Mean-Shift Visual Tracking with NP-Windows Density Estimates

Chuan Zhao
zhao@robots.ox.ac.uk
Ian Reid
ian@robots.ox.ac.uk

Robotics Research Group
University of Oxford
Oxford, UK

Abstract

The mean-shift algorithm is a robust and easy method of finding local extrema in the density distribution of a data set. It has been used successfully for visual tracking in which the target is modelled using a colour histogram, and the image window with best matching histogram is sought. However a histogram is potentially a poor estimate of the underlying colour distribution: it is not invariant to the image scale, the number of histogram bins or the number of samples, and this can have an adverse affect on the speed and accuracy of convergence of the mean-shift algorithm.

We apply a general non-parametric PDF estimation method [7] to replace the histogram in mean-shift tracking to improve its accuracy. This algorithm uses an interpolation scheme which fits piecewise functions to the signal samples, calculating the PDF by accumulating the contribution of each piecewise function. Its accuracy is dependent only on the accuracy of the piecewise representation, not the number of samples or the number of bins. Experiments are conducted to demonstrate that we improve mean-shift visual tracking both in accuracy and speed.

1 Introduction

Mean-shift is a popular mode-finding algorithm applied to various aspects of computer vision. In arguably the most compelling application of the algorithm, [4] showed how it can be used for fast effective visual tracking of coloured blobs; here a local maximum of a similarity measure between two colour distributions (PDFs) is sought. The estimation of these colour distributions in mean-shift visual tracking is essential since the goodness of convergence largely depends on them, and so if the accuracy of the distribution estimation can be improved, the tracking performance will also potentially improve, and we would reduce the risk of losing the target from accumulation of bias.

In the standard mean shift algorithm, the simplest non-parametric representation, namely a histogram, is used as the estimate of the colour probability distribution for both the target and candidate window. It satisfies the low computational cost imposed by real-time processing. However, it is potentially an impoverished representation, being neither invariant to the image scale, the number of histogram bins nor the number of samples. An alternative non-parametric PDF estimate algorithm is Parzen windowing, which avoids arbitrary bin assignments and leads to smoother PDFs. However, a suitable kernel shape and size must be chosen, not always a trivial task.

BMVC 2010 doi:10.5244/C.24.103

Kadir and Brady [7] noted that, when forming the distribution of a *sampled signal* (such as an image), two important properties of signals can be used to improve the PDF estimate: first, the samples are ordered; second, if the sampling rate was sufficiently high, then the samples contain all the information required to reconstruct the original continuous (band-limited) signal exactly. Using these facts they proposed general non-parametric technique for accurate and stable PDF estimation, which calculates the PDF of neighbouring samples directly in closed-form to accumulate the full PDF of the signal.

The key contribution of our paper is to use Kadir's non-parametric windows method to replace the histogram for estimating the colour probabilities in mean-shift visual tracking. We show, in particular, that this leads to faster and more accurate convergence, and by making use of a (separable) Gaussian kernel rather than the more usual Epanechnikov kernel, that the actual run-time can be reduced as well.

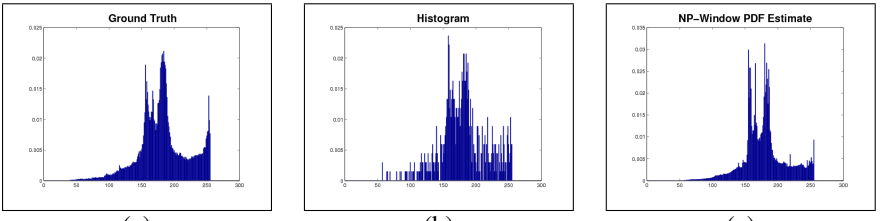
2 Non-parametric (NP) Windows for PDF Estimation

Accumulating a histogram is a standard and expedient method to obtain an estimate of the probability density from which a data set was generated. The NP-Windows algorithm, developed by Kadir and Brady in [7], is a method for estimating the underlying probability density of a signal which is more accurate than a histogram for a particular class of data set, namely samples from a band-limited signal. It was inspired by the observation that upsampling of a signal can be used to deal with an impoverishment of samples from which to generate a histogram, since the upsampled signal has many more samples. Kadir *et al.* noted that for band-limited signals which have been critically sampled (*i.e.*, at faster than the Nyquist rate), upsampling can be achieved to arbitrary accuracy, and that therefore, for this class of data-set, the estimation of the underlying probability density of the data could be much more accurate than could be achieved by treating each data point as an independent sample and accumulating a histogram.

In this section, we briefly recap Kadir's NP-windows method. In this method, the original signal is not directly upsampled using interpolation; rather since the final goal is to estimate the PDF, the re-sampling step is avoided, and the PDF is instead calculated directly in closed-form by considering the analytic contributions from each pair of neighbouring samples (or in 2D, each 4-connected neighbourhood). The main process consists of three main steps: first, between each pair of neighbouring samples (x_i, y_i) and (x'_i, y'_i), the coefficients of a polynomial interpolation $(y = Y(x))$ are calculated; second, the PDF of y, $p_y(\cdot)$ is calculated analytically by transforming a uniform distribution on x through the transformation $Y(x)$; finally, bins at an arbitrary precision are populated from the $p_y(\cdot)$ for each pair of samples in order to generate the PDF for the whole target region. Although the last step generates the PDF by populating bins, the accuracy of this depends only on the accuracy of the piecewise representation, not the number of samples nor the number of bins.

The signals we are dealing with are images, and we use bilinear interpolations. Kadir *et al.* showed that it is necessary to introduce a dummy variable to compute the contribution of a bilinearly interpolated 4-neighbourhood (which is subsequently marginalised out). The limits of this integration involve a number of special cases: Kadir *et al.* noted 24 (calculated as 4!) such cases, and we refer the reader to their papers [7, 8] rather than reproduce that detail here.

By way of example, Figure 1 shows the performance of different PDF estimators for real images; the middle one is standard histogram and the right one is generated from NP-



(a) (b) (c)

Figure 1: The ground truth of PDF estimate on a real image (left), the histograms from standard method (middle) and NP-Windows model (right). Note the zero bins in the histogram on the middle graph.

windows on the same image. On the left, it is the ground truth by histogramming an interpolated version of the image using a FIR interpolator. Note, in particular the numerous bins with zero probability in the standard histogram on the middle, which compromise the accuracy of the representation of the underlying colour distribution.

3 Speeding the Convergence of Mean Shift

The mean-shift vector can be generally derived as the normalised density gradient estimate in [6] and also discussed in [3]:

$$m(\mathbf{x}) = \frac{h^2 c_g}{2c_k} \frac{\nabla F_K(\mathbf{x})}{F_G(\mathbf{x})} \tag{1}$$

Here, $F_K(\mathbf{x})$ is the kernel density estimate obtained with kernel $K(\mathbf{x})$ and window radius h; $\nabla F_K(\mathbf{x})$ is the gradient of the kernel density estimate which is (by associativity of the grad operator and the kernel) is also the estimate of the density gradient; $F_G(\mathbf{x})$ is the density estimate at $\mathbf{x}$ computed with the kernel G, where G is determined as follows. The *profile* of kernel K, $k(x)$, is defined by the equation

$$K(\mathbf{x}) = c_k k(\|\mathbf{x}\|^2),$$

G is then defined as the kernel having profile $g(x) = -k'(x)$; that is

$$G(\mathbf{x}) = c_g g(\|\mathbf{x}\|^2)$$

where c_k and c_g are the corresponding normalisation constants. $K(\cdot)$ is called the shadow of $G(\cdot)$. For the Epanechnikov kernel, it is the shadow of the uniform kernel, while the Gaussian kernel and its shadow have the same expression.

Comaniciu's original work in [4] made use of an Epanechnikov kernel K because of its optimality to minimise the average global error between the estimate and the true density, and its computational efficiency since the kernel evaluations involve only pairs of neighbouring points within distance smaller than the bandwidth rather than all pairs of points. Moreover, it leads to a particularly simple and beautiful form for the mean shift vector (because the magnitude of its derivative is one, so G collapses). When applying the Epanechnikov kernel, convergence is achieved in a finite number of iterations owing to its finite support.

Nevertheless, subsequent work [2] has demonstrated that the use of a Gaussian kernel can have better performance than the Epanechnikov kernel. In particular, use of the Gaussian kernel can decrease the number of iterations required for convergence (providing a lower bound

for the magnitude of the mean shift vector is set), since the kernel G imposes a weighting on the data points according to the distance from its center. Furthermore, since the Gaussian kernel is separable, there are computational gains in using it in preference to the Epanechnikov kernel; *viz*, the 2D kernel density estimate can be computed efficiently using two independent 1D Gaussian symmetrical masks, one in the x direction and one in the y direction.

The computational gains of this approach are not immediately obvious when considering convergence of the standard mean shift, because there, each iteration is almost trivial, so performing a few less iterations doesn't have a measurable effect on the total run time of the optimisation. However, in our case, each iteration requires calculation of the histogram using NP-Windows, which introduces considerable additional computational effort per iteration. We find, therefore, significant gains in using the separable Gaussian in preference to the Epanechnikov kernel, as well as improved accuracy (see Section 5).

Much has been written [1, 5] about the importance of bandwidth selection within mean-shift. We have previously presented a hybrid tracking approach in which we measure the change in scale of a target using affine structure [9]. In the present work we use the output of this calculation to set the bandwidth of the kernels; i.e., we set the bandwidth of the kernels to match the target scale as computed using affine structure.

4 Algorithm Description

Our modified algorithm for mean-shift tracking is summarised as follows: Given the initialised location of the target $\mathbf{y}_0$, using the NP-windows to estimate the distribution $\{q_u\}_{u=1\ldots m}$ of the target model on the very first frame, then:

1. Initialise the location of the target in the current frame with $\mathbf{y}_0$, compute the distribution $\{p_u(\mathbf{y}_0)\}_{u=1\ldots m}$ using NP-windows method in section 2:

$$p_u(\mathbf{y}_0) = C_h \sum_{i=1}^{n} f_i(u)$$

   where C_h is the normalization constant, n is the number of the four 2×2 neighbouring pixel squares, $f_i(\cdot)$ is the PDF estimate of the ith square. Then evaluate the Bhattacharyya coefficient ρ_0.

2. Calculate the weights $\{\omega(\mathbf{x}_i)\}_{i=1\ldots n}$ as derived in [4].

3. Using a 2D Gaussian kernel to estimate the mean-shift vector, derive the new location of the target, according to the method mentioned in section 3:

$$\mathbf{y}_1 = \frac{h^2}{2} \frac{\sum_{i=1}^{n} \omega(\mathbf{x}_i)\mathbf{x}_i g(\| \frac{\mathbf{x}-\mathbf{x}_i}{h} \|^2)}{\sum_{i=1}^{n} \omega(\mathbf{x}_i) g(\| \frac{\mathbf{x}-\mathbf{x}_i}{h} \|^2)}$$

4. Update $\{p_u(\mathbf{y}_1)\}_{u=1\ldots m}$, and evaluate the Bhattacharyya coefficient ρ_1.

5. While $\rho_1 < \rho_0$ Do $\mathbf{y}_1 \leftarrow \frac{1}{2}(\mathbf{y}_0 + \mathbf{y}_1)$.

6. If $\|\mathbf{y}_1 - \mathbf{y}_0\| < \varepsilon$ Stop. Otherwise, set $\mathbf{y}_0 \leftarrow \mathbf{y}_1$ and go to Step 1.

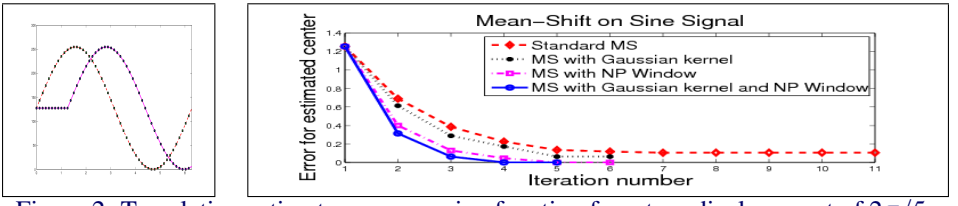


Figure 2: Translation estimate errors on sine function for a true displacement of $2\pi/5$.

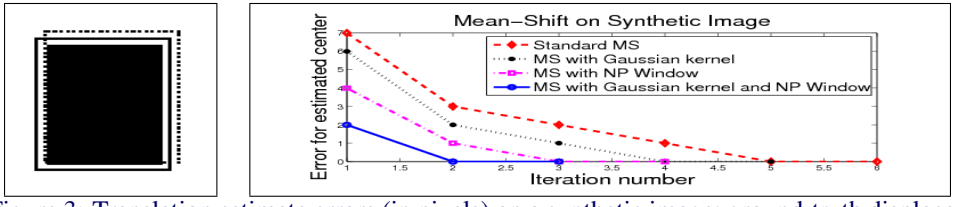


Figure 3: Translation estimate errors (in pixels) on a synthetic image; ground-truth displacements are 7 in x-axis and -9 in y-axis.

5 Experiments and Results

We applied our model on 1D signals, 2D synthetic images, 2D real images as well as sequences. In all the cases we employ a 256 bin histogram. The first experiment compares the accuracy for four models: mean-shift with Epanechnikov kernel (we refer to this hereafter as "standard mean-shift"), mean-shift with Gaussian kernel, mean-shift with NP-windows and mean-shift with both NP-windows and Gaussian kernel. The objective is to estimate the displacement of the shifted signals from the original signals. The 1D signals are one period of the sine function, $127.5\sin(x+\phi)+127.5$, on which we choose 50 samples, a synthetic image and a real image, using four different models of mean-shift. The displacement is $2\pi/5$ for the sine function and $\Delta x = 7, \Delta y = -9$ for the synthetic image, while for the real image, the displacement in x direction is 15 and 20 in y direction. The accuracies are calculated by evaluating the fixation error between the measurement and the ground truth location.

The computation time for the four models running on the sine signal were 22ms, 20ms, 48ms and 22ms respectively, thus with only NP-windows the computation time is about twice of the standard mean-shift, (though the number of iterations is half of the standard mean-shift), but with Gaussian kernel, the time is decreased so that it is comparable to the original mean-shift. Figure 2 shows the result of all four models on the sine function; the x axis is the iteration number, and the y axis is the accuracy measured by calculating the distance between the zero shifted signal and that resulting from the translated signal for each method at every iteration. Figure 3 and Figure 4 show the results on a simple black-and-white synthetic image and on a real image respectively. All three figures demonstrate that mean-shift with Gaussian kernel and NP-windows yields the best accuracy with the fewest iterations.

Next we examine the performance of the proposed technique for video sequences as follows. Two sequences are filmed at 15 fps, and each is 640×480 pixels resolution. We measure the accuracy by calculating the distance in pixels between the ground truth (hand-labelled) and resulting position estimated from the four methods, normalised by the target scale (*i.e.*, the error quoted is the pixel error divided by half the target size). Speed is measured both in the absolute computation time and numbers of iterations for the mean-shift

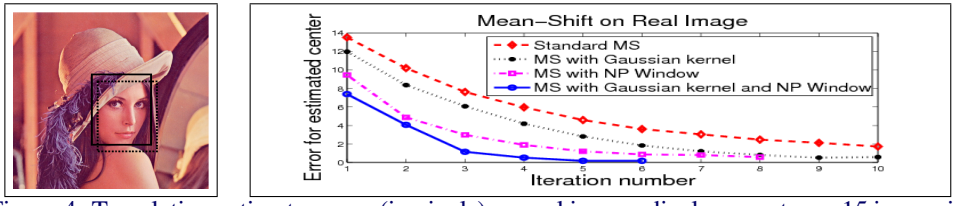
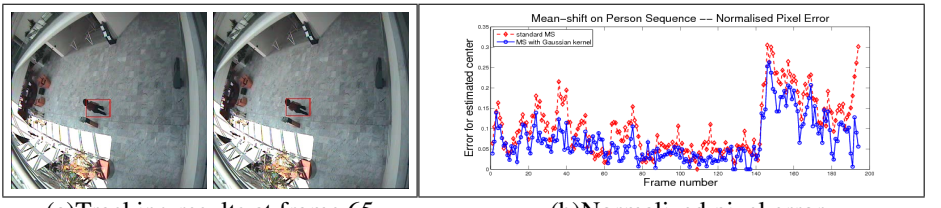
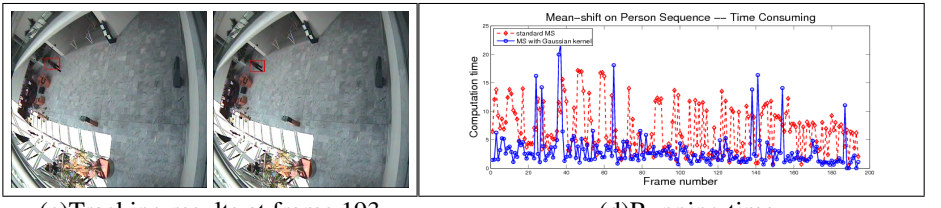


Figure 4: Translation estimate errors (in pixels) on real image, displacements are 15 in x-axis and 20 in y-axis.



(a)Tracking results at frame 65 (b)Normalized pixel error



(c)Tracking results at frame 193 (d)Running time

Figure 5: Tracking results of lobby sequence at frame 65 (a) and 193 (c): standard MS(left) MS with Gaussian kernel and NP-windows(right); (b) shows normalized pixel error: red dashed line is MS and blue real line represents MS with Gaussian kernel and NP-windows.

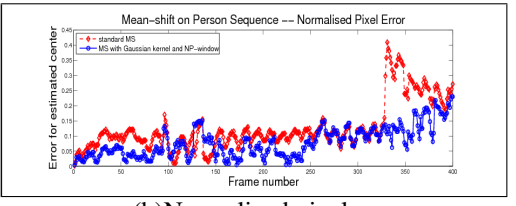
process. Table 1 gives the numerical results, showing the average error, running time and number of iterations for the different models. Figure 7 shows the results for the person sequence and Figure 8 for the vehicle sequence.

Standard mean-shift tracking is notoriously poor at dealing with changes in target scale. In order to prevent scale changes biasing the results for the different methods, we use the technique described in [9] to precompute the scale using affine structure. This scale sets the window size and kernel bandwidth for each frame of each sequence, and these same values are used when running each algorithm.

We also compared our combined model with standard mean-shift using the dataset from the standard "CAVIAR" surveillance video dataset (http://homepages.inf.ed.ac.uk/rbf/CAVIAR/). Figure 5(a) and (c) shows tracking results on a sequence of a person walking taken by a wide angle lens in a lobby. The left subfigures in Figure 5(a) and (c) are from standard mean-shift, while the right subfigures are from mean-shift with both NP-windows and Gaussian kernel. The normalized pixel error (compared to the ground truth) and running time are showed in Figure 5(b) and (d) respectively. More results in Figure 6 demonstrate using these two methods to track a person walking in a group in a shopping center sequence. Like as Figure 5, Figure 6 (a) and (b) shows tracking results from both models. Normalized pixel error is in Figure 6 (c) and running time is in Figure 5(d).

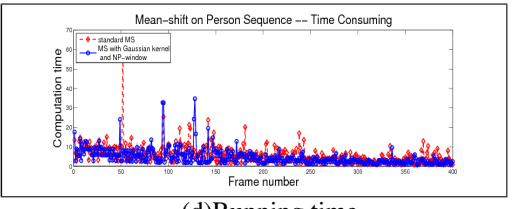
For the above cases, in the large majority of frames, the model with NP-windows and Gaussian kernel has the lowest fixation error, runs in the fewest iterations and in the least

(a)Tracking results at frame 150



(b)Normalized pixel error

(c)Tracking results at frame 350



(d)Running time

Figure 6: Tracking results of shopping center sequence at frame 150 (a) and 350 (c): standard MS(left) MS with Gaussian kernel and NP-windows(right); (b) shows normalized pixel error: red dashed line is MS and blue real line represents MS with Gaussian kernel and NP-windows.

time.

Table 1: Average results on normalised pixel error, the number of iterations, and running time (in milliseconds) for standard MS, MS with Gaussian kernel, MS with NP-windows density estimate and MS with both.

(a) Results on person sequence

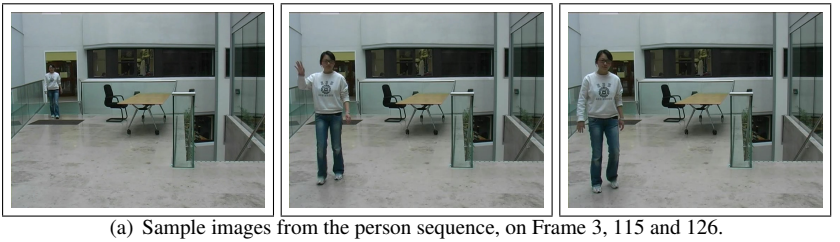
Algorithm	Error	Iteration	Time
MS+Epanechnikov	0.0885	4.2100	33.4857
MS+Gaussian kernel	0.0871	1.9650	22.5025
MS+NP Windows	0.0421	2.5000	40.2398
MS+Gaussian kernel+NP Windows	0.0419	1.8900	12.4759

(b) Results on vehicle sequence

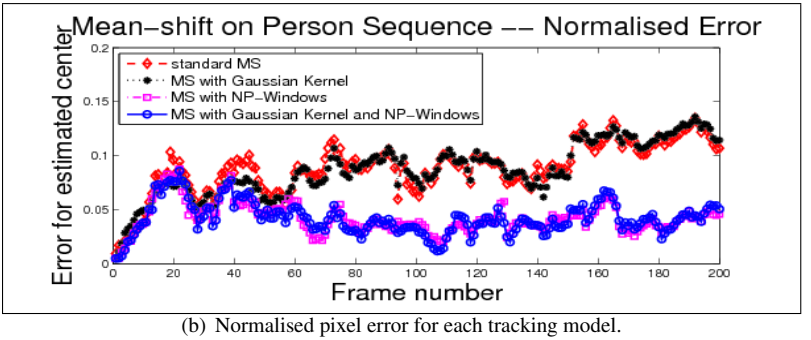
Algorithm	Error	Iteration	Time
MS+Epanechnikov kernel	0.0748	3.9520	29.9621
MS+Gaussian kernel	0.0789	1.4040	10.9280
MS+NP Windows	0.0341	1.6800	10.6157
MS+Gaussian kernel+NP Windows	0.0356	1.3320	8.7649

6 Conclusion

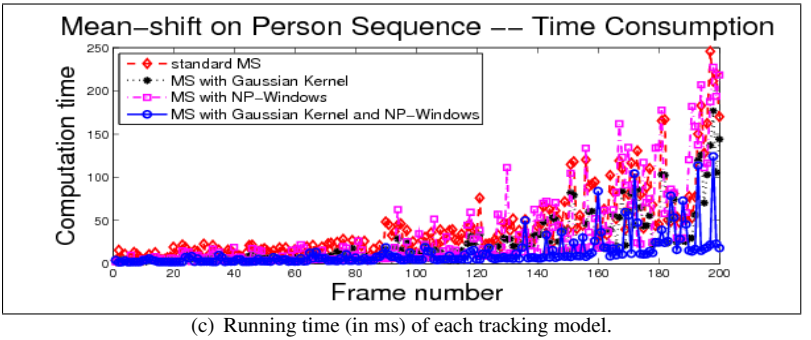
We have presented a method for improving the accuracy as well as the convergence speed of mean-shift visual tracking process, through use of a better approximation of the underlying colour distribution. Though direct application of NP-windows to mean-shift tracking does increase the computational time required, we have further shown that use of the 2D Gaussian



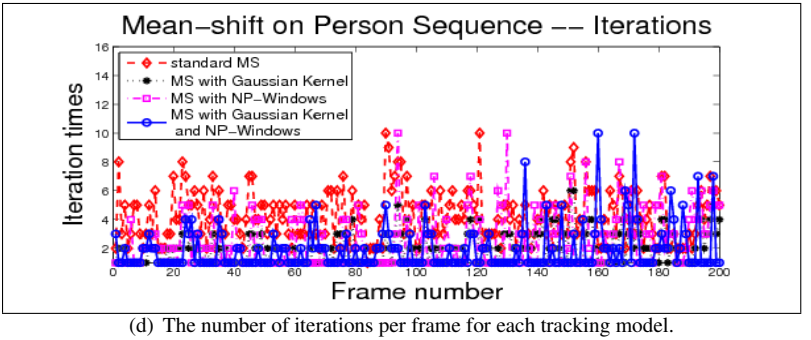
(a) Sample images from the person sequence, on Frame 3, 115 and 126.



(b) Normalised pixel error for each tracking model.



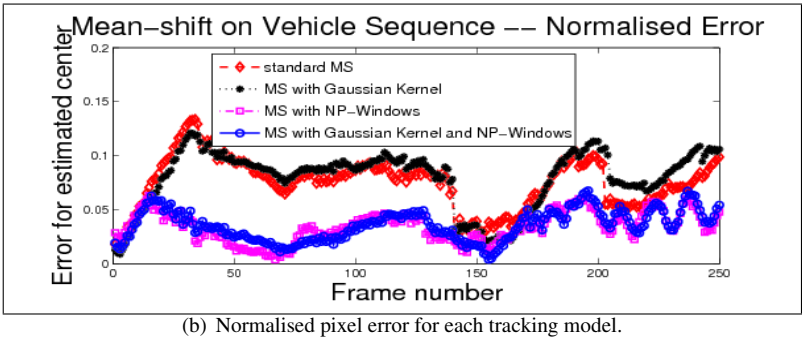
(c) Running time (in ms) of each tracking model.



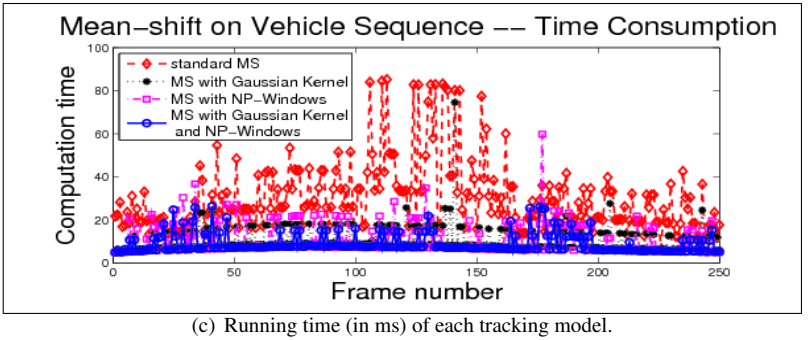
(d) The number of iterations per frame for each tracking model.

Figure 7: Tracking results on person sequence of four models: standard MS, MS with Gaussian kernel, MS with NP-windows and MS with Gaussian kernel and NP-windows.

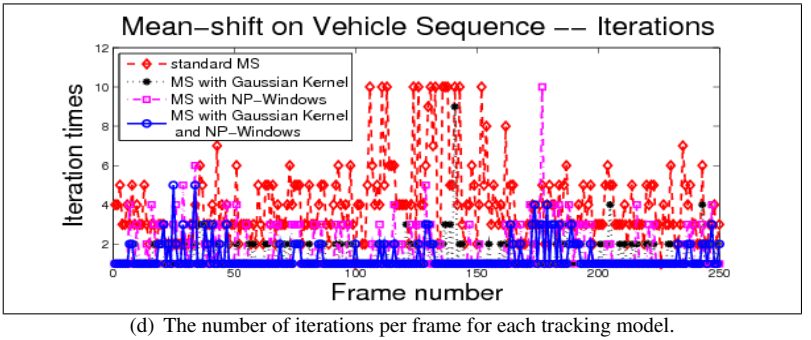
(a) Sample images from vehicle sequence, on Frame 1, 57, 128 and 200.



(b) Normalised pixel error for each tracking model.



(c) Running time (in ms) of each tracking model.



(d) The number of iterations per frame for each tracking model.

Figure 8: Tracking results on vehicle sequence of four models: standard MS, MS with Gaussian kernel, MS with NP-windows and MS with Gaussian kernel and NP-windows.

kernel into separable 1D masks can compensate by reducing the run-time to compute the mean shift vector.

In considering the reasons for the improved performance, we first note that NP-windows effectively uses the available subpixel information; the colour distribution in NP-windows is not only calculated from individual pixels, but from the contributions of their neighbouring pixels and their relationships. Moreover, compared to the standard histogram, NP-windows is invariant to the bin and template size, so that scale changes of a target have less effect on the quality of the colour distribution estimate than when using standard counting to build a histogram. Finally, the convergence of Mean-Shift is based on the distance of the two distributions (template and candidate). Since the density estimation from NP-windows is much closer to the ground truth than a standard histogram, the optimisation is comparing two better approximations of the underlying distribution and so is more likely to converge to the correct result.

As noted in Kadir's original work, NP-windows is a general technique that applies whenever a histogram is being constructed from a signal. The widespread use of this operation in visual tracking means that the technique could be applied, not only in mean-shift tracking, but in any other method for visual tracking that involves building a histogram colour model, with likely improvement in accuracy and robustness.

References

[1] Dorin Comaniciu. An algorithm for data-driven bandwidth selection. *IEEE Trans. PAMI*, 25(2):281–288, 2003.

[2] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Trans. PAMI*, 24:603–619, 2002.

[3] Dorin Comaniciu and Peter Meer. Mean shift analysis and applications. In *Proc. IEEE ICCV*, volume 2, page 1197, 1999.

[4] Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. Real-time tracking of non-rigid objects using mean shift. In *Proc. IEEE CVPR*, volume 2, pages 142–149, 2000.

[5] Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. The variable bandwidth mean shift and data-driven scale selection. In *Proc. IEEE ICCV*, pages 438–445, 2001.

[6] Keinosuke Fukunaga and Larry D. Hostetler. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Trans. on Information Theory*, 21(1):32–40, 1975.

[7] Timor Kadir and Michael Brady. Non-parametric estimation of probability distributions from sampled signals. Technical Report 2283/05, Robotics Research Laboratory, University of Oxford, 2005.

[8] Timor Kadir and Michael Brady. Estimating statistics in arbitrary regions of interest. In *Proc. BMVC*, volume 2, pages 589–598, 2005.

[9] Chuan Zhao, Andrew Knight, and Ian Reid. Target tracking using mean-shift and affine structure. In *Proc. IEEE ICPR*, pages 1–5, 2008.