values

If one pixel is considered to be a foreground pixel

- Its neighbor is also likely to be a foreground pixel
- If its neighbor is not considered to be a foreground pixel, one of the two might be wrong
- Neighboring pixels are highly correlated (similar)

1. Calculate mean and variance for each pixel
2. Capture camera image
3. Subtract image (= motion)
4. **Weight the distances** (new)
5. Threshold according to variance
6. Delete noise
Weight the Distances

What does a pixel say about a foreground pixel, that is further away?

• Pixels with an increasing distance to each other are saying less about each other

• The correlation between pixels decreases with distance

1. Calculate mean and variance for each pixel
2. Capture camera image
3. Subtract image (= motion)
4. **Weight the distances (new)**
5. Threshold according to variance
6. Delete noise
Weight the Distances

Use a Gaussian for Weighting

To test a pixel $I(x,y)$

- Center a Gaussian on this pixel and weight the neighboring pixels accordingly => Convolution!

1D example:

<table>
<thead>
<tr>
<th>Signal:</th>
<th>60</th>
<th>48</th>
<th>2</th>
<th>3</th>
<th>1</th>
<th>222</th>
<th>100</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaussian filter:</td>
<td>0.25</td>
<td>0.5</td>
<td>0.25</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output:</td>
<td>39.5</td>
<td>13.75</td>
<td>2.25</td>
<td>56.75</td>
<td>136.3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Learn the background and its variations

Likelihood image or Probability image, containing the likelihood for each pixel of being a background/foreground pixel

1. Calculate mean and variance for each pixel
2. Capture camera image
3. Subtract image (= motion)
4. Weight the distances (new)
5. Threshold according to variance
6. Delete noise
A little detour to Statistics
Statistics

Mean

• Center of gravity of the object

\[ x_{\text{mean}} = \frac{1}{N} \sum_{i=1}^{N} x_i \]
\[ y_{\text{mean}} = \frac{1}{N} \sum_{i=1}^{N} y_i \]

Variance

• The Variance measures the variations of the object pixel positions around the center of gravity

\[ x_{\text{var}} = \frac{1}{N} \sum_{i=1}^{N} (x_i - x_{\text{mean}})^2 \]
\[ y_{\text{var}} = \frac{1}{N} \sum_{i=1}^{N} (y_i - y_{\text{mean}})^2 \]

\[ x_{\text{var}} \text{ is big. } y_{\text{var}} \text{ is small} \]
Statistics

Standard deviation: sigma (σ) \( \chi_{\text{sigma}} = \sqrt{\chi_{\text{var}}} \)

Normal distribution = Gaussian distribution

\[
\mathcal{N}(x|\mu, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp \left\{ -\frac{1}{2\sigma^2} (x - \mu)^2 \right\}
\]

<table>
<thead>
<tr>
<th>Range</th>
<th>Samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \sigma )</td>
<td>68.26 %</td>
</tr>
<tr>
<td>( 2\sigma )</td>
<td>95.44 %</td>
</tr>
<tr>
<td>( 3\sigma )</td>
<td>99.73 %</td>
</tr>
<tr>
<td>( 4\sigma )</td>
<td>99.99 %</td>
</tr>
</tbody>
</table>
How to use it

- "Automatic" thresholding based on statistics

Example: the color of the hand

- Training =>mean color
- Algorithm: hand pixel if: $TH_{\text{min}} < \text{pixel} < TH_{\text{max}}$
- How do we define $TH_{\text{min}}$ and $TH_{\text{max}}$?
- Use statistics: $TH_{\text{min}} = \text{mean}-2\sigma$ and $TH_{\text{max}} = \text{mean}+2\sigma$
Very Advanced Background Subtraction

TH should be different for each pixel

![Image](none)

Histograms for positions #1 and #2.
Threshold According to Variance

**Threshold** can be chosen depending on the variance
- A local threshold

**Standard Deviation**
For example:
- If $\text{Th}_{\text{min}} < \text{dist} < \text{Th}_{\text{max}}$ => object pixel
- $\text{Th}_{\text{min}} = \text{mean} - K\sigma$
- $\text{Th}_{\text{max}} = \text{mean} + K\sigma$
- $K$ is set by you! ($K=2$ usually)

\[
\sigma(x, y) = \sqrt{\text{var}(x, y)}
\]

<table>
<thead>
<tr>
<th>Range</th>
<th>Samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sigma$</td>
<td>68.26 %</td>
</tr>
<tr>
<td>$2\sigma$</td>
<td>95.44 %</td>
</tr>
<tr>
<td>$3\sigma$</td>
<td>99.73 %</td>
</tr>
<tr>
<td>$4\sigma$</td>
<td>99.99 %</td>
</tr>
</tbody>
</table>

1. Calculate mean and variance for each pixel
2. Capture camera image
3. Subtract image (= motion)
4. Weight the distances (new)
5. **Threshold according to variance**
6. Delete noise
Each pixel modeled with a mixture of Gaussians
Flexible to handle variations in the background

\[
\hat{p}(\bar{x} | \mathcal{X}_T, BG+FG) = \sum_{m=1}^{M} \hat{\pi}_m \mathcal{N}(\bar{x}; \hat{\mu}_m, \hat{\sigma}_m^2 I)
\]
Tracking
Tracking

Follow the object(s) over time

- Finding the trajectory (curve connecting the positions over time)

Simple tracking:

Advanced tracking:

- Cluttered background and multiple objects
Prediction

Given the position of the object in previous images, where do we think (predict) the object will be in the current image?

We need a **motion model**

The size of the search region depends on:

- The uncertainty of the prediction
- The framerate
- How fast can the object max. move?
Motion Model

Predicted position at time $t$:

**Brownian Motion**: According to a Gaussian model

0’th order: $(x_t, y_t) = (x_{t-1}, y_{t-1})$

1’th order: $x_t = \frac{(x_t, y_t)}{t-1} \cdot \Delta t + x_{t-1}$
- Similar for $y$
  $\dot{x}_{t-1}$: velocity in $x$ at time $t-1$

2’th order: $x_t = \frac{1}{2} \cdot \ddot{x}_{t-1} \cdot \Delta t^2 + \dot{x}_{t-1} \cdot \Delta t + x_{t-1}$
- Similar for $y$
  $\ddot{x}_{t-1}$: acceleration in $x$ at time $t-1$

Many other types exist: look at your application!
Readings

• Navneet Dalal and Bill Triggs, *Histograms of Oriented Gradients for Human Detection*

  Chris Stauer & W.E.L. Grimson, *Adaptive Background Mixture Models for Real-Time Tracking*

Thank you!!!