Computer Vision

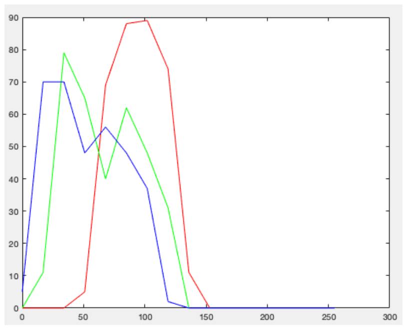
Alessia Paccagnella

December 16, 2019

1 Condensation Tracker Based On Color Histograms

1.1 Color histogram

In order to calculate the histogram of RGB colors occurring in the bbox defined by the inputs `[xMin, xMax]x[yMin, yMax]` in a certain video frame, I filled the function `color_histogram`. First of all, I checked if the bounding box was outside the frame: in that case, I cut it. Then I use the matlab function `imhist` to create a histogram plot by defining `hist_bin` equally spaced bins, where `hist_bin` is given as input of the function I implemented. I do it for every channel RGB and then return the histogram including the three of them, one per each row.
When I plot it, for example for the first frame, I obtain the following:



1.2 Derive matrix A

In this assignment, we consider two prediction models:
1. No motion.
2. Constant velocity motion.
We had to derive the matrix A for both of the models, in order to use them in the propagation step. In fact, we know that:

$$s_t = \mathrm{A}s_{t-1} + w_{t-1}$$

In the No motion model, we only have equations for x and y:

$$x_t = x_{t-1} + w_{x,t-1}$$
$$y_t = y_{t-1} + w_{y,t-1}.$$

Therefore, the A matrix has dimension 2x2 and is equal to the identity matrix:

$$\mathrm{A} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

In the constant velocity model though, we also have the two components velocities $\dot{x}$ and $\dot{y}$. Therefore, now the equations are 4:

$$\begin{aligned} x_t &= x_{t-1} + \dot{x}_{t-1}\ \mathrm{dt} + w_{x,t-1} \\ y_t &= y_{t-1} + \dot{y}_{t-1}\ \mathrm{dt} + w_{y,t-1} \\ \dot{x}_t &= \dot{x}_{t-1} + w_{\dot{x},t-1} \\ \dot{y}_t &= \dot{y}_{t-1} + w_{\dot{y},t-1} \end{aligned}$$

Therefore, the matrix A in this case is:

$$\mathrm{A} = \begin{bmatrix} 1 & 0 & dt & 0 \\ 0 & 1 & 0 & dt \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(I put dt = 1).

1.2.1 Propagation

Now that I derived the matrix A for both motion model, I had to use it to implement the function `propagate` that, given the particles, the sizes of the frame and the chosen parameters as input, check what is the actual model to be used (no motion or constant velocity) and propagates the particles. If the particles are not inside the frame after propagating, I put them in the edges of the frame.

1.2.2 Observation

Now we have to compute, for all the particles, their color histograms describing the bbox, so then we can update the weights particles using the following equation:

$$\pi = \frac{1}{\sqrt{2\pi}\sigma_{observe}}\ \mathrm{e}^{\frac{\chi^2(CH_{curr}-CH_{target})^2}{2\sigma^2_{observe}}}$$

We know that:
- $\sigma_{observe}$ is the normal variance defining the prob of each sample
- CH$_{curr}$ is the current color histogram of the sample we are considering
- CH$_{target}$ is the color histogram of the target

I looped all over the particles and computed the color histogram for each of them and the χ ^{2} distance use the provided function `chi2_cost`.
I filled the particles weights structure at every iteration.
At the end, before returning, I normalized the weights of the particles.

1.2.3 Estimation

Now we want to estimate the mean state given the particles and their weights. I did that by implementing the function `estimate`, in which I sum, for each particle, the product of the weights and the position.

1.2.4 Resample

The last function we needed to implement was the `resample` one. This function had to resample the particles based on their weights, and returning the new particles with their weights. I used a low variance resampling algorithm found on the internet.

1.3 Experiments

1.3.1 Experiment video 1

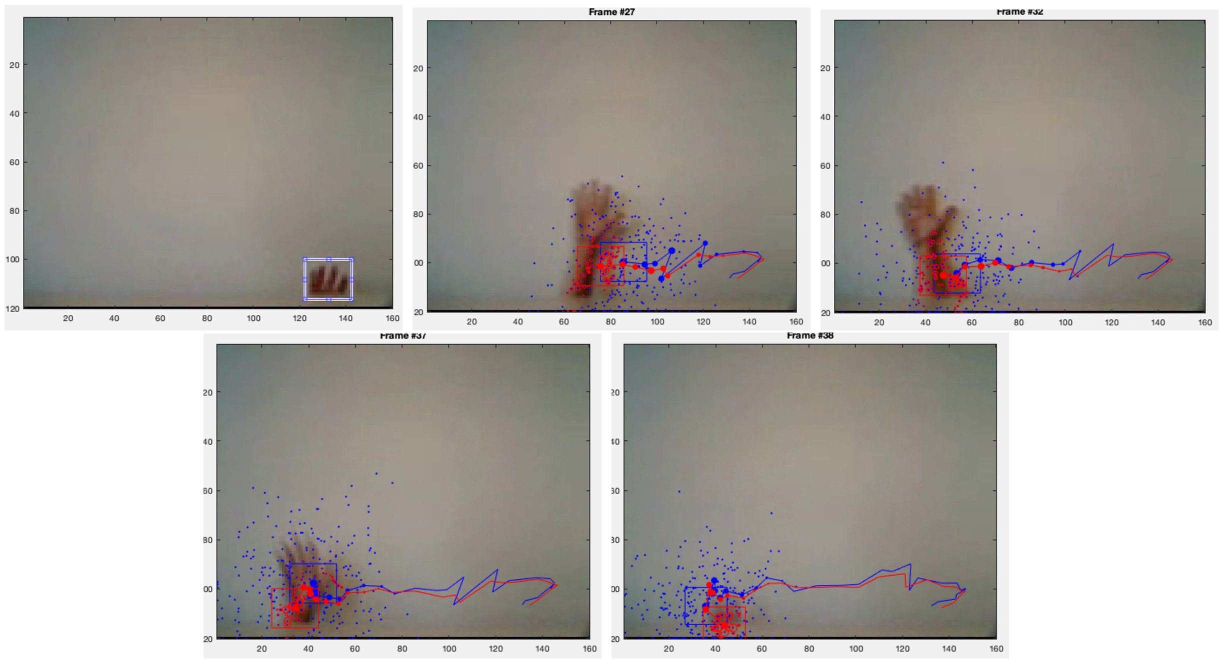
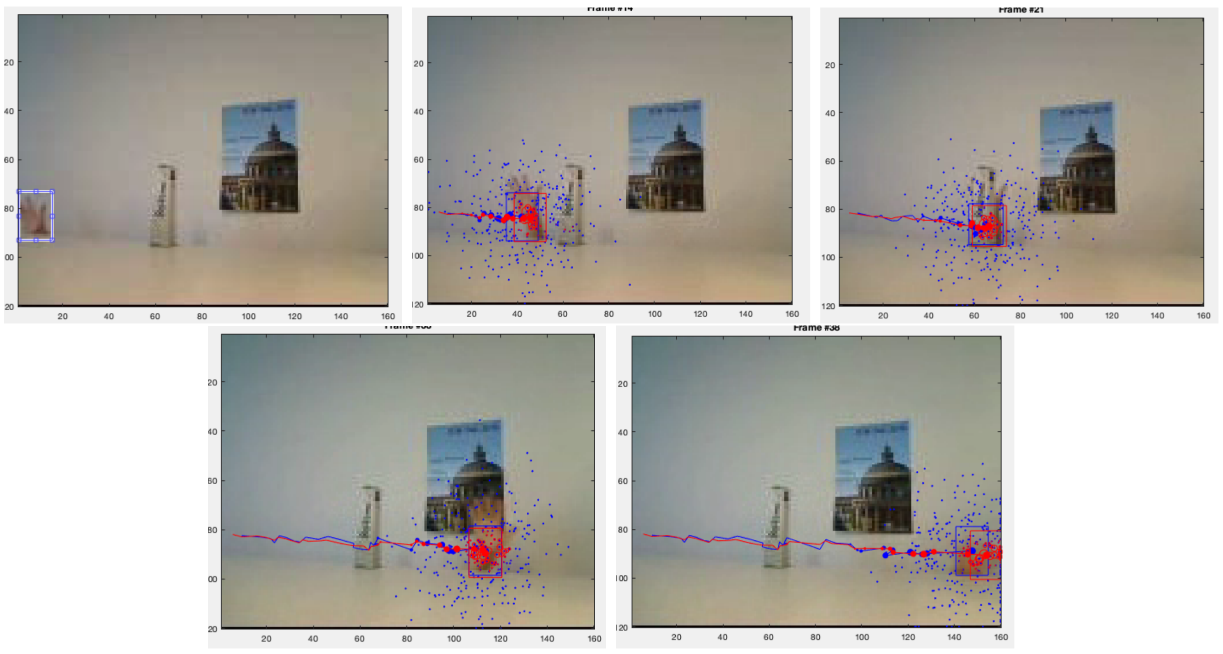


Figure 1: Blue points represent a priori particles, red points represents a posteriori particles. Blue lines represent the states of the a priori particles, red lines represent the states of the a posteriori particles.

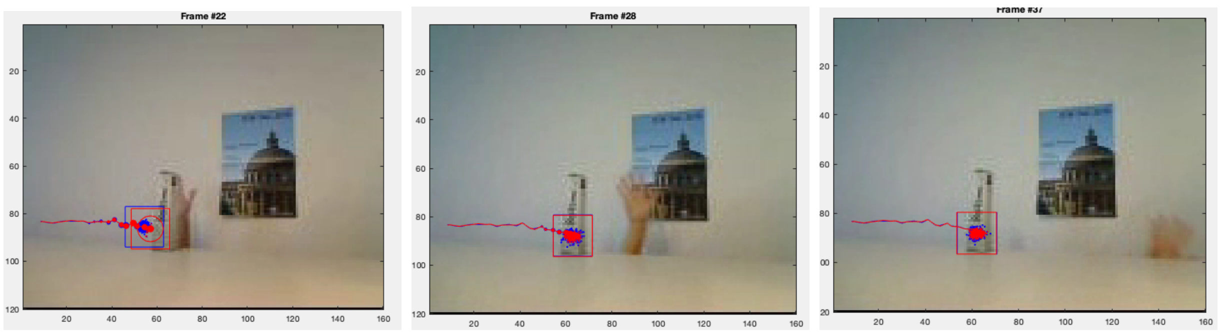
I used the default params given.
Particles follow the wrist of the person in a good way, but they don't specifically follow his hand. The reason for this is probably that on the first moving frames the fingers become lighter while the rest of the arm gets higher contrast.

1.3.2 Experiment video 2

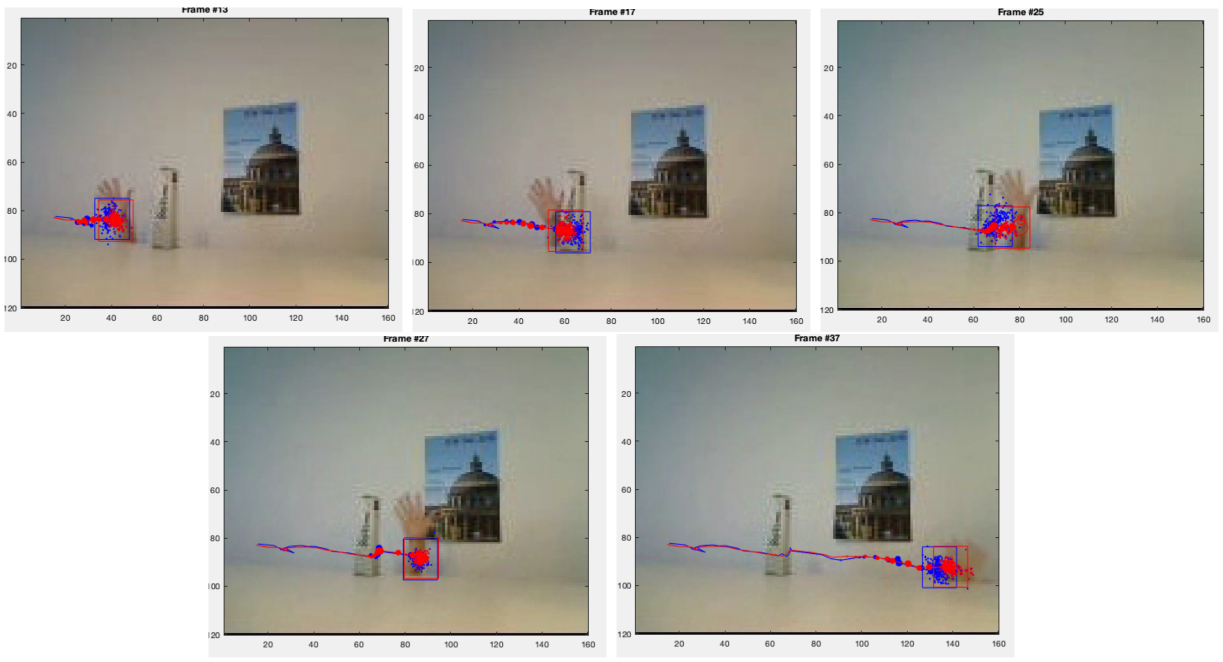
For the first trial that I did, I chose to use the **no motion model** with $\sigma_{position} = 15$.



For the second trial that I did, I chose to use the **no motion model** with $\sigma_{position} = 1$.



We can see clearly the difference: while in the first case, by using a $\sigma_{position} = 15$, the model is still able to track the hand for the whole video, by using $\sigma_{position} = 1$ the model stops tracking it at the first occlusion. In the second model in fact we can see that the particles are super close to each other, making it impossible to move after the hand reappears. With the first model instead, the particles are more spread in space, so some of them will be on the hand, letting the tracking work until the end.
For the third trial that I did, I chose to use the **costant velocity motion model** with $\sigma_{position} = 1$ and $\sigma_{velocity} = 1$



Now the movement of the hand is tracked until the end of the video.
For all these experiments, I always used $\sigma_{observe} = 0.1$.
I then tried to change the values of $\sigma_{observe}$. For $\sigma_{position}$, I used again the initial value 15. Here we can see the results:

Figure 2: first image $\sigma_{observe} = 0.6$; second image $\sigma_{observe} = 0.2$;

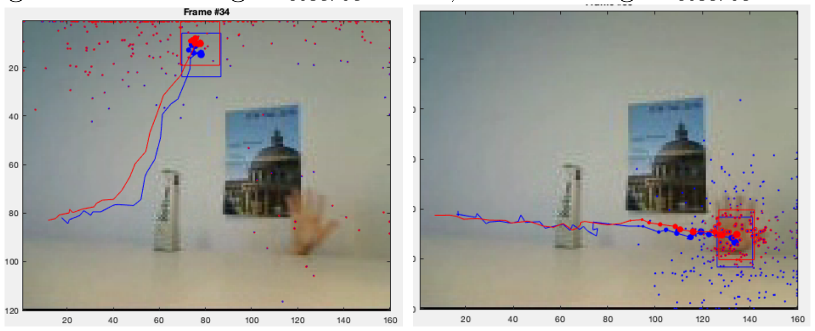
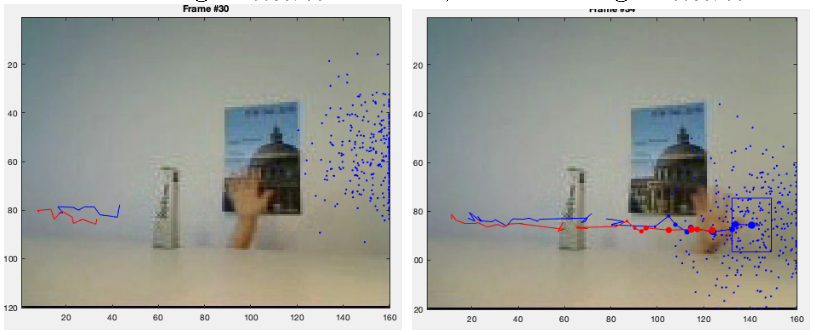


Figure 3: first image $\sigma_{observe} = 0.004$; second image $\sigma_{observe} = 0.008$;

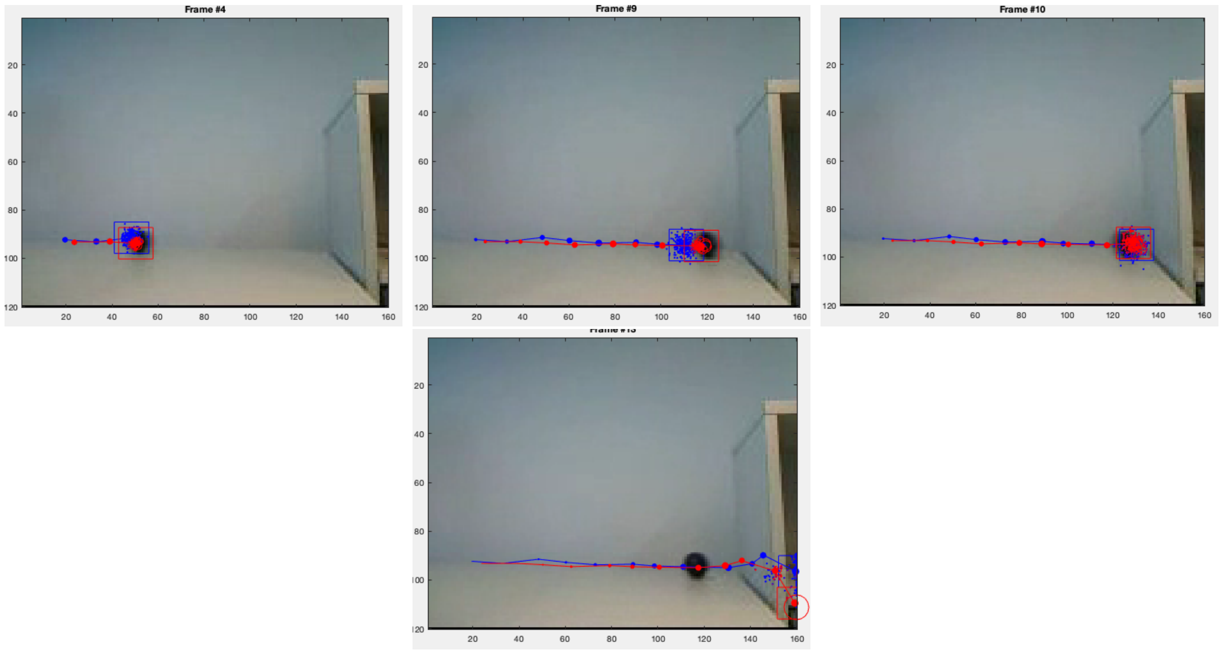


We can observe that for values close to 0.1 (that we used before) like 0.2, the result is quite

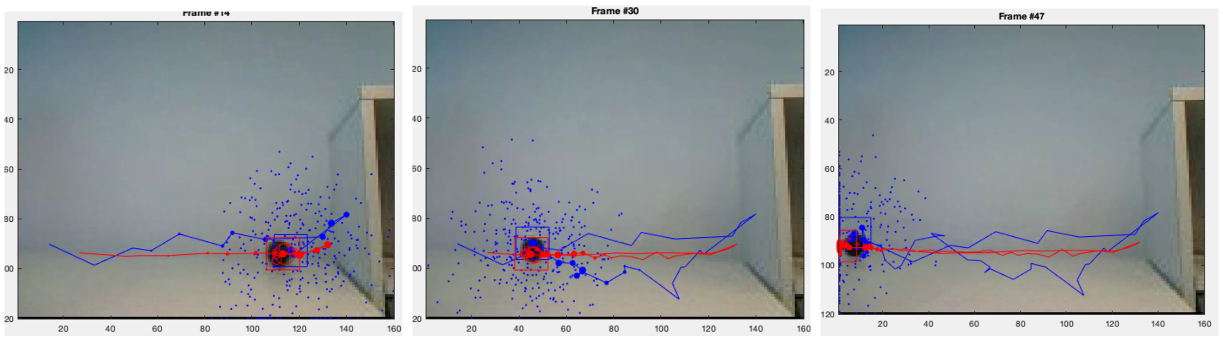
good. However, if we increase the value of $\sigma_{observe}$ too much (e.g 0.6), the weights are spread widely, therefore it is given importance to particles that should not have. Indeed we can see that the the hand is not tracked in a good way.
If we descrease the value of $\sigma_{observe}$ too much, we can see that the tracking is lost easily because particles have very small weights. The tracker in this case is not flexible at all for changes of the bbox.

1.3.3 Experiment video 3

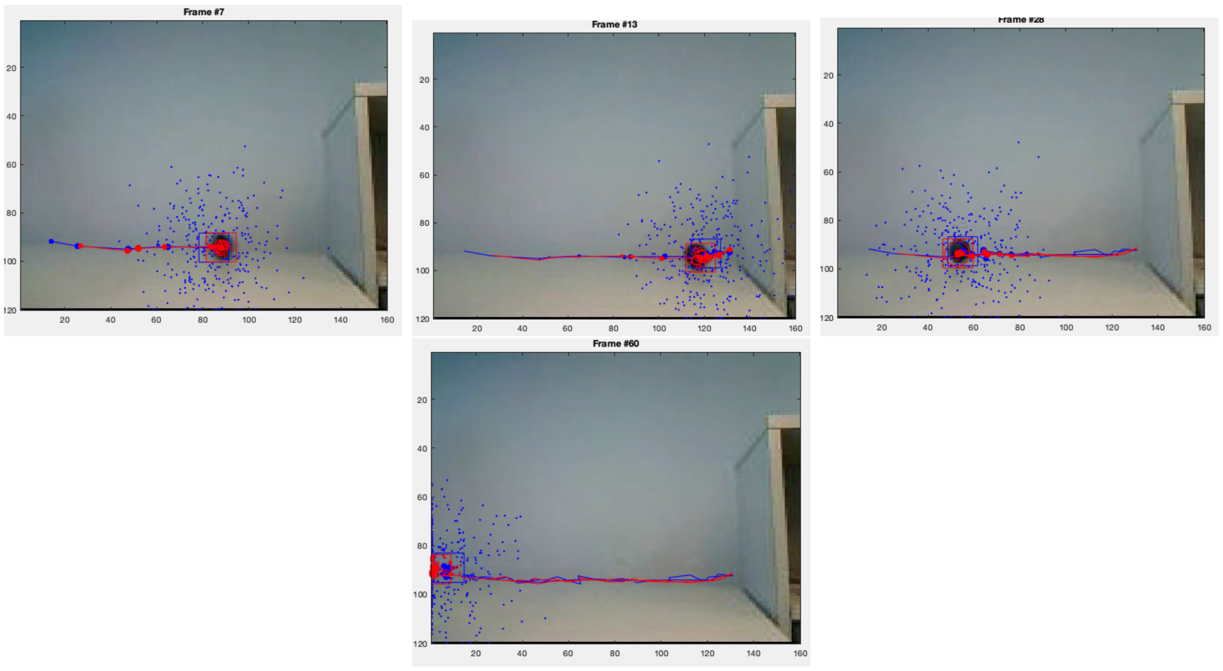
I first used the best parameters tried with video 2: $\sigma_{observe} = 0.1$, and velocity =[8,0], **constant velocity model**, and $\sigma_{position} = 1$.



We see however that with these values we keep track of the ball until it bounces on the wall, and then the tracking is lost.
I tried to put a smaller velocity [1,0] and increasing the $\sigma_{velocity}$ to 5, to let the particle velocity to change. I obtained a better result: the ball is tracked also after the bounce. However the tracking lines changes very randomly and suddently:



Therefore I tried with the **no motion model** and by putting $\sigma_{position} = 15$ just like default values, and I obtained the following:

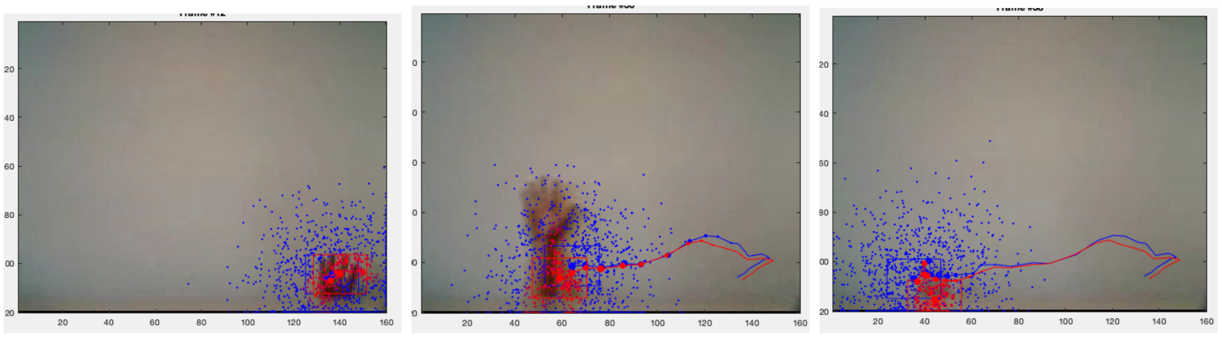


The ball is this way successfully tracked for the entire scene, probably because the area where the particles propagate is larger.

1.4 Questions

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

Increasing the number of particles leads to a more precise mean state, therefore the tracking results of a better quality and more robust. If we use too less particles, the tracking fails, mainly after the occlusion, and it's also very random. We can see it from here:



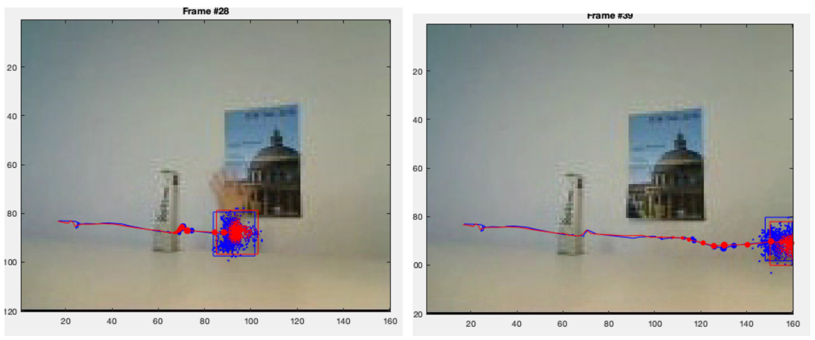


Figure 4: Using num of particles = 1000

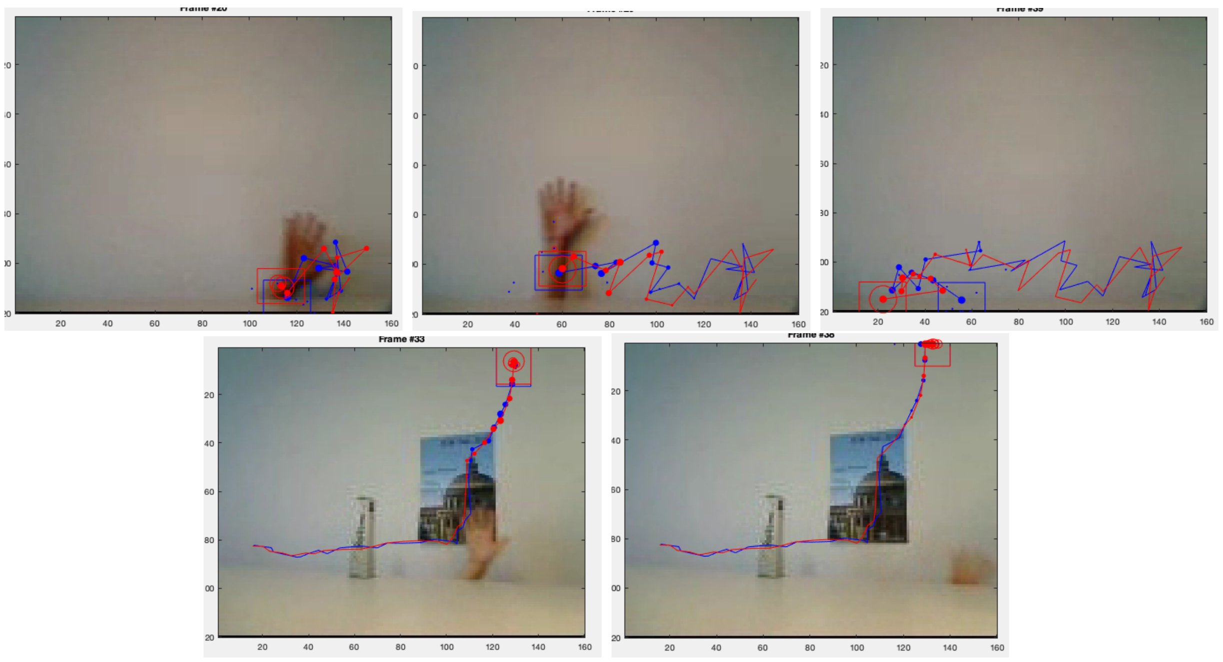


Figure 5: Using num of particles = 10

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

Increasing the number of bins in the histogram color model leads to a better quality of the tracking. The histogram are indeed more precise. With too few bins, the histograms would be too similar with each other, leading to less accuracy. However, having a small number of bins makes the model more robust to noise. For example:

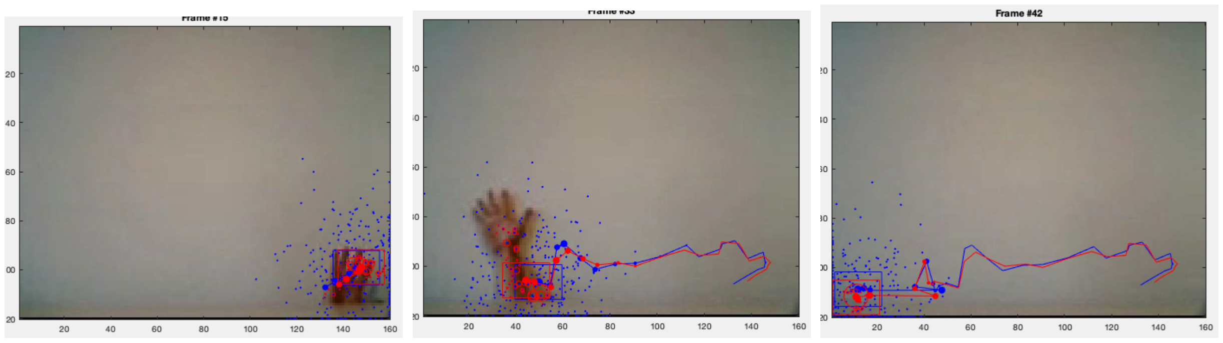


Figure 6: Using num of bins = 200

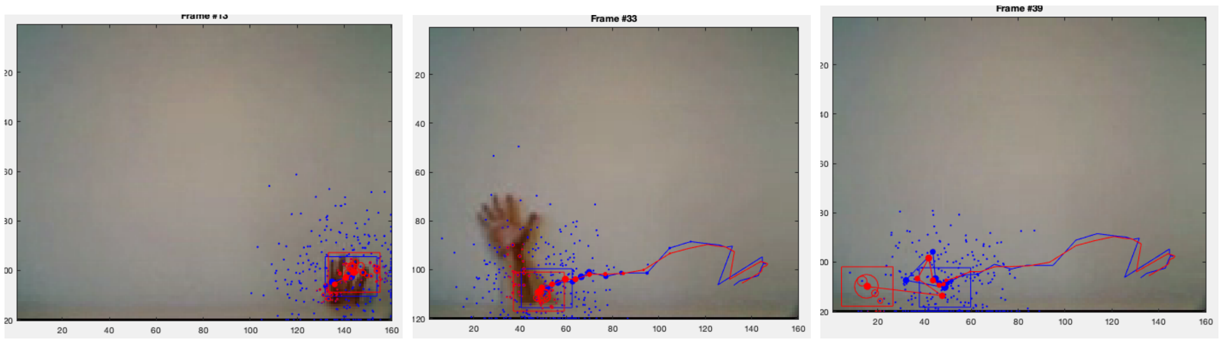


Figure 7: Using num of bins = 100

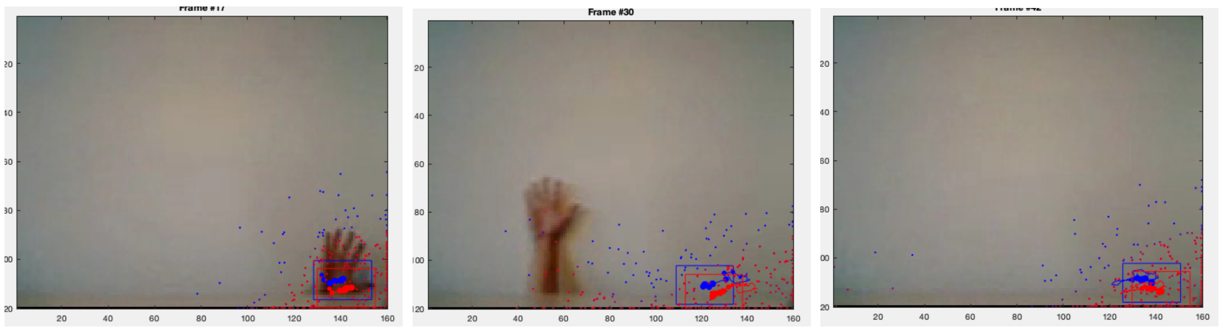


Figure 8: Using num of bins = 2

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

Appearance model updates with alpha. If we increase alpha the tracker is more sensible to luminosity changes and can adapt to them. For example, this could help in the first case of tracking of the wrist instead of the hand. However, a possible disadvantage could be that a slow moving object with a hand tracker could trick the tracker by making it believe that the histogram of the object represents a hand, which would cause the tracker to follow the object instead of the hand for example.