Object Tracking with OpenCV Template Matching

Changsoo Jung, Vishwajeet Bhosale, Amanda Cohen
Colorado State University

Introduction

We used six template matching methods from OpenCV - TM_CCOEFF, TM_CCOEFF_NORMED, TM_CCORR, TM_CCORR_NORMED, TM_SQDIFF, and TM_SQDIFF_NORMED [1]. These methods easily missed the targets and tracked wrong objects. To reduce the false positives, we set the thresholds; 0.5 for the minimum value and 0.9 for the maximum value. For our experiments, we used normalized versions of methods since thresholding can only adapt to normalized methods. We tested methods on pure frames and adjusted frames. Also, we tracked objects in several environments with various lights and sizes. We chose a candle cover, baseball, and handkerchief as objects for testing. The candle cover which has a similar color to the background was chosen to see how the methods track between similar colors. Baseball and handkerchief were chosen to check how robust the methods are on different rotation and light conditions.

Methods

To improve tracking, we adjusted template images and frames. After getting an object image, we colored the image with the dominant colors of the object. The dominant colors are calculated through OpenCV's K-Means clustering, and we set the K value to three, to get three dominant colors of the object [2]. Then, we added edges of the object to highlight the object's characteristics such as a pattern. We assumed adding edges will help to track in various light conditions. To get an object's edges, we used Canny algorithm on the blurred grayscale image and eliminated noise by thresholding. We made a template of the object by summing the dominantly colored image and then extracted edges. For template matching, we added edges calculated by the above method into each original frame.

Results

	TM_CCOEFF_NORMED	TM_CCORR_NORMED	TM_SQDIFF_NORMED
Basic Template Matching	3.95	4.08	4.06
Adjusted Template Matching	3.91	3.98	4.00

Table 1 | FPSs on Multiple Methods. The FPS value are averages of three trial per method.

TM_CCOEFF_NORMED and TM_SQDIFF_NORMED didn't exceed our threshold or generate false positives. Only TM_CCORR_NORMED satisfied our threshold and tracked correct targets on most of the frames. However, it also produced some false positives if the lighting condition or size of an object were changed. Although we used the algorithm to extract edges for every frame, the values of frames per second (FPS) were close to the basic template matching method's FPS values. In addition, the FPS value wasn't affected by methods since these three methods were computed using similar amounts of data [3].

To compare the performance between basic template matching and adjusted template matching, we tested objects with TM_CCORR_NORMED in various rotations, sizes and light. Tracking performed poorly on candle cover when size and lighting were varied due to its material, that reflected object color. The candle cover only performed in perfect condition, which is the same as its template, so it produced a lot of false positives in most frames even if we used adjusted template matching. The baseball was tracked well in basic template matching except light changing since the matching algorithm relies on the pixel comparison [3]. However, the adjusted template matching performed well in changing light and tracked well in most conditions. The reason is that the colored template with object's dominant colors is less sensitive to change in lighting than the pure template which is composed of various colors. Also, the object's edges help in tracking when the light conditions are changed. The handkerchief showed robustness of adjusted template matching on various light conditions since its edges include a clear pattern that helped to track the object. However, the adjusted template matching couldn't track the objects when edges were messed up due to downsizing.

Conclusion

Overall, TM_CCORR_NORMED performed well by satisfying our threshold, while other methods produced a lot of false positives. Our method showed stable tracking in various conditions through coloring the object's template with the dominant colors of the object. Also, by using the edges, our methods produced better performance than the basic template matching methods in different light conditions. In addition, our adjusting algorithm didn't reduce FPS, so we can increase the performance with simple computations.

1. https://docs.opencv.org/master/d4/dc6/tutorial_py_template_matching.html
2. https://docs.opencv.org/master/d1/d5c/tutorial_py_kmeans_opencv.html
3. https://docs.opencv.org/master/df/dfb/group__imgproc__object.html