A Simple Oriented Mean-Shift Algorithm For Tracking

Jamil Draréni and Sébastien Roy

DIRO, Université de Montréal, Canada.
{drarenij,roys}@iro.umontreal.ca

Abstract. Mean-Shift tracking gained a lot of popularity in computer vision community. This is due to its simplicity and robustness. However, the original formulation does not estimate the orientation of the tracked object. In this paper, we extend the original mean-shift tracker for orientation estimation. We use the gradient field as an orientation signature and introduce an efficient representation of the gradient-orientation space to speed-up the estimation. No additional parameter is required and the additional processing time is insignificant. The effectiveness of our method is demonstrated on typical sequences.

1 Introduction

Object tracking is a fundamental and challenging task in computer vision. It is used in several applications such as surveillance [1], eye tracking [2] and object based video compression/communication [3].

Although many tracking methods exist, they generally fall into two classes, *bottom-up* and *top-down* [4]. In a bottom-up approach, objects are first identified and then tracked. The top-down approach instead, uses hypotheses or signatures that discriminate the object of interest. The tracking is then performed by hypotheses satisfaction.

Recently, a *top-down* algorithm based on mean-shift was introduced for blob tracking [5]. This algorithm is non-parametric and relies solely on intensities histograms. The tracking is performed by finding the mode of a statistical distribution that encodes the likelihood of the original object's model and the model at the probing position. Because it is a *top-down* approach and it does not rely on a specific model, the mean-shift tracker is well adapted for real-time applications and robust to partial occlusions.

In [6], an extension was proposed to cope with the scale variation. However, little has been done to extend the tracker for rotational motions[7]. In fact, the original mean-shift tracker as proposed in [5] is invariant to rotations and thus, does not provide information on the target's orientation. This property is induced by the inherent spaceless nature of the histograms. While this may not be problematic for objects with symmetrical dimensions like circles or squares, it is no longer valid when the tracked objects are "thin" [7]. An example of a tracked thin object (an arm) is illustrated in Fig.1.

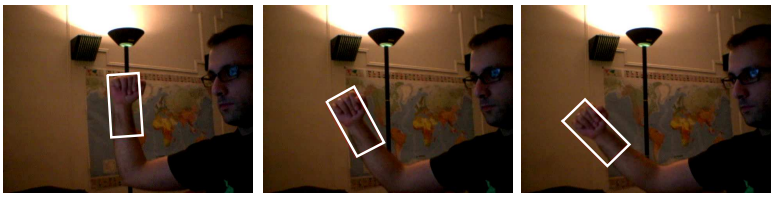

Fig. 1. Result of tracking an arm using the presented oriented mean-shift tracker.

In [7], the authors used a simplified form of *correlograms* to encode pixels positions within the region of interest. Pairs of points at an arbitrarily fixed distance along the principal axis vote with their joint intensities and their angle relative to the patch's origin to generate an orientation-intensity correlogram. Once the correlogram is estimated, it is used in the mean-shift's main loop just like a regular histogram. Unfortunately no method was proposed to automatically select the fixed distance for pairs sampling. Furthermore, since the pairs are only picked along the principal axis of the object, the generated correlogram does not encode a global representation of the object.

In this paper, we propose a fast and simple algorithm for an oriented mean-shift tracking. We use the original mean-shift formulation to estimate the object's translation and for the rotational part, a histogram of the orientations of the spatial gradients (within the region of interest) is used to assign an orientation according to a previously computed set of possible orientations' histogram. The effectiveness of the proposed method is demonstrated in experiments with various types of images. Our method can also be applied to video stabilization as our experiments will show. Real-time applications are still possible since the additional processing time is negligible.

The rest of the paper is organized as follows, in section 2, mean-shift tracking is summarized. Section 3, presents the gradient-orientation representation using histogram's LUT. The implementation of the proposed method is described in section 4. The experiments and results are reported in section 5 and we finally summarize our conclusion in section 6.

2 Mean-shift and Limitations

The mean-shift algorithm, as initially proposed in [8], is a non-parametric method to estimate the mode of a density-function. Let $S = \{x_i\}_{i=1..n}$ a finite set of r-dimensional data and $K(x)$ a multivariate kernel with window radius h. The sample mean at x is defined as:

$$m(x) = \frac{\sum_{x_i \in S} K(x - x_i).x_i}{\sum_{x_i \in S} K(x - x_i)}$$

The quantity $m(x) - x$ is called the mean-shift vector. It has been proven that if $K(x)$ is an isotropic kernel with a convex and monotonic decreasing profile, the

mean-shift vector always points in the direction of the maximum increase in the density. Thus, following this direction recursively leads to the local maximum of the density spanned by S. Examples of such kernels are the gaussians and Epanechnikov kernels. The reader is referred to [8, 9, 5] for further details on the mean-shift algorithm and related proof of convergence.

2.1 Mean-Shift for Tracking

Comaniciu et al.[5] took advantage of the mean-shift's property and proposed an elegant method to track blobs based on intensities histograms. The algorithm finds the displacement $\triangle y$ of the object of interest S as a weighted sum:

$$\triangle y = \frac{\sum_{x_i \in S} w_i.K(x - x_i).x_i}{\sum_{x_i \in S} w_i.K(x - x_i)}$$

Where w_i are weights related to the likelihood of the model and the target's intensities histograms. The estimation is recursive until the displacement's magnitude $||\triangle y||$ vanishes (or reaches a predefined value).

Unfortunately, the mean-shift tracker can not infer the orientation of an object based on its intensity histogram. To overcome this limitation, the tracker must use clues related to the spatial organization of the pixels or parameters that describe textures. Among those clues, image gradients are good candidates because their orientations vary when the image undergoes a rotation and are easy to compute.

2.2 Gradients and Gradients Histogram

Let I be an image. The first order gradient of I at position (x, y), noted ∇I_{xy} is defined as:

$$\nabla I_{xy} = [I_x, I_y]^T = \begin{bmatrix} I(x+1, y) - I(x-1, y) \\ I(x, y+1) - I(x, y-1) \end{bmatrix} \quad (1)$$

The orientation and the magnitude of the gradient vector I_{xy} are given by:

$$\theta(x, y) = tan^{-1}\left(\frac{I_y}{I_x}\right) \; ; \; mag(x, y) = \sqrt{I_x^2 + I_y^2} \quad (2)$$

It is clear that the orientation is independent of the image translation. However, a rotation of the image yield a rotation of the gradient field by the same amount. This property can be used to assign an orientation to the object of interest. Instead of keeping track of the gradient field itself, it is more convenient to build a histogram of gradient's orientations. This representation has been used in Lowe's SIFT [10] to assign an orientation to the keypoints.

In the present work, the m-bin orientation histogram O of an object is computed as:

$$O_m = C \sum_{i=1}^{i=n} mag(p_i) \cdot \delta[\theta(p_i) - m] \tag{3}$$

Where $p_0, p_1, ...p_n$ are the n pixels of the object of interest and the normalization constant C is computed as to insure that $\sum_{u=1}^{u=m} O_u = 1$. δ is the Kronecker delta function.

$\theta(p_i)$ and $mag(p_i)$ are functions that return the orientation and the magnitude of the gradient at pixel p_i as defined in (2) As opposed to a regular intensity histogram, each sample modulates its contribution with its magnitude. The reason behind this choice is two-fold: first, we generally observe that gradients with larger magnitudes tend to be more stable ; second, the gradient is known to be very sensitive to noise, thus weighting the votes with their magnitudes is like privileging samples with a good signal to noise ratio.

As opposed to [10], we do not extract a dominant orientation from the histogram. Rather, we keep the whole histogram as an orientation signature.

Histograms and bin width One of the major problem that arises when estimating a histogram (or any density function) from a finite set of data is to determine the bin width of the histogram. A large bin width gives an over-smoothed histogram with a coarse block look, whereas a small bin width results in an under-smoothed and jagged histogram [11]. In [12], Scott showed that the optimal bin width W, which provides an unbiased estimation of the probability density is given by:

$$W = 3.49 \cdot \sigma \cdot N^{-1/3} \tag{4}$$

Where N is the number of the samples and σ is the standard deviation of the distribution. We used a more robust formulation described in [13]:

$$W = 2 \cdot IQR \cdot N^{-1/3} \tag{5}$$

The interquartile range (IQR) is the difference between the 75^{th} and 25^{th} percentile of the distribution. Note that (5) does not contain σ, thereby reducing the risk of bias. The bin width computed with (5) is the one we use throughout our experiments.

3 Tracking with Gradient Histograms

A single orientation histogram encodes only the gradient distribution for one specific orientation. Thus, to infer the orientation from a gradient histogram, a LUT of gradient histograms corresponding to all image orientations must be built beforehand (at the initialization step). During the tracking process, the gradient histogram of the object must be compared against the histograms in the LUT. The sought orientation is the one that corresponds to the closer histogram in the LUT. A histogram's likelihood can be computed in different ways. We used

the histogram intersection as introduced in [14] for its robustness and ease of computation.

The intersection of two m-bins histograms h_1 and h_2 is defined as:

$$h_1 \cap h_2 = \sum_{i=1}^{i=m} Min(h_1[i], h_2[i]) \tag{6}$$

Where $Min()$ is a function that returns the minimum of its arguments. It is clear that the closer the histograms, the bigger the intersection score. The look-up table of histograms captures the joint orientation-gradient space of the object and can also be seen as a 2D histogram.

In the following subsections, two methods are introduced to construct a histogram gradient table: *Image-Rotation Voting* and *Gradient-Rotation Voting*.

3.1 Image-Rotation Voting

This is the simplest way to gather histograms of gradients for different orientations. The image of the tracked object is rotated by 360° around its center. The rotation is performed by a user-defined step (2° in our experiments) and an orientation histogram is computed at each step. The resulting histograms are stored in a stack and they form the gradient's histograms LUT. To reduce noise due to the intensity aliasing , rotations are performed with a bi-cubic interpolation. This method is outlined in the algorithm below:

Given: Original image, target's pixels $\{p_i\}_{i=1...n}$ and a rotation step $\triangle rot$.

1. $step \leftarrow 0$, $ndx \leftarrow 0$.
2. Apply a gaussian filter on $\{p_i\}_{i=1...n}$ to reduce noise (typically 3×3).
3. Compute $\{mag_i\}_{i=1...n}$ and $\{\theta_i\}_{i=1...n}$ the orientation and magnitudes of gradients at $\{p_i\}_{i=1...n}$ according to (1).
4. Derive the orientation histogram O_m using $\{mag_i\}$ and $\{\theta_i\}$ according to (3).
5. $LUT[ndx] \leftarrow O_m$
6. $ndx \leftarrow ndx + 1$, $step \leftarrow step + \triangle rot$.
7. Rotate $\{p_i\}_{i=1...n}$ by $step$ degrees.
8. If $step < 360$ go to step 3.
9. return LUT

3.2 Gradient-Rotation Voting

The second method is faster and produces better results in practice. Instead of rotating the image itself, the computed gradient field of the original image is incrementally rotated and the result of each rotation votes in the proper histogram. Note that due to histogram descretization, rotating a gradient field is not exactly equivalent to shifting the histogram by the same amount. This is due to the fact that histogram sampling is generally not the same as the rotation sampling. For instance, after rotating the gradient field some samples that vote

for a specific bin might still vote for the same bin whereas others may jump to an adjacent bin. They would be equivalent if the gradient histogram had a bin width of 1 (i.e 360 bins), which is not the case in practice. The gradient-rotation voting is outlined below:

Given: Original image, target's pixels $\{p_i\}_{i=1...n}$ and a rotation step $\triangle rot$.

1. $step \leftarrow 0$, $ndx \leftarrow 0$.
2. Apply a gaussian smoothing on $\{p_i\}_{i=1...n}$ to reduce noise.
3. Compute $\{mag_i\}_{i=1...n}$ and $\{\theta_i\}_{i=1...n}$ the orientation and magnitudes of gradients at $\{p_i\}_{i=1...n}$ according to (1).
4. Derive the orientation histogram O_m using $\{mag_i\}$ and $\{\theta_i\}$ according to (3).
5. $LUT[ndx] \leftarrow O_m$
6. $ndx \leftarrow ndx + 1$, $step \leftarrow step + \triangle rot$.
7. For each $\{\theta_i\}_{i=1...n}$
   Do $\theta_i \leftarrow \theta_i + \triangle rot$
8. If $step < 360$ go to step 4.
9. return LUT

---

4 Implementation

We implemented the proposed oriented mean-shift tracker as an extension to the original mean-shift tracker. The user supplies the initial location of the object to track, along with its bounding-box and an initial orientation. Images are first smoothed with a gaussian filter to reduce the noise (typically a 3×3 gaussian mask). Note that the smoothing is only applied in the neighborhood of the object. Orientations's look-up table are generated using the *Gradient-Rotating* method with a 2° step. The orientation estimation can either be nested within the original mean-shift loop or performed separately after the estimation of the translational part. The complete algorithm is outlined below:

Given: The original sequence, the initial object's position (y_0) and orientation (θ_0).

1. Compute the LUT of histograms orientations at y_0 (see section 3).
2. Initialize the mean-shift algorithm.
3. For each frame f_i
   (a) Update the object's position using the original mean-shift.
   (b) compute the gradient and estimate H the orientation histogram using (3).
   (c) $h_{max} \leftarrow \ Max \ \ [H \cap h_i]; \ h_i \in LUT$.
   (d) update the object's orientation by the orientation that corresponds to h_{max}.

Notice that the orientations are estimated relatively to the initial orientation θ_0.

Even though histogram intersection is a fast operation, processing time can still be saved at step 4.c by limiting the search in a specific range instead of the entire LUT. Typical range is $\pm 20^\circ$ from the object's previous orientation.

Fig. 2. Some frames from the manually rotated sequence (with a fixed background).

5 Experimental Results

We tested the proposed oriented mean-shift algorithm on several motion sequences. Since we propose an orientation upgrade to the original mean-shift tracker, we mostly considered sequences with dominating rotational motion. We first ran our tracker on a synthetic sequence that was generated by fully rotating a real image (a chocolate box). The figure fig.2 shows some frames from the synthetic sequence.

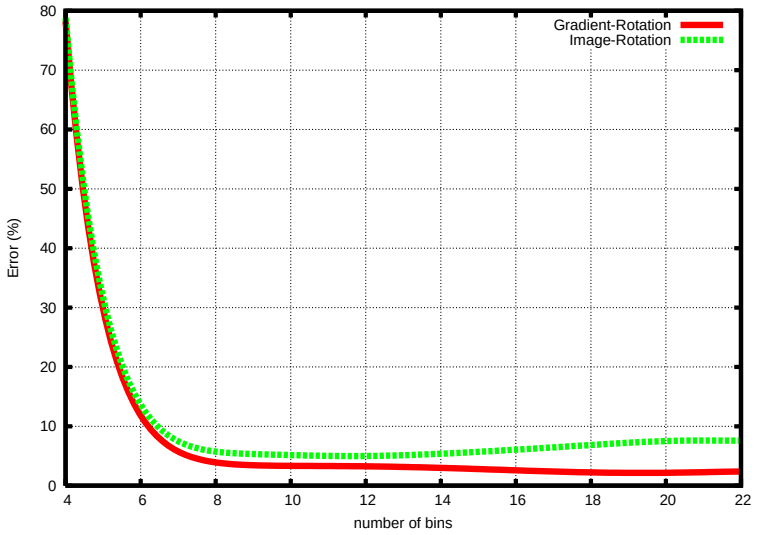

Fig. 3. Errors in orientation estimation as a function of histogram samples.

The error of rotation estimation using different bin size is reported in figure fig.**??**. The green curve represents the error with a LUT generated by the *image-rotation* method whereas the red curve is the error using a LUT generated by the *gradient-rotation* method. In both cases the computed optimal bin size was

14. As we can see, the *gradient-rotation* method gives better results and is less sensitive to the bin size variation. For the rest of the experiments, gradient's LUT were generated using the *gradient-rotation* method.

Fig. 4. Results of tracking a rotating face. Sample frames: 78, 164 and 257

We further tested our method for face tracking. As the face underwent an almost perfect roll, we computed the orientation estimations at each frame. We observe that the face is tracked accurately, although no exact ground truth is available in this case. The results are shown in figures fig.4 and fig.5.

Aerial surveillance is another field where the tracking is useful. Due to the rectangular shape of common vehicles, an oriented tracking is suitable as shown in figure fig.6. However, notice that the orientation is not truly 2D, as the view angle induces some perspective distortions that is not handled in our method.

Finally, we illustrate the effectiveness of the proposed method for video rectification. A hand-held camera was rotated by hand around its optical axis while gazing at a static scene (see figure fig.7, left column). We tracked a rigid object attached to the scene and used the recovered motion to rectify and cancel the rotation in the video sequence. The results of tracking/rectification are shown in the figure fig.8 and the estimated orientations are plotted in the figure fig.7. The rotation is well recovered, as can be seen in the estimated curve of figure fig.7 and the rectified images of figure fig.8. Notice that the rectified images are sometimes distorted by parallax effects that are not modelled by our algorithm.

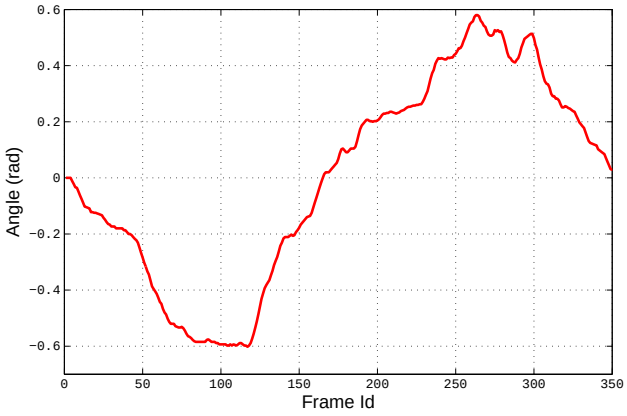

Fig. 5. Estimated orientation for the rotating face sequence.

Fig. 6. Tracking results for the car pursuit sequence.

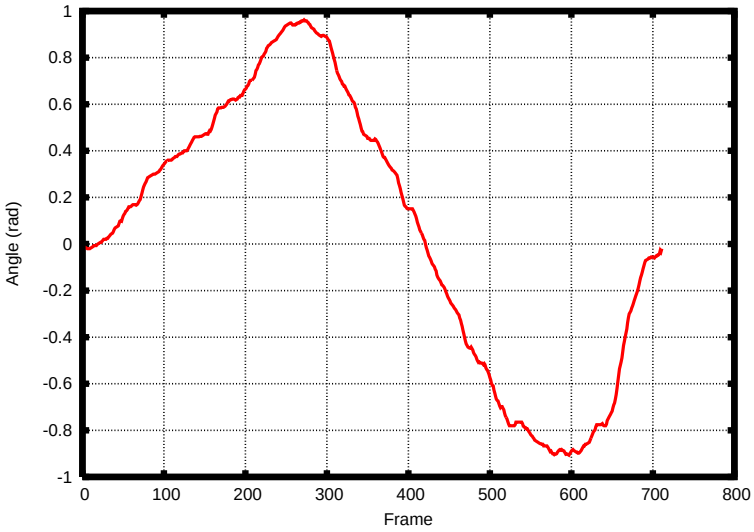

Fig. 7. Estimated orientation for the shelf sequence.

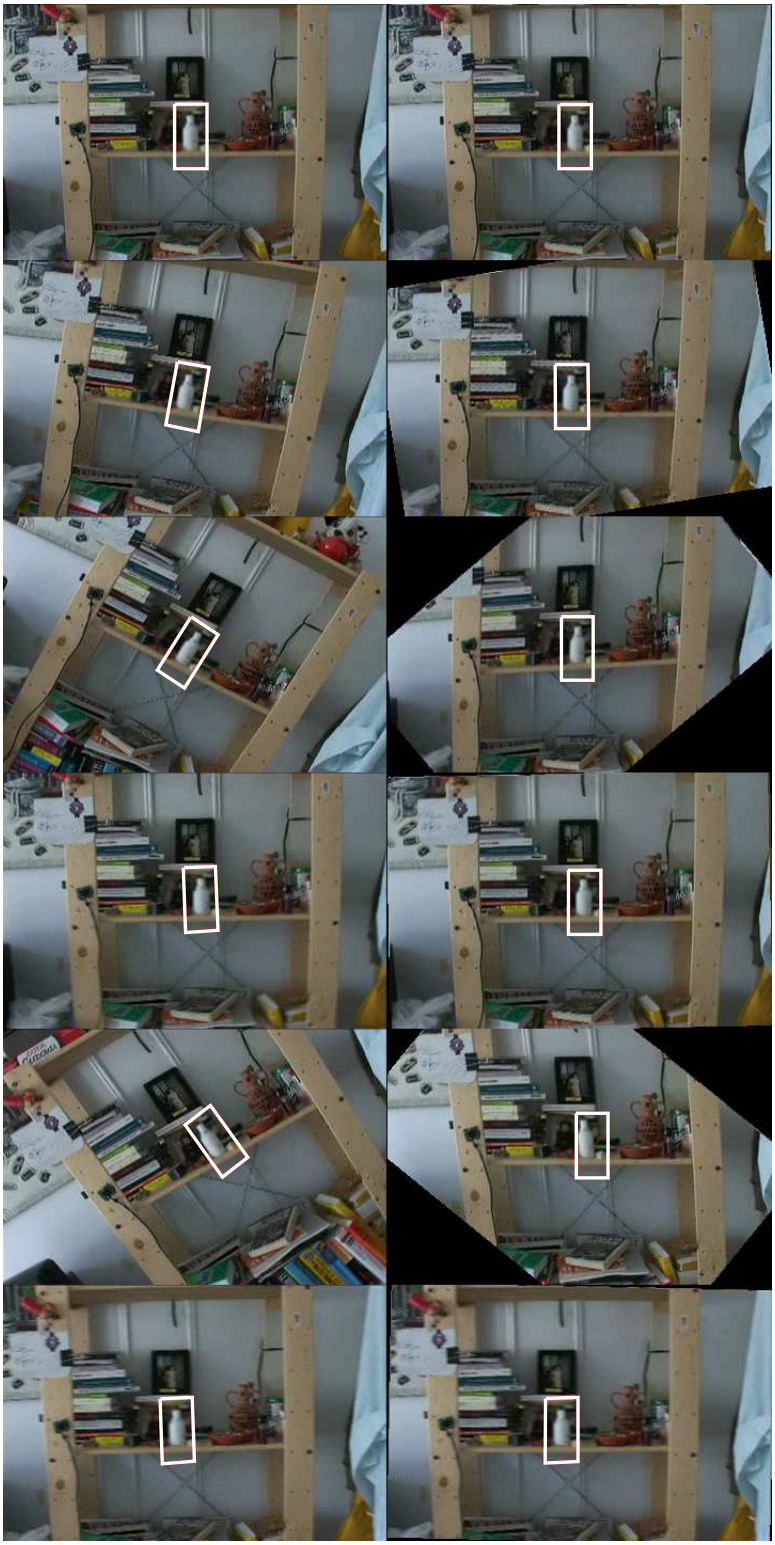

Fig. 8. Results of tracking and rectifying images from a rolling camera sequence. *left*) results of the original tracking. *right*) rectified sequence after rotation cancellation. Shown frames are 0, 69,203,421,536 and 711.

6 Conclusion

We have presented a fast and simple extension to the original mean-shift tracker, to allow the estimation of the orientation. This rotation parameter is crucial when the tracked objects have a "thin" shape. We introduced the idea of the gradient-orientation space represented by the gradient look-up tables. Of course, the LUT can be extended to other cues related to the texture or pixels positions. This representation proved to be efficient as our experiments depicted. The proposed method ran comfortably on a regular PC in real time. Tracking was performed at 10-25 frames per second for typical 2000 pixels objects. In our implementation, the orientation was estimated independently from the translation shift. However, performing a combined mean-shift on histograms intensities and gradient LUT is possible. In the future, we plan on adding support for perspective deformation to better handle different type of rotations.

References

[1] J. Segen and S. G. Pingali. A camera-based system for tracking people in real time. In *International Conference on Pattern Recognition*, page 63, 1996.

[2] Zhiwei Zhu, Qiang Ji, and Kikuo Fujimura. Combining kalman filtering and mean shift for real time eye tracking under active ir illumination (oral). 2002.

[3] J. Crowley and K. Schwerdt. Robust tracking and compression for video communication. In *IEEE Computer Society Conference on Computer Vision, Workshop on Face and Gesture Recognition*, 1999.

[4] Katja Nummiaro, Esther Koller-Meier, and Luc J. Van Gool. An adaptive color-based particle filter. *Image Vision Comput.*, 21(1):99–110, 2003.

[5] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. In *Conference on Computer Vision and Pattern Recognition*, pages 142–151, 2000.

[6] R. T. Collins. Mean-shift blob tracking through scale space. In *Conference on Computer Vision and Pattern Recognition*, volume 2, pages 234–240, 2003.

[7] Qi ZHao and Hai Tao. Object tracking using color correlogram. In *Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pages 263–270, 2005.

[8] K. Fukunaga and L. Hostetler. The estimation of the gradient of a density function with applications in pattern recognition. In *IEEE Transactions on Information Theory*, pages 32–40, 1975.

[9] Yizong Cheng. Mean shift, mode seeking, and clustering. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 17(8):790–799, 1995.

[10] D. Lowe. Distinctive image features from scale-invariant keypoints. In *International Journal of Computer Vision*, volume 20, pages 91–110, 2003.

[11] M. P. Wand. Data-based choice of histogram bin width. *The American Statistician*, 51(1):59, 1997.

[12] D. Scott. On optimal and data-based histograms. Number 66, 1979.

[13] A. J. Izenman. Recent developments in nonparametric density estimation. volume 86, 1991.

[14] Michael J. Swain and Dana H. Ballard. Color indexing. *International Journal of Computer Vision*, 7(1):11–32, 1991.