Online Object Tracking Based on Depth Image with Sparse Coding

Shan-Chun Shen[1], Wei-Long Zheng[1], and Bao-Liang Lu[1,2,⋆]

[1] Center for Brain-Like Computing and Machine Intelligence,
Department of Computer Science and Engineering
Shanghai Jiao Tong Unviersity, Shanghai 200240 China
[2] Key Laboratory of Shanghai Education Commission for
Intelligent Interaction and Cognitive Engineering
Shanghai Jiao Tong University, Shanghai 200240 China
bllu@sjtu.edu.cn

Abstract. Online object tracking is a challenging problem because of changing environment including diverse illumination and occlusion conditions. The emergence of commercial real-time depth cameras like Kinect make online RGBD-based object tracking algorithm become a focus of research. In this paper, we propose a robust online depth image-based object tracking method with sparse coding. We introduce sigmoid normalization for local depth patch. In order to recovery from tracking failure in condition of heavily occlusion. we present a detection module based on PCA bases. Experiments show that our method exceeds original color image-based method in case of environment changes.

Keywords: object tracking, depth image, sparse coding, normalization.

1 Introduction

Object tracking is one of the key problems in computer vision and it has broad practical scenarios such as activity recognition, motion analysis and image compression. Although the performance of object tracking algorithm has been much improved recently, it's still a great challenge to develop a robust object tracking algorithm considering some problems caused by illumination varying and target object occlusions.

The object tracking algorithm generally consists of three basic modules [1]: 1) object shape representation; 2) image features that hold the characteristic of target object; 3) strategies for detection the objects in a scene. The availability of high quality and inexpensive video cameras has improved the development of a great amount of object tracking methods based on color image features. In this paper, we propose a robust object tracking method based on depth image. Hence, we only discuss key issues related to image type.

[⋆] Corresponding author.

C.K. Loo et al. (Eds.): ICONIP 2014, Part III, LNCS 8836, pp. 234–241, 2014.

Generally speaking, there are four types of common visual features extracted from color image including color, edges, optical flow and texture. Numerous algorithms based on these features performance well in some constrained situation. Paschos proposed a color based object tracking solution in RGB color space [2], but color features are easily influenced by illumination. Object boundaries often located where image intensities strongly change. The new variational framework for detecting and tracking multiple moving objects is a very popular edge detection approach [3], it uses a statistical framework based on a mixed model. It is robust to illumination change but when occlusions occur the edge based method would lose target.

Color camera can real-time collect color image stream at the cost of losing information by projection 3D to 2D. As a result, color image based features would easily crash with changes of illumination. A new device Kinect can real-time acquire both color and depth image stream. A face tracking method integrated color and depth image stream is implemented with ASM model and statistical methods [4].

An online depth image based face tracking method is proposed on the assumption that face shape is an ellipse in [5]. However when occlusion occurs, the tracking method will lose target.

In this paper, we propose a general object tracking method based on single depth image, which is robust to occlusion and illumination changes. Compared with color imaged methods, our algorithm is less influenced by illumination change. With sparse coding representation, we can keep tracking the target object until the occlusion area reaches 50% of the target object.

The rest of this paper is organized as follows. In Section 2, we review the theory of sparse coding in object tracking; in Section 3, we introduce our tracking method; in Section 4, we present qualitative and quantitative results of our tracker on a number of challenging image sequences. Finally we conclude the paper in Section 5.

2 Sparse Coding

Sparse coding is a popular solution to object tracking problems recently. We can simply classified it into three forms: 1) appearance modeling based on sparse coding (AMSC); 2) target searching based on sparse coding(TSSR); 3) combination of both. Jia and colleagues proposed a structural local sparse coding model [6]. Mei and Ling solved most challenges like occlusion through a set of positive and negative trivial templates [7]. By transferring object tracking problem to a sparse approximation problem, they proposed a robust algorithm. Wang and colleagues proposed a new method that views coefficients of trivial templates as a single factor of tracking performance [8]. Studies mentioned above have proved that sparse coding is a good solution to color image based object tracking. In this paper we apply sparse coding to depth image based tracking algorithm. As shown in Fig. 1, each target is represented by some bases and templates with sparse coding.

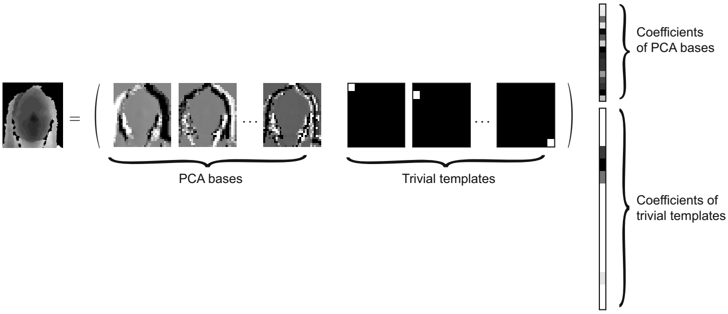

Fig. 1. sparse coding

In sparse coding framework, tracking problem is casted to finding the most likely patch among candidates by

$$y = Uz + e = [A\ I]\begin{bmatrix} z \\ e \end{bmatrix} = Bc \tag{1}$$

where y indicates the object vector, U denotes templates matrix, z represents coefficients of bases vectors and e is the coefficients of trivial templates. As is shown in Fig 1 , we assume that target object is sparsely represented by bases and trivial templates. We solve Eq. (1) via ℓ_1 minimization:

$$\min_{z,e} \frac{1}{2} \parallel y - Uz - e \parallel_2^2 + \lambda \parallel e \parallel_1 \tag{2}$$

where $\parallel \cdot \parallel_2^2$ and $\parallel \cdot \parallel_1$ are the ℓ_2 and ℓ_1 normal forms, respecitively. Several works have been done on online subspace learning by learning and updating bases represented by A such as PCA and ICA. With an iteration algorithm, optimal z and e for each candidate are computed.

After getting the optimal z and e for each candidate, the object tracking problem is transferred to a statistical inference problem.

3 Tracking Algorithm

In this paper, we applied sparse coding to depth imaged object tracking. To some degree, depth image is the same as color image except for the meaning of each pixel. In color image, pixels represent the color of this point while in depth image they represent the distance from the point to camera. Aiming to reduce the influence of illumination change, we try to develop depth image based tracking methods, and in order to solve the occlusion problem we incorporate sparse coding. So we should design a new algorithm to adapt to depth image. The workflow of our algorithm is described in Fig. 2.

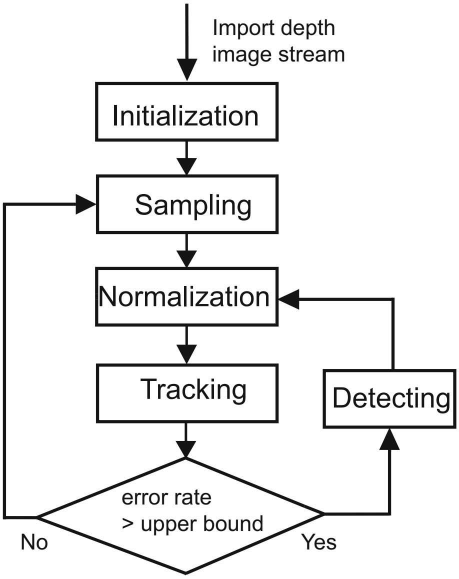

Fig. 2. Workflow of depth-image-based tracking algorithm with sparse coding

We adopt sparse coding method to tracking object. Firstly, we initialize the tracking by manually calibrating target position, computing PCA bases and setting other parameters such as patch size and bases number. Secondly, we sample in original depth image according to sampling parameters. The samples are size-adjustable to suit for demand of object front-back moving. To speed up the proposed tracking algorithm, we transfer all the samples to the same size patches. Then we consider the object tracking as a Bayesian task. By evaluating every patch, we find the patch with the highest posterior probability and return its location as the target. During the process, we compute the occlusion rate by coefficients of trivial templates. If the occlusion rate exceeds the upper bound, we discard the result and regard it as losing target. Then we startup the detection module. If not, we update the bases and go to next loop.

3.1 Alternative Box Sampling

Candidates are patches with size of 32*32, which is the result of trade-off between algorithm efficiency and accuracy rate. But it doesn't mean every patch is exactly a copy of a 32*32 patch in original depth image. According to the perspective relation, nearby object looks larger than distant one with the same size. So during the tracking process, the tracking sampling alternative boxes should be adjustable.

In detail, there are 5 parameters in tracking sampling stage: x and y denote transformation in plane, α and β are scale variation, and θ is angle rotation. Alternative boxes are uniformly distributed around the target. To adapt to the characteristics of the depth map, we set the α and β a little bigger. But too big α

and β mean more alternative boxes to be computed and slower processing speed. To speed up the tracking algorithm, we transfer all the samples to the same size by interpolation. In sampling stage, we don't concern the size of alternative boxes with regard to specific depth. This problem will be solved in the next section.

3.2 Depth Image Normalization

The meaning of pixels in depth image is the distance between camera and the point on object. The whole depth image represents the shape of the target. Transformation in the same depth can remain both pixel values and pattern. But once target moves front and back, the pattern is remained but the pixel values will shift .

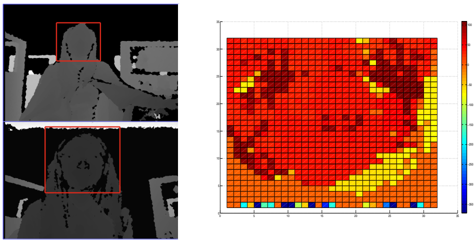

Fig. 3. Pixel value shifts of two frames. Left top is the face far from the camera. Left down is the nearer one. Right is the pixel value of two patches after interpolating.

As shown in Fig. 3, we should normalize the patch to eliminate the offset. The most common method of normalization is min-max normalization. If there are noises, they often deviate from average and become peaks of the image and finally their deviation results in extreme minimum or maximum. The existing of noise limits the performance of min-max normalization. So we adopt the sigmoid filter to normalize the patches [9]. The sigmoid function is a S-type function :

$$y = f(x) = \frac{1}{1 + e^{\frac{x-\beta}{\alpha}}} \tag{3}$$

In our method, α equals 1 and β is set to this median of each patch. β is set to the value because the median of a patch is not sensitive to noise. And a little peaks would not change the median much.

3.3 Restarting by Detecting

In this paper we solve the problem of occlusion by sparse coding. We estimate the occlusion by η which is the ratio of non-zero pixels and the number of occlusion map pixels. η is put forward to deal with partial model updating problem [8]. In our paper, we apply η to restart the tracking model when tracking failure occurs.

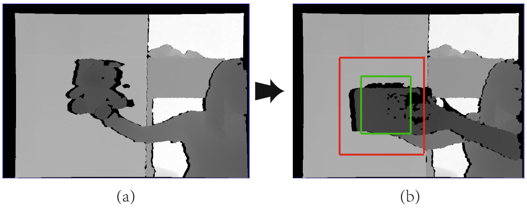

Fig. 4. Detecting sample method. (a) Image without occlusion and target is a bear; (b) Green box is the tracking sample range while the larger red one is for detecting sample range.

We set an upper bound and lower bound for η. Different values correspond to different tracking results. If η is larger than the upper bound, we view this situation as tracking failure, and we restart the tracking module. Since we are updating the bases of target, we don't adopt other detecting method. Instead, our detecting method is based on the recorded bases. Once η becomes larger than the upper bound, we startup the detecting module.

Our detecting method has similar idea as tracking method. Their difference lies on the sampling stage as shown in Fig. 4. On the assumption that when losing the target we still can find it in a wider scope centered on the original position, the sampling scope spreads to three times of the size of the original one. After sampling stage, the rest stages are the same as tracking method. By computing coefficients of bases and solving Bayesian task, we find the most likely patch among candidates. We compute error ratio η in detecting module. If η becomes lower than its upper bound we start the tracking module.

4 Experiment and Result

Our method is implemented in MATLAB on a Triple-Core Processor 2.10GHz with 6GB memory. The speed of our algorithm is related to sampling number. More sampling candidate boxes would slow down the processing speed. As a trade-off between computational efficiency and effectiveness, sampling number is set to 600. Our method is mainly compared with the original algorithm on the RGB image. We use 5 image sequences of a public dataset Princeton Tracking Benchmark [10] to test our algorithm as shown in Fig. 5. The challenges of each sequence and results are listed in Table 1. Our results are evaluated by average center error of pixels.

As shown in Table 1, the original color image based object tracking method and the depth image based object tracking method provide different performances in different cases.

- In cup sequences, challenge in this sequence is moving back and front. Their average center error of pixels are around 13. It means they both track the target closely.

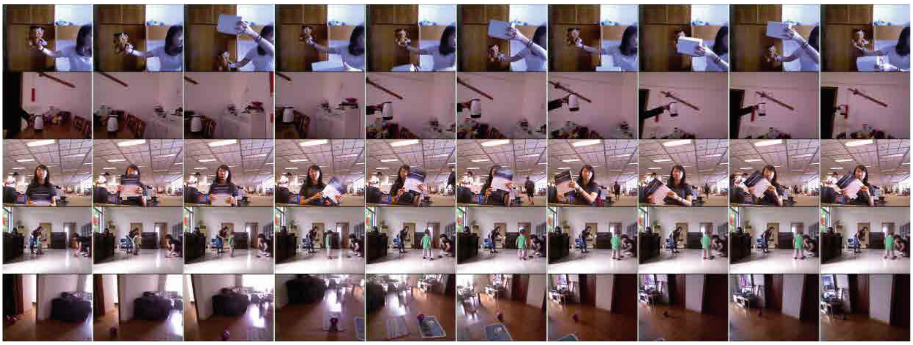

Fig. 5. Image sequences of bear, cup, face, child and ball are listed from top to bottom (only RGB images listed)

Table 1. Results of experiments

test sequence	frame number	challenge	color image error	depth image error
cup	368	move back and front	13.93	**12.83**
face	330	occlusion	**15.30**	17.52
ball	117	illumination change	263.60	**14.49**
bear	281	heavily occlusion	192.99	**46.23**
child	164	no-rigid	**47.57**	135.34
average			106.67	**45.282**

- In face sequences, challenge is occlusion. A book may occlude most part of target. From Table 1 we can find that they provide good performance in face sequences. Because both methods are based on sparse coding.
- In ball sequence, we can find that the illumination changes when the ball rolls around. In color image sequence the method loses target in the fortieth frame as the ball rolls to another brighter room. While in depth image sequence, our method keeps tracking the ball through out the whole sequence.
- In bear sequences, heavily occlusion is the main challenge when a book occludes the target bear for a while. Heavily occlusion leads to losing target in color image and without restarting module in the rest images it fail to find the target again. In our methods we add the detecting module to detect the losing target and keep tracking again.
- Finally, in child sequences, no-rigid target tracking is the main challenge. From Table 1, we can find that both methods performs bad with large error pixels numbers. It illustrates that both of them lose target. It is because the target child is not a rigid object and his movements result in changing of target's shape.

5 Conclusions and Future Work

This paper proposes a robust tracking method based on the depth image with a sparse coding representation. We improve the performance of the color image based object tracking method with sparse coding representation by applying sigmoid normalization algorithm and by designing the detecting module. The two modifications are designed to acquire stable performance when illumination changes or occlusion occurs.

But we still leave the no-rigid object tracking problem unsolved, because our tracking patches are decomposed into PCA bases with different weights. The tracking method of no-rigid object is limited by the characteristics of the PCA bases. As a consequence, we plan to improve our method by adopting other type sparse representation in the future.

Acknowledgments. This work was partially supported by the National Natural Science Foundation of China (Grant No. 61272248), the National Basic Research Program of China (Grant No. 2013CB329401) and the Science and Technology Commission of Shanghai Municipality (Grant No. 13511500200).

References

1. Yilmaz, A., Javed, O., Shah, M.: Object tracking: A survey. Acm computing surveys (CSUR) 38(4), 13 (2006)
2. Paschos, G.: Perceptually uniform color spaces for color texture analysis: an empirical evaluation. IEEE Transactions on Image Processing 10(6), 932–937 (2001)
3. Paragios, N., Deriche, R.: Geodesic active contours and level sets for the detection and tracking of moving objects. IEEE Transactions on Pattern Analysis and Machine Intelligence 22(3), 266–280 (2000)
4. Cai, Q., Gallup, D., Zhang, C., Zhang, Z.: 3D deformable face tracking with a commodity depth camera. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part III. LNCS, vol. 6313, pp. 229–242. Springer, Heidelberg (2010)
5. Cao, Y., Lu, B.-L.: Real-time head detection with kinect for driving fatigue detection. In: Lee, M., Hirose, A., Hou, Z.-G., Kil, R.M. (eds.) ICONIP 2013, Part III. LNCS, vol. 8228, pp. 600–607. Springer, Heidelberg (2013)
6. Jia, X., Lu, H., Yang, M.-H.: Visual tracking via adaptive structural local sparse appearance model. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1822–1829. IEEE (2012)
7. Mei, X., Ling, H.: Robust visual tracking and vehicle classification via sparse representation. IEEE Transactions on Pattern Analysis and Machine Intelligence 33(11), 2259–2272 (2011)
8. Wang, D., Lu, H., Yang, M.-H.: Online object tracking with sparse prototypes. IEEE Transactions on Image Processing 22(1), 314–325 (2013)
9. Pei, S.-C., Lin, C.-N.: Image normalization for pattern recognition. Image and Vision Computing 13(10), 711–723 (1995)
10. Song, S., Xiao, J.: Tracking revisited using rgbd camera: Unified benchmark and baselines. In: ICCV (2013)