Elliptical Head Tracker using Intensity Gradients and Texture Histograms

Sriram Rangarajan,
Dept. of Electrical and Computer Engineering,
Clemson University, Clemson, SC 29634
srangar@clemson.edu

December 2004

1. Abstract

Various algorithms have been developed in the recent past for tracking a person's head. Here, an algorithm based on the elliptical head tracker described in [2] is introduced which uses intensity gradients and texture histograms to track a person's head. The algorithm models the head as a vertical ellipse with a fixed aspect ratio whose position and size are continually updated by the combined output of two modules, one focusing on the intensity gradients around the perimeter of the ellipse and the other on the texture histogram inside the ellipse. Texture histograms are obtained by convolving the image with a bank of log-Gabor filters. A model histogram is generated before the start of tracking and target histograms obtained from each frame are compared to generate a likelihood for the position of the head in the frame. This likelihood function is combined with the likelihood function from the intensity gradients to obtain the location of the head in the current frame. To facilitate a simple combination of the outputs from the two modules, each likelihood is normalized to a percentage score.

2. Introduction

Reliable,robust visual tracking of an object like a person's head is an interesting problem in computer vision. Though this involves integration of several modules, an interesting approach was used by [2] to simplify this problem. The use of two modules that are orthogonal to each other and also symmetric in operation produces a robust tracking algorithm which is insensitive to out-of-plane rotations, small lighting variations and occlusions.

Based on [1, 2], the head is approximated by a simple two-dimensional model namely an ellipse.The two modules used in this work,one which matches intensity gradients along an object's boundary and the other which matches the texture histogram of the object's interior, are orthogonal to each other in operation which provides robust tracking, since one module works even when the other completely fails. Though orthogonal in terms of operation, the modules are symmetric which makes the combination step trivial as suggested by [2].

The idea of using texture for tracking is based on the knowledge that very few trackers have been implemented with texture as a cue. Texture is similar to color in many ways and several successful trackers have made use of color histograms, due to their robustness to changing pose and object shape [2, 10, 13, 5, 6, 4, 14]. Based on this, the idea of using texture histograms to track a person's head was developed. The algorithm presented here, using texture histograms and intensity gradients, is insensitive to 360 degrees out-of-plane rotation of the target and small lighting changes.

3. Texture

Though texture does not have a formal definition, intuitively texture provides a measure of properties such as smoothness, coarseness and regularity etc. Extracting texture from images has been done using a variety of wavelet-based filters such as Gabor filters, log-Gabor filters,Haar wavelets and steerable filters [7]. Texture has been widely used in combination with color for image retrieval applications. log-Gabor filters have been selected for texture analysis here for reasons mentioned in subsequent sections.

3.1. Texture Histograms for tracking

Texture has been used for pattern recognition tasks like face recognition, iris recognition and image retrieval [9, 8, 11]. Though Color histograms have been used successfully to track a person's head, tracking based on texture has remained a less-explored area. Texture histograms have been used in image retrieval systems but have not been implemented for tracking. In the field of tracking, template-based tracking using texture has been implemented [3]. The head is modeled as a 3-D cylinder and the target is tracked by template matching. The idea of using a texture histogram

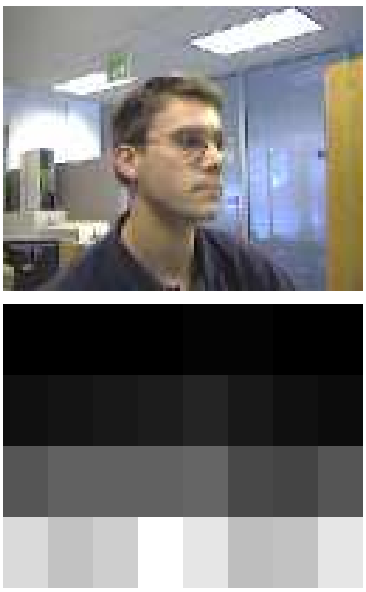

Figure 1: Image used to obtain the Model Histogram and The Model Texture Histogram

though has not been widely explored in tracking. The idea of using texture histograms for tracking is also based on the fact that texture is similar to color and computing a texture histogram is computationally less expensive compared to template-based methods.

3.2. Gabor Filters and log-Gabor Filters

Gabor filters have been used in texture analysis due to the fact that the real part of complex Gabor functions fits the receptive field weight functions found in simple cells in cat striate cortex. The Gabor function in space domain is given by

$$g(x, y) = s(x, y)w_r(x, y) \tag{1}$$

where $s(x, y)$ is a complex sinusoidal, known as the carrier,and $w_r(x, y)$ is a 2-D Gaussian known as the envelope. For a given image I(x,y) with size $(P \times Q)$ its discrete Gabor wavelet transform is given by a convolution:

$$G(x, y) = (\Sigma_s \Sigma_t I(x - s, y - t)\Psi^*_{mn}(s, t) \tag{2}$$

where, s and t are the filter mask size variables, and Ψ^*_{mn} is the complex conjugate of Ψ_{mn}, the frequency response of the Gabor function.

Gabor filters are a traditional choice for obtaining localized frequency information. They offer the best simultaneous localization of spatial and frequency information. However they have two main limitations. The maximum bandwidth of a Gabor filter is limited to approximately one octave and Gabor filters are not optimal if one is seeking broad spectral information with maximal spatial localization.

An alternative to the Gabor function is the log-Gabor function proposed by Field [1987]. Log-Gabor filters can be constructed with arbitrary bandwidth and the bandwidth can be optimized to produce a filter with minimal spatial extent. Field suggested that natural images are better coded by filters that have Gaussian transfer functions when viewed on the logarithmic frequency scale. On a linear frequency scale, the log-Gabor function has a transfer function of the form

$$G(w) = e^{(-log(w/w_0))/(2(log(k/w_0)^2)} \tag{3}$$

where w_0 is the filter's center frequency. The log-Gabor filter by definition does not have a DC component and has an extended tail at the high frequency end. Field suggested that log-Gabor functions, having extended tails, should be able to encode natural images more efficiently than ordinary Gabor functions, which would over-represent the low frequency components and under-represent the high frequency components in any encoding.Another point in support of the log-Gabor function is that it is consistent with measurements on mammalian visual systems which indicates that the cell responses are symmetric on the log frequency scale.

4. Tracking using Texture

Solving the tracking problem using texture histograms takes the same approach as that of tracking using color histograms. A model histogram is generated and then compared with the target histogram. The number of bins in the texture histogram is indirectly defined by the user, which depends on the scale and frequency settings for the filter bank. The number of scales and frequencies for the filter bank are user-defined and remain constant throughout tracking.The person's head is modeled by a vertical ellipse represented by $S = (x, y, \sigma)$ with a fixed aspect ratio of 1.2 which is updated by the likelihood function. Here, x and y represent the center of the ellipse and σ is the minor axis of the ellipse.

4.1. Acquiring the Model Histogram

Tracking a person is done over a sequence of images and the approach taken is similar to that of tracking the head using color histograms. A model histogram of the person's head to be tracked is generated by convolving the image with a three-quarters view of the head with the log-Gabor filter back. The three-quarters view of the head is chosen in order to capture both skin and hair.Before convolving with the filter bank, an ellipse mask is generated such that the resulting image contains only the head of the person.The histogram is then generated for four scales and eight orientations after convolution with the filter bank. The number of scales and orientations for the filter bank is set at the start which defines the size of the model histogram.

The model histogram obtained using texture is given as

$$n_b(S,O) = \sum_X \sum_Y abs(G_{S,O}(X,Y)) \tag{4}$$

The model histogram is represented by n_b, S represents the number of scales specified for the filter bank,O represents the number of orientations, and $G_{S,O}(X,Y))$ is the result of convolving the image with the filter bank.

From 4, it is clear that the histogram for each scale and orientation is the sum of the resulting filter outputs for that particular scale and orientation. Here, only the real part of the filter outputs are taken into consideration. The image used for obtaining the Model histogram and the model histogram itself are shown in figure 1.

4.2. Acquiring the Target Histogram

The texture histogram of the target is obtained in the same way as the model histogram. Tracking the person's head is achieved by combining the output of the texture module with the intensity gradient module. To achieve this, the model histogram is compared with the target histogram. For generating the target histogram, an ellipse mask is created around the ROI (region of interest) in the current image. The Target histogram(n'_b) is then computed by

$$n'_b(S,O) = \sum_X \sum_Y abs(G'_{S,O}(X,Y))$$

where $G'_{S,O}(X,Y)$ is the output of the filter bank after convolution with the current image.

4.3. Comparing Texture Histograms

As with color histograms, the texture histograms can be compared using a number of techniques such as histogram intersection or using Bhattacharayya coefficients, but histogram intersection is used to obtain the likelihood map for the module. The similarity of between the model and target histograms provides the likelihood Φ_t which can be calculated by histogram intersection [12, 2],

$$\Phi_t(n_b, n'_b) = \frac{\min(n_b, n'_b)}{\sum_{j=1}^{B} n_j} \tag{5}$$

The similarity between the two texture histograms may also be found using the Bhattacharayya coefficient [6]:

$$\Phi_S(n_b, n'_b) = \frac{\sqrt{n_b n'_b}}{\sqrt{\left(\sum_{j=1}^{B} n_j\right)\left(\sum_{j=1}^{B} n'_j\right)}}. \tag{6}$$

One problem that arises here is that the target and the model histogram, though having the same number of bins, cannot be compared directly. This is due to the fact that σ of the tracker's state given by (x, y, σ) is varied. This in simple terms means that the scale of the ellipse used for tracking is varied. To overcome this problem, the target histogram is normalized with respect to the texture histogram. The normalization of the target histogram is given by

$$\hat{n'_b} = n'_b \times \frac{\sum_{i=1}^{N} n_b}{\sum_{i=1}^{N} n'_b}$$

Once the likelihood is obtained, it is then normalized and converted to a percentage score to enable a combination with the likelihood generated from the intensity gradients module. The normalized likelihood for texture is given by

$$\bar{\Phi}_t(S) = \frac{\Phi_t(S) - min_{S_i \in S}\Phi_t(S_i)}{max_{S_i \in S}(\Phi_t(S_i)) - min_{S_i \in S}\Phi_t(S_i)} \tag{7}$$

5. Combining with intensity gradients

A simple tracker described by [1] is used as a base for the current tracker. The base is an intensity gradient module which works by computing the gradients along the boundary of an ellipse defined by the state S given by (x,y,σ) where x and y are the coordinates of the center of the ellipse and σ is the length of the minor axis with a fixed aspect ratio of 1.2. The intensity gradients can be computed as a normalized sum of gradient magnitudes along the boundary as

$$\Phi_g = \frac{1}{N_\sigma} \sum_{i=1}^{N_\sigma} |g_s(i)| \tag{8}$$

Alternatively, the normalized dot product of the gradient magnitudes can be used which provides better results compared to (8). This is given by

$$\Phi_g = \frac{1}{N_\sigma} \sum_{i=1}^{N_\sigma} |n_\sigma . g_s(i)| \tag{9}$$

The likelihood thus obtained is normalized as

$$\bar{\Phi}_g(S) = \frac{\Phi_g(S) - min_{S_i \in S}\Phi_g(S_i)}{max_{S_i \in S}(\Phi_g(S_i)) - min_{S_i \in S}\Phi_g(S_i)}$$

Once the likelihoods from the intensity gradient and the spatiogram modules are obtained, they are combined to obtain the best likelihood which is used to update the state S. The updated state is given by

$$S^* = arg(max_{S_i \in S}\{\bar{\Phi}_g(S_i) + \bar{\Phi}_t(S_i)\}) \tag{10}$$

To enable the tracker to handle moving targets, a constant velocity prediction scheme described in [1] is employed. This removes the restraint on lateral velocity of the object and only the amount of acceleration of the target is limited.

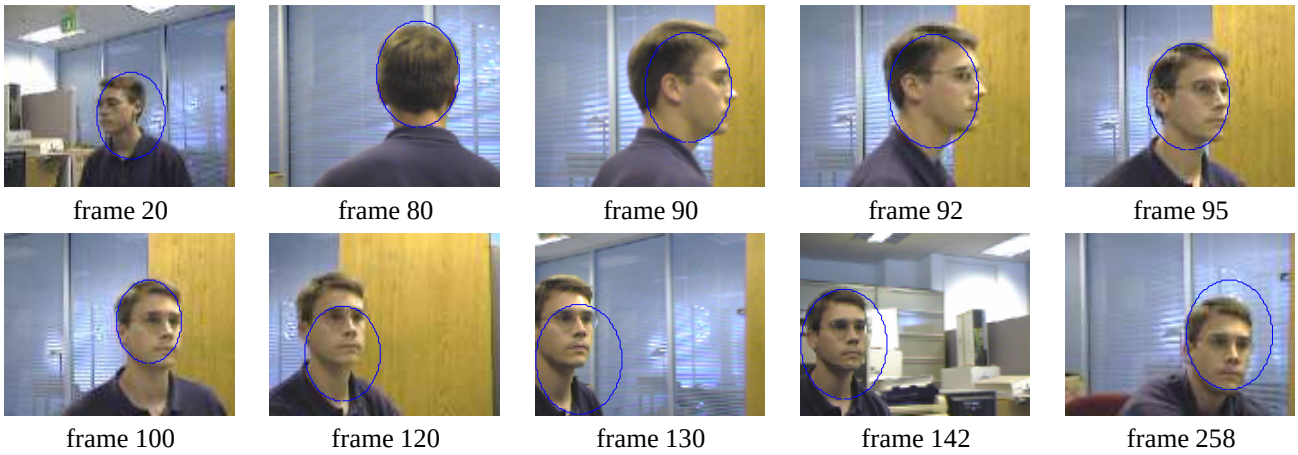

Figure 2: Tracking results using Texture Histograms and Intensity Gradients.

6. Overview of the algorithm

A simple step by step description of the algorithm using texture histograms and intensity gradients for tracking a person's head is given below.

1. Place ellipse on person's head using initial state S obtained manually.
2. Generate an ellipse mask so that image contains only the face of the target with three-quarters view of the person's head.
3. Compute model texture histogram by convolving the masked image with Log-Gabor filter bank.
4. Store model texture histogram for comparison with target histogram
5. For each image in the sequence,generate an ellipse mask using current state S to mask out only the ROI.
6. Convolve masked image with log-Gabor filter bank to get target histogram.
7. Compare target histogram with model histogram for best match.
8. After comparing, compute Likelihood to give location for best match
9. Obtain likelihood for location of the person's head from the intensity gradients module by computing gradients along the boundary of the head bounding box (ellipse).
10. Normalize both likelihoods and combine them to get a final likelihood map.
11. Obtain best match for head from final likelihood map.
12. Once a best match is obtained, use it to update current state S.

7. Experimental results

An exhaustive local search with a $\pm 4 \times \pm 4 \times \pm 1$ search window in x, y, and scale, over an image sequence[1] was done with the texture histogram algorithm combined with intensity gradients. The tracker is initialized manually by the user on the first frame of the image sequence. The model histogram is generated from this image. This requires that the first image in the sequence should be a three-quarters view of the person's head. From the results shown in figure 2, it can be seen that the tracking is not perfect. The intensity gradient module is distracted by the collar of the shirt (shown in Frames 120 and 130), but the texture module does not produce a strong enough likelihood that can pull the ellipse back to the center of the head. This causes the tracker to fail in subsequent frames. Although the tracker does not completely lose the person's head, the ellipse is not centered on the head. However, close to frame 256 the tracker reacquires the person's head(shown in frame 258). The tracker is capable of handling a moving target, but fails when the movement of the target is sudden and fast. The tracker, as seen in frames 90 - 120, is not distracted by skin-colored background or by large gradients

8. Conclusion

We have presented a novel algorithm that uses texture histograms for tracking a person's head. Though incapable of robust tracking on its own, the texture module when combined with intensity gradients produces satisfactory tracking of a person's head. The tracker is capable of handling

[1] from `http://www.ces.clemson.edu/~stb/research/headtracker`

360 degrees out-of-plane rotation of the target and small lighting changes. The tracker is also capable of handling a moving target well but fails when the target accelerates suddenly.

References

[1] S. Birchfield. An elliptical head tracker. In *Proceedings of the 31st Asilomar Conference on Signals, Systems and Computers*, volume 2, pages 1710–1714, 1997.

[2] S. Birchfield. Elliptical head tracking using intensity gradients and color histograms. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 232–237, 1998.

[3] L. Brown. 3-d head tracking using motion adaptive texture-mapping. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, page 998.

[4] R. T. Collins. Mean-shift blob tracking through scale space. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2003.

[5] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 142–149, 2000.

[6] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):564–577, May 2003.

[7] W. Freeman and E. H. Adelson. The design and use of steerable filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:891–906, 1991.

[8] J. Huang, S. R. Kumar, M. Mitra, W.-J. Zhu, and R. Zabih. Image indexing using color correlograms. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1997.

[9] Y. Rubner and C. Tomasi. Texture-based image retrieval without segmentation. In *Proceedings of the International Conference on Computer Vision*, pages 1018–1024, 1999.

[10] L. Sigal, S. Claroff, and V. Athitsos. Estimation and prediction of evolving color distributions for skin segmentation under varying illumination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2000.

[11] J. R. Smith and S.-F. Chang. Automated image retrieval using color and texture. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1996.

[12] M. Swain and D. Ballard. Color indexing. *International Journal of Computer Vision*, 7(1):11–32, 1991.

[13] Y. Wu and T. S. Huang. A co-inference approach to robust visual tracking. In *Proceedings of the 8th International Conference on Computer Vision*, pages 26–33, 2001.

[14] Z. Zivkovic and B. Kröse. An EM-like algorithm for color-histogram-based object tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2004.