Mean-shift Blob Tracking through Scale Space *

Robert T. Collins
Carnegie Mellon University

Abstract

The mean-shift algorithm is an efficient technique for tracking 2D blobs through an image. Although the scale of the mean-shift kernel is a crucial parameter, there is presently no clean mechanism for choosing or updating scale while tracking blobs that are changing in size. We adapt Lindeberg's theory of feature scale selection based on local maxima of differential scale-space filters to the problem of selecting kernel scale for mean-shift blob tracking. We show that a difference of Gaussian (DOG) mean-shift kernel enables efficient tracking of blobs through scale space. Using this kernel requires generalizing the mean-shift algorithm to handle images that contain negative sample weights.

1. Introduction

The mean-shift algorithm is a nonparametric statistical method for seeking the nearest mode of a point sample distribution [3, 6]. The algorithm has recently been adopted as an efficient technique for appearance-based blob tracking [1, 4]. In the blob tracking scenario, sample points are regularly distributed along the image pixel grid, with a pixel's value $w(a)$ being the *sample weight* of the point at that location. This sample weight is chosen such that pixels on the foreground blob have high weight, and pixels in the background have low weight. The mean-shift algorithm specifies how to combine the sample weights $w(a)$ in a local neightborhood with a set of kernel weights $K(a)$ to produce an offset that tracks the centroid of the blob in the image (Figure 1).

The problem we address in this paper is selecting the scale of the mean-shift kernel, which directly determines the size of the window within which sample weights are examined. This size should of course be proportional to the expected image area of the blob being tracked. Although kernel scale is a crucial parameter for the mean-shift algorithm, there is currently no sound mechanism for choosing this scale within the framework. We show how to combine a well-developed theory of feature scale selection due to Lindeberg with the mean-shift algorithm, resulting in a method that represents and tracks blobs as the modes of a differential function in scale space.

*This work is supported in part by DARPA/IAO HumanID under ONR contract N00014-00-1-0915, and by DARPA/IPTO MARS contract NBCHC020090.

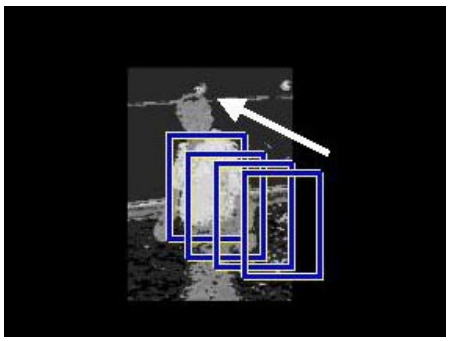


Figure 1: Two-dimensional blob tracking by applying the mean-shift algorithm to an image where pixel values represent likelihood of being on the tracked object. The mean-shift algorithm is a non-parametric method for climbing to the nearest mode of this likelihood image.

In this paper we do not specify how the sample weight image $w(a)$ is produced. Typically, weights are determined using a color-based object appearance model. In [1] a histogram of skin color in HSV color space is used to determine the likelihood of skin occuring at each pixel, using histogram backprojection to replace each pixel with the frequency (probability) associated with that HSV value in the skin color histogram. In [4] the appearance model is a distribution of colors represented by a histogram m, which is compared with a histogram of colors d observed within the current mean-shift window. The sample weight at each pixel with color i is then set to $\sqrt{m_i/d_i}$, a value related to the Bhattacharyya coefficient of histogram similarity. The mean-shift tracking paradigm applies equally well to sample weight images computed using other features besides color, such as texture similarity, background subtraction results, or the output of intensity template correlation.

In Section 2 we review the mean-shift algorithm and its relationship to non-parametric mode seeking. We also discuss how to generalize the algorithm to handle images with negative sample weights. In Section 3 we review Lindeberg's theory of feature scale selection. This theory represents image features of different scales by local maxima of a differential operator in scale-space. Section 4 combines

the two methods to show how mean-shift tracking of a blob that changes in scale can be performed by tracking the blob feature's mode through scale space. The paper closes by showing an illustrative example.

2. Mean-shift and Its Limitations

The heart of the mean-shift algorithm is computation of an offset from location vector x to a new location $x' = x + \Delta_x$, according to the mean-shift vector

$$\Delta_x = \frac{\sum_a K(a-x)w(a)a}{\sum_a K(a-x)w(a)} - x = \frac{\sum_a K(a-x)w(a)(a-x)}{\sum_a K(a-x)w(a)} \quad (1)$$

where K is a suitable kernel function and the summations are performed over a local window of pixels a around the current location x. A "suitable" kernel K is one that can be written in terms of a *profile* function k such that $K(y) = k(\|y\|^2)$ and profile k is nonnegative, nonincreasing, piecewise continuous, and $\int_0^\infty k(r)dr < \infty$ [Cheng].

An important theoretical property of the mean-shift algorithm is that the local mean-shift offset Δ_x computed at position x using kernel K points opposite to the gradient direction of the convolution surface

$$C(x) = \sum_a H(a-x)w(a) \quad (2)$$

computed with a kernel H, which is called the *shadow* of kernel K. This property makes the mean-shift procedure an efficient iterative method for gradient ascent to a local mode of the convolution surface without having to compute the full convolution.

Kernels K and shadow kernels H must satisfy the relationship

$$h'(r) = -ck(r) \quad (3)$$

where h and k are the respective profiles of H and K, $r = \|a-x\|^2$, and $c > 0$ is some constant [3]. Equation 3 guarantees that the mean-shift vector Δ_x computed from equation 1 points in the opposite direction to the gradient $\nabla C(x)$ of the surface in equation 2 for all x. For example, consider the Gaussian shadow kernel $H(x) = \exp\{-\|a-x\|^2 / (2\sigma^2)\}$, which has profile $h(r) = \exp\{-r/(2\sigma^2)\}$. Since $h'(r) = -\frac{1}{2\sigma^2}\exp\{-r/(2\sigma^2)\}$ we find from equation 3 that $k(r) = \exp\{-r/(2\sigma^2)\}$, and thus the associated kernel is $K(x) = \exp\{-\|a-x\|^2 / (2\sigma^2)\}$, showing that the Gaussian kernel is its own shadow. Another popular kernel-shadow pair is the flat kernel $\{K(x) = 1, \|x\| < 1\}$ and its associated shadow $\{H(x) = 1 - \|x\|^2, \|x\| < 1\}$.

Note that the mean-shift vector computed by equation 1 is invariant to scaling of the sample weights $w(a)$ by a positive constant c, that is, if each $w(a)$ is replaced by $cw(a)$ for $c > 0$. Note that the mean-shift vector is **not** invariant to a constant offset $w(a) + c$. However, all mean-shift kernels explored to date are symmetric about the origin, such that $K(x) = K(-x)$, in which case

$$\Delta_x = \frac{\sum K(a-x)(w(a)+c)(a-x)}{\sum K(a-x)(w(a)+c)} \quad (4)$$

$$= \frac{\sum K(a-x)w(a)(a-x) + c\sum K(a-x)(a-x)}{\sum K(a-x)(w(a)+c)} \quad (5)$$

$$= \frac{\sum K(a-x)w(a)(a-x)}{\sum K(a-x)(w(a)+c)} \quad (6)$$

showing that the direction of the mean-shift vector is invariant, and just the step size changes.

2.1 Mean-shift and negative weights

One limitation of the classic mean-shift algorithm is that the sample weights $w(a)$ have to be nonnegative. Indeed, empirical observation of mean-shift performance in the presence of negative sample weights shows that the method breaks down, with the algorithm sometimes seeming to converge to almost anything *except* the blob being tracked. To explain this behavior, consider what happens when c in equation 6 is a negative number. Although the direction of the mean-shift offset (numerator) is still valid, if c is larger than some of the values $w(a)$, the step size (denominator) can become negative, and the algorithm takes a step *away* from the mode. Once this begins to happen, the algorithm quickly diverges.

Why might we want to use negative sample weights? When comparing similarity of a model color histogram m with data histogram d, an intuitive alternative to $\sqrt{m_i/d_i}$ would be the log likelihood value $\log(m_i/d_i)$, which is negative when $d_i > m_i$. Accumulating log likelihood values with a flat kernel would then amount to computing the KL-divergence between the histograms m and d, that is

$$\sum_a log\frac{m_i}{d_i} = n\sum_{i=1}^{n} d_i log\frac{m_i}{d_i} \ .$$

A second reason for wanting to use negative values is that we might try to make the mean-shift vector invariant to both scale AND offset of the sample weights $w(a)$. We can easily do this by subtracting the sample mean $\overline{w}$ from the values $w(a)$ within the kernel window, thereby performing mean-shift on the offset-normalized image $w(a) - \overline{w}$. A third reason for allowing negative weights is that we might want to use mean-shift kernels that have negative weights. This last point is explored in Section 4.

Before changing the mean-shift algorithm to handle negative sample weights, we must first determine in what way negative weights make sense. If we interpret the mean shift quantity (Eq 1) as a sample center of mass, then a negative mass at any point doesn't make any sense. Instead,

we should interpret a point's weight as a vote for the direction and magnitude of the mean shift offset vector towards or away from that point. For a single point p, a positive weight w specifies that the offset vector at neighboring points should be directed towards p, with a magnitude w. If instead we have a negative weight w, the offset vector at neighboring points is now directed away from p, but with magnitude $|w|$. The mean-shift equation can be interpreted as forming the superposition of all these point-wise offset votes to produce an overall average offset vector.

We now see how to modify the mean-shift equation (1) to make sense for negative weights. The numerator of that equation votes for both the magnitude and direction of point-wise offset vectors, so the negative weights should stay. However, the denominator normalizes by the overall total magnitude of the votes, and therefore we must sum only the magnitude (the absolute value) of each term. The modified equation is

$$\Delta_x = \frac{\sum_a K(a-x)w(a)(a-x)}{\sum_a |K(a-x)w(a)|} \tag{7}$$

With this modification, mean-shift remains well-behaved on images that contain negative pixel values.

2.2 Mean-shift and scale selection

Kernel scale is a crucial parameter to the performance of the mean-shift algorithm. If the kernel size is chosen too large, the tracking window will contain many background pixels as well as the foreground object pixels. This is a problem when a data histogram collected within the window is compared to a model histogram describing the appearance of the foreground object, since the data histogram is then "diluted" with noisy background pixels. A window size that is too large can also cause the tracker to become more easily distracted by background clutter. Finally, a kernel window that is too large may cause convergence to an area *between* multiple modes, rather than converging to just one of the modes (see Figure 2, left).

Choosing a kernel size that is too small is also a problem. Kernels that are too small can "roam" around on a likelihood plateau around the mode, leading to poor object localization (Figure 2, right). This is particularly bad when mean-shift tracking results are being used to point an active pan/tilt camera. In this case, even when the object is not moving, the pan/tilt head will randomly jump around as the mean-shift window moves around the likelihood plateau.

The is no natural mechanism within the mean-shift framework for choosing the kernel scale or adapting it over time. [1] uses moments of the sample weight image to compute blob scale and orientation. This is only appropriate if there is a single foreground object in the scene. [4] suggests repeating the mean-shift algorithm at each iteration using

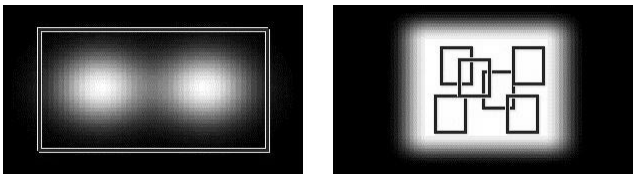


Figure 2: Mean-shift kernels that are too big (left) include background clutter as well as the foreground object pixels, and can also fail by encompassing multiple modes. Kernels that are too small (right) can "roam" around on a likelihood plateau around the mode, leading to poor localization.

window sizes of plus or minus 10 percent of the current size, and evaluating which scale is best using the Bhattacharyya coefficient. We have experimented with this approach and found it lacking. In particular, the approach does keep the window from growing too large, but unfortunately does not stop the window from shrinking too small. Looking at the right hand side of Figure 2, we see that in a uniformly colored region, any location of a window that is too small will yield the same value of the Bhattacharyya coefficient, as will shrinking the window even more. In short, this method has no "expansion force" to keep the window inflated. In [5], methods for adaptive kernel size selection are explored in the context of clustering point sample distributions for image segmentation, however we believe these methods are currently too computationally expensive for real-time tracking applications.

3. Scale Space Blobs

In this section we consider a method for automatically choosing an appropriate scale for blob tracking. The work of Lindeberg provides an elegant theory for selecting the best scale for describing features in an image [8, 2]. A multi-scale image description is formed (at least conceptually) by convolving the image with Gaussian filters of increasing spatial variance to form a *scale-space* representation with two spatial dimensions and a third dimension representing scale or resolution. Lindeberg then proposes to detect image features as points in scale-space where various differential operators achieve a local maxima with respect to both space and scale. This approach has been shown to produce feature descriptions that are well-localized both spatially and at an appropriate level of resolution.

For blob-like features, the differential operator used is the Laplacian, which when applied over Gaussian scale-space leads to a multi-resolution image description based on convolution with Laplacian-of-Gaussian (LOG) filters at varying scales. The 2D LOG filter at scale σ has the form

$$LOG(x;\sigma) = \frac{2\sigma^2 - \|x\|^2}{2\pi\sigma^6} e^{\frac{-\|x\|^2}{2\sigma^2}} \tag{8}$$

Let the convolution of image $f(x)$ with a LOG operator of scale σ be denoted as $L(x;\sigma) = \sum_a LOG(a-x;\sigma) f(a)$. Then blob features at various scales can be detected as points (x_0, σ_0) that are local maxima both spatially and in scale, i.e.

$$(\nabla_x L)(x_0;\sigma_0) = 0 \text{ and } (\delta_\sigma L)(x_0;\sigma_0) = 0 \ . \quad (9)$$

Let image $f(x)$ contain a blob with scale σ, centered at x_0, so that $L(x_0;\sigma)$ is a local maximum in the scale space representation of f. Consider a new image $g(y) = f(y/s)$ that is a spatially scaled version of f. As we would then expect, $L(sx_0;s\sigma)$ is a local maxima in the scale space representation of g. However, the magnitudes of the two filter responses are different, which makes it hard to compare feature decriptions across different scales. Assuming without loss of generality that the blob is centered at zero, and performing a change of variables $y = sx$ on $L(0;\sigma)$ we find that

$$\begin{aligned} L(0;\sigma) &= \sum_x \frac{2\sigma^2 - \|x\|^2}{2\pi\sigma^6} e^{\frac{-\|x\|^2}{2\sigma^2}} f(x) && (10)\\ &= \sum_y \frac{2\sigma^2 - \|y/s\|^2}{2\pi\sigma^6} e^{\frac{-\|y/s\|^2}{2\sigma^2}} f(y/s)\, |\frac{1}{s^2}| && (11)\\ &= s^2 \sum_y \frac{2(s\sigma)^2 - \|y\|^2}{2\pi(s\sigma)^6} e^{\frac{-\|y\|^2}{2(s\sigma)^2}} g(y) && (12)\\ &= s^2 L(0;s\sigma) && (13) \end{aligned}$$

where the $|1/s^2|$ term at the end of Eq. 11 is the Jacobian of the transformation $x = y/s$. In general, Lindeberg shows that $L(x;\sigma) = s^2 L(sx;s\sigma)$, and defines a new normalized operator $\sigma^2 LOG(x;\sigma)$ whose magnitude stays invariant across changes of image scale [8].

In the present work, we propose to replace the LOG scale space with one based on Difference-of-Gaussian (DOG) filters. It is well-known that the LOG filter can be well-approximated with a difference of two centered Gaussians $G(x;0,\sigma_1) - G(x;0,\sigma_2)$ with scales related by $\sigma_2/\sigma_1 = 1.6$ [7]. The DOG filter that approximates the LOG filter at scale σ thus has the form

$$DOG(x;\sigma) = \frac{1}{2\pi\sigma^2/1.6} e^{\frac{-\|x\|^2}{2\sigma^2/1.6}} - \frac{1}{2\pi\sigma^2(1.6)} e^{\frac{-\|x\|^2}{2\sigma^2(1.6)}} \quad (14)$$

One reason to prefer the DOG approximation over LOG is that there exist fast techniques for generating Gaussian convolution pyramids. We prefer the DOG operator in the present context because it is more clearly related to the mean-shift algorithm using a Gaussian kernel, a connection that will be explored in the next section.

It is interesting to note that the magnitude of the DOG operator is already invariant across scales, thus the operator does not have to be normalized by scale parameter σ^2, as the LOG filter does. That is, if the convolution of image $f(x)$ with a DOG operator is written as $D(x;\sigma) = \sum_a DOG(a-x;\sigma) f(a)$, then performing a change of variables procedure as above shows that $D(x;\sigma) = D(sx;s\sigma)$, so we do not have to normalize the DOG operator to achieve invariance across scales.

Why does the LOG filter have to be normalized to give an invariant response across scales, while its approximation, the DOG filter, does not? The DOG operator does not approximate the LOG in the traditional sense of $DOG(x) \approx LOG(x)$. Rather, the two functions are approximately equal up to a scale factor, such that $DOG(x)/LOG(x) \approx$ constant. Specifically, by considering the values of the two functions at the origin $x = 0$, it is easy to see that

$$\text{DOG}(x;\sigma) \ \approx \ 0.4875\,\sigma^2\,\text{LOG}(x;\sigma) \ .$$

The extra σ^2 factor is what makes the DOG operator invariant across scales. This precise manner in which the DOG operator "approximates" the LOG operator seems not to be widely discussed. The two are typically presented in the context of locating zero crossings, where the scale factor does not matter.

4. Mean-shift in Scale Space

We adapt feature scale selection theory to create a mechanism for adapting mean-shift kernel size while tracking blobs through changes in scale. Intuitively, we will track a blob's location and scale by using the mean-shift algorithm to track the local maximum (Eq 9) that represents the blob feature in the scale space generated by the DOG operator.

Define a 3D shadow kernel $H(x,\sigma)$ with two spatial dimensions x and one scale dimension σ. At any given scale σ_0, the 2D marginal kernel $H(x,\sigma=\sigma_0)$ will be a spatial filter $\text{DOG}(x,\sigma_0)$. The set of DOG filters at multiple scales form a scale-space filter bank, which is convolved with a sample weight image where each pixel is proportional to the likelihood that it belongs to the object being tracked. See Figure 3. The results of the DOG filters at multiple scales are then convolved with an Epanichikov shadow kernel in the scale dimension. The result is a 3D scale space representation in which modes represent blobs at different spatial positions and scales. We want to design a mean-shift filter to track modes through this scale space representation.

More formally, define a set of kernel scales around the current scale σ_0 as

$$\{\sigma_s = \sigma_0 * b^s, \ \text{ for } -n \le s \le n\} \quad (15)$$

where $b > 1$ specifies the base of a logarithmic coordinate scale and n limits the range of scales to explore around the current scale σ_0. We typically choose $b = 1.1$ and $n = 2$. The shadow kernel in the scale dimension is the Epanichnikov kernel

$$H_s(s) = 1 - (s/n)^2 \quad (16)$$

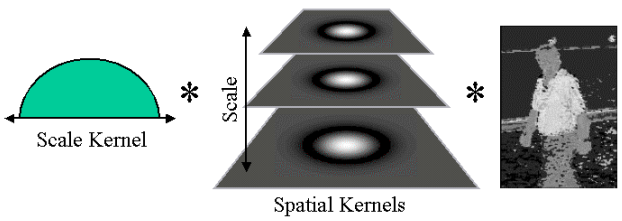


Figure 3: Shadow kernel for mode-seeking in scale space. This 3D kernel takes the form of a filter bank of centered DOG spatial kernels at different scales, convolved with a 1D Epanichnikov kernel across the scale dimension.

A filter bank of spatial shadow kernels is also defined, one for each scale σ_s. Each is defined as a DOG kernel around the current object location x_0

$$H_x(x,s) = DOG(\text{diag}(1/w, 1/h)(x - x_0), \sigma_s) \quad . \tag{17}$$

The scale factors w and h are introduced to allow elliptical spatial kernels (the DOG operator is circular). Finally, the 3D scale space is generated from a sample weight image $w(x)$ as

$$E(x,s) = \sum_s H_s(s) \sum_x H_x(x,s) w(x) \tag{18}$$

We now wish to design a mean-shift procedure that finds local modes in the scale space defined by the 3D convolution of Eq. 18. We simplify by considering two interleaved mean-shift procedures, one spatial and one in scale, that iterate to find scale space modes. To find the mean-shift kernels K_s and K_x in image space that correspond to shadow kernels H_s and H_x in scale space, we use the kernel-shadow relationship constraint of Eq. 3.

4.1 Spatial kernel

First, we treat the current scale σ_0 as constant, and consider the spatial shadow kernel $H_x(x,s)$. Since the DOG operator is a linear combination of Gaussians, and since the Gaussian kernel is its own shadow, we should expect the mean-shift kernel $K_x(x,s)$ to also be a linear combination of Gaussians. Indeed, after applying Eq. 3 with respect to $r = (x - x_0)^2$ we find that

$$K_x(x,s) = G(x; x_0, \sigma_1)/\sigma_1^2 - G(x; x_0, \sigma_2)/\sigma_2^2 \tag{19}$$

with $\sigma_1 = \sigma_s/1.6$ and $\sigma_2 = 1.6\sigma_s$. This kernel is also a DOG kernel, but not one that approximates a Laplacian – it has a more pronounced peak and a shallower negative basin around the peak. The mean-shift kernel corresponding to $E(x,s)$ in Eq 18, for fixed σ_0 is therefore

$$K_E(x) = \sum_s H_s(s) K_x(x,s) \quad . \tag{20}$$

This is a linear combination of filter bank responses, with the highest weight at the current scale σ_0, and weights dropping off as the scale of the filter increases or decreases.

One snag is that each $K_x(x,s)$ is a non-convex kernel that contains negative weights, and it therefore violates the assumptions that are typically placed on mean-shift kernels. The practical difficulty of this is that during computation of the mean-shift offset (Eq 1), some pixel values will be multiplied by negative kernel weights. However, we have already dealt with this problem in Section 2, where we considered the related problem of negative sample weights. The solution is the same: during computation of the mean-shift offset vector, we divide by the sum of the absolute values of all weight terms, instead of just the sum of the terms. With this modification, which was shown in Eq. 7, mean-shift with the $K_x(x,s)$ kernel is well-behaved.

4.2 Scale kernel

Now consider holding the location x_0 fixed, and looking for the best scale σ_s. The mean-shift kernel corresponding to the Epanichnikov shadow in the scale dimension will be a flat kernel. This kernel is applied to the values $\sum_x H_x(x,s)w(x)$, for the range of scales $-n \le s \le n$. Intuitively what is happening is that, for a given spatial location x_0, a filter bank of DOG operators is applied at varying scales, centered on that location, and then mean-shift is performed on the 1D array of results to locate the scale mode.

4.3 Algorithm summary

To summarize, we use two interleaved mean-shift procedures to track modes in scale space, which represent the spatial location and scale of blobs in an image. The algorithm proceeds as follows

1. Input is a sample weight image $w(a)$ and an estimate of the blob's current scale space location, represented by (x_0, σ_0), a parameter vector that combines both spatial and scale terms.

2. Hold σ_0 fixed, and perform a spatial mean-shift procedure using the equation

$$\Delta_x = \frac{\sum_s H_s(s) \sum_a K_x(a - x_0, s) w(a) (a - x_0)}{\sum_s H_s(s) \sum_a |K_x(a - x_0, s) w(a)|} \tag{21}$$

where $-n \le s \le n$ defines a range of scales σ_s as in Eq 15.

3. Let x_0 now represent the value that the spatial mean-shift procedure in step 2 converges to. Holding x_0 fixed, perform a mean-shift procedure using the equation

$$s' = \frac{\sum_s \sum_x H_x(x,s) w(x) s}{\sum_s \sum_x H_x(x,s) w(x)}. \tag{22}$$

This procedure is iterated until convergence. Set the new value of σ_0 to $\sigma_0 * b^{s'}$.

4. Iterate by interleaving steps 2 and 3 until both $\|s'\| < \varepsilon_s$ and $\|\Delta_x\| < \varepsilon_x$.

5. Examples

Figure 4 will help to better understand the motivation behind the scale selection approach presented in this paper. The top of Figure 4 shows three blobs (squares) of different sizes. The bottom of Figure 4 shows one slice through the 3D scale space generated by convolution with the DOG-Epanichnikov filter bank defined in the previous section. The modes in this scale space clearly localize each of the blobs both spatially, and in the scale dimension, as expected from the feature scale selection theory of Lindeberg. In the previous section we designed an interleaved mean-shift procedure that can find/track these modes without having to explicitly generate the scale space.

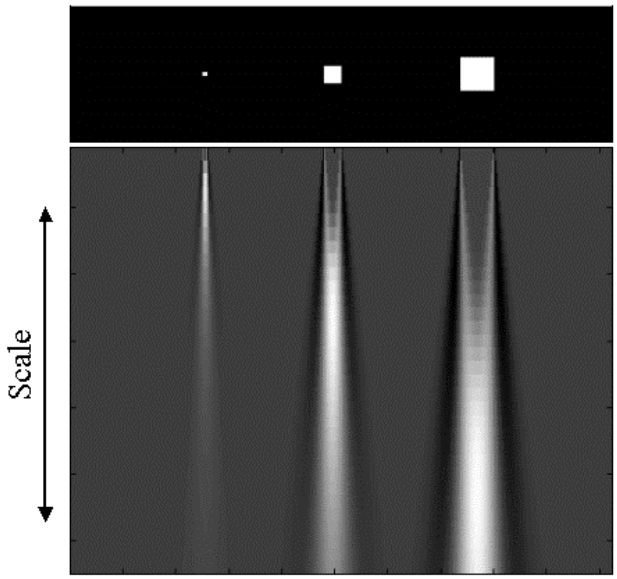


Figure 4: Intuitive illustration of scale space modes. Top: three blobs (squares) of different sizes. Bottom: scale space generated by the DOG-Epanichnikov filter bank defined in this paper. Each blob is well localized both spatially and in the scale dimension by a mode in this scale space.

Figure 5 shows blob tracking results on a real image sequence of a person walking around a test trajectory. The sequence comes from a database of gait test sequences collected at NIST as part of the DARPA Human Identification program. An image rectangle is selected by hand in the first frame, and a color histogram m_i is acquired to model the color distribution within this rectangle. Sample weight images are generated with respect to a data histogram d_i of colors within the current tracking window, with each pixel of color i being set to $\sqrt{m_i/d_i}$. This is the same method used in [4], who show that these weights are related to maximizing the similarity of histograms d_i and m_i as measured by the Bhattacharyya coefficient.

Sample frames of the tracking results from three different algorithms are presented. Figure 5A shows tracking results from the classic mean-shift algorithm without any changes of scale. The person is successfully tracked throughout the sequence, but the fixed scale of the tracking kernel provides poor localization of the centroid of the person as their size increases near the end of the sequence.

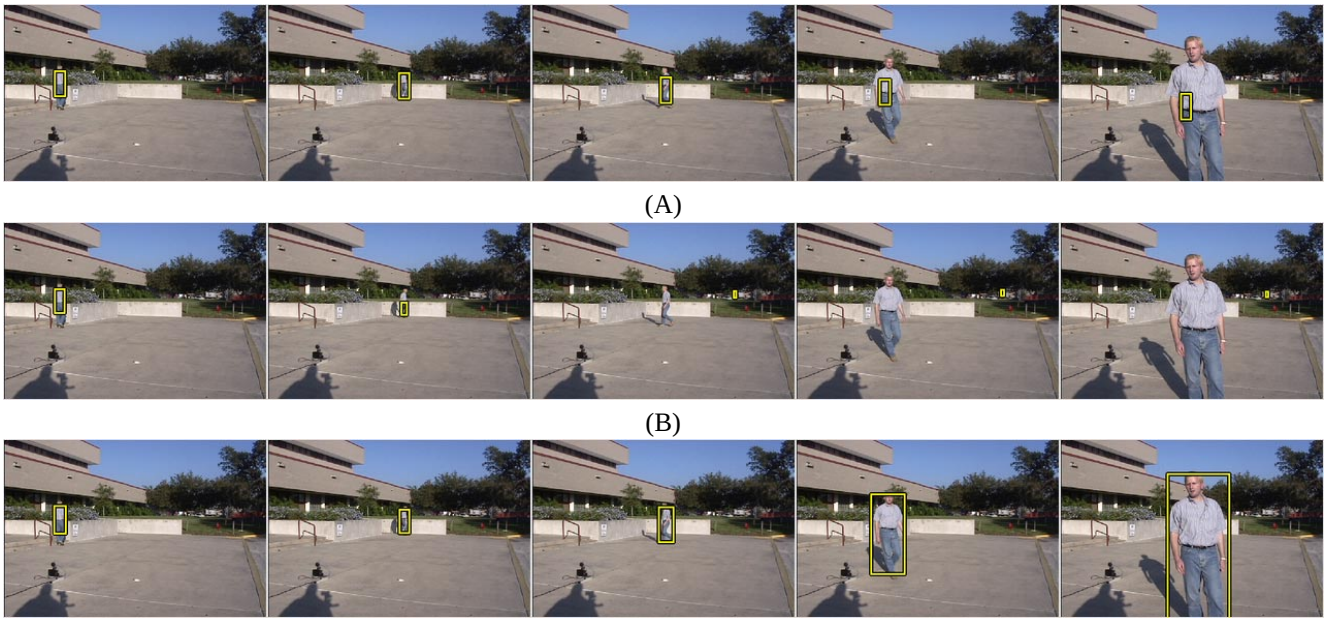
Figure 5B shows tracking results when the scale is adapted using the approach suggested in [4]. At each iteration, the mean-shift algorithm is run three times, once with the current scale, and once with window sizes of plus or minus 10 percent of the current size. For each, the color distribution observed within the mean-shift window after convergence is compared to the model color distribution using the Bhattacharyya coefficient, and the window size yielding the most similar distribution is chosen as the new current scale. We see that the window quickly shrinks too much, a common failure of this scale selection approach. At roughly a third of the way through the sequence, the tracking is lost.

Finally, Figure 5C shows the results of our kernel size selection method, described in the last section. The person is consistently tracked, both in image location AND in scale. Note in particular the correct adaptation of kernel scale as the person's size rapidly expands near the end of the sequence.

6. Conclusion

We have addressed the problem of choosing the correct scale at which to track a blob using the mean-shift algorithm. We start with Lindeberg's theory of blob scale detection using differential scale-space filters, and define an efficient mean-shift procedure that tracks the modes in this scale space. Each mode represents the spatial location and scale of an image blob.

Conceptually, the scale space is generated by convolving a filter bank of spatial DOG filters with a sample weight image. The results are then convolved with an Epanichikov kernel in the scale dimension. Explicit generation of these convolutions in order to find the modes would be very expensive. However, by exploiting the relationship between shadow kernels and mean-shift kernels, we can derive "designer" mean-shift kernels that find the modes much more efficiently. A two-stage mean-shift procedure is developed that interleaves spatial and scale mode-seeking to track a local mode in scale space. The result is stable and natural selection of an appropriate kernel size for appearance-based image blob tracking.



(A)

(B)

(C)

Figure 5: Tracking examples. (A) Using a fixed-scale mean-shift kernel. The person is tracked through the sequence, but localization is poor when the person's size increases. (B) Using the plus or minus 10 percent scale adaptation method (see text). The kernel soon shrinks too much, leading to tracking failure. (C) Using the scale-space mode-tracking method presented in this paper. The person is tracked well, both spatially and in scale.

References

[1] Bradski, G.R., "Computer Vision Face Tracking for Use in a Perceptual User Interface," *IEEE Workshop on Applications of Computer Vision*, Princeton, NJ, 1998, pp.214-219.

[2] Bretzner, L. and Lindeberg, T., "Qualitative Multi-scale Feature Hierarchies for Object Tracking," *Journal of Visual Communication and Image Representation*, Vol 11(2), June 2000, pp.115-129.

[3] Cheng, Y., "Mean Shift, Mode Seeking, and Clustering," *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol 17(8), August 1995, pp.790-799.

[4] Comaniciu, D., Ramesh, V. and Meer, P., "Real-Time Tracking of Non-Rigid Objects using Mean Shift," *IEEE Computer Vision and Pattern Recognition*, Vol II, 2000, pp.142-149.

[5] Comaniciu, D., Ramesh, V., Meer, P., "The Variable Bandwidth Mean Shift and Data-Driven Scale Selection," *International Conference on Computer Vision*, Vol I, pp.438-445.

[6] Fukanaga, K. and Hostetler, L.D., "The Estimation of the Gradient of a Density Function, with Applications in Pattern Recognition," *IEEE Trans. Information Theory*, Vol 21, 1975, pp.32-40.

[7] Hildreth, E.C., "The Detection of Intensity Changes by Computer and Biological Vision Systems," *Computer Vision, Graphics and Image Processing*, Vol 22(1), April 1983, pp.1-27.

[8] Lindeberg, T., "Feature Detection with Automatic Scale Selection," *International Journal of Computer Vision*, Vol 30(2), November 1998, pp.79-116.