Statistical Patch-Based Observation for Single Object Tracking

Mohd Asyraf Zulkifley and Bill Moran

Department of Electrical and Electronic Engineering
The University of Melbourne
Victoria 3010 Australia
m.zulkifley@student.unimelb.edu.au, wmoran@unimelb.edu.au

Abstract. Statistical patch-based observation (SPBO) is built specifically for obtaining good tracking observation in robust environment. In video analytics applications, the problems of blurring, moderate deformation, low ambient illumination, homogenous texture and illumination change are normally encountered as the foreground objects move. We approach the problems by fusing both feature and template based methods. While we believe that feature based matchings are more distinctive, we consider that object matching is best achieved by means of a collection of points as in template based detectors. Our algorithm starts by building comparison vectors at each detected point of interest between consecutive frames. The vectors are matched to build possible patches based on their respective coordination. Patch matching is done statistically by modelling the histograms of patches as Poisson distributions for both RGB and HSV colour models. Then, maximum likelihood is applied for position smoothing while a Bayesian approach is applied for size smoothing. Our algorithm performs better than SIFT and SURF detectors in a majority of the cases especially in complex video scenes.

Keywords: Neyman-Pearson, Tracking observation, Poisson modelling, Maximum likelihood.

1 Introduction

Object detection algorithms have evolved from the implementation of simple edge matching to complex feature matching. In multiple objects tracking systems, good matching technique is important for distinguishing objects between consecutive frames. Robust matching algorithms provide better measurement inputs for updating tracking algorithms. Early feature based object matching algorithms are built on edge information such as Canny [1], Sobel [2] and Roberts [3] edge operators. The detected edges are compared to find similarity traits between objects. In order to improve the detection accuracy, corner detectors such as the algorithms of Harris [4], and Shi and Tomasi [5] are used for finding corners which serve as the points of interest. However, the resulting points are not very distinctive and perform poorly under illumination change. In 1999, Lowe introduced the Scale Invariant Feature Transform (SIFT), [**?**], that generates robust features that can cater for problems of rotation, scaling and moderate illumination

G. Maino and G.L. Foresti (Eds.): ICIAP 2011, Part II, LNCS 6979, pp. 119–129, 2011.

change. Ke and Suthankar [6] improved SIFT's feature distinctiveness by applying principal components analysis. Later, Burghouts and Geusebroek [7] fused SIFT with their colour invariance algorithm [8] to achieve robustness to the illumination change. Performance evaluation of SIFT and its variants are explained in detail by Mikolajczyk and Schmid [9]. They also introduced their own feature descriptor's algorithm, GLoH that enhanced SIFT by integrating more spatial properties during histogram accumulation. In 2006, Bay et al. [10] introduced Speeded Up Robust Features (SURF) which has similar detection performance to SIFT but with a much lower computational burden. However, all these algorithms are based on fixed size kernel matching in which each pixel descriptor is obtained based on certain pixel distance from the anchor pixel. For example, SIFT uses fix kernel size of 16×16 pixels. Recognition performance degrades when the objects are blurred, moderately deformed or possess homogenous texture. Another approach is use template based matching, which requires several templates to be matched by correlation over the search regions. This approach allows a collection of pixels to be compared and matched as a group. This idea is basically the antithesis of a content-based image retrieval (CBIR) approach and provides better robustness in detecting moderately deformed objects. However, its disadvantages include the fact that the search area is usually big and the chosen templates need to be able to cater all possible transformations of the object between frames.

In order to overcome the weaknesses of both approaches, we have fused both feature and template based methods together. The aim of this paper is to find the matches of foreground objects between consecutive video frames especially in complex scenes for updating tracking algorithms observation. We assume that the size of the objects between the consecutive frames does not change too much. The advantage of using fusion approach is it manages to obtain good detection even for the case of object deformation as normally occurs during human movement. Use of the template-based approach alone results in algorithms that have a diminished distinctiveness property because most objects are nonrigid; while for the implementation of feature based detector alone, it is bound to fail as some points are covered or hidden in the later frames. One major difference between image processing and video analytics is the sharpness of the captured image. Usually, in the former one assumes that the image captured has sharp boundaries, while this is not necessarily the case for video. Most feature based algorithms are not able to obtain a match in this context. They also perform poorly in matching objects under low ambient illumination.

Our algorithm is intended to increase robustness of detecting tracking observation in situations where there is a blurring effect, illumination change, low illumination surroundings, homogenous texture and moderate deformed objects. The main idea is to match the same points of interest between frames and use this to develop possible patches for object matching. However, this is performed only to selected points of interest so that the calculation burden is reduced compared to a simple template based approach. To improve detection accuracy, position smoothing is performed by aligning the bounding box to the object centroid. Lastly, size adjustment is performed to cater for small change in object size. Any object will become bigger as it moves closer to the camera and smaller as it recedes. In addition, deformation usually forces a new

bounding box size. Finally, the output patch is fed into any filter-based tracker as the measurement input.

2 Statistical Patch-Based Observation (SPBO)

SPBO is built specifically for video applications that require object matching between consecutive frames. This algorithm has a moderate distinctiveness property yet better recognition accuracy for most applications. The challenge lies in recognizing objects under illumination change, blurring effects, moderate deformation and in handling objects with homogenous texture. The main components of the algorithm are 1) generate possible patches, 2) undertake patch matching, 3) perform position smoothing and 4) perform size smoothing.

2.1 Generate Possible Patches

Point of interest are first used to find the locations to generate vector descriptors. These descriptors are matched between frames for building possible patches. The original location of the object or the original patch in the first frame is initialized by the user. The importance of this patch is that it serves as the reference for building the statistical data used in matching and smoothing procedures; in particular the reference histograms. Moreover, the size of the first frame patch is an indicator of the size of the original object. Let P_w and P_h denote the width and height of the user defined bounding box, while (x, y) represents the coordinate location and t is the time of the frame. Both, the first frame ($F_{r,g,b}^{x,y,t}$) and the second frame ($F_{r,g,b}^{x,y,t+1}$) are transformed to greyscale space ($F_I^{x,y,t}$, $F_I^{x,y,t+1}$). Corner detectors as defined by Shi and Tomasi [5] are applied to find the possible points of interest. The algorithm was chosen because of its ability to generate the points even during low ambient illumination and for low textured objects. For the first frame, the points are generated inside the predefined patch only, while for the second frame, the points are generated for the whole image. A vector descriptor, $\mathbf{V}$ is used for matching the points of interest between frames. Possible bounding boxes are generated at each corner where the vectors are matched. There will be 3 sets of vectors for each point of interest, ($\mathbf{V}_R^{x,y,t}$, $\mathbf{V}_G^{x,y,t}$, $\mathbf{V}_B^{x,y,t}$). Let i denote the channel type and define

$$\mathbf{V}_i^{x,y,t} = \{F_i^{x-1,y,t} - F_i^{x,y,t}, F_i^{x,y-1,t} - F_i^{x,y,t}, F_i^{x+1,y,t} - F_i^{x,y,t}, F_i^{x,y+1,t} - F_i^{x,y,t}\} \quad (1)$$

The vectors are generated by finding the colour difference between the anchor pixel and its selected neighbourhood pixels as shown in Figure 1. Then the vector components are

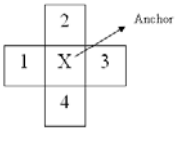

Fig. 1. Neighbourhoods pattern used for vectors generation

sorted from the lowest to the highest value. Each vector from an initial frame is compared with each vector in the next frame. The decision rule (dr) for vectors matching is shown in equation 2 where the differences between each vector component are summed up and the final value is obtained by combining all 3 channels differences. Then it is compared with a predefined threshold, γ_1 which was found by experiment to be optimal in the range of 11 to 13. Let $L_1^{x,y,t}$ denotes the label which takes the value 1 when the vectors are matched and 0 for unmatched vectors.

$$dr = \sum_{i=R}^{B} \left| \mathbf{V}_i^{x,y,t} - \mathbf{V}_i^{x,y,t-1} \right| \tag{2}$$

$$L_1^{x,y,t} = \begin{cases} 1 \text{ if } dr < \gamma_1 \\ 0 \text{ if } dr \geq \gamma_1 \end{cases} \tag{3}$$

All of the matched vectors are candidate for locations at which patches are built. Patches for the second frame are generated around the location of the matched vector in the first frame with respect to the original bounding box. Figure 2 shows an example of how the bounding box is generated. Initially, the size of the object is assumed to remain constant between frames.

A subsequent test for distinguishing overlapping patches is performed after all patches have been assigned location and size. This is done in order to reduce the calculation burden by reducing the number of patches. Patch smoothing is performed if the overlapping area is more than 70% of the original patch size.

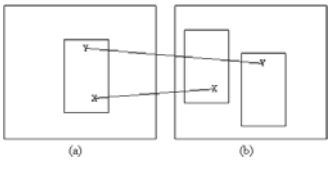

Fig. 2. Examples of constructing the new patches between the frames. The bounding boxes are aligned with respect to the matched vectors in the first frame. (a) First frame (b) Second frame.

2.2 Patch matching

Patch matching is performed to find the patch where the object most likely resides. The match is done by comparing the histograms of the first and second frame patches. Two colour models are considered: RGB and HSV colour spaces. For the case of no illumination change, use of RGB colour space gives a better histogram comparison. When illumination change occurs, the hue channel from the HSV colour model gives better comparison since the hue channel remain invariant but the distinctive feature are degraded. For RGB colour space, a 3-dimensional histogram is built for each patch while a 1-dimensional histogram is built for the hue channel. All histogram matching is done by modelling the relationship between two histograms as a Poisson distribution as in 4 and 5. We chose Poisson distribution as it gives good probability density function for histogram matching which later will be integrated into Bayesian-based decision. Let

N_b be the number of histogram bins in 1-dimensional, n_i and m_i denotes the i^{th} bin value of the first and second frame histograms respectively.

For the 1-dimensional histogram:

$$P(n_{(i)}, m_{(i)}) = \prod_{i=1}^{N_b} \left(\frac{\exp^{-n_{(i)}} n_i^{m_{(i)}}}{m_{(i)}!} \right), \tag{4}$$

and for the 3-dimensional histogram:

$$P(n_{(i,j,k)}, m_{(i,j,k)}) = \prod_{i=1}^{N_b} \prod_{j=1}^{N_b} \prod_{k=1}^{N_b} \left(\frac{\exp^{-n_{(i,j,k)}} n_{(i,j,k)}^{m_{(i,j,k)}}}{m_{(i,j,k)}!} \right) \tag{5}$$

A maximum likelihood approach is used to find the matched patch for both colour models. The likelihoods are modelled by equations 4 and 5 where $\hat{\beta}$ denotes the matched patch and **x** represents the observation.

$$P(\mathbf{x}|\beta) = \begin{cases} P(n_{(i)}, m_{(i)}) \text{ for HSV colour model} \\ P(n_{(i,j,k)}, m_{(i,j,k)}) \text{ for RGB colour model} \end{cases} \tag{6}$$

$$\hat{\beta} = \underset{\forall \beta}{\operatorname{argmax}}\, P(\mathbf{x}|\beta) \tag{7}$$

There are two candidates for the most likely patch. The decision to choose hue over the RGB colour model is decided by using a Neyman-Pearson hypothesis testing [11]. Let $P(\mathbf{x}; \mathcal{H}_0) = P(\hat{\beta}_{\text{RGB}})$, $P(\mathbf{x}; \mathcal{H}_1) = P(\hat{\beta}_{\text{hue}})$ and λ_1 represent the threshold for the Neyman-Pearson hypothesis testing. If the test favours $\mathcal{H}_0$, then an indicator, ϵ is initialized as 1, while if $\mathcal{H}_1$ is chosen, ϵ is equal to 0. The parameter ϵ is the indicator for deciding which colour space is used for the position and size smoothing. The resultant patch (β_{fin_1}) from the test will be the final matched patch.

$$\text{NP}_1 = \frac{P(\mathbf{x}; \mathcal{H}_1)}{P(\mathbf{x}; \mathcal{H}_0)} = \frac{P(\hat{\beta}_{\text{hue}})}{P(\hat{\beta}_{\text{RGB}})} > \lambda_1 \tag{8}$$

$$\beta_{\text{fin}_1} = \begin{cases} \hat{\beta}_{\text{RGB}} \text{ if } P(\hat{\beta}_{\text{hue}}) < \lambda_1 P(\hat{\beta}_{\text{RGB}}) \\ \hat{\beta}_{\text{hue}} \text{ if } P(\hat{\beta}_{\text{hue}}) \geq \lambda_1 P(\hat{\beta}_{\text{RGB}}) \end{cases} \tag{9}$$

2.3 Position Smoothing

Position smoothing is used to adjust the centroid of the patch to accurately align with the centroid of the object. Sometimes, the calculated patch is slightly misaligned with the original object which is prevalent during illumination change or in low ambient illumination. There are two techniques which the patch position is adjusted depending on the value of ϵ with the RGB histogram applied for $\epsilon = 1$ and the hue histogram for $\epsilon = 0$. Firstly, the step size used for adjusting the patch translation is determined, $\delta = \alpha(\min(P_w, P_h))$. Let α denote a weight factor which is found experimentally to be optimal within $[0, 0.5]$ based on the assumption that the object size does not change

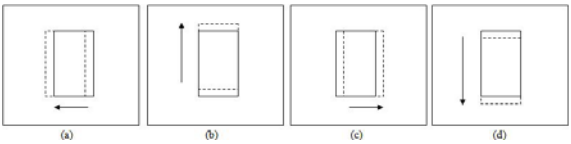

Fig. 3. Patches coordination for location smoothing (a) Left side translation (b) Upward translation (c) Right side translation (d) Downward translation

abruptly between two consecutive frames. Four new candidate patches are created for adjusting the patch position as shown in Figure 3, representing translations in four directions of the pivot patch ($\beta_{\text{fin}_1}^{\text{old}_0}$): leftward ($\beta_{\text{fin}_1}^{\text{new}_1}$), upward ($\beta_{\text{fin}_1}^{\text{new}_2}$), rightward ($\beta_{\text{fin}_1}^{\text{new}_3}$) and downward ($\beta_{\text{fin}_1}^{\text{new}_4}$). The histograms of each of the five patches including the original position patch are obtained, and again maximum likelihood is used to find the new location. No histogram normalization is needed in this subsection as both original and candidate patches are of the same size. Likelihood is derived from the relationship between the first and second frame histograms as in equations 4 and 5. Let $\hat{\beta}_{\text{fin}_2}$ denotes the output of the position smoothing.

$$P(\mathbf{x}|\beta_{\text{fin}_1}) = \begin{cases} P(n_{(i)}, m_{(i)}) \text{ if } \epsilon = 0 \\ P(n_{(i,j,k)}, m_{(i,j,k)}) \text{ if } \epsilon = 1 \end{cases} \tag{10}$$

$$\beta_{\text{fin}_2} = \underset{\forall \beta_{\text{fin}_1}}{\text{argmax}}\, P(\mathbf{x}|\beta_{\text{fin}_1}) \tag{11}$$

For each iteration, the pivot position is reinitialized by letting $\beta_{\text{fin}_1}^{\text{old}_0} = \beta_{\text{fin}_2}$, so that 4 new translated patches for the next iteration are built around β_{fin_2}. The algorithm is iterated until the estimated patch position remain the same as shown by the decision rule L_2.

$$L_2 = \begin{cases} \beta_{\text{fin}_2} = \beta_{\text{fin}_1}^{\text{old}_0} \text{ stop the iteration} \\ \beta_{\text{fin}_2} \neq \beta_{\text{fin}_1}^{\text{old}_0} \text{ continue the iteration} \end{cases} \tag{12}$$

2.4 Size Smoothing

This section focuses on adjusting the size of the patch so that it provides a good fit to the object. Generally, the image of the object becomes bigger as it moves closer to the camera and smaller as it moves away. However, the size increment and decrement between the frames will not be very large. We limit the scale change for size smoothing by at most a factor of $\frac{1}{2}$. Eight new patches with different sizes are used for the size smoothing test. The same δ used in position smoothing is used to adjust the patch size. Four shrinkage and four expansion pattern patches are obtained either by subtracting from or adding to one of the patch corners by a step size value. Figure 4 shows a shrunk patch, while an expanded patch is shown in Figure 6. $\hat{\beta}_{\text{fin}_2}$ is the pivot point for creating all the new patches. A Bayesian approach is used to decide the final patch size ($\hat{\beta}_{\text{fin}_3}$) from among the nine patches including the original patch ($\hat{\beta}_{\text{fin}_2}$).

$$P(\beta_{\text{fin}_2}|\mathbf{x}) \propto P(\mathbf{x}|\beta_{\text{fin}_2})P(\beta_{\text{fin}_2}) \tag{13}$$

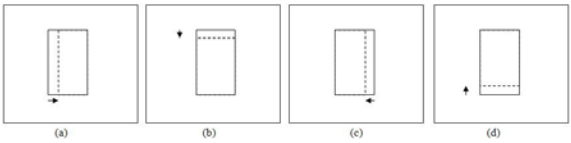

Fig. 4. New shrinking patches pattern (a) Left side shrinkage (b) Upper side shrinkage (c) Right side shrinkage (d) Lower side shrinkage

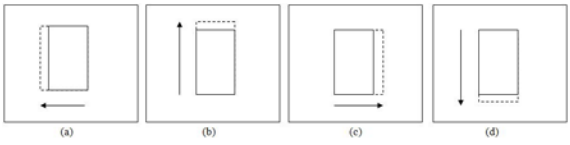

Fig. 5. New expanding patches pattern (a) Left side expansion (b) Upper side expansion (c) Right side expansion (d) Lower side expansion

The ϵ value determines what type of histogram is built. If $\epsilon = 0$, a 1-dimensional hue histogram is used while for $\epsilon = 1$, a 3-dimensional RGB histogram is applied. Before any comparison is performed, the histogram size needs first to be normalized. Let S_1 and S_2 denote the number of pixels in the patches in the first and second frames respectively. Each histogram bin value, H is adjusted by the ratio of sizes of S_2 and S_1, $H^{\text{new}} = \left(\frac{S_2}{S_1}\right) H^{\text{old}}$. Once again, the histogram relationship between the first and second frame patches are modelled by a Poisson distribution.

$$P(\mathbf{x}|\beta_{\text{fin}_2}) = \begin{cases} P(n_{(i)}, m_{(i)}) \text{ if } \epsilon = 0 \\ P(n_{(i,j,k)}, m_{(i,j,k)}) \text{ if } \epsilon = 1 \end{cases} \tag{14}$$

Two sets of prior probabilities are used. These are dependent on whether the size of the detected object inclines towards expansion ($P(\beta_{\text{fin}_2}^{\text{big}_p})$) or shrinkage ($P(\beta_{\text{fin}_2}^{\text{small}_p})$) where p^{th} denotes the patch under consideration. The selection of a suitable prior is very important as the shrinkage likelihoods are usually large even when the object expands. Thus we apply lower prior probabilities to shrinkage candidates if the size is increasing. In order to determine which set of the priors to be used, again a Neyman-Pearson hypothesis test is implemented where $\mathcal{H}_0$ and $\mathcal{H}_1$ represent the expansion and shrinkage hypotheses. Only eight candidate patches are used (4 shrinkage patches + 4 expansion patches) for this test where the same Poisson distribution is used as in equation 14. The maximum probability among the expansion patches represents the $\mathcal{H}_0$ probability while the maximum probability among the shrinkage patches represents the $\mathcal{H}_1$ probability.

$$P(\mathbf{x}; \mathcal{H}_0) = \max_{\forall \text{big}} P(\mathbf{x}|\beta_{\text{fin}_2}) \tag{15}$$

$$P(\mathbf{x}; \mathcal{H}_1) = \max_{\forall \text{small}} P(\mathbf{x}|\beta_{\text{fin}_2}) \tag{16}$$

Let λ_2 be the threshold for the Neyman-Pearson test.

$$\text{NP}_2 = \frac{P(\mathbf{x};\mathcal{H}_1)}{P(\mathbf{x};\mathcal{H}_0)} > \lambda_2 \tag{17}$$

$$P(\beta_{\text{fin}_2}) = \begin{cases} P(\beta_{\text{fin}_2}^{\text{big}}) \text{ if } \mathcal{H}_0 \text{ is true} \\ P(\beta_{\text{fin}_2}^{\text{small}}) \text{ if } \mathcal{H}_1 \text{ is true} \end{cases} \tag{18}$$

After the prior is obtained, the posterior probabilty $P_{i^{th}}(\beta_{\text{fin}_2}|\mathbf{x})$ of each of the nine patches is calculated. Each side of the bounding box can be expand or shrink independently based on the L_3 decision rule. Each side size is altered depending on whether the new posteriors exceed the original size posterior. $L_3 = 1$ indicates that the size is updated while $L_3 = 0$ indicates that the size change remain constant and i^{th} takes value from 1 to 8.

$$L_3 = \begin{cases} 0 \text{ if } P_{i^{th}}(\beta_{\text{fin}_2}|\mathbf{x}) \leq P_{0^{th}}(\beta_{\text{fin}_2}|\mathbf{x}) \\ 1 \text{ if } P_{i^{th}}(\beta_{\text{fin}_2}|\mathbf{x}) > P_{0^{th}}(\beta_{\text{fin}_2}|\mathbf{x}) \end{cases} \tag{19}$$

Figure 6 shows two examples of how the size of the object is updated. The iteration is terminated if no size change is detected.

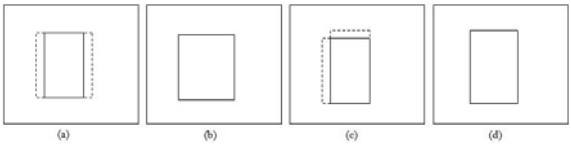

Fig. 6. Example of patch expansion (a)(c) Original patch (b) Result if the right and left side expansion are true (d) Result if the left and upper side expansion are true

3 Results and Discussion

SPBO has been tested on several video sequences that contain moving objects in various frame sizes. SIFT and SURF are chosen as the benchmarks for performance comparison. The implementation of SURF algorithm is based on OpenSURF by Evans [12] while SIFT is applied based on the original Lowe algorithm [**?**]. The performance is measured by calculating the Euclidean distance, $\mathcal{D}$ between the centroid (Ω_{sim}) of the simulation result and the manually determined ground truth centroid (Ω_{truth}) of the detected object.

The 4 corners of SPBO bounding box are used as the reference points for the centroid calculation. While the centroid for SIFT and SURF is generated by constructing a bounding box which uses the extreme points in 4 directions as shown in Figure 7. Then, the generated bounding box corners are used for the centroid calculation. Table 1 shows the distance error analysis among the methods. SPBO performance is the best with 43.3% of the detected object centroid have less than 10 pixels distance from the

Table 1. Comparison of the centroid distance among SPBO, SIFT and SURF

Method	Distance error (pixel)										
	0-9	10-19	20-29	30-39	40-49	50-59	60-69	70-79	80-89	90-99	>99
SPBO	52	30	19	13	3	0	1	0	0	0	2
SIFT	40	19	9	3	6	8	2	0	2	2	30
SURF	30	15	8	6	5	3	3	1	0	2	47

ground truth centroid. SIFT based recognition manages 33.3% detection with less than 10 pixels distance error while the worst is SURF with just 25% detection. The distance error relative to the patch size is given by this equation, $err = \frac{\mathcal{D}}{\max(P_w^{\text{new}}, P_h^{\text{new}})} \times 100\%$ where SPBO error rate is just 7% while SIFT is 27.1% and SURF is 42.6%. Figure 8 shows some of the results of applying SPBO in various scenes and situations. The images in 8(a) to 8(i) show the results of applying each method to 3 sets of video sequences. All methods obtain good recognition for the first set of video sequences [8(a)-8(c)]. For the second set of images [8(d)-8(f)], only SPBO recognizes the person under an illumination change. While, SIFT gives wrong matching and SURF provides no matching at all. In the third set, only SPBO 8(d) is able to recognize the object under blurring noise because of the fast movement of the ball. On the other hand, both SIFT and SURF did not detect any matched points. Examples of successful implementation of SPBO in various scenes and situations are given in the images 8(j) to 8(o) which include the problems of size change, blurring effect, homogenous texture, deformed object and illumination change.

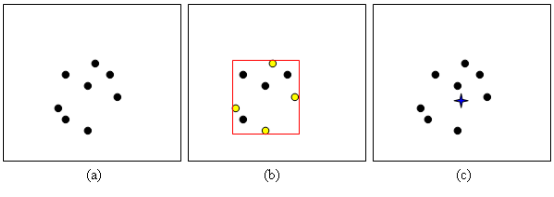

Fig. 7. Generating centroid for the SIFT and SURF algorithms (a) Matched points of interest (b) Constructing a bounding box (c) Centroid location is indicated by the blue star

4 Conclusion

In conclusion, we have shown that SPBO works well for recognizing the objects through video sequences. The problems of image sharpness, moderate deformation, illumination change, blurring, small size change and homogenous texture are solved by fusing both feature and template based approaches. The feature detector is obtained to generate the possible bounding boxes while the matching is done by taking a collection of pixels instead of a single point. This method is suitable for application in a system that requires object recognition in complex scenes.

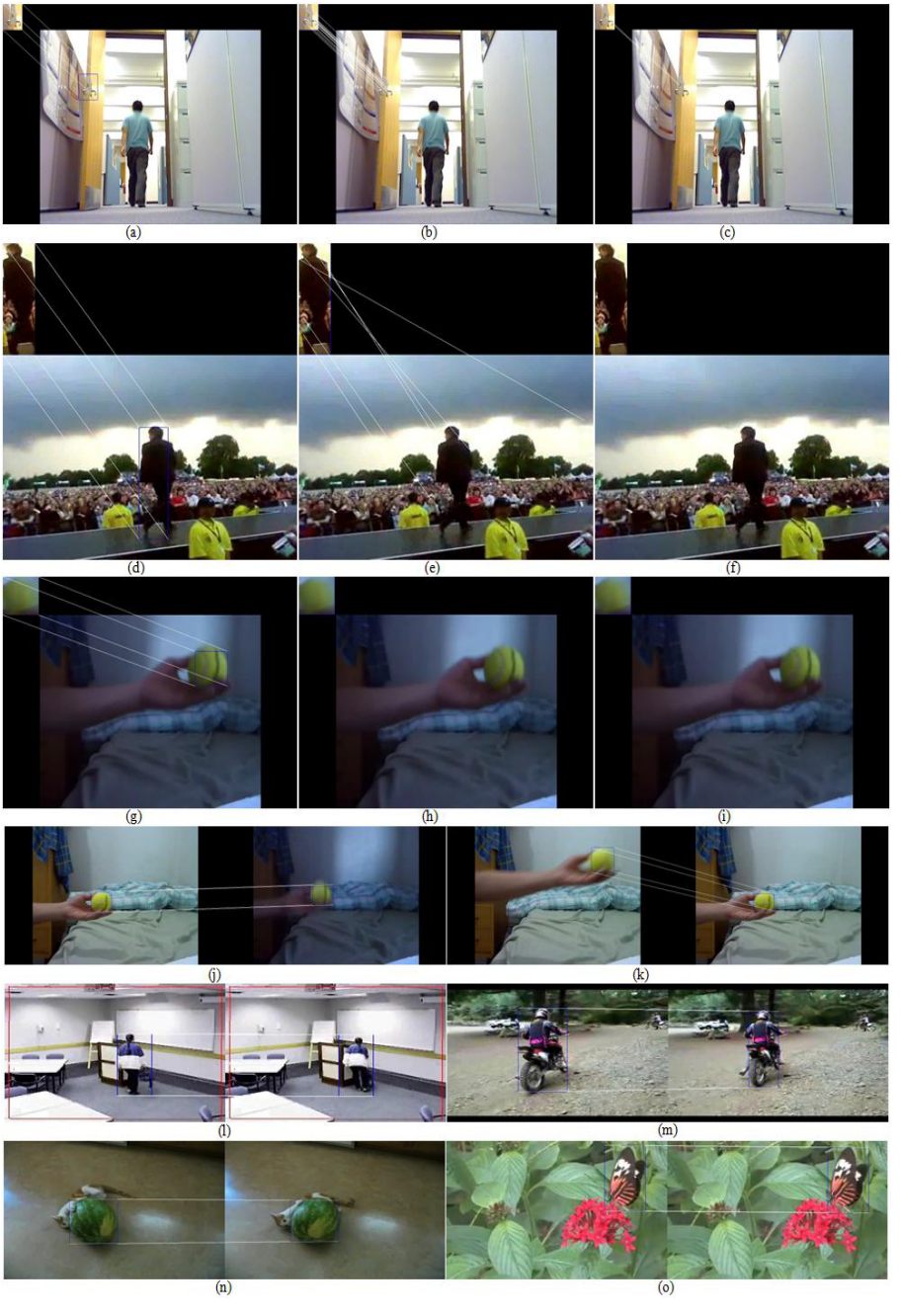

Fig. 8. Results of applying SPBO, (a)-(f): Detection comparison among SPBO [a][d], SIFT [b][e] and SURF [c][f], (g)-(n): Samples of SPBO result (left image is the 1^{st} frame, right image is the 2^{nd} frame)

References

1. Canny, J.: A computational approach to edge detection. IEEE Transactions on Pattern Analysis and Machine Intelligence 6, 679–698 (1986)
2. Duda, R.O., Hart, P.E.: In: Sobel, I., Feldman, G. (eds.) A 3x3 Isotropic Gradient Operator for Image Processing, pp. 271–272 (1973)
3. Roberts, L.G.: Machine perception of three-dimensional solids. PhD thesis, Dept. of Electrical Engineering, Massachusetts Institute of Technology (1963)
4. Harris, C., Stephens, M.: A combined corner and edge detector. In: Proceedings of the 4th Alvey Vision Conference, pp. 147–151 (1988)
5. Shi, J., Tomasi, C.: Good features to track. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 593–600 (1994)
6. Ke, Y., Suthankar, R.: Pca-sift: a more distinctive representation for local image descriptors. In: IEEE Computer Society Conference Computer Vision and Pattern Recognition, vol. 2, pp. 506–513 (2004)
7. Burghouts, G.J., Geusebroek, J.M.: Performance evaluation of local colour invariants. Computer Vision and Image Understanding 113, 48–62 (2008)
8. Geusebroek, J.M., Smeulders, A.W.M., Boomgaarda, R.V.D.: Color invariance. IEEE Transactions on Pattern Analysis and Machine Intelligence 23, 1338–1350 (2001)
9. Mikolajczyk, K., Schmid, C.: A performance evaluation of local descriptors. IEEE Transactions on Pattern Analysis and Machine Intelligence 27, 1615–1630 (2005)
10. Bay, H., Ess, A., Tuytelaars, T., Van Gool, L.: Speeded-up robust features (surf). Computer Vision and Image Understanding 110, 346–359 (2008)
11. Kay, S.M.: Fundamentals of Statistical Signal Processing, Detection Theory, vol. 2. Prentice Hall, Englewood Cliffs (1998)
12. Evans, C.: Notes on the opensurf library. Technical Report CSTR-09-001, University of Bristol (January 2009)