Exercise 2: Mean-Shift tracking

Matej Miočić, 63180206, mm9520@student.uni-lj.si

I. Introduction

In this exercise we look into the mean shift algorithm. It's a technique for locating the maxima of a density function. We implemented an object tracking algorithm using this technique. We show examples, failures and parameter impact on multiple video sequences.

II. Experiments

A. Mean-shift algorithm

We illustrate the mean shift algorithm on a toy problem. We select a starting point and ascent to the mode of a simple generated function. We show that starting point and window size of the searching region impact the convergence speed and the final solution. Convergence speed obviously depends on the starting point. The closer to the final solution we start, the faster we converge. If we start too far away, we might not converge at all since algorithm won't know which direction to go. This also depends on the window size. If the window size is big enough to see the ascending values, function will start to converge. If the window is too small we could get stuck in local maxima (see Figures 1, 2). Larger window sizes give us faster convergence, since we see more area, but we might not get as accurate results (see Figures 1, 2). We can see that both convergence points for window size 31x31 are not at maxima but slightly to top-right. This is because the other local maxima is impacting the weights.

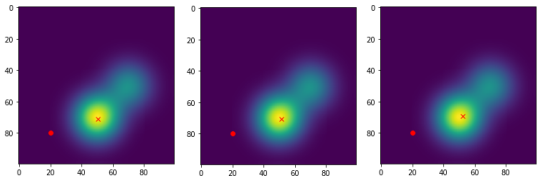

Figure 1: Successful convergence for all window sizes (5x5, 15x15, 31x31).

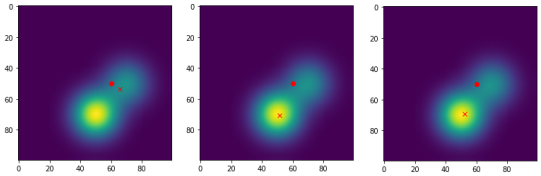

Figure 2: Failure of convergence for window size 5x5.

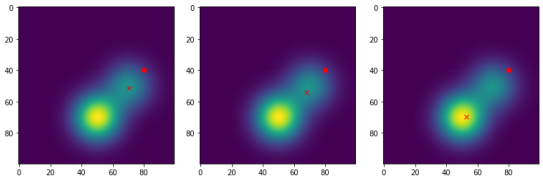

Figure 3: Failure of convergence for window sizes 5x5 and 15x15.

For stopping criteria we used euclidian distance. For Figure 1 algorithm needed 190 iterations for window size 5x5, 21 iterations for window size 15x15 and only 6 iterations for window size 31x31.

We also implemented our own pdf function to test the mean-shift algorithm (see Listing 1 and Figure 4). We select 3 to 10 random points and set their values between 0.3 and 0.5. Then we set a peak randomly somewhere in the region. Finally we use gaussian kernel to smooth our points.

Listing 1: Our own pdf generator.

```
responses = np.zeros((100,100))
for i in range(1, rand.randint(3,10)):
    responses[rand.randint(20,80),
              rand.randint(20,80)] =
    rand.uniform(0.3,0.5)

responses[rand.randint(20,80),
          rand.randint(20,80)] = 1
return gausssmooth(responses, 10)
```

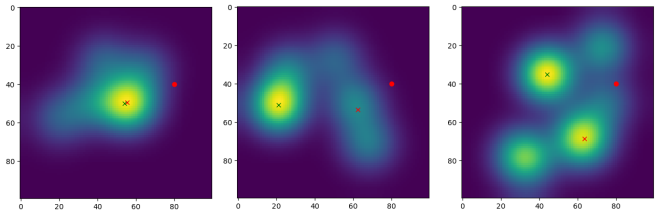

Figure 4: Mean-shift algorithm examples on our of pdf generator where red dot is the starting point, red cross convergence and green cross optimal solution.

B. Mean-shift tracker examples

We now show examples of mean-shift tracker at work from multiple sequences from VOT14 (see Table I). We have an annotated bounding box of a target and we try to follow it with mean-shift algorithm using color histogram and histogram backprojection. Algorithm succeeds most of the time, but fails if it comes close to an object that is similar to our target or if our target becomes occluded. We used the following parameters for all sequences (except parameter testing):

1) Smoothness of our kernel is $\sigma = 0.5$
2) Number of bins used for histograms = 16
3) Update parameter $\alpha = 0$

Table I: Obtained results on sequences from VOT14 using mean-shift tracker.

Sequence	Number of failures	FPS
basketball	1	205.4
bolt	0	665.0
skating	7	239.7
sunshade	0	440.3
torus	5	911.7

Our target can also change throughout the sequence. To tackle this problem there is an idea to update our color

histogram based on α parameter with each iterations. We tested this with multiple sequences and we have seen some improvements for some sequences but not for all. Mean-shift tracker fails where there are a lof of ambient changes and rotation of the target. Using $\alpha = 0.05$, to update histogram each iteration we used the tracker on *skating* sequence and obtained only 3 errors compared to 7.

C. Method parameters and improvements

We decided to use sequence *hand2* to show how parameter choice affects mean-shift tracking. We also implemented integrated feature selection with modeling background color distribution. For the starting region we obtained a histogram of the background which is 3 times larger than the target patch. With this method we try to tell our algorithm to focus more on our object than on the background since the background can change if our target is moving. With our choice of parameters we obtain 4 errors, but with background information we lower this number to 3 (for more information see Figures 5, 6, 7).

1) Alpha: We see that for this sequence alpha parameter does not change number of errors so much (see Figure 5). Even though the hand is rotating, updating the histogram does not help. This might be due to the fact that rotating the hand does not introduce big colour changes since our skin is relatively the same colour from both sides. When combining histogram update with background information the algorithm performs worse with bigger α parameter. This is because we only take information of the initial background. If we update our histogram, but not the background we're not helping the algorithm. Taking information of the background every iteration is computationally too expensive so we did not use it.

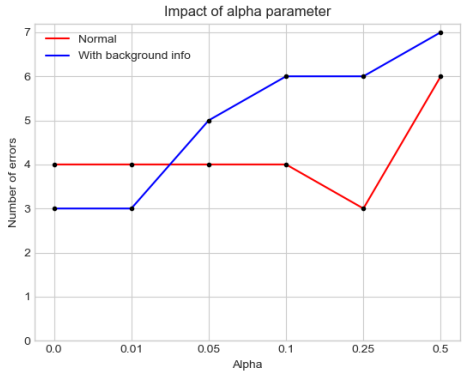

Figure 5: Number of errors based on **alpha** parameter on sequence *hand2*.

2) Number of bins: With number of bins we decide to how many colours we will reduce our problem. With more colours the problem becomes computationally more expensive. With our observations (see Figure 6) with increasing number of bins, we do not see improvement so we decided to use 16 bins.

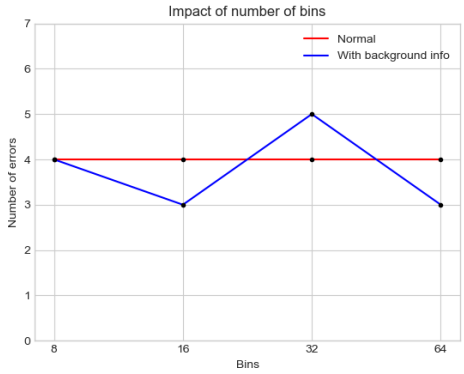

Figure 6: Number of errors based on **number of bins** on sequence *hand2*.

3) Sigma: With increasing the σ parameter our normal mean-shift tracker became gradually worse. Our explanation is that the more we smooth the image, the more likely it is that the tracker will mistake something with its target, since the image becomes less distinct. But in Figure 7 we can see how background information improves our tracking when we change σ. Even with bigger smoothing parameter, the tracker manages to tell the difference between the background and its target.

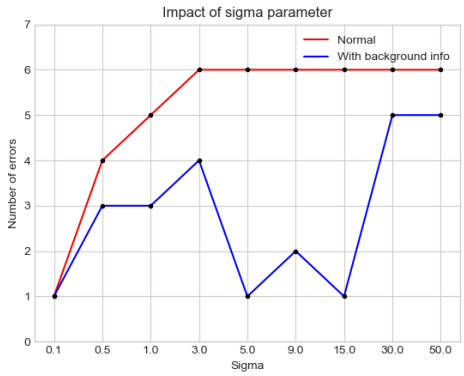

Figure 7: Number of errors based on **sigma** parameter on sequence *hand2*.

III. Conclusion

We have explored the mean-shift algorithm and implemented it in a tracking object algorithm. We have shown the mean-shift algorithm on a toy problem and implemented our own pdf generator function. We have shown how parameters affect convergence and its speed. Finally we have implemented a mean-shift tracker and tested it on various sequences. We explored how parameters affect tracking in terms of number of errors. We also implemented an improvement which used background information as feature selection.