CS512 Project Report:

Object Tracking Using Camshift With Weighted Back Projection

Chen Xu A20377739, Yizhi Hong A20386348

PROBLEM STATEMENT

Object tracking is a crucial task within the computer vision field. It has been widely applied in applications. In order to detect the object and promptly tracking in each frame, the selection of the algorithm will be the crucial part of the object tracking. The speed and the accuracy are the two main criteria for measuring the performance of target recognition algorithms.There are many different approaches for tracking an object, and mean shift algorithm is one among them aiming at real-time object tracking which is widely used in the object tracking area.CAMShift stands for Continuously Adaptive Mean Shift and it aiming at efficient head and face tracking in a perceptual user interface. It is one approach to realize real-time object tracking based on mean shift algorithm. The OpenCV library provides an implementation of CAMShift algorithm, which uses 1-D hue histogram and realizes adaptive scale and orientation for view-changing objects.

However, Since Meanshift or CAMShift establish the target model by the color histogram, the model only roughly estimates of the characteristics. If the background and target are similar, the target recognition might interfere the tracking. At this time, the tracking performance of the Meanshift and CAMShift tracking method will be poor. In the actual tracking process, the edge of the target has a high coefficient with the similarity of the background and the gradient of the image will be lower than what we expected. Thus, the CAMShift algorithm being affected.

In this project, the CAMShift algorithm apply the weighted back projection will be implemented, a back projection weighting strategy is proposed as a solution towards the above situations.

PROPOSED SOLUTION

The performance becomes unreliable while tracking multi-colored objects or if the background interferes the tracked object. In this project, in order to ignore the insignificant colors within the track window, we set a threshold to the RGB histogram to get rid of the interference of the colors which occupy only small parts of the target region but would affect the performance significantly if not eliminated. In our implementation, half of the maximum value

of the histogram is taken as a typical threshold for general cases. Also, whiter pixels correspond to higher possibility of being part of the tracked object. Here is the approach:

1. Select the tracking object.

User will capture the track window by click and drag the mouse. We will get the object window we want to check initially.

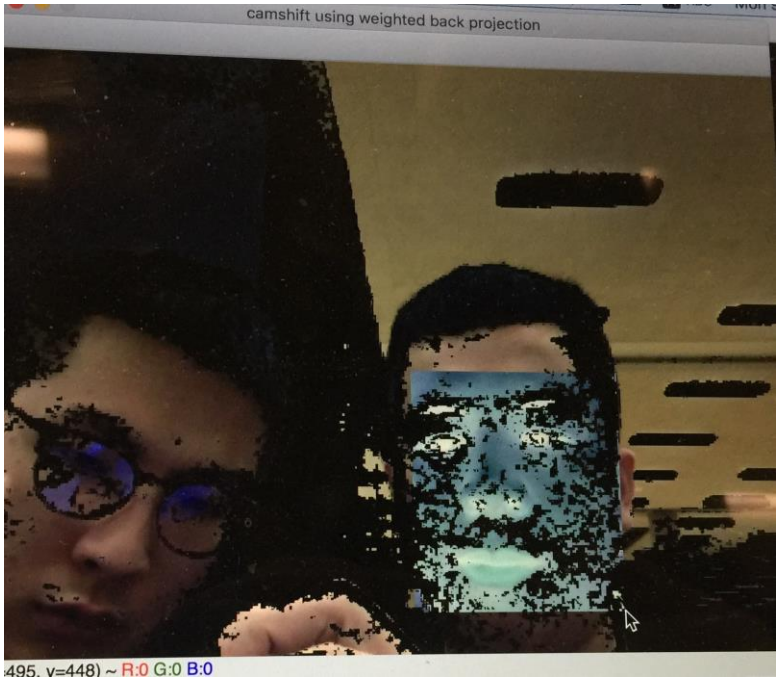


Figure 1. Shows how to click and drag a windows.

2. Calculate the weighted backprojection by histogram.

Back projection is a primitive operation that associates the probability of being part of the tracked object in the image of each pixel with the value of the calculated color histogram. First, the frame is converted to HSV color space. Second, color histogram of the target object is computed using single hue value in HSV color space of the frame. Let {xi}i=1...n denotes the pixels in the region of tracked object, and m is the number of quantized bins. The histogram is calculated as

$$q_u = \sum_{i=1}^{n} \delta[b(x_i) - u] \,, \quad \text{u} = 1 \ldots \text{m.}$$

Iteratively convert each frame from BGR to HSV color space in order to get the HSV image and mask. Apply the size of the window to capture object's HSV and mask.

A. Thresholded multi-dimensional histogram

Since The original backprojection is 1-D dimension. the hue channel of HSV color space alone is insufficient to distinguish general objects, multi-dimensional histogram is used in our proposed method. And RGB color space is chosen as the feature space. In our implementation, we use 3-D RGB histogram quantized into sized bins. We include Hue, Value, Saturation as our 3-D RGB histogram. For each one we do a back projection. In order to ignore the insignificant colors within the track window, we set a threshold to the histogram to get rid of the interference of the colors which occupy only small parts of the target region but would affect the performance significantly if not eliminated.

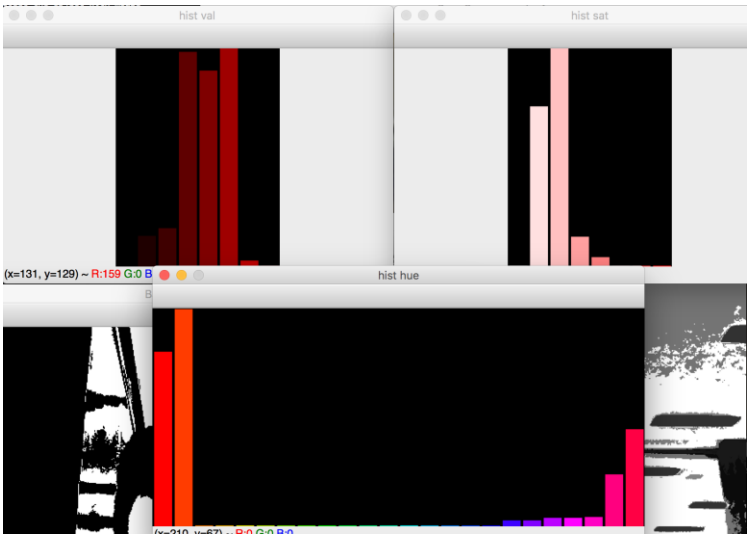


Figure 2. The 3D- histogram for Hue, Value, Saturation.

B. weighted back Projection

back projection image is calculated using the histogram calculated in the above step and the probability value is rescaled to be within the pixel value range. For example, for 8-bit hues, the range is between 0 and 255. The innermost frame shows the tracking window where the target object is supposed to be. For each backprojection, we weighted with threshold. And we weighed the hue weighted 60% and value 40%, and weighed the hue weighted 60% and saturation 40%. This will change weighting region, in which all pixels are weighted using an isotropic kernel. In our implementation, the Epanechnikov kernel as follows is used. The images show below the difference after applying the weight comparing to original Camshift:

Figure 3. Back projection image (left:original back projection, right:weighted back projection)

Obviously, the weighted back projection tends to emphasize the tracking object. Which will improve the Camshift.

3. Apply the new backprojection Image into Camshift algorithm.

After the new back projection image is calculated, mean shift procedure is applied to find the mass center of the current tracked region. And the calculated mass center will be used as the initial center of the next frame. The steps are described as follows algorithm:

First, initialize the window using the hand-selected location.

Second, calculate the mass center using the statistical moments of the search window. Let I(x, y) be the intensity of the back projection image at location (x, y), then the zeroth moment and first moments for x and y are calculated as

$$M_{00} = \sum_x \sum_y I(x, y),$$

$$M_{10} = \sum_x \sum_y x * I(x, y),$$

$$M_{01} = \sum_x \sum_y y * I(x, y).$$

Then the mass center location is calculated as

$$x_c = \frac{M_{10}}{M_{00}}, \quad y_c = \frac{M_{01}}{M_{00}}.$$

Third, move the window center to the calculated mass center.

And then repeat above steps until convergence.

Algorithm 1: meanshift

First, get the first frame and initialize the state.

Second, apply mean shift procedure to locate the tracked object within the frame and store the zeroth moment (it is used as the window size of the next frame) and the mass center location. At last, set the calculated result as the initial state of the next frame and repeat. In this way, the target object in video sequences can be tracked.
The CAMShift procedure flowchart is shown as Fig. 4.

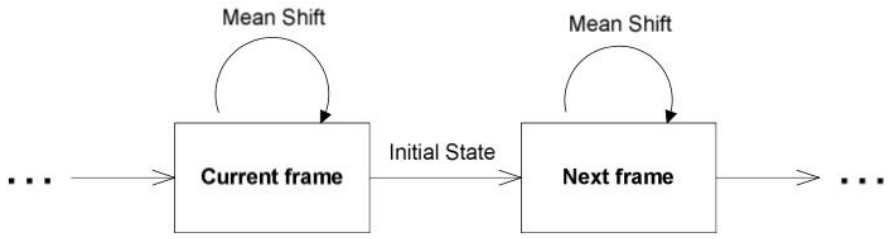


Figure 4: The flowchart of the camshift.

4. **Iteratively tracking object by the track windows.**

IMPLEMENTATION DETAILS

1. The process of computing the weighted back projection is computing intension, it may be more efficient if focus on tracking a specific part of video.
2. If there are too many light sources in the background, the performance of camshift algorithm will be unreliable.
3. Threshold is the key point of the weighted back projection, if the threshold is too big or too small ,it may not as good as regular CAMshift.
4. Since the hue channel of HSV color space alone is insufficient to distinguish general objects, multi-dimensional histogram is used in our proposed method. And RGB color space is chosen as the feature space. In our implementation,we use 3-D RGB histogram.
5. CAMShift is not a general-purpose tracker since it originally targets at head and face tracking, which is mostly uniformly colored. It is suitable for uniform-colored object tracking, but it may fail to track multi-colored object.
6. The high resolution of the video might fail to tracking since the computation are intensitive.

RESULTS AND DISCUSSION

RESULTS:

Test environment: Macbook Pro 13.3
Processor 2.7 GHz Intel Core i5 memory 8 GB 1867 MHz DDR3
Situation 1: Object tracking in similar content.

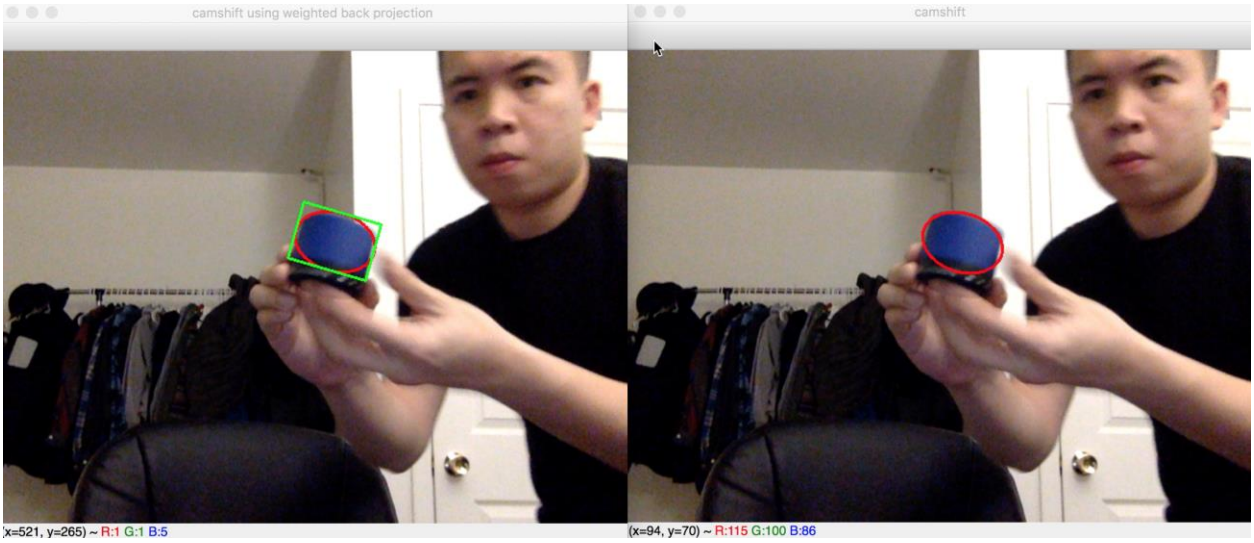


Figure 5. Dark color object moving in frame 37 (left: camshift using weighted back projection, right: camshift)

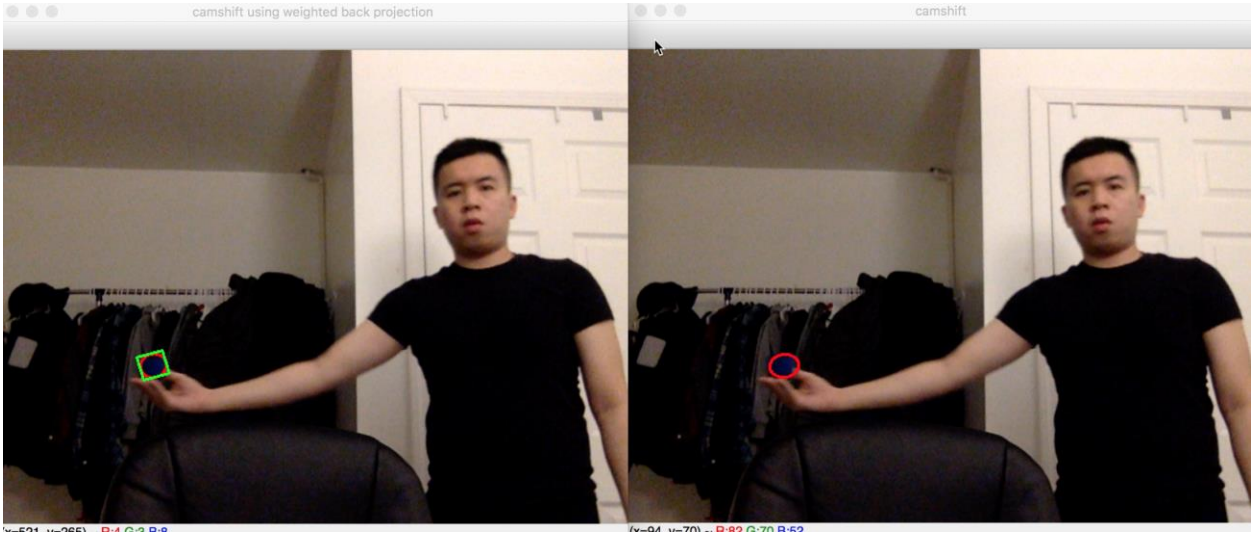
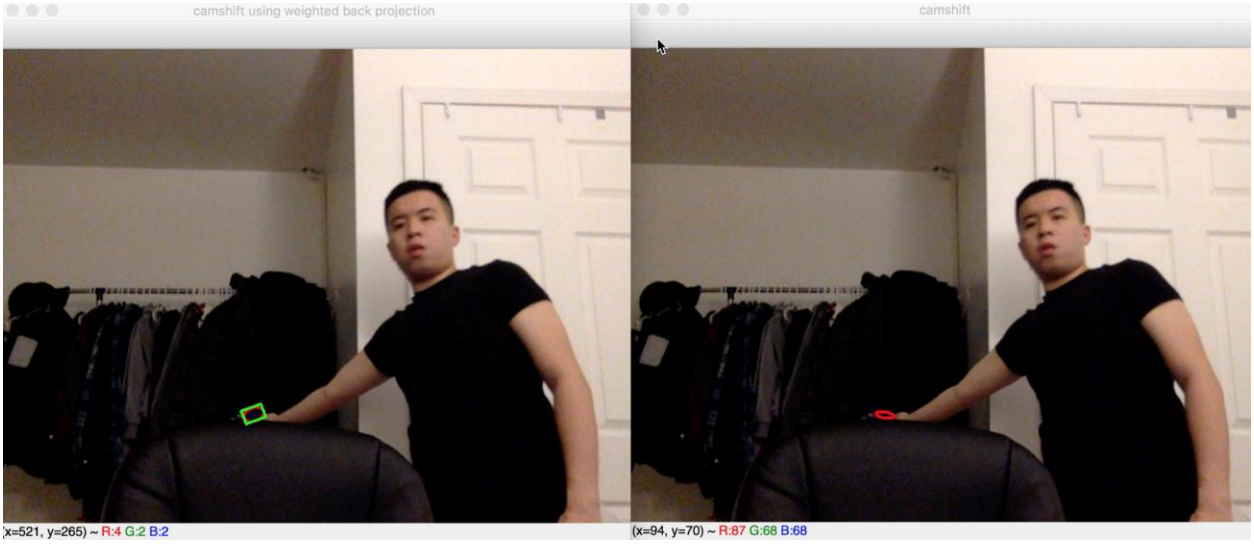


Figure 6. Dark color object moving in frame 168, In this frame, The original camshift and weighted camshift still works good in the dark background with similar content. (left: camshift using weighted back projection, right: camshift)



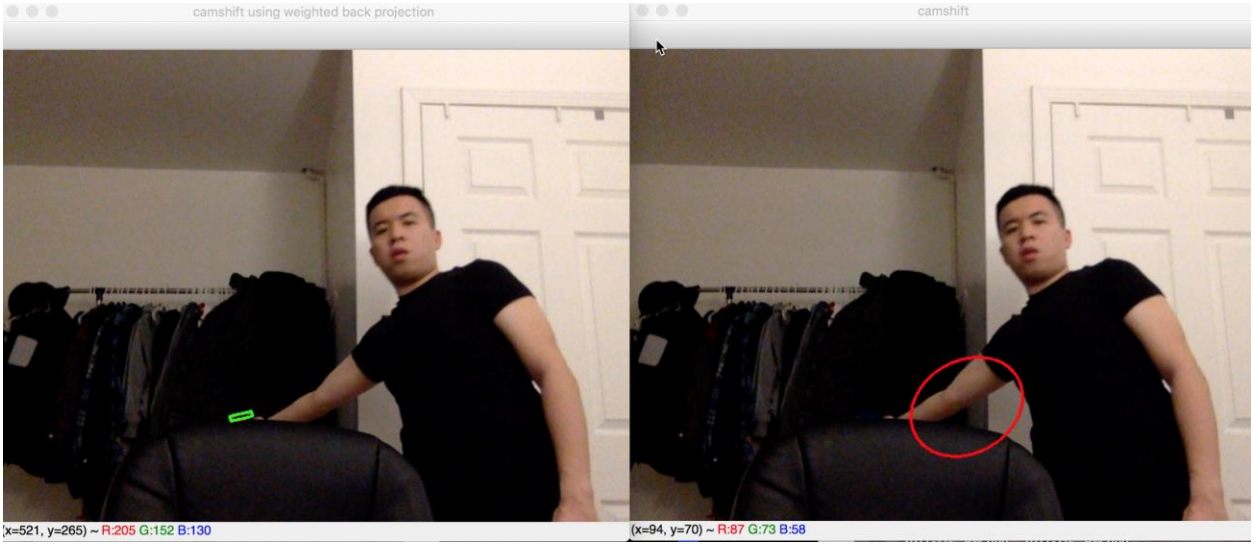


Figure 7. Dark color object moving in frames 182, 189. In those frames, the original camshift begin locate to wrong object, but the imported camshift still tracking the object even it is small and blend in the background. (left: camshift using weighted back projection, right: camshift)

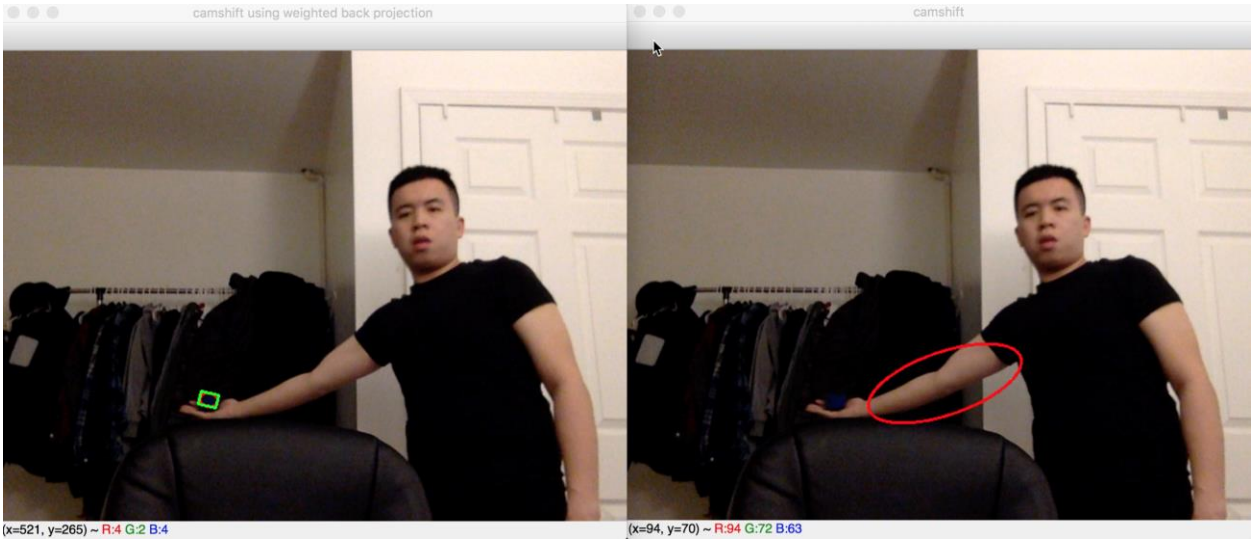


Figure 8. Dark color object moving in frames 206. In these frames, The original camshift lost the target, but the imported camshift will relocate and tracking the object normally. (left: camshift using weighted back projection, right: camshift)

Situation 2: Object tracking in different light source and changing environment.

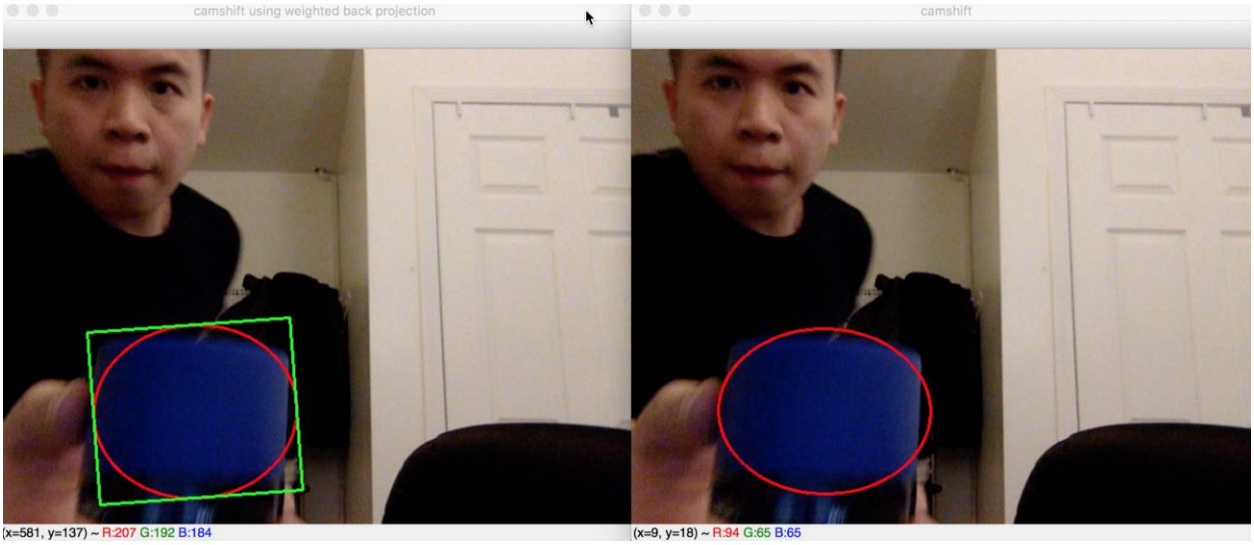


Figure 9. Initial tracking the object in both same frame 2. (left: camshift using weighted back projection, right: camshift)

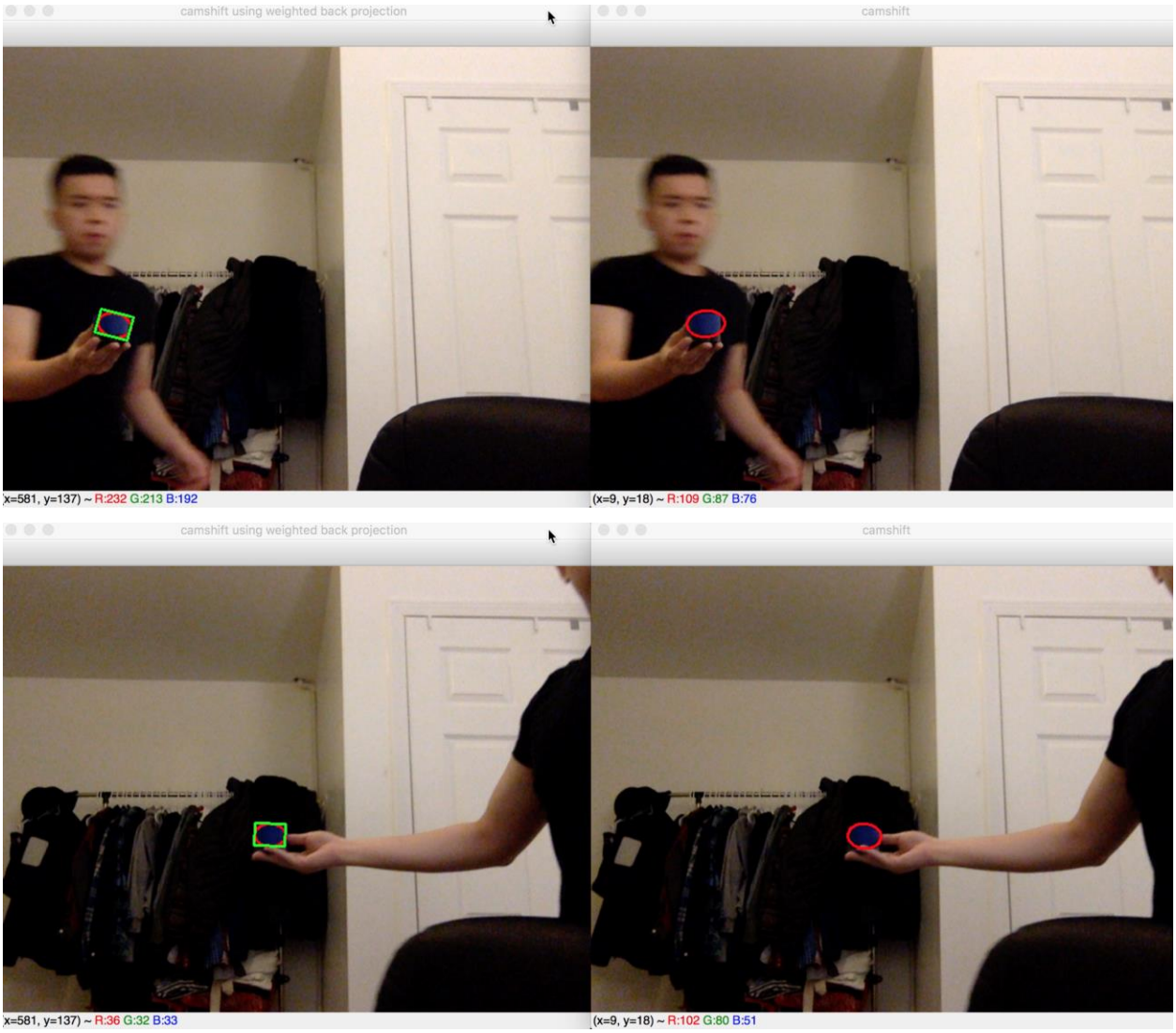


Figure 10. In frames 107, 167, The both camshift trackers keep tracking the object perfectly, the object is keep moving from frame to frame. (left: camshift using weighted back projection, right: camshift)

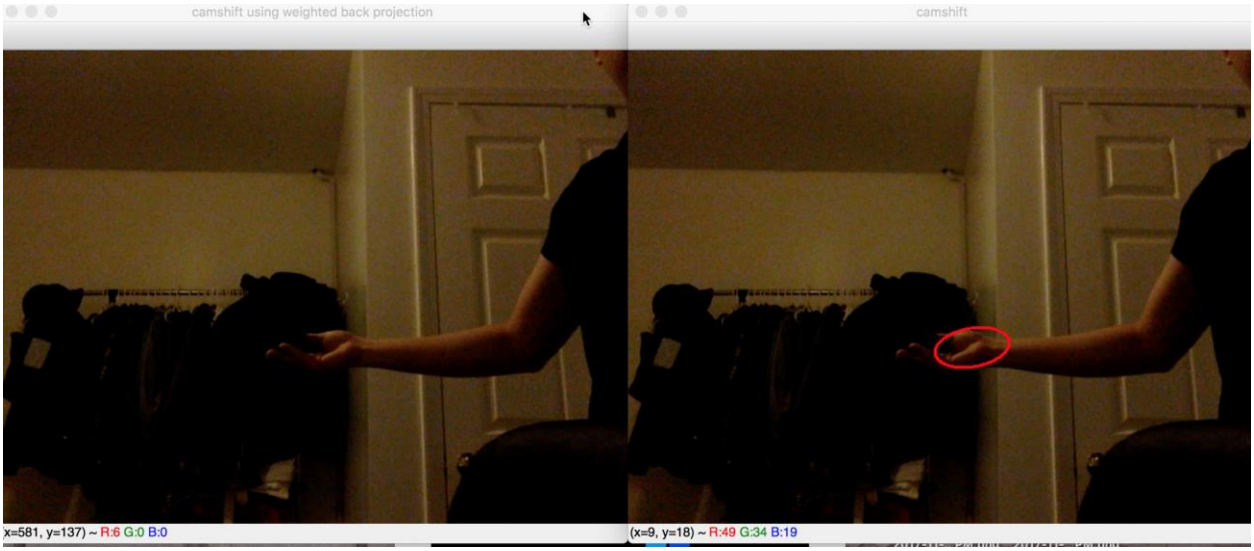
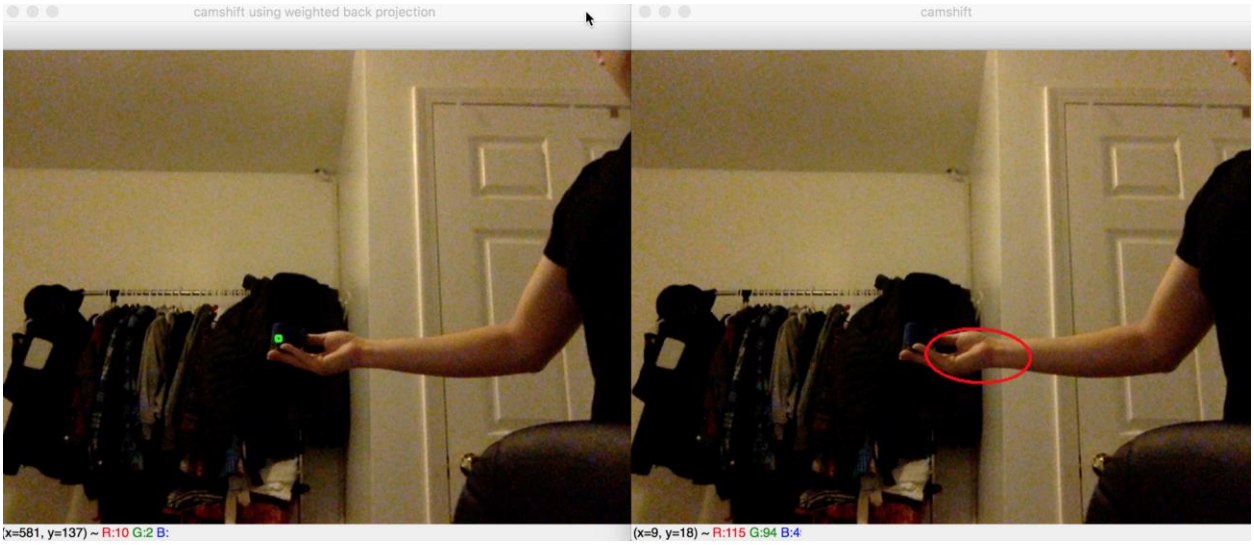


Figure 11. In frame 172. I close the light suddenly. And the imported camshift seen lost the target, the original camshift keep tracking but is not the object itself, is the object reflecting different light. In this case. Is my hand (left: camshift using weighted back projection, right: camshift)



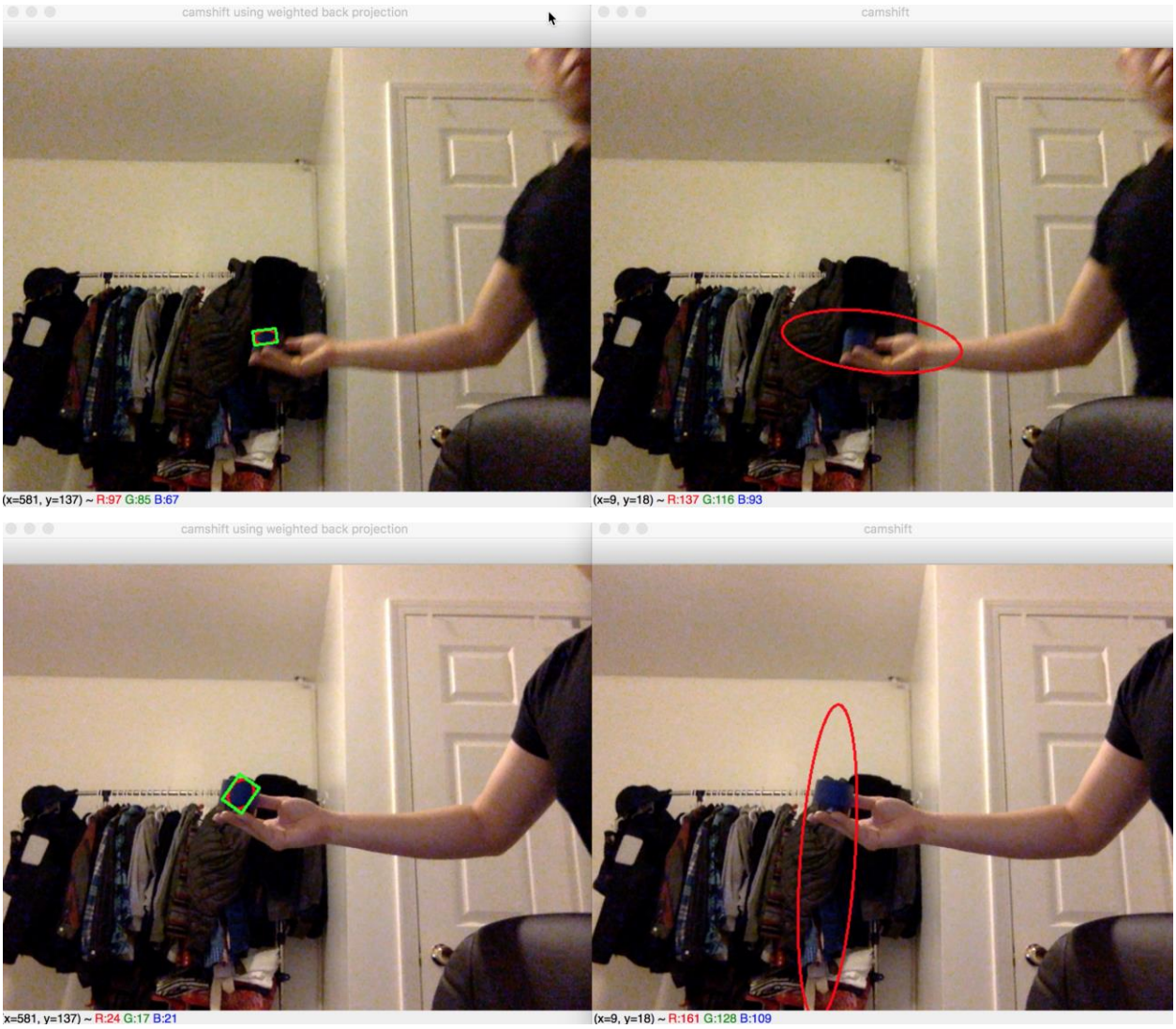


Figure 12. In frames 175, 192, 240. The light source back, the original camshift diverge after tracking wrong object. The camshift using weighted back projection back to tracking the object perfectly (left: camshift using weighted back projection, right: camshift)

Result Discussion:

As we can see, the camshift using weighted back projection outperform the original camshift in the slightly illumination change and similar color background. But It lose the target occasionally since the high intensitive computation. If object is too fast, it might be perform worst than the original camshift.

Conclusion

The CAMShift algorithm is simple and stable if the scene is not demanding (uniform-colored object & discriminating background). It can deal with slightly illumination and appearance change. But the performance becomes unreliable while tracking multi-colored objects or if the background interferes the tracked object. In this paper, we proposed a new method to improve the performance for this situation. Through experiments we can see that the proposed method improves the performance while keep the processing speed real-time.

REFERENCES

[1]Lei Sun, Bingrong Wang, Takeshi Ikenaga, "Real-time Non-rigid Object Tracking Using CAMShift with Weighted Back Projection", Graduate School of Information, Production and Systems Waseda University Kitakyushu-shi, Fukuoka-ken, Japan
[2]Abdul Basit, Matthew N. Dailey, Pudit Laksanacharoen and Jednipat Moonrinta, "Fast Target Redetection for CAMShift using Back-projection and Histogram Matching", Department of Computer Science and Information Management, Asian Institute of Technology, Klong Luang (12120), Pathumthani, Thailand
[3] Zheng Han, Rui Zhang, Linru Wen, Xiaoyi Xie, Zhijun Li, "Moving Object Tracking Method Based on Improved CAMShift Algorithm", School of Automation, Wuhan University of Technology, Wuhan 430070, China
[4] Richard Szeliski," Computer Vision:Algorithms and Applications", September 3, 2010 draft
[5]Improved Camshift by EfforiaKnight
https://github.com/EfforiaKnight/ImprovedCamshift
[6] Meanshift and CAMShift:
https://docs.opencv.org/trunk/db/df8/tutorial_py_meanshift.html
[7] Centroid method:
https://iliauk.com/2016/03/02/centroids-and-centres-numpy-r/