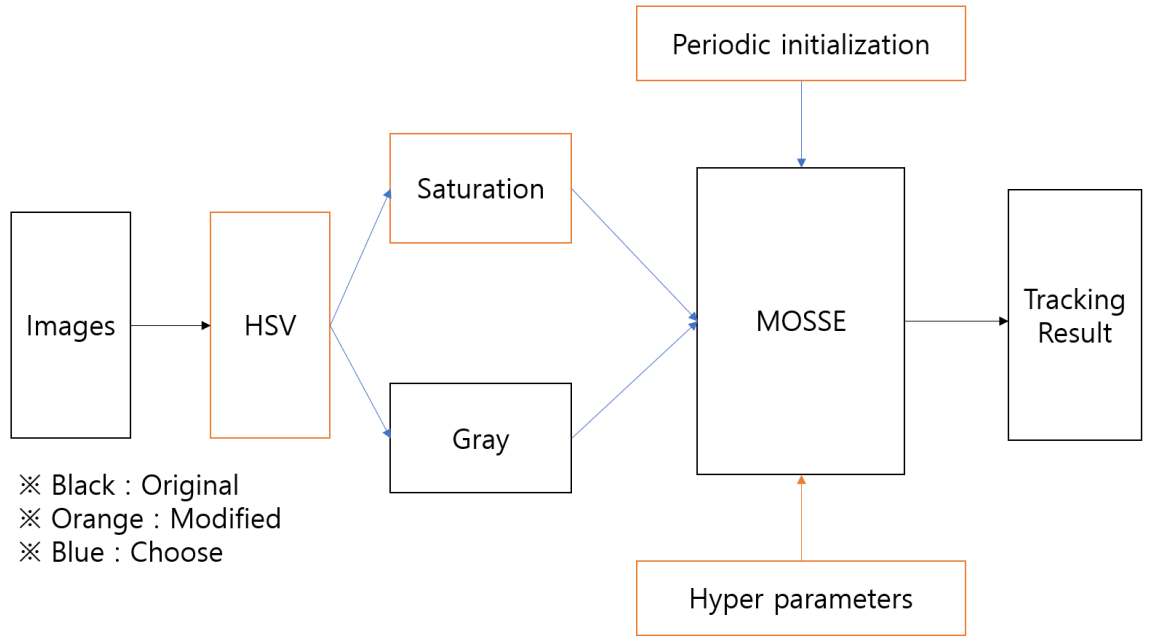
Title: Code Improvement with Basic Configuration : Image Tracking Using Correlation Filter

Author: 이민영(Minyoung Lee)

Teaser Figure (Overall concept, target problem, etc.)

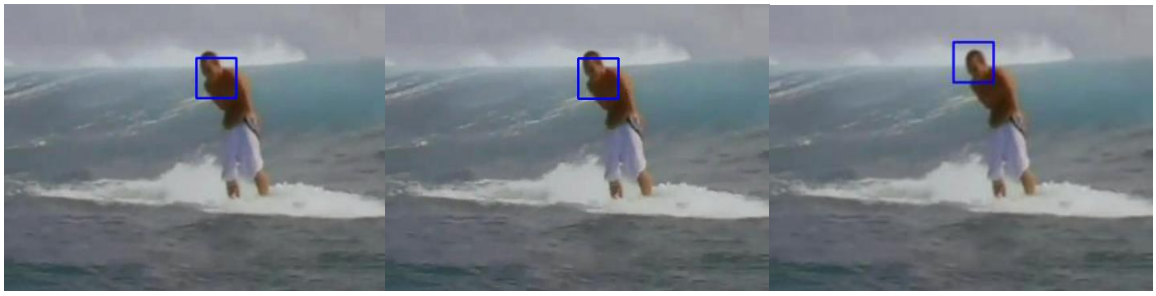
Framework Figure



Experiment Figure (qualitative results, ablation tests, analysis, etc.)

[250, 100, 120, 180] [220, 100, 160, 200]

Gray Scale HSV(Saturation) Filter Recentering



Gray Scale HSV(Saturation) Filter Recentering

*more results in the Result section

General description of your target problem

I chose the 'Image Processing 2 : Image Tracking' algorithm for the final project. This algorithm seeks to track a given target by using correlation filter.

The limitation of the target problem that you want to solve

The underlying algorithm reveals several limitations when executed to track the whole body or head of the surfer in the given image dataset. First, in the process of tracking the body, the output box pointing out the target subject gets drifts off the target, if the body is tilted or about half of the body is submerged. When tracking only a specific part of the body, the head for example, it was also found that the tracking target often moved to the body part, not the head. This was especially common in images where the scale of head changes. It is presumed to be a problem caused by the lack of a function to recognize a scale change in this code, but in this task, I tried to solve the limitations using existing functions as much as possible instead of adding new functions to the algorithm.

The solution you've found to solve the limitation

The first method attempted is the adjustment of the hyperparameters. The accuracy of the algorithm results is largely determined by how appropriate the parameters initially provided are. Therefore, to increase the accuracy of algorithmic results, I attempted to track the surfer better by fine-tuning the learning rate, and location or size of the ground box. Alternatively, I have adjusted the criteria for tracking in existing algorithms. The basic algorithm converts the given image into gray scale, a black-and-white information value, and tracks the target based on the filter data given at the starting step. However, in the case of a given image, there is a noticeable contrast of color between the background and the target human, and the background is relatively stable as target continues to move around. Therefore, I adjusted the given data to use the color data and applied recentering of the filter.

The detailed description of your framework

1. Adjustment of Hyperparameters

```
# alpha value
lr = 0.05 #0.125
# variance for the target Gaussian response
sigma = 100
# number of frames to be used for the pre-training
num_pretrain =  1
# rotation augmentation?
rotate = False #True
```

It is the learning rate that determines the speed at which learning proceeds in the tracking algorithm. If the learning rate is reduced, the learning speed decreases as the iteration to find the convergence point decreases, but the possibility of error, or divergence also falls. However, if the learning rate becomes too small, there may be no improvement in performance compared to the longer learning time, or it might converge to the local minimum. Therefore, the learning rate was adjusted from 0.125 to 0.05, the appropriate value found by various trail. The method of rotating a given image to diversify the learning data did not help tracking the given data. Adjusting from True to False rather showed a slight improvement in the accuracy.

Secondly, the ground box is a very important initial value when using the correlation filter for image tracking. Since this ground box determines the value of the initial tracking target filter, the setting of an appropriate ground box is an essential in improving learning and tracking accuracy. Therefore, after several attempts, the location and size of the ground box were adjusted from [250, 100, 120, 180] to [220, 100, 160, 200], which had the highest accuracy. The size of the box was set to be larger than the size of the body in the initial image because the accuracy was reduced when a part of the body escaped from the box due to body movement. (220, 100) is the starting point for the box,

and (160, 200) is the size of the box. As the value was set to the place the body at the center of the box as much as possible, the box was decided based on the tilted body with every body parts.

```
img_path = '.'
tracker = mosse(lr, sigma, num_pretrain, rotate, img_path)
tracker.start_tracking([260, 120, 50, 50])
```

In the case of tracking only the head, the most appropriate ground box was [260, 120, 50, 50].

2. Filter Criteria Adjustment

1) RGB to HSV

```
def start_tracking(self, init_gt):
    # get the image of the first frame...
    init_img = cv2.imread(self.frame_lists[0])

    init_frame = cv2.cvtColor(init_img, cv2.COLOR_BGR2HSV) #read as HSV scale image
    h2, init_frame, v2 = cv2.split(init_frame)
    #init_frame, s2, v2 = cv2.split(init_frame)

    init_frame = init_frame.astype(np.float32)
```

```
# start the tracking...
for idx in range(len(self.frame_lists)):
    current_frame = cv2.imread(self.frame_lists[idx])

    #frame_gray = cv2.cvtColor(current_frame, cv2.COLOR_BGR2GRAY)
    frame_gray = cv2.cvtColor(current_frame, cv2.COLOR_BGR2HSV)

    #frame_gray, s2, v2 = cv2.split(frame_gray)
    h2, frame_gray, v2 = cv2.split(frame_gray)
```

As briefly explained earlier, there is a clear contrast in the color between the blue sea and the relatively red person in the given surfer images. Therefore, judging that processing image data on a color-based scale rather than a gray scale would be effective for learning, I converted the given image on various scales using the cv2.cvtColor() function, including the HSV scale and Ycbcr scale. Since the previous gray scale was information of 1 channel, when converted to a different scale, the number of channels did not match due to the addition of information. Both HSV scale and Ycbcr scale had three channels. At first, I attempted to increase the number of channels in which the learning was conducted, but there were many difficulties in modifying the given code. Therefore, I chose the alternative to use only one value of the scales using the cv2.split() function in each. As the Ycbcr scale didn't make much improvement for the performance, I chose to use the H(Hue) or S(Saturation) value of the HSV scale which showed the best results. The given image above shows the code to use the saturation data.

2) Filter Recentralization Based on Initial Frame

```
if idx!=0 and idx%150 == 0:
    g = response_map[init_gt[1]:init_gt[1]+init_gt[3], init_gt[0]:init_gt[0]+init_gt[2]]
    fi = init_frame[init_gt[1]:init_gt[1]+init_gt[3], init_gt[0]:init_gt[0]+init_gt[2]]
    print(idx)
    G = np.fft.fft2(g)
    # start to do the pre-training...
    Ai, Bi = self._pre_training(fi, G)
```

The given correlation filter proceeds with learning by updating the filter map when moving from the previous frame to the next frame. This is an important requirement for tracking a moving target but repeating this process may cause the target to move to around the edge of the box or the box to drift away from the target entirely due to loosened original target feature. Since the background of the image is monotonous and stable compared to the target, I thought that if the value of the filter map is initialized to the value of the ground specified at the first frame, the target could be reset to the center of the box, resulting to improve the accuracy of tracking. Therefore, when the index reached a certain period based on the initial frame, the code to initialize the filter was written and added as in the image above.

Experimental results

1. Result of Hyperparameter Adjustment

First frame

[250, 100, 120, 180]

[220, 100, 160, 200]

Middle Frame

[250, 100, 120, 180] [220, 100, 160, 200]

Last frame

[250, 100, 120, 180] [220, 100, 160, 200]

As the ground box was adjusted to include most of the body parts for the majority of the images, it is shown that it stably tracks the whole body throughout the process. The limitation of not recognizing the body for the last frame when the body half submerged in water and tilted, has been solved, which was the most noticeable problem of the existing algorithm.

2. Result of Scale Change and Filter Recentering

1) Body tracking

The experiment for the scale adjustment used the complemented hyperparameters specified above.

When HSV color space is used, there was no noticeable difference in performance between Hue and Saturation, so the results are compared based on Saturation. In common, for body recognition, the upper body of the tracking target was most stably placed in the center of the box, based on the gray scale.

Gray Scale HSV(Saturation) Filter Recentering

Gray Scale HSV(Saturation) Filter Recentering

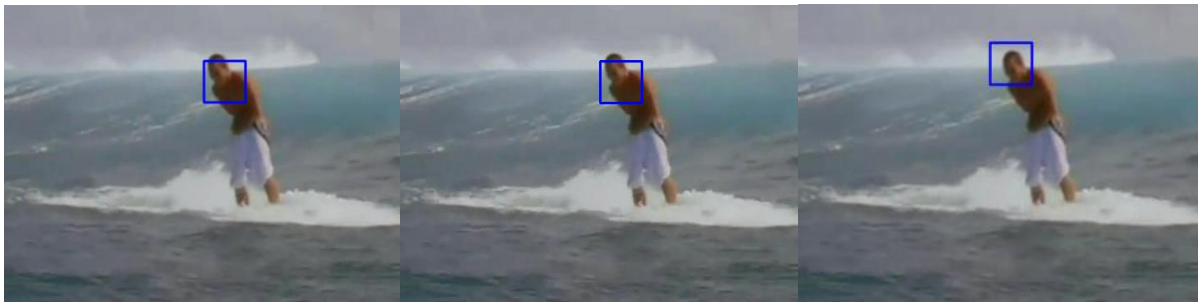
The original gray scale tracked the surfer's body most stably, as the hand or head was often slightly out of the box with the modified codes. Between the HSV version and Filter Recentering version, HSV showed better performance.

However, on the other hand, for most of the trials, the box for the modified codes contained more leg parts, especially with the filter recentering code. This can be interpreted that the performance was not poor, in that the leg part was recognized as part of the body, even though the body was separated by the white pants. It could be seen that by recentering the filter based on the initial frame, the target information was to include the arms and legs, and this affected the result of the filter recentering algorithm. This might have made the filter to predict that the upper body to be at the higher part of the box, resulting with flaws when the legs are concealed by the sea.

2) Head Tracking

Gray Scale HSV(Saturation) Filter Recentering

Shown above is the result of the three versions of codes, the early images before the filter recentering occurred. The code using saturation scale shows a noticeable improvement with tracking the head, comparing to the gray scale. As recentering didn't occur yet, the third code doesn't show any difference with the original code.

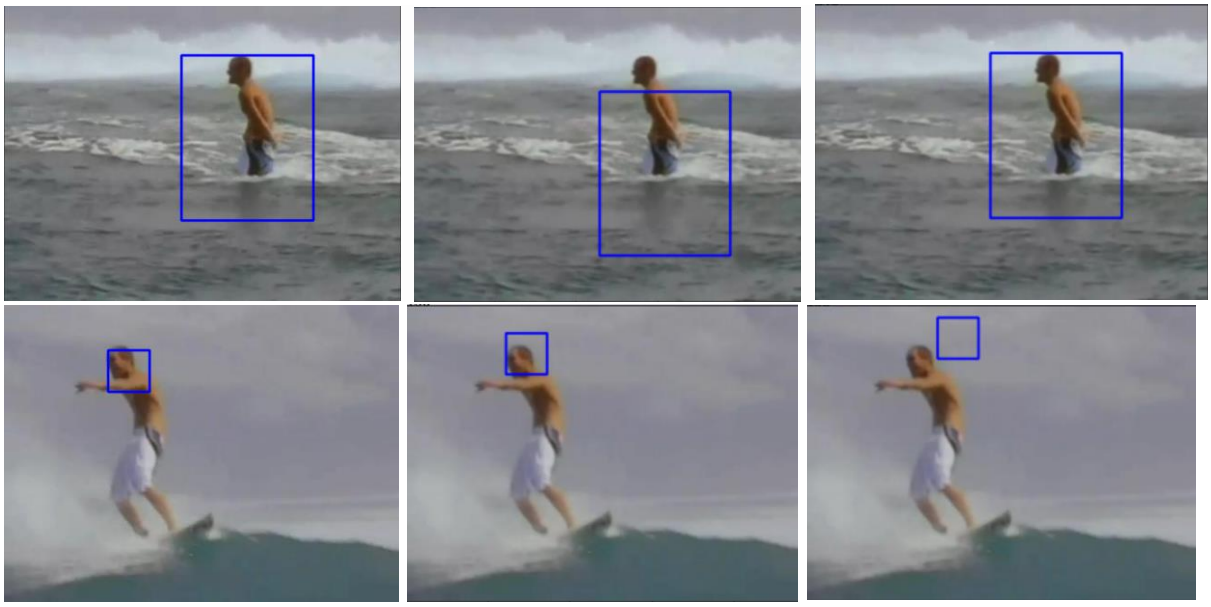


Gray Scale HSV(Saturation) Filter Recentering

However, when analyzing the results for the later images, the situation is reversed. The Saturation code is pointing the head and the body together, not accurately tracking the head, as same as the gray scale does. This shows the limitation of box drifting away from the target due to target movement. On the contrast, the filter recentering code is showing the head to be at the center of the box, producing the intended result. This represents that recentering the filter is effective, as the drifted filter map returns to the initial value and gives the result as centered target.

3) HSV + Filter Recentering

As scale change is effective with the early results and filter recentering is effective with the later results, it could be thought that by combining the two adjustments, they would complement each other.



Gray Scale | Filter Recentering | HSV + Filter Recentering

But as experimented, it improved the performance for the full body track, but worsened the head tracking result as the box often drifted away from the target.

Overall, changing the scale of the filter and recentering the filter are both a usable implementation for improving the tracking performance. But combining the two of them might require further adjustments.

Conclusion & Future works

Adjusting the hyperparameters can significantly improve the performance of the tracking algorithm. Also, setting suitable criteria for the given data has the effect of increasing tracking performance for a target with clear features. Each tuning method does not always yield good results, but it shows limited performance improvements. Therefore, if these hyperparameters can be adaptively adjusted according to the characteristics of the image, they will continue to show good performance. In addition, as shown in the HSV experiment, it was confirmed that tracking in color space containing other color components in addition to the gray scale could produce better results.

Although I have improved the performance of the algorithm with several hyperparameters, more complicated algorithm such as adaptive boundary box scaling can improve the performance further.

It may be possible to use clustering or other methods to highlight the boundaries to determine the area of the target, and then adjust the box to form a more flexible filter map for the movement or rotation of the target later.