ETH zürich | Computer Vision and Geometry Lab

December 28, 2018

Computer Vision

Lab 11: Condensation Tracker

Sumeet Gyanchandani

In this lab, I implemented a CONDENSATION tracker based on color histograms using MATLAB. The objects to be tracked are represented by bounding-boxes. Here, we are tracking just one object and considering just two prediction models (modeled by matrix A): (i) no motion at all i.e. just noise; and (ii) constant velocity motion model. We also assume that the width and height describing the object bounding-box do not change over time.

1 Condensation Tracker

1.1 Color Histograms

This function calculates the normalized histogram of RGB colors occurring within the bounding box defined by *[xMin, xMax] x [yMin, yMax]* within the current video frame. The histogram is obtained by binning each color channel (R, G, B) into *hist_bin* bins. Here, for every pixel, I divide its value in each color channel to the number of bins, if the component fails into a particular bin, I increment its value by 1.

1.2 Deriving matrix A

For the first prediction model, i.e., no motion at all or just noise, we need an identity matrix, so that it is able to maintain the x and y values. Therefore, I made it equal to [1 0; 0 1]. For the second prediction model, i.e., constant velocity motion model, I selected $[[1,0]', [0,1]', [1,0]', [0,1]']$, so that the velocity component of each state is also considered.

1.3 Propagation

For this section, we propagate the particles given the system prediction model (matrix A) and the system model noise represented by *params.model*, *params.sigma_position* and *params.sigma_velocity*. I used MATLAB function **normrnd** to generate noise and then calculate the new particle locations.

1.4 Observation

This function makes observations, i.e. computes for all particles, its color histogram describing the bounding box defined by the center of the particle and W and H. These observations are then used to update the weights *particles_w* using *eq. 6* based on the χ^2 distance between the particle color histogram and the target color histogram. In order to compute the χ^2 distance, I use the function *chi2_cost.m*. Finally, I normalize the particle weights.

1.5 Estimation

In this function, I just estimate the mean state given the particles and their weights.

1.6 Resampling

This function resamples the particles based on their weights, and return these new particles along with their corresponding weights. I use the MATLAB function - *randsample*, to pick a random sample from the current set of particles, considering their weights as a probability distribution. Finally, I normalize the particle weights.

2 Experiments

You can run **demo.m** to see results similar to the ones I have described below.

2.1 Video 1

Fig. 1 depicts the result of tracking the hand in the first video with *params.model* $= 0$, i.e, no motion. I observed that the noise only model was sufficient in tracking the hand here! I didn't observe any improvement by using the constant velocity model, the tracking results were rather the same. I guess this is because of the slower motion of the hand and the uniform background that we were able to track the hand by using only the noise.

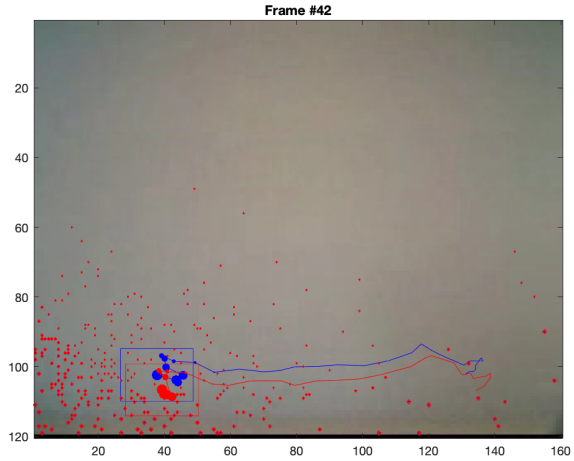

Figure 1. Tracking trajectories in video 1 for no motion model

2.2 Video 2

Fig. 2 shows the results of tracking the hand in the second video with params.model = 0, i.e, no motion. This time around the noise only model failed at the point where the hand was occluded by the statue! Here, the tracker confused the statue with the hand and doesn't move after the intersection and the hand moves away without getting tracked.

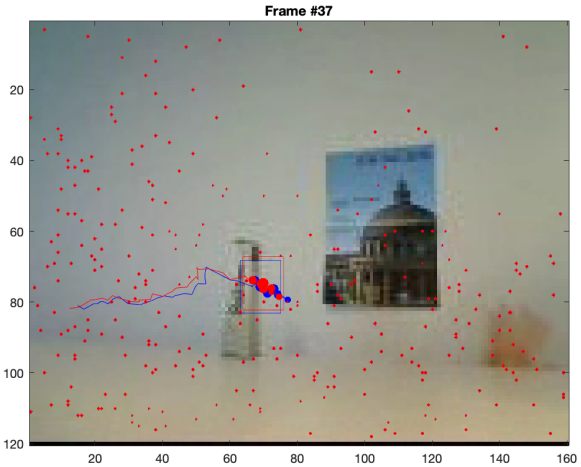

Figure 2. Tracking trajectories in video 2 for no motion model

I would attribute this to the fact that the histogram of the hand and that of the statue are similar. Using the constant velocity model and some quick changes in parameters like *sigma_observe* $= 0.2$ and *initial_velocity* $= [8,0]$ made the tracker overcome this challenge. Fig 3. depicts the resulting trajectory.

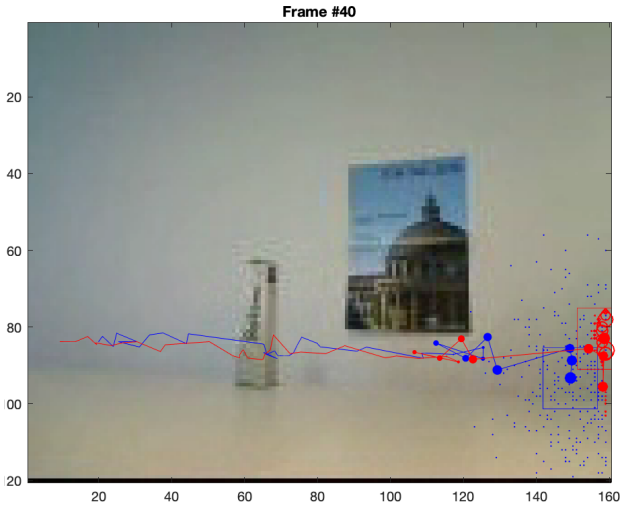

Figure 3. Tracking trajectories in video 2 for constant velocity model

The effect of using constant velocity is that the tracker gets additional help in guessing the location of the object, especially in cases where it is occluded. The bounding box moves in the direction of the velocity, even if the object is no more visible.

The effect of increasing the system noise is more robust tracking under conditions like occlusions. Decreasing it would reduce the robustness, but we get smoother tracking.

The effect of increasing the measurement noise is that we get more robust tracking for confusing backgrounds. Decreasing it would again lead to less robust but smooth tracking.

2.3 Video 3

If I use the same parameters that I used for the second video, I fail to track the ball. I am able to track the motion before the bounce, but after that, the tracker loses the ball. Fig 4. depicts the trajectory for this. I attribute this to less amount of system noise, that couldn't understand the ball hitting the wall and suddenly changing the direction of its velocity.

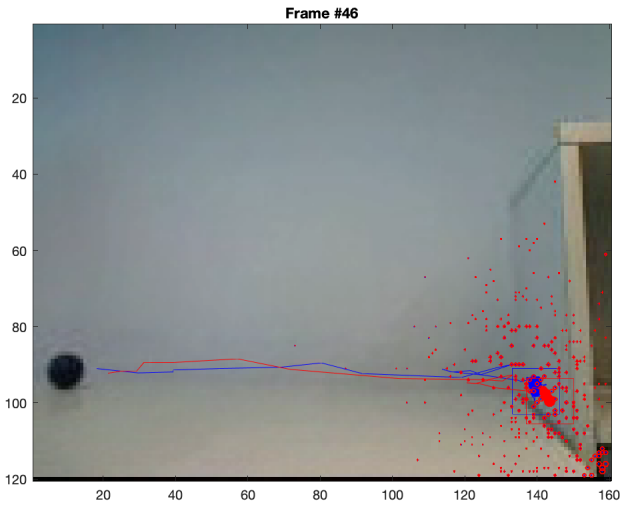

Figure 4. Tracking trajectories in video 3 for constant velocity model

To overcome this challenge, I increased the *sigma_observe* parameter from 0.2 to 0.5. I also increased the velocity a bit. The result is shown in Fig. 5.

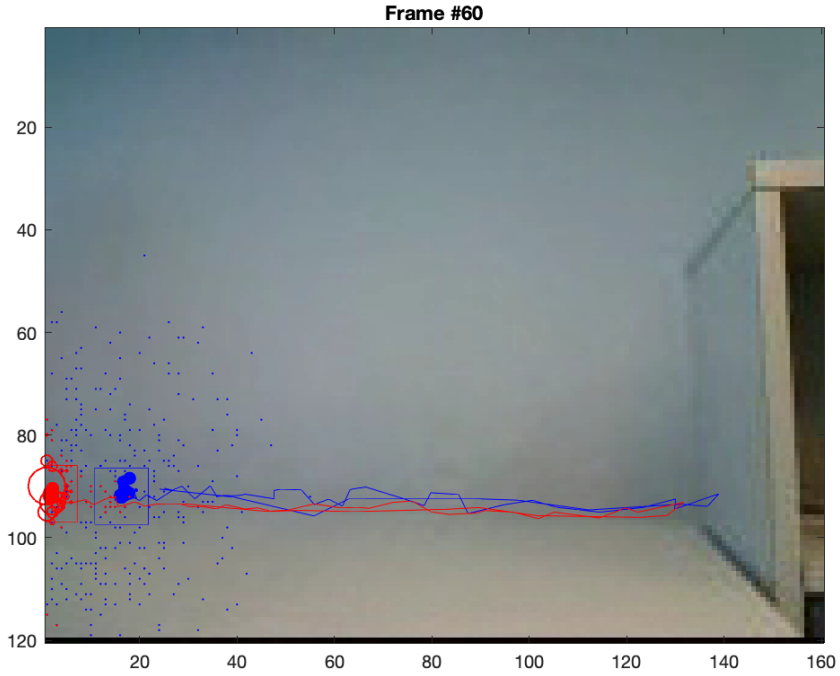

Figure 5. Tracking trajectories in video 3 for constant velocity model

The effects of increasing/decreasing the system/measurement noise that I mentioned above stand true for this video as well. But, the effect of whether or not we use the constant velocity motion model is different here! It is because there is no occlusion in this video. I was amazed by the fact that tracking was better in the no motion model than in the constant velocity motion model for the same parameters in the case of the third video. Fig. 6 shows the result of this tracking.

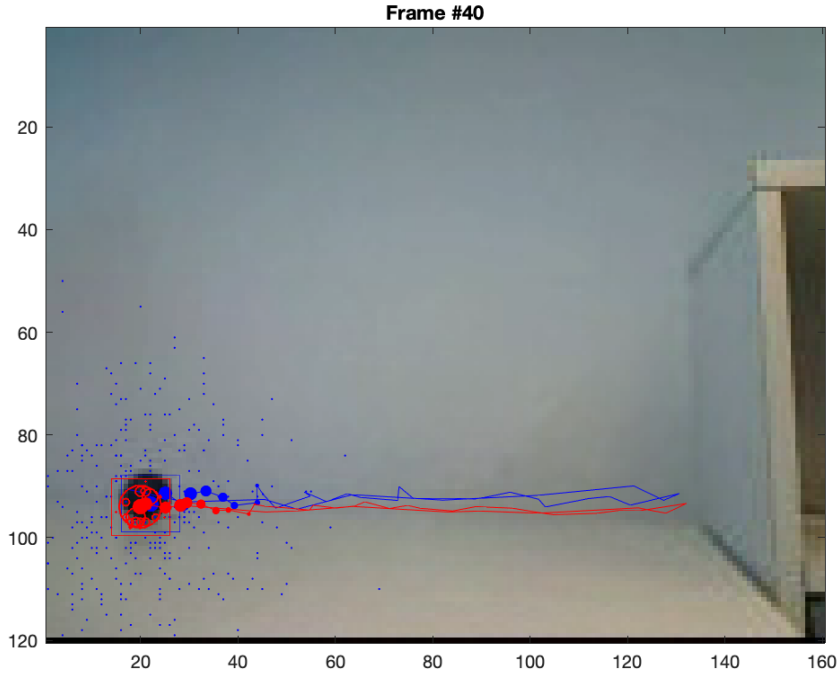

Figure 6. Tracking trajectories in video 3 for no motion model

2.4 Effects of parameter tuning

2.4.1 What is the effect of using more or fewer particles?

Fewer particles lead to bad hypotheses, while a large number of particles will lead to cancellation of the effect of noise. Too many particles also lead to scattering of them in bimodal or multi-modal distributions (or sometimes just skewed towards some random background noise). For instance, when the hand crosses the statue in video 2, we have a bimodal distribution where almost half the points are on the statue and half the points are on the hand. This is good because the tracker is able the model a bimodal hypothesis, but this is also bad because we average the positions of these points, resulting in the inferred position to be neither on the statue nor on the hand, but somewhere between them.

2.4.2 What is the effect of using more or fewer bins in the histogram color model?

More bins mean more precise calculation of histograms distances between the predicted and the target histograms. Increasing the number of bins will make the tracker computationally and spatially expensive.

2.4.3 What is the advantage/disadvantage of allowing appearance model updating?

Appearance model updating helps in the situations where there is a sudden change in the illumination or the motion the object in the scene. For instance, the object disappearing/teleporting from the scene or as in the case of video 3, sudden reversal of the direction of the motion. Apperance model updating result in better tracking in such situations.