Line Histogram: A Fast Method for Rotated Rectangular Area Histogramming

Zhiqiang Hou[1], Sion Hannuna[2], Xianghua Xie[3] and Majid Mirmehdi[2]
[1]*Xi'an Jiaotong University*
[2]*Department of Computer Science, University of Bristol*
[3]*Department of Computer Science, University of Swansea*
{*zqhou@mail.xjtu.edu.cn, hannuna@cs.bris.ac.uk, x.xie@swansea.ac.uk, majid@cs.bris.ac.uk*}

Keywords: Line Histogram, Integral Histogram, Efficient Histogramming

Abstract: We propose a novel approach to executing exhaustive histogram search, which incorporates a process to rotate the template used in a computationally efficient manner, e.g. for tracking applications where object rotation will require template rotation. The method is particularly applicable to rectangular templates as any rotation in the region of interest will greatly reduce the potential for a satisfactory match. We present a computational analysis of our proposed method, followed by comparative experimental results.

1 Introduction

The histogram is an important tool in computer vision and is used for a variety of purposes, such as image segmentation (Puzicha et al., 1999; Tobias and Seara, 2002), image retrieval (Brunelli and Mich, 2001; Huang et al., 1997), object detection (Laptev, 2009; Viola and Jones, 2001; Viola and Jones, 2004) and visual tracking (Comaniciu et al., 2003; Czyz et al., 2007; Nummiaro et al., 2003; Perez et al., 2002). This paper is concerned with efficient histogram construction for exhaustive image search, with flexible image regions such as rotated rectangles in arbitrary directions. The proposed technique benefits any algorithm which extracts histograms from regions that are rotating over time. For example, in aerial imaging (Doretto and Yao, 2010), video and images tend to be acquired at a fixed altitude with minimal pan or tilt taking place and the primary motions producing translation and rotation of the image plane. Another example is in human pose estimation where body parts are often modelled as cylinders which have corresponding rectangular regions in the image plane, e.g. Ramanan and Forsyth (Ramanan and Forsyth, 2003) use colour histograms within this type of paradigm to learn human appearance. The line histogram approach described here could speed up many such applications.

One of the major impediments in direct histogram construction is its computational complexity, and while an exhaustive search is guaranteed to find the global minimum, it is often avoided by performing localised searches, e.g., in particle filtering (Czyz et al., 2007; Nummiaro et al., 2003; Perez et al., 2002) and mean shift (Comaniciu et al., 2003; Zhao and Tao, 2009), at the expense of accuracy, i.e. only local minima are obtained (Porikli, 2005; Sizintsev et al., 2008).

The integral histogram is rooted in computer graphics, where Crow (Crow, 1984) first introduced the summed-area table method for texture mapping. This inspired Viola and Jones (Viola and Jones, 2004) to propose a new image representation called an integral image in order to efficiently extract features from rectangular regions in images. The integral image representation, which can be calculated in linear time, is a cumulative function where each element contains the sum of the pixel values above and to the left of the location of the element. This representation was adopted by Porikli et al. (Porikli, 2005) to construct the so called integral histogram (IH) for efficient histogram-based exhaustive search. For each bin of the histogram, an integral image was constructed to count the accumulative number of pixels falling into the bin. The histogram of rectangular regions was then efficiently obtained by accessing those integral images using simple arithmetic operations. Regional statistics, such as covariance, can also be efficiently computed using integral histograms (Porikli and Tuzel, 2006). Further computational improve-

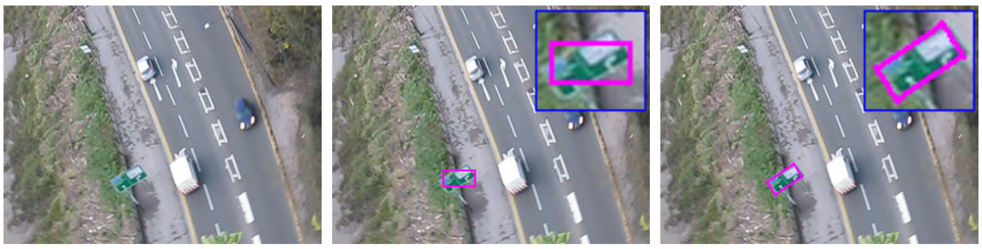

(a) Original image (b) no rotation (CH, IH, or DH) (c) with rotation (LH)

Figure 1: Example of conventional histogram (CH) template match (or IH (Porikli, 2005) or DH (Sizintsev et al., 2008)) against the proposed method (LH).

ment can be achieved through, for example, adaptive approximation (Muller et al., 2008).

Earlier works in efficient median filtering, such as (Huang et al., 1979), also inspired the development of efficient exhaustive search algorithms. In (Huang et al., 1979), Huang et al. made use of overlapping pixels between adjacent windows to address the redundancy along a row in computing the histograms for median filtering. However, no redundancy was exploited between rows. Perreault and Hébert (Perreault and Hebert, 2007) extended (Huang et al., 1979)'s work by removing redundancies both within row and between rows based on the principle of *histogram distributivity*, that is the histogram of two disjoint regions is simply a summation of histograms of those individual regions. The rectangular region of interest was broken up into a union of its columns, each of which maintained its own histogram. The histogram for the region was then computed by summing those adjacent column histograms. This work was followed by Sizintsev et al. (Sizintsev et al., 2008) who constructed the distributive histogram (DH) based on the principles laid in (Perreault and Hebert, 2007) and used it for very fast, exhaustive histogram search for object detection. They also showed that this distributivity can be useful in computing regional statistics, such as mean and covariance matrices.

Recently, Wei and Tao (Wei and Tao, 2010) optimised histogram based searches by calculating the objective function in an incremental fashion. Their research complements Porikli et al.'s (Porikli, 2005) IH and Sizintsev et al.'s (Sizintsev et al., 2008) DH fast exhaustive searches. However, histogram construction using the IH or DH methods is generally carried out on regular rectangular regions, i.e. they are explicitly designed for unrotated rectangular regions. For example, in the case of IH, the integral images for histogram bins are constructed as integration along image coordinates. This may force the exhaustive search to result in undesirable local minima, since the region of interest is subject to arbitrary rotations. A square shape is thus often used; however, this may compromise the representativeness of the target histogram since it can include more irrelevant regions due to a sub-optimal bounding box. The DH suffers from similar difficulties, since there is also no mechanism in DH to take into account object rotations. The histogram initialisation process of DH would have to be repeated for each possible rotation, which is cumbersome to say the least. Here, we propose a method that can handle arbitrary rotations, and can be used to build on the localisation performance of IH and DH to generate even more optimal matches, e.g. see Fig. 1. Indeed, as it will be shown later, disregarding object rotation can in fact lead to complete tracking failure. Furthermore, rotational information is important in determining orientation of objects, e.g. (Zhao and Tao, 2009). In (Lienhart and Maydt, 2002), Lienhart and Maydt used the rotated summed area table or integral image to compute histograms of rectangular regions with 45° rotation. However, this strategy is impractical and inefficient for arbitrary rotations. Moreover, the shape can take any arbitrary form.

Here, we present a simple, flexible and efficient histogram-based exhaustive search method. We also exploit the distributivity of histogram representation, by using vertical or horizontal line integration to obtain rotated histogram representation, instead of confining the process to regular rectangular regions. Hence, we refer to our histogram representation as the *line histogram* or LH. One significant difference between the proposed LH and IH is that the IH prepares histogram initialisation using rectangular areas whereas ours can use rows or columns to conveniently form arbitrarily shaped regions. This leads to a significant advantage in handling non-rectangular region histograms in a computationally cheap and memory efficient manner. However, without losing generality and for the ease of comparison with IH and DH, in this paper we use rotated rectangular area histograms. The proposed method also allows adaptive change of template size in exhaustive search without re-initiating the histogram generation process as in the DH method.

Note in this work we develop the LH in the spirit of the works that introduced the IH (Porikli, 2005) and DH (Sizintsev et al., 2008) where a template of an object is available and one or more unseen frames are searched to optimally locate the template - for object detection or tracking purposes. The rest of the paper is organized as follows. Section 2 describes the proposed LH method. In Section 3, we present a computational analysis of LH. Experimental results are shown in Section 4 and we conclude the paper in Section 5.

2 Line Histogram

Image histograms provide a quantized representation of an image's intensity or colour distribution, the granularity of which is determined by the number of bins used. An important property is that histograms for non-intersecting regions may be summed (Perreault and Hebert, 2007; Sizintsev et al., 2008) as

$$S(A \cup B) = S(A) + S(B) \tag{1}$$

where $S(.)$ is a histogram and A and B are two image regions which may or may not be connected. This property forms the foundation of the integral histogram and distributive histogram techniques and also underpins the line histogram technique proposed here.

When obtaining a single histogram for a whole image, or subset of an image, it is optimal to generate the distribution directly from the pixel data. However, when performing a task that requires histograms to be calculated for overlapping regions, e.g. in an exhaustive search, significant computational savings may be gained by precomputing an image's histogram. For example, the IH (Porikli, 2005) extends the concept of the integral image (Crow, 1984) by precomputing histograms at every pixel location such that the stored histogram at a given pixel describes the distrbution of the subimage contained within the upper left of that location. This allows an arbitrarily sized and positioned histogram to be efficiently calculated from a combination of four previously determined histograms. The DH (Sizintsev et al., 2008) precomputes column histograms for a subset of an image, such that it is specifically tailored to an exhaustive search. The proposed LH method is, in some sense, an extension of the DH in that column histograms are also precomputed, but for full length columns, that span the entire image.

Our LH approach has three phases: an exhaustive search, followed by line histogram initialisation, and finally rotational search. The exhaustive search generates similarity scores for each pixel location. Initialisation precomputes column histograms for the entire image, with each location in a column containing the histogram of all the pixels from the first position to the current position. For rotational search, the template is rotated through $1^\circ \rightarrow 180^\circ$ and the orientation and position yielding the best similarity score is selected.

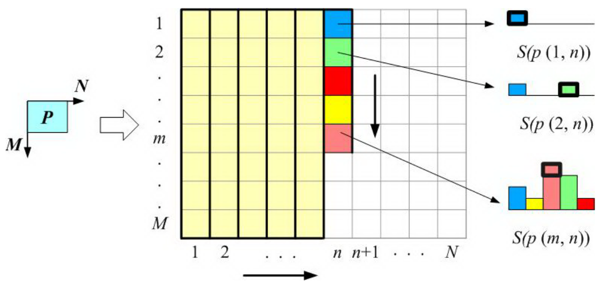
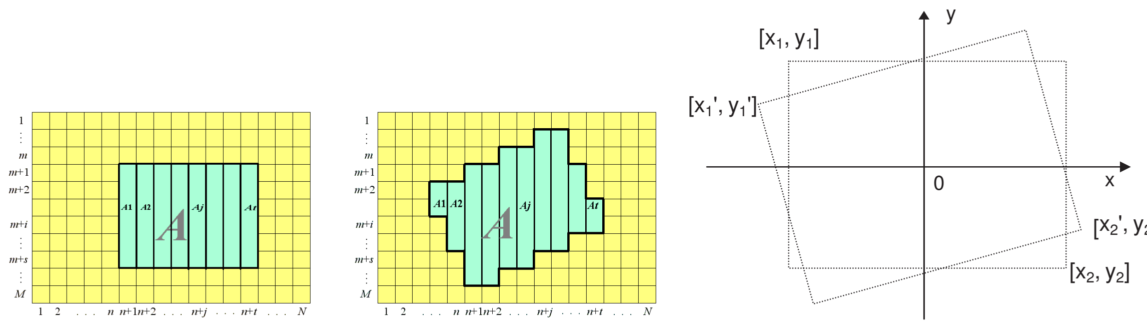


Figure 2: Histogram initialisation.

Exhaustive Search - The first stage of the algorithm is to perform an exhaustive search of the entire image, retaining a subset of image locations most closely matching the unrotated search template. The DH is an extremely fast technique and is applied to this end in our proposed LH method. The question of how many close matches to retain for further attention is not trivial and has thus far been determined empirically. By enforcing a minimum distance between matches, based on the image size, it is possible to ensure that all the matches are not so tightly clustered that potential matches are overlooked. If only a few matches are retained, or if they are localized to a sub-area of the image, it is possible to initialise the LH for small areas containing the matches. This potentially offers additional computational savings. We will touch on this again but later below LH initialisation is explained for the whole image. To perform histogram matching in general, there are many measures that can be used, e.g. the L_1 norm, Bhattacharrya distance, Earth Mover's distance, or χ^2 difference. Here, we use the L_1 norm on 64-bin histograms.

Histogram Initialisation - Consider an image, P, with M rows and N columns. The inititialized LH will then be a $M \times N \times B$ matrix, where B is the number of bins. Starting at the first row, histograms are calculated at each pixel location along each column, with each location's histogram describing the distribution of the pixel located at that position and all those above it on that column (see Fig. 2). In practice, this involves performing a memory copy of the histogram immediately above the current location and then incrementing the bin corresponding to the current pixel. Note the histogram of the first pixel marked $p(1,n)$ in column n is $S(p(1,n))$ and includes one pixel $p(1,n)$, the histogram of the second pixel marked $p(2,n)$ in column n is $S(p(2,n))$ and includes two pixels $p(1,n)$,



(a) Unrotated template (b) Rotated template (c) Rotated template

Figure 3: (a) and (b) Area histogram calculation for unrotated and rotated templates respectively, and (c) effect of rotating template on horizontal displacement..

$p(2,n)$, and the histogram of the m^{th} pixel marked $p(m,n)$ in column n is $S(p(m,n))$ and includes m pixels. This process is repeated for each column in the image so that location M in each column contains the histogram for that entire column. This could be performed for rows instead of columns, but for the sake of clarity, columns are considered in the proceeding discussion.

Rotational Search - Once the LH has been initialized, (1) may be exploited to efficiently calculate histograms for arbitrarily orientated and sized regions. Fig. 3(a) illustrates a rectangular region, A, subdivided into columns A_1 to A_t. Each column's histogram is obtained by subtracting the histogram stored immediately above the start of a column from that at the last pixel in that column:

$$S(A_j) = S(p(m+s, n+j)) - S(p(m, n+j)) \quad (2)$$

where (m,n) is the origin of the area histogram A, $j = 1,\ldots,t$, and s is the length of column j. Next, the sub-histograms, $S(A_j)$, are summed to yield the histogram for A:

$$S(A) = S(A_1) + S(A_2) + \cdots + S(A_j) + \cdots + S(A_t) \quad (3)$$

The discussion above refers to the unrotated rectangular template, but is equally applicable to a rotated one in Fig. 3(b). All that is required are the start and end points for each column across the template's width. In our implementation this is achieved by considering the line equations for the rectangle's four sides at 0° to 90° when its centroid is at the origin. Start and end points are computed for all orientations and these may be used by simply adding the current search coordinates to them.

3 Computational Analysis

The proposed LH method for rotated rectangular histograms can offer significant computational savings over a naive approach where histograms are calculated from scratch, pixel by pixel. Combining histograms requires that corresponding bin counts are added or subtracted from one another. Hence, the greater the number of bins used, the more computationally expensive the LH technique becomes. The naive approach does not involve combining histograms, so is independent of bin count. It is, however, very sensitive to template size as the number of operations required goes with the square of its length. In simple terms, the LH approach offers the greater savings for larger templates and fewer histogram bins.

We next consider the cost of generating histograms for our rotated rectangular search templates. The cost of calculating pixel by pixel conventional histograms (CH) is independent of rectangle orientation because the cost is proportional to search area, which is not a function of angle. For LH, the width of the template and hence the number of columns required is dependent on the rectangle's orientation (see Fig. 3(c)). Note that for anticlockwise rotation by an angle θ ($0 < \theta < \frac{\pi}{2}$), the maximum width, w_{rot}, of the template is $x_2' - x_1'$. For rotation about the origin

$$x_2' - x_1' = (x_2 - x_1)\cos\theta - (y_2 - y_1)\sin\theta \quad (4)$$

If w and h are the height and width of the search template, (4) simplifies to:

$$w_{rot} = w\cos\theta + h\sin\theta \quad (5)$$

Note the sign has changed because $(y_2 - y_1) < 0$. The mean value of w_{rot} is:

$$\overline{w}_{rot} = \frac{2}{\pi}\int_0^{\frac{\pi}{2}} w\cos\theta + h\sin\theta\, d\theta \quad (6)$$

which resolves to $\frac{2}{\pi}(w+h)$. Hence, for a unit length square template the mean number of columns to be considered, $\overline{w}_{rot}$, over a full $\frac{\pi}{2}$ anticlockwise rotation is the mean value of this function: $\frac{4}{\pi}$, which is approximately 1.27. For more elongated templates, e.g. where the height is 10 and the width 50, $\overline{w}_{rot}$ is around 38.2. This is important as it shows that less column histograms have to be summed, on average, than for the full width of the template, reducing cost. Of course the conventional cost is also lower in this case as the area is less.

Without loss of generality, consider a square image P of size $M \times M$ and histogram with B bins, and rectangular template T of size $w \times h$. Using the same entities operated in (Sizintsev et al., 2008), we set d for division, f for floor and type conversion, and a for addition or subtraction.The CH approach operates on wh pixels for each template orientation and location. Operations include: 1 division and 1 floor to find the histogram bin, and this bin is incremented using 1 addition. So the total computational cost of the CH approach for a single orientation and location is $wh(d+f+a)$ and for multiple locations and orientations is

$$n_{loc} n_{angle} wh(d+f+a), \tag{7}$$

where n_{loc} and n_{angle} are the number of locations and angles examined respectively. The LH method requires 1 histogram addition and 1 subtraction to generate histograms for each column and then $\overline{w}_{rot}$ additions to combine them to produce the histogram for the entire area. Hence for n_{loc} locations and n_{angle} angles the operations are:

$$n_{loc} n_{angle} 2\overline{w}_{rot} Ba \tag{8}$$

However, there is also the LH initialisation cost. Excluding the cost of the memory copies, when moving down each column, this is

$$M^2(d+f+a) \tag{9}$$

The costs deduced in the above discussion are summarized in Table 1. Recall that $\overline{w}_{rot}$ is a linear combination of w and h so as they are increased LH becomes cheaper relative to CH. Conversely as B is increased, the relative cost for LH goes up. Furthermore, to benefit from the relative cheapness of LH for rotation, n_{loc} and n_{angle} must be high enough to offset the initialisation cost of LH. In practical terms CH and LH take the same time to execute, where $B = 64$, $w = 33$, $h = 19$, $n_{loc} = 5$, $n_{angle} = 180$ and $M = 277$. Note, the relative savings can be massively increased by reducing M, initialising LH for a subset of the image, or if many locations are being searched, reducing B, using a more compact colour space representation, e.g. HSV, which was used by Sizintsev et al (Sizintsev et al., 2008), who set B to 16.

Memory Requirements - For an $M \times M$ image and histogram representation with B bins, our LH approach, like the IH, requires M^2B units of memory. However, for small images with height < 256 pixels,

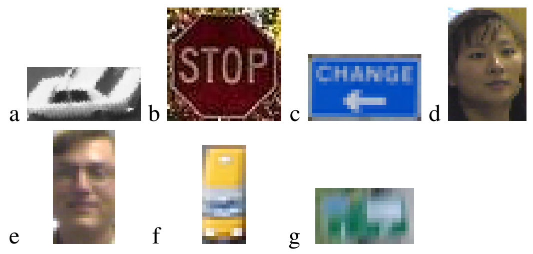


Figure 4: Templates: (a) *Taxi* sequence, (b) and (c) square and rectangular templates, (d) and (e) for face tracking in *Girl* and *Villains*, (f) and (g) for tracking multiple objects in *Road*.

it is only necessary to use unsigned chars for each unit and unsigned short integers will always provide ample capacity for larger images. This contrasts with the IH approach which can be subject to overflow errors (Sizintsev et al., 2008) and requires 4 byte integers as storage units. Furthermore, it is not always necessary to initialise the LH for the entire image region so other potential avenues for reducing memory requirement are available.

4 Experimental Results

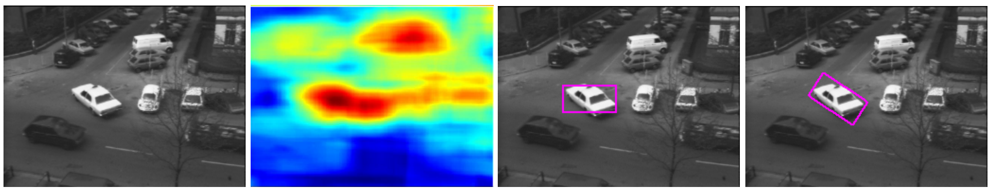
This section has two main objectives: firstly to demonstrate how detection accuracy is improved using a rotating rectangular template and secondly to show how and when the line histogram method is computationally more efficient than the conventional approach. Fig. 4 shows the templates used in the various examples presented in this paper for reference.

Fig. 5 presents two frames of the *Taxi* sequence where the benefits of employing a rotating template is illustrated. The match maps in Figs. 5(b) and 5(f) clearly show two spatially disparate local minima which leads to incorrect detection in Fig. 5(g) for the unrotated histogram. Furthermore, note how the template boundary completely encompasses the object of interest in 5(d) for the proposed method, whereas in Fig. 5(c) the fit is less accurate without rotation, i.e. as the result of IH or DH.

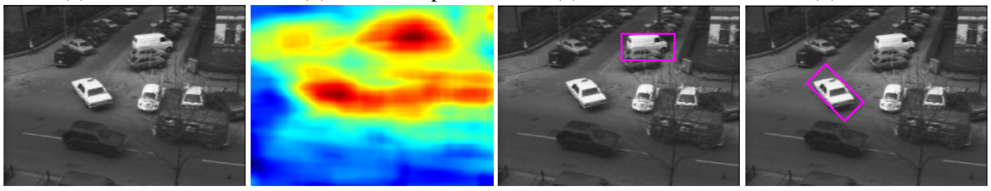
Fig. 6 compares the relevance of using rotated templates for square and rectangular regions at sample rotations. The top row shows the conventional output, i.e. again for IH, or DH, and the bottom row presents the results for the proposed LH method. The contrast in the results is far more striking when a rectangular region is being detected as there is no way to effectively align elongated regions at different orientations. The accuracy of the detection combined with its discovered orientation can benefit higher level recognition processes. Fig. 7 shows histogram sim-

Table 1: Comparison of computational cost.

Method	sequences**Initialisation**	**Rotated Template Histogram Calculation**
CH	0	$n_{loc}n_{angle}wh(d+f+a)$
LH	$M^2(d+f+a)$	$n_{loc}n_{angle}2\overline{w}_{rot}Ba$

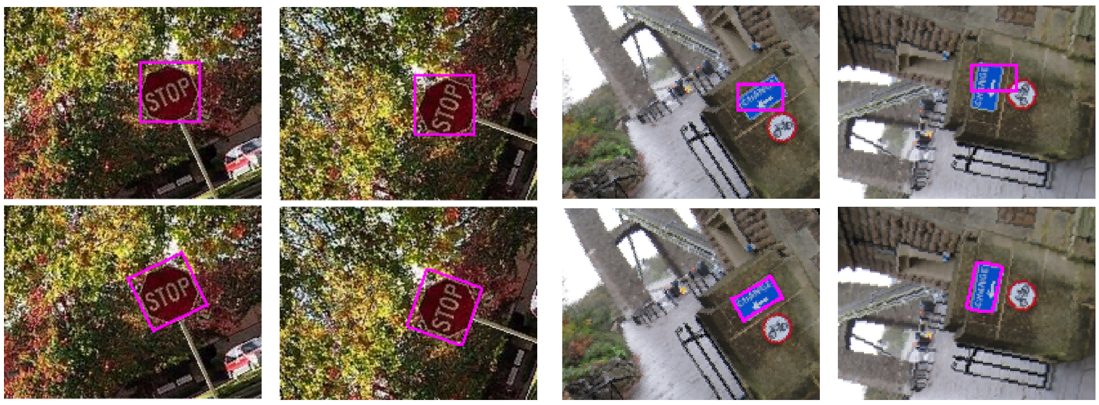


(a) Frame 26 (b) match map (c) Unrotated (d) Rotated



(e) Frame 39 (f) match map (g) Unrotated (h) Rotated

Figure 5: Sequence *Taxi*, comparing results on two different frames. Note the inaccuracy or loss of representation for the conventional unrotated approaches.



(a) 25° (b) 70° (c) 35° (d) 80°

Figure 6: Finding a rotated object using *square* and *rectangular* templates at various rotations - top row: unrotated template approaches (IH or DH), bottom row: LH.

ilarity distances at different rotation angles for the square and rectangular templates in Fig. 6. Note how far greater returns are gained for the rectangular template relative to the square template. The latter shows maximum error at around 40° whereas the former peaks at around 90°.

In Fig. 8, the results of applying histogram detection to face tracking are shown in frames from two different sequences, *Girl* and *Villains*. In both sequences superior alignment is achieved with LH's rotating template and the face orientation is revealed. Figs. 8(i) and 8(j) illustrate how varying rotational angle yields a minimal similarity distance for a fixed location for the frames used in Fig. 8. Furthermore, Table 2 lists the improved similarity scores and agreement with handlabeled groundtruth using rotating templates in the two frames.

We have so far illustrated the benefits of rotating a rectangular search template when performing an exhaustive histogram-based search. Next, we demonstrate empirically the savings made when calculating rotated histograms using the LH approach on a 3.2GHz CPU PC under Win XP, with 3GB RAM. All the code was written in C++.

Table 3 shows the computational times for the various stages of an exhaustive search, including rotation, for a single frame for different sequences and template shapes. The number of locations searched, n_{loc}, is 5, the number of orientations examined, n_{angle}, is 180, and the number of bins, B, is 64. For each of the four sequences, running times are provided for rotated templates calculated in a pixel by pixel fashion (CH) and using LH. Note that the DH search time is the same for CH and LH as it is the optimal method for finding candidate locations to apply rotation. These results show excellent agreement with the computational costs determined analytically in Section 3 and summarized in Table 1. Larger images require longer LH initialisation times whilst larger templates are relatively expensive to calculate using CH. For example, the 'Girl' sequence is processed five times as quickly using LH as the image is small and the template relatively large. Indeed, using LH in conjunction with DH, a rate of 75fps is achieved. On the other hand, for the 'Rectangle' sequence, LH and CH take around the same time to execute as the image is larger and the template smaller.

5 Conclusion

We proposed the Line Histogram, a new method to compute a rotated histogram which is based on sub-histograms along columns of a region. The method is fast and can be used in real-time applications for object detection and tracking, single, or multiple (e.g. see Fig. 9). We outline its computational performance analytically and through several experiments.

6 Acknowledgements

This paper is supported by National Natural Science Foundation of P.R.China(60805015,61175029) and Shaanxi Province Natural Science Foundation(2011JM8015).

REFERENCES

Brunelli, R. and Mich, O. (2001). Histograms analysis for image retrieval. *PR*, 34(8):1625–1637.

Comaniciu, D., Ramesh, V., and Meer, P. (2003). Kernel-based object tracking. *IEEE T-PAMI*, 25(5):564–577.

Crow, F. (1984). Summed-area tables for texture mapping. *ACM Computer Graphics*, 18(3):207–212.

Czyz, J., Ristic, B., and Macq, B. (2007). A particle filter for joint detection and tracking of color objects. *IVC*, 25(8):1271–1281.

Doretto, G. and Yao, Y. (2010). Region Moments: Fast invariant descriptors for detecting small image structures. In *IEEE CVPR*.

Huang, J., Kumar, R., Mitra, M., Zhu, W., and Zabih, R. (1997). Image indexing using color correlograms. In *IEEE CVPR*, pages 762–768.

Huang, T., Yang, G., and Tang, G. (1979). A fast two-dimensional median filtering algorithm. *IEEE T-ASSP*, 27(1):13–18.

Laptev, I. (2009). Improving object detection with boosted histograms. *IVC*, 27(5):535–544.

Lienhart, R. and Maydt, J. (2002). An extended set of haar-like features for rapid object detection. In *IEEE ICIP*, pages 900–903.

Muller, T., Lenz, C., Barner, S., and Knoll, A. (2008). Accelerating integral histograms using an adaptive approach. In *ICISP*, pages 209–217.

Nummiaro, K., Koller-Meier, E., and Gool, L. (2003). An adaptive color-based particle filter. *IVC*, 21(1):99–110.

Perez, P., Hue, C., Vermaak, J., and Gangnet, M. (2002). Color-based probabilistic tracking. In *ECCV*, pages 661–675.

Perreault, S. and Hebert, P. (2007). Median filtering in constant time. *IEEE T-IP*, 16(9):2389–2394.

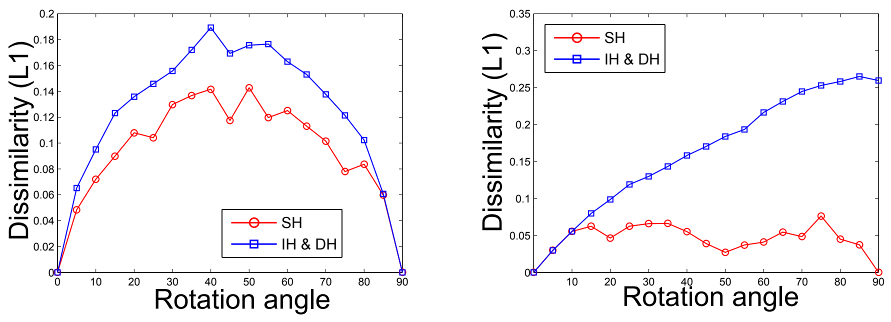
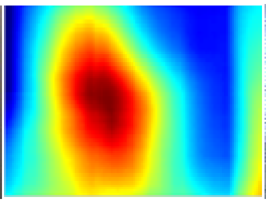


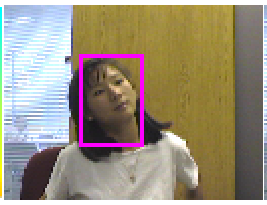
Figure 7: The comparative histogram similarity results for the square (left) and rectangular (right) templates in Fig. 6 with and without rotation.

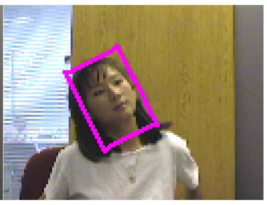
Table 2: Comparison of groundtruth and predicted positions for unrotated and rotated templates. The similarity distance is under the ($L_1 norm$) column

Image	Groundtruth	Method	Predicted	Angle	L_1 norm
Girl frame 69	(46,54)	**Unrotated**	(48,54)	0°	0.144
		Rotated	(47,54)	27°	0.127
Villains frame 14	(51,61)	**Unrotated**	(52,62)	0°	0.363
		Rotated	(51,61)	−45°	0.294

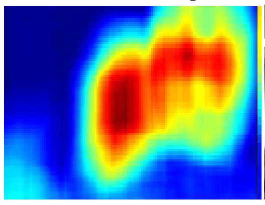
(a) frame 69

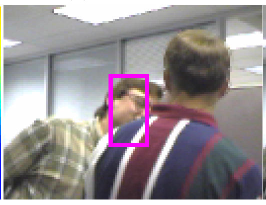

(b) match map

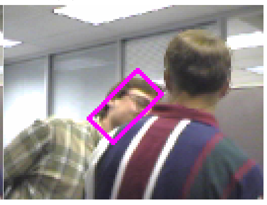

(c) Unrotated

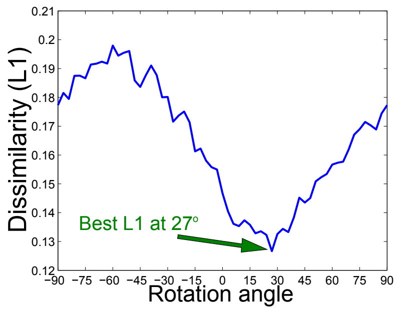

(d) Rotated: LH

(e) frame 14

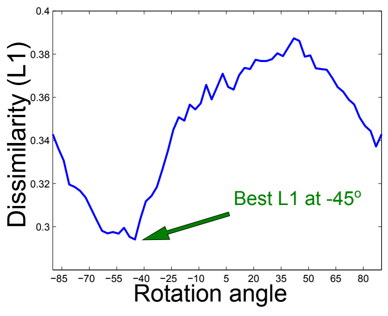

(f) match map


(g) Unrotated


(h) Rotated: LH



(i) at pixel (47,54)



(j) at pixel (51,61)

Figure 8: Face tracking examples, sequences *Girl* and *Villain*, (top and middle rows, from left) original, histogram distance map, unrotated result e.g. IH or DH, and LH's rotated result. (bottom row) minimum dissimilarity measure for each frame.

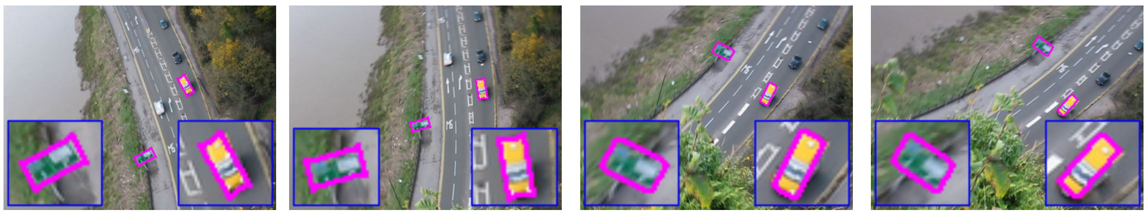

(a) Frame 10 (b) Frame 28 (c) Frame 57 (d) Frame 66

Figure 9: Sequence *Road*, an example of tracking multiple objects.

Table 3: Running time (in *ms*) comparisons for various templates.

Sequence	Image size	Template size	DH search time	Init. time	Rotate time	Total
Girl (CH)	128×96	31×45	1.78	0	64.41	66.19
Girl (LH)	128×96	31×45	1.78	2.45	9.11	13.34
Villains (CH)	128×96	19×35	2.26	0	30.98	33.24
Villains (LH)	128×96	19×35	2.26	2.55	6.10	10.91
Square (CH)	480×360	47×47	54.10	0	307.03	361.13
Square (LH)	480×360	47×47	54.10	51.22	26.68	132.00
Rectangle (CH)	320×240	35×19	25.50	0	24.73	50.23
Rectangle (LH)	320×240	35×19	25.50	20.71	5.16	51.37

Porikli, F. (2005). Integral histogram: A fast way to extract histograms in cartesian spaces. In *IEEE CVPR*, pages 829–836.

Porikli, F. and Tuzel, O. (2006). Fast construction of covariance matrices for arbitrary size image windows. In *IEEE ICIP*, pages 1581–1584.

Puzicha, J., Hofmann, T., and Buhmann, J. (1999). Histogram clustering for unsupervised image segmentation. In *IEEE CVPR*, pages 602–608.

Ramanan, D. and Forsyth, D. (2003). Finding and tracking people from the bottom up. In *IEEE CVPR*, pages 467–474.

Sizintsev, M., Derpanis, K., and Hogue, A. (2008). Histogram-based search: a comparative study. In *IEEE CVPR*, pages 1–8.

Tobias, O. and Seara, R. (2002). Image segmentation by histogram thresholding using fuzzy sets. *IEEE T-IP*, 11(12):1457–1465.

Viola, P. and Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *IEEE CVPR*, pages 511–518.

Viola, P. and Jones, M. (2004). Robust real-time face detection. *IJCV*, 57(2):137–154.

Wei, Y. and Tao, L. (2010). Efficient histogram-based sliding window. In *CVPR*, pages 3003–3010.

Zhao, Q. and Tao, H. (2009). A motion observable representation using color correlogram and its applications to visual tracking. *CVIU*, 113(2):273–290.