Using Histograms to Detect and Track Objects in Color Video

Michael Mason
Zoran Duric

Department of Computer Science
George Mason University

The problem:

Static camera, but the background is not entirely stable

Outdoor Scenes

-leaves and trees moving

Indoor Scenes

-varying lighting conditions

-cluttered backgrounds

Applications

Surveillance and safety:

- Security cameras, children, pedestrians

Related work

Background subtraction on a pixel by pixel basis

- Very sensitive to any movements of the camera or the background scene
- Required to “learn” the background (statistics)
- Many different methods have been developed for background / foreground segmentation

Results of previous work

Motivations

- Want to detect that something is happening in a video sequence without too many false positives.
- Do not want to spend a lot of time "learning" the background (or doing post-processing)
- Would like a fast and reliable method (accuracy is not paramount)

Approach

- Compare color histograms computed for small regions of the "background" image and the current (new) image (reduced color/ 12 bit bit representation)
- Compare color edge histograms computed for small regions of the "background" image and the current image (36 bin quantization)

Color Histograms

Reduced color representation =

C = (R/16) * 256 + (G/16)*16 + (B/16)

(This results in a 24 -> 12 bit color depth reduction)

This results in a 4096 bin histogram

- lowest 4 bits are less useful
- requires less storage
- faster implementation - easier to compare histograms

Color Edge Histograms

For each frame:

- Use the Sobel edge detector to compute edges in each color band $(r_x, r_y, g_x, g_y, b_x, b_y)$
- Combine the three color bands into the structure matrix, S, to compute the color edge response
- The edge strength is computed as the larger of the two eigenvalues of S, and the orientation is given by the corresponding eigenvector
- Histogram bin index is determined using edge orientation (36 bins total), and the bin count is incremented using the edge magnitude

Color Edge Histograms (cont.)

$$S = \begin{pmatrix} r_x^2 + g_x^2 + b_x^2 & r_x r_y + g_x g_y + b_x b_y \\ r_x r_y + g_x g_y + b_x b_y & r_y^2 + g_y^2 + b_y^2 \end{pmatrix}$$

S has two eigenvalues $\lambda_1 \geq \lambda_2 \geq 0$

If $\lambda_2 \approx 0$, we have an edge

λ_1 is a squared edge magnitude and the edge direction is given by the correponding eigenvector e_1

Histogram Matching

- Histogram Intersection

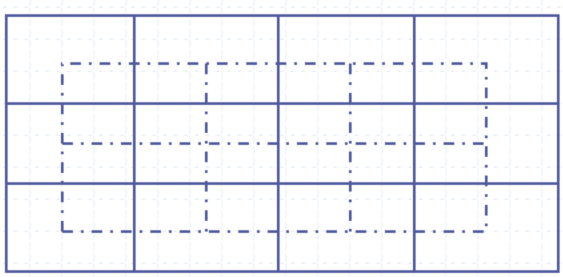
$$\frac{\sum_i \min\{h_c(i), h_b(i)\}}{\sum_i h_b(i)}$$

- Chi Squared Formula

$$X^2 = \sum_i 2\frac{(h_c(i) - h_b(i))^2}{h_c(i) + h_b(i)}$$

Overall control

- We divide each frame into *40x40* pixel blocks
- To make sure that we do not miss objects on grid block boundaries we tile the frame by overlaying two grids, one of which is shifted by *20* pixels in *x* and *y* directions

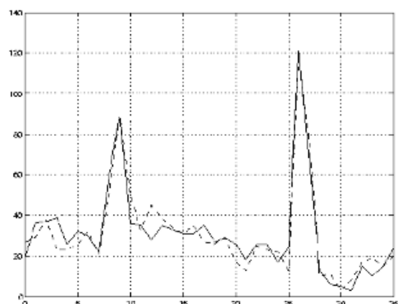


Criteria for block activation

- On a block by block basis, similarity measures between background and foreground histograms are computed
- For histogram intersection: If the similarity is below a threshold, *T*, then the block contains a foreground object and is activated for display
- For chi squared: If the X^2 measure is greater than a threshold, *T*, then the block contains a foreground object and is activated for display

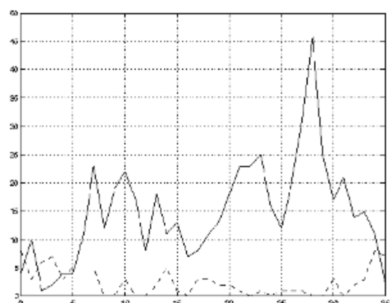
Examples of edge histograms

similar histograms



Similarity (inters.) = 92%
$X^2 = 61$

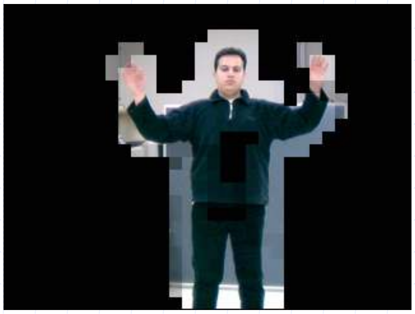
different histograms



Similarity (inters.) = 22%
$X^2 = 828$

Results…

Edge based detection

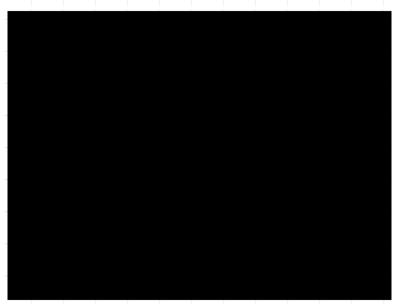


Moving person in a cluttered scene

Color histogram based detection

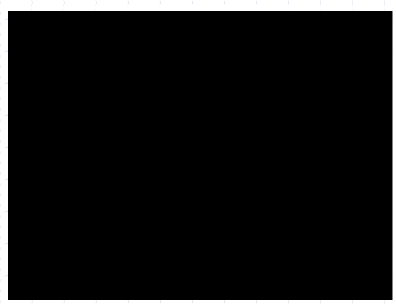
Edge histogram based detection

Surveillance: dropping an object



Surveillance: removing an object

Surveillance: Interacting people



Conclusions and future work

- Presented two histogram-based methods of detecting (tracking) objects in video
- The methods are fast and reliable and do not require prior training (learning of the scene)
- We are currently investigating:
  - Different histogram matching methods (unfolding histograms)
  - Tracking detected objects in video
  - Recognizing object classes from histograms of detected objects

Questions?