Exercise 2: Mean-Shift tracking

Matic Isovski, mi6568@student.uni-lj.si

I. Introduction

In this paper we present a Mean-Shift algorithm for mode seeking and object tracking. We also present results we obtained when testing both scenarios. We tried different kernels, kernel sizes, termination criteria, etc.

II. Experiments

In the first part, we tested different parameters for ms mode seeking method. It was quite easy to 'break' the flow. One of the parameters is the starting position. If it is set too far, for example in a corner or any other area with only zeros, there is no way of convergence. The other 'trap' is to set the starting position closer to local maximum than the global. The second parameter is neighborhood (or kernel) size. Bigger the size, less short-sited our method was. Size also affects the number of iterations needed to converge and of course time complexity due to computation increase. Lastly, we have termination criteria. If the threshold is set too low, there is a high probability that it will not converge, but if it is set too high, it might stop near the maximum.

In Table I we can see results for different kernel sizes on our distribution function shown in Figure 1 (red cross is the starting position). We can clearly see how the kernel size

Table I

kernel size	iterations	result (x, y)
5	2	$(74, 88)$
7	1	$(74, 88)$
9	150	$(60, 117)$
11	31	$(60, 119)$

improves performance. With small kernel size, method return local maximum p(74, 88), but with increased size the ms is shifting towards the global maximum. The real maximum point is set to p(60, 120), so our best (and fastest) test is with kernel size of 11.

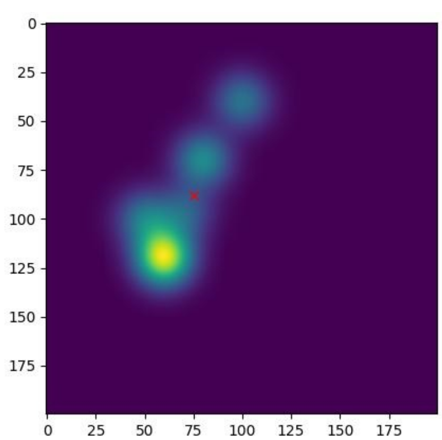

Figure 1. Custom probability distribution function.

In the second part, we tested ms tracker on 5 sequences, result are shown in Table II.

Table II

sequence	failures
ball	0
basketball	4
bicycle	1
bolt	2
car	1

The most common reason for failure is having two (overlapping) areas with similar color distributions. In Figure 2 we can see that the tracker switch to the other player's legs when they overlap. The same happened in basketball sequence (Figure 3); when the players were close, tracker switched because they look similar (skin color, jersey, ..). The solution for that would be to check the object's velocity vector. If the change is significant, then there is probably the mentioned failure case.

Figure 2. Example of a tracking failure on bolt sequence.

Figure 3. Example of a tracking failure on basketball sequence.

We tried to find good setting of the parameters, but we could not find such parameters, that would be universal for optimal performance. For instance when tracking bolt sequence, changing the number of bins from 16 to 8 gave us one failure less. We also obtained better results (better overlap with ground truth data) when using Gauss kernel, but time performance worsened. Interestingly, we could not find a case, where setting larger alpha brings better result, when updating q.

III. Conclusion

For simple sequences, ms algorithm works well and fast. For other sequences, there might be failures because of method's weakness when two similar areas overlap.