Computer Vision
Assignment# 6

Submitted by
Shubham Kumar(21-944-491)

24 December, 2021

Contents

List of Figures

List of Tables

1 Implementation

1.1 Color Histogram

`hist = color_histogram(xmin, ymin, xmax, ymax, frame, hist_bin)`

In *color_histogram* the x and y limits are used to extract the patch (*frame_patch*) from the *frame* for which histogram will be computed. After extracting the frame, for each of the 3 channels, a histograms with *hist_bin* number of bins is computed, and are concatenated to form a vector with $hist\_bin \times 3$ elements. The concatenated histogram vector is normalized by dividing it by its sum of elements. Later in the pipeline, this vector is used as feature for finding the degree of similarity between the patches.

1.2 Derive matrix A

Q. *In this exercise we consider two prediction models: (i) no motion at all i.e. just noise; and (ii)constant velocity motion model. For both, derive the dynamic matrix A and explain how you worked them out*

(i) *no motion at all*:
For the first case, there is "no motion" but only noise. Also, for such model state has 2 elements (elements corresponding to velocity is not modeled). So the state exploration equation essentially becomes:

$$x_t^{(n)} = x_{t-1}^{(n)} + w_{t-1,1}^{(n)} \tag{1}$$

and

$$y_t^{(n)} = y_{t-1}^{(n)} + w_{t-1,2}^{(n)} \tag{2}$$

In matrix form

$$\begin{aligned} \begin{bmatrix} x_t^{(n)} \\ y_t^{(n)} \end{bmatrix} &= \begin{bmatrix} x_{t-1}^{(n)} \\ y_{t-1}^{(n)} \end{bmatrix} + \begin{bmatrix} w_{t-1,1}^{(n)} \\ w_{t-1,2}^{(n)} \end{bmatrix} \\ \implies \begin{bmatrix} x_t^{(n)} \\ y_t^{(n)} \end{bmatrix} &= \mathbb{I}_2 \begin{bmatrix} x_{t-1}^{(n)} \\ y_{t-1}^{(n)} \end{bmatrix} + \begin{bmatrix} w_{t-1,1}^{(n)} \\ w_{t-1,2}^{(n)} \end{bmatrix} \\ \implies s_t^{(n)} &= \mathbb{I}_2 s_{t-1}^{'(n)} + w_{t-1}^{n} = A s_{t-1}^{'(n)} + w_{t-1}^{(n)} \\ \implies A &= \mathbb{I}_2 \qquad \text{(where } \mathbb{I}_2 \text{ is a } 2 \times 2 \text{ identity matrix)} \end{aligned}$$

(ii) ***constant velocity model*:**
For the second case, there is "motion with constant velocity". So the state exploration equation becomes:

$$\begin{aligned} x_t^{(n)} &= x_{t-1}^{(n)} + \dot{x}_{t-1}^{(n)}\Delta t + w_{t-1,1}^{(n)} \\ &= x_{t-1}^{(n)} + \dot{x}_{t-1}^{(n)} + w_{t-1,1}^{(n)} \\ &\quad \text{Since velocity is measured here in pixels per frame,} \\ &\quad \text{and observations are made for each frame.} \\ &\quad \text{Therefore, we can take } \Delta t = 1 \text{ frame} \end{aligned} \tag{3}$$

Similarly,

$$y_t^{(n)} = y_{t-1}^{(n)} + \dot{y}_{t-1}^{(n)} + w_{t-1,2}^{(n)} \tag{4}$$

For velocities,

$$\begin{aligned} &\dot{x}_t^{(n)} = \dot{x}_{t-1}^{(n)} + w_{t-1,3}^{(n)} \\ &\text{Since velocity stays constant, except for added noise} \end{aligned} \tag{5}$$

Similarly,

$$\dot{y}_t^{(n)} = \dot{y}_{t-1}^{(n)} + w_{t-1,4}^{(n)} \tag{6}$$

Writing equations [3, 4, 5, 6] in matrix form yields:

$$\begin{bmatrix} x_t^{(n)} \\ y_t^{(n)} \\ \dot{x}_t^{(n)} \\ \dot{y}_t^{(n)} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{t-1}^{(n)} \\ y_{t-1}^{(n)} \\ \dot{x}_{t-1}^{(n)} \\ \dot{y}_{t-1}^{(n)} \end{bmatrix} + \begin{bmatrix} w_{t-1,1}^{(n)} \\ w_{t-1,2}^{(n)} \\ w_{t-1,3}^{(n)} \\ w_{t-1,4}^{(n)} \end{bmatrix} \tag{7}$$

Therefore,

$$A = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{8}$$

1.3 Propagation

```
particles = propagate(particles, frame_height, frame_width, params)
```

In this function, based on the model code the dynamic matrix (A) is selected. To get the next state (predicted state before adjustment), A is multiplied with transposed *particles* (of size *num_particles* $\times$ *state_len*) and then a random normal noise vector is added with the mean 0 and standard deviation *sigma_position* (first two vector elements). In case of constant velocity model, the last two noise vector elements are sampled with mean 0 and standard deviation *sigma_velocity*. Helper function *clip_values* is used to clip the values such that the new particles are inside the frame.

1.4 Observation

```
particles_w = observe(particles, frame, bbox_height,
                 bbox_width, hist_bin, target_hist,
                 sigma_observe)
```

Here for each of the particle , colour histogram is computed over a patch of given size (*bbox_height* × *bbox_width*) with particle as the patch center. In case the patch boundary gets outside the given frame, only the part of the patch inside the frame is considered. Once all the histograms have been computed (and stored in *particles_hist*), in *compute_particle_weights* these are compared with the *target_hist* using the provided *chi2_cost* function. The distance (*chi_sqr_dist*) so computed, is further used to compute the new particle weights (*particles_w*) using the equation (6) in the handout.

1.5 Estimation

```
mean_state = estimate(particles, particles_w)
```

In this function, the weighted mean (*expected_particle*) of *particles* is computed with *particles_w* as the weight. This estimated mean is used as the center of the bounding box later.

1.6 Resampling

```
particles, particles_w = resample(particles, particles_w)
```

In this function, *num_samples* index are selected (drawn) independently from the given distribution *particles_w*, to get an array of sampled indices (*sampled_indices*). The new *particles* array is formed by taking the elements pointed to by *sampled_indices*. Similary new *particles_w* is formed (*particles_w*[*sampled_indices*]), and it is scaled properly so that the sum of its elements add up to 1.

2 Experimentation

For all the experiments *.avi* files were used (not the *.wmv* ones). Tables [1], [2] and [3] list the hyper-parameters used for the three videos (in order). The column *Sr.* in the tables represent experiment number.

2.1 video1

Sr.	N_{bins}	model	α	$N_{particles}$	$\sigma_{position}$	$\sigma_{velocity}$	V_0	$\sigma_{observation}$
1.1	16	$M_{NoMotion}$	0.1	300	15	-	-	0.1
1.2	16	$M_{NoMotion}$	0.1	300	8	-	-	0.1
1.3	16	$M_{NoMotion}$	0.1	300	2	-	-	0.1
1.4	32	$M_{NoMotion}$	0.1	300	15	-	-	0.1
1.5	64	$M_{NoMotion}$	0.1	300	15	-	-	0.1
1.6	16	$M_{ConstVelocity}$	0.1	300	15	1	[1, 10]	0.1
1.7	16	$M_{ConstVelocity}$	0.1	300	15	1	[-1, -10]	0.1
1.8	16	$M_{ConstVelocity}$	0.1	300	15	1	[5, -10]	0.1

Table 1: Experiment 1 hyperparameter values

With hyperparameters of experiment 1.1 the tracking was successful but as the time passed, the bounding box drifted below the palm. With small $\sigma_{position} = 2$, in experiment 1.3 the tracking was not successful (please refer to figure [1]). Tracking performance was the best for the hyperparameters setting of experiment 1.8, where the model was "constant velocity model" and the initial velocity was adjusted close to the actual initial velocity. In this case too the bounding box drifted slightly below the palm but this was relatively smaller than the experiment 1.1.

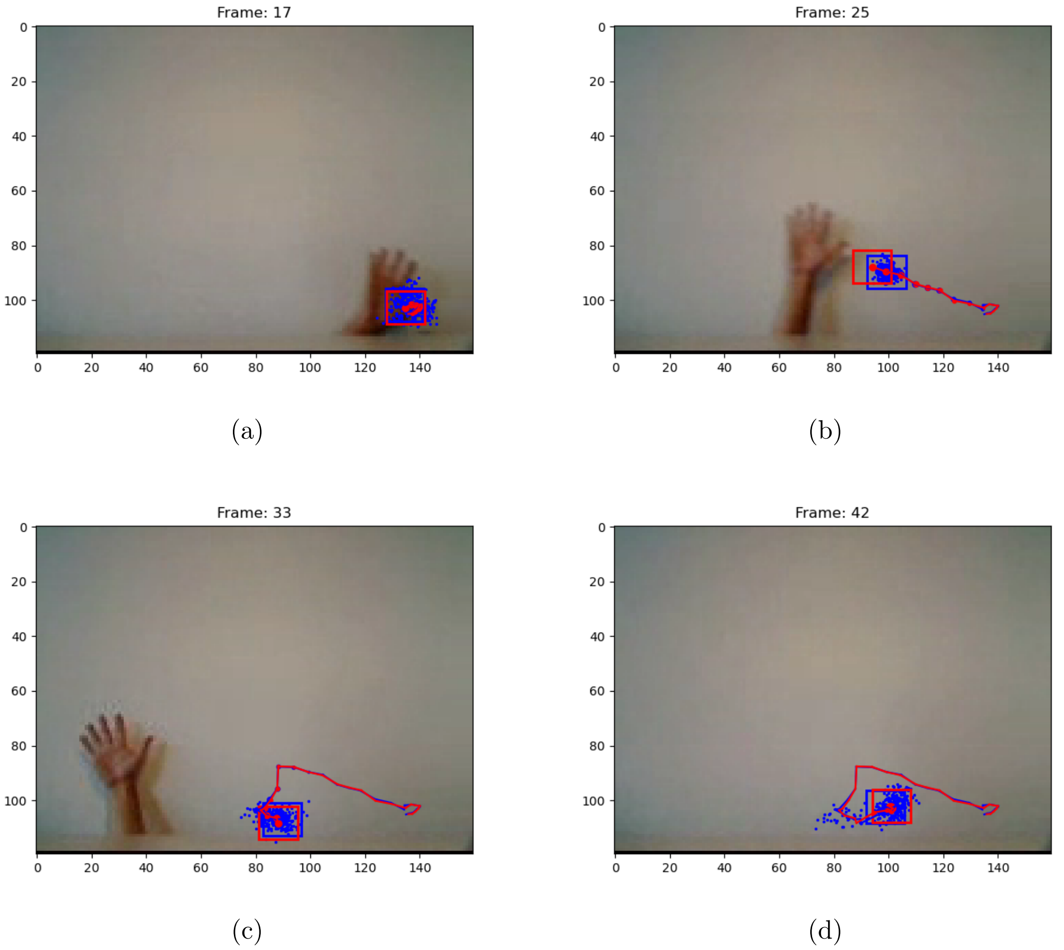


Figure 1: Experiment 1.3 Output (Please refer to table 1 for hyperparameters)

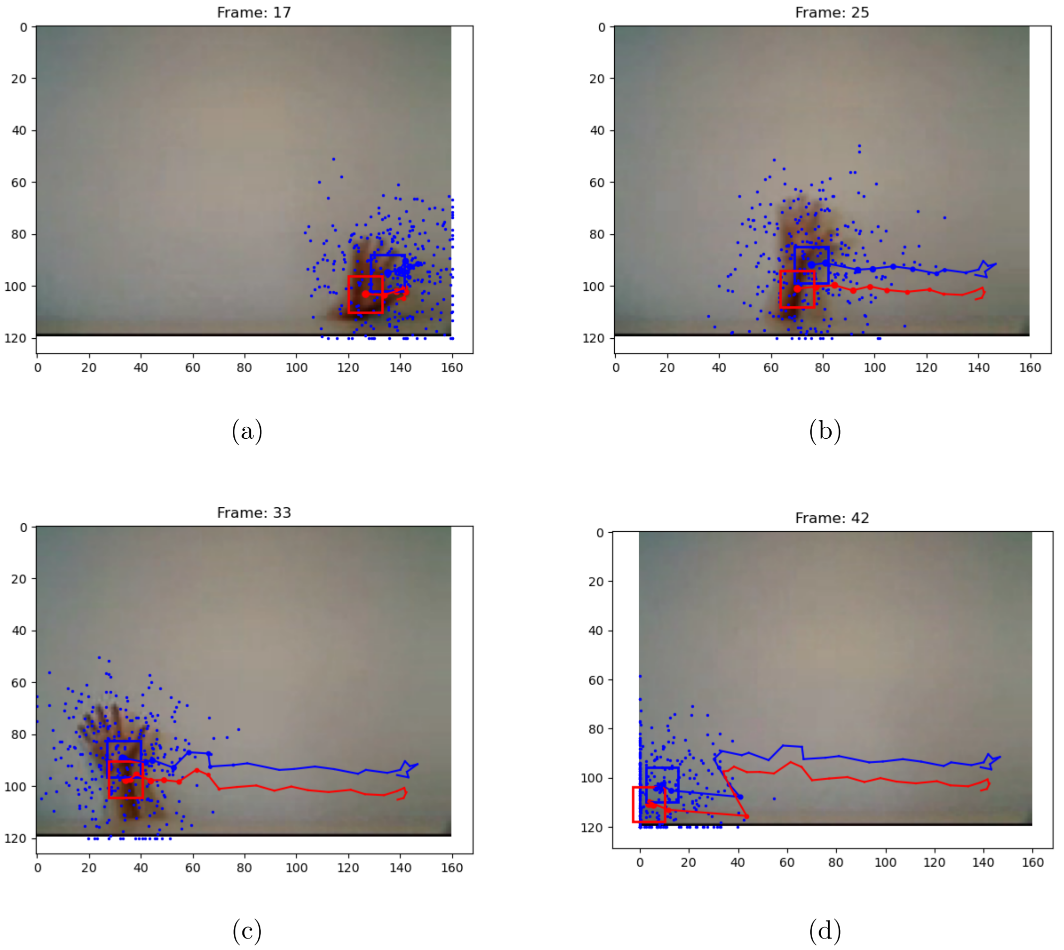


Figure 2: Experiment 1.8 Output (Please refer to table 1 for hyperparameters)

2.2 video2[1]

Sr.	N_{bins}	model	α	$N_{particles}$	$\sigma_{position}$	$\sigma_{velocity}$	V_0	$\sigma_{observation}$
2.1	16	$M_{NoMotion}$	0.1	300	15	-	-	0.1
2.2	16	$M_{ConstVelocity}$	0.1	300	15	1	[1, 10]	0.1
2.3	16	$M_{ConstVelocity}$	0.1	300	15	10	[1, 10]	0.1
2.4	16	$M_{ConstVelocity}$	0.1	300	5	1	[1, 10]	0.1
2.5	16	$M_{ConstVelocity}$	0.1	300	20	1	[1, 10]	0.1
2.6	16	$M_{ConstVelocity}$	0.1	300	5	1	[1, 10]	10
2.7	16	$M_{ConstVelocity}$	0.1	300	5	1	[1, 10]	5
2.8	16	$M_{ConstVelocity}$	0.1	300	5	1	[1, 10]	0.05
2.9	16	$M_{ConstVelocity}$	0.1	300	5	1	[1, 10]	0.09
2.10	32	$M_{ConstVelocity}$	0.1	300	5	1	[1, 10]	0.1
2.11	8	$M_{ConstVelocity}$	0.1	300	5	1	[1, 10]	0.1
2.12	16	$M_{ConstVelocity}$	0.1	300	5	1	[1, 10]	0.001

Table 2: Experiment 2 hyperparameter values

- ***What is the effect of using a constant velocity motion model?***
  The dynamic model helps in estimating our prior belief where the next state could be. Therefore having a dynamic model which properly models the reality is better for tracking, as the predicted prior state to be explored will be closer to the actual next state. Therefore, for moving objects, when the constant velocity model is used, the prior and posterior (adjusted state after observation has been made) mean positions (bounding boxes) are closer to each other and the tracking is smoother.

- ***What is the effect of assuming decreased/increased system noise?***
  System noise represents the uncertainty in our dynamic model. In a way system noise can also be considered as the degree of exploration we want to allow (lower noise models lower degree of exploration and higher degree of belief in the state predicted by the dynamic model and vice versa).

  So, when in experiment 2.3 when $\sigma_{velocity}$ is set to a large value (= 10), the blue track (prior) is zig-zag (as can be seen in figure [3]). This random looking track is observed as a result of higher degree of exploration (higher deviation from noise-free dynamic model output). Similarly with lower noise levels we observe a smoother blue track (prior) but it may drift away from the actual

[1] Please note: The *last_frame* for this video (video2.avi) was set to (38) in the code, because the default value (40) resulted in "empty" frames. Error: (-215:Assertion failed) !_src.empty() in function 'cvtColor'

object or may get stuck at some patch other than the target object, because the degree of exploration is low.

- ***What is the effect of assuming decreased/increased measurement noise?***
Since, particle's probability ($\pi^{(n)}$) is modeled by equation [9]. The measurement noise directly controls the spread of the probability distribution over the distance between the histogram corresponding to particle (as patch center) and the target histogram.

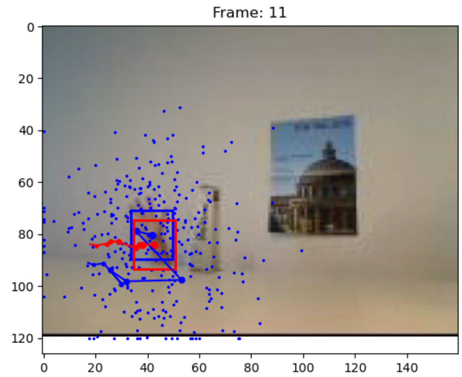
$$\pi^{(n)} = \frac{1}{\sqrt{2\pi}\sigma_{observation}} \cdot e^{-\frac{\chi^2(CH_{s^{(n)}} - CH_{target})}{2\sigma^2_{observation}}} \tag{9}$$

Therefore, if the $\sigma_{observation}$ is too small, the probability would be concentrated around zero distance, and hence only the particles with histograms very close to the target histogram would be considered for mean posterior state calculation.

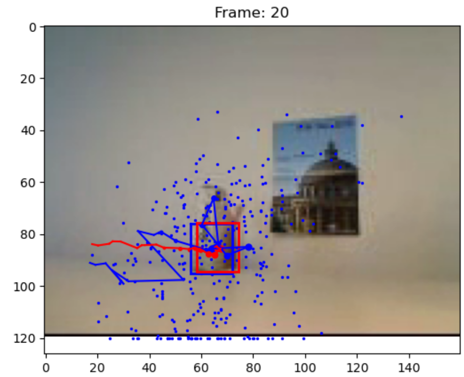
When $\sigma_{observation}$ is large, significant probability mass is also given to the particles even if whose corresponding histogram is a bit further from the target histogram. Very large $\sigma_{observation}$ would essentially mean that all the candidate particles are equally likely and we do not gain any information from the observation.

As seen in figure [4] (of experiment 2.6), with high $\sigma_{observation}$ value (= 10), the tracking is not successful. The model loses track of the hand, as it gives significant weight to the particles whose histograms are only slightly matching with the target histogram (of the hand).

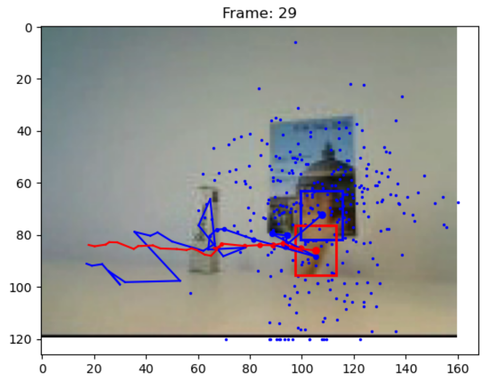
Similar trajectory was observed for experiment 2.12 (please refer figure [4]) , with low $\sigma_{observation}$ value (= 0.001). In this case, in the new frame it may happen that the object being tracked has slightly different histogram (due to brightness change, colour inconsistency *etc.*) so the un-normalized $\pi^{(n)}$ value would be 0 (owing to the precision limit of the machine) (for even the particle with the most similar histogram to the target). So essentially, once we normalize to get new *particles_w* array, all the $\pi^{(n)}$ values would be same. Which is once again akin to not getting any information from the observation.



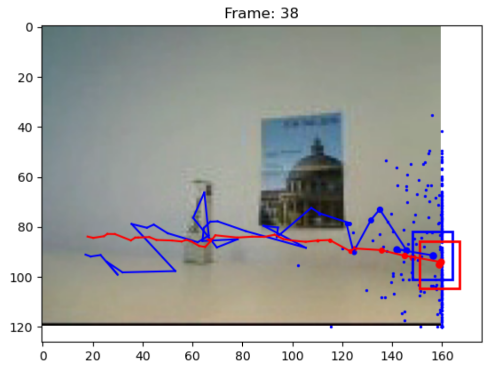
(a)



(b)

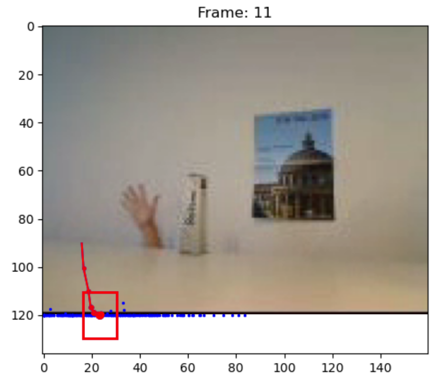


(c)

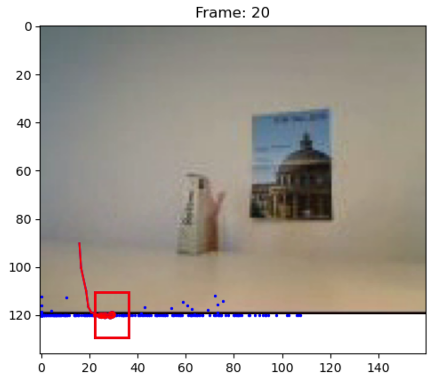


(d)

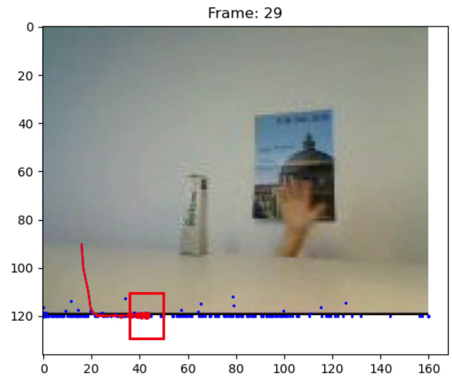
Figure 3: Experiment 2.3 Output (Please refer to table 2 for hyperparameters)



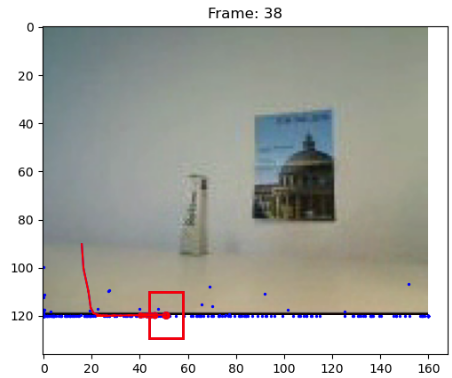
(a)



(b)

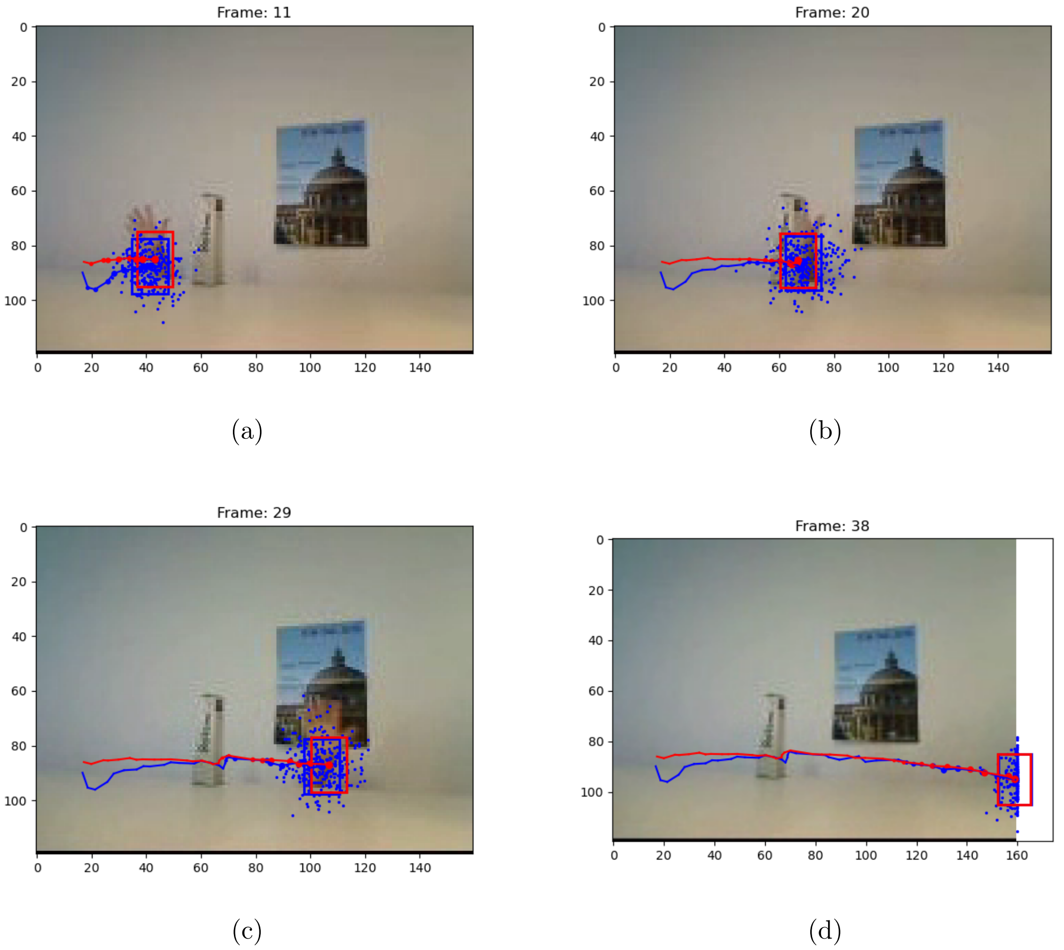


(c)



(d)

Figure 4: Experiment 2.6 Output (Please refer to table 2 for hyperparameters)



(a) (b)

(c) (d)

Figure 5: Experiment 2.4 Output (Please refer to table 2 for hyperparameters)

2.3 video3

Sr.	N_{bins}	model	α	$N_{particles}$	$\sigma_{position}$	$\sigma_{velocity}$	V_0	$\sigma_{observation}$
3.1	16	$M_{ConstVelocity}$	0.1	300	5	1	[1, 10]	0.1
3.2	16	$M_{ConstVelocity}$	0.1	300	15	1	[1, 0]	0.1
3.3	16	$M_{ConstVelocity}$	0.1	300	15	1	[1, 0]	10
3.4	16	$M_{ConstVelocity}$	0.1	300	15	5	[1, 0]	0.1
3.5	16	$M_{ConstVelocity}$	0.1	300	15	3	[1, 0]	0.1
3.6	16	$M_{ConstVelocity}$	0.1	300	25	3	[1, 0]	0.1
3.7	16	$M_{ConstVelocity}$	0.1	300	15	1	[1, 0]	0.05
3.8	16	$M_{NoMotion}$	0.1	300	15	-	-	0.1
3.9	16	$M_{ConstVelocity}$	0.1	50	15	3	[1, 0]	0.1
3.10	16	$M_{ConstVelocity}$	0.1	300	15	1	[1, 0]	0.001
3.11	16	$M_{ConstVelocity}$	0.1	200	10	1	[1, 0]	0.1
3.12	16	$M_{ConstVelocity}$	0.1	300	0.1	0.1	[1, 0]	0.1
3.13	16	$M_{ConstVelocity}$	0.1	300	10	1	[1, 0]	0.001

Table 3: Experiment 3 hyperparameter values

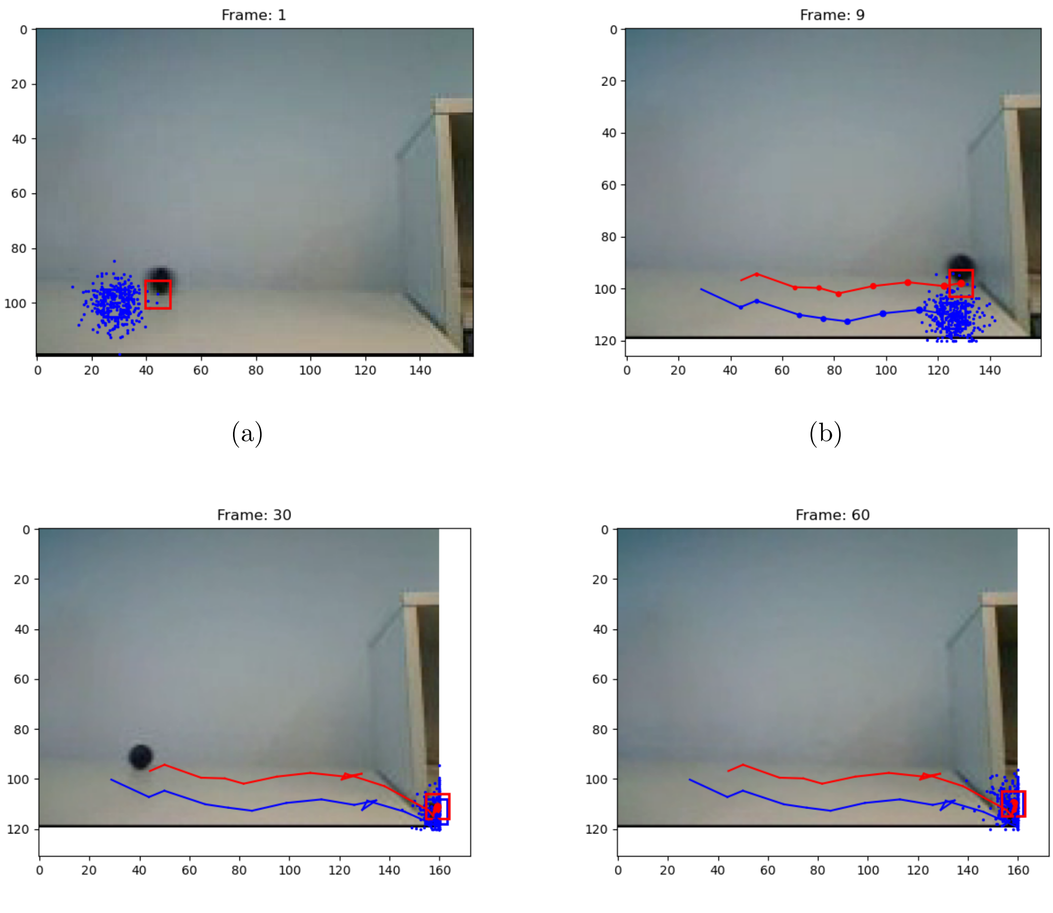
The best hyperparameter values for the video2 did not work for this video. It was due to $\sigma_{position}$ being too small (= 5) (please refer experiment 3.1). As seen in figure [6(c) and (d)], the model loses track of the ball and gets stuck in the similar looking corner instead. It is due to the fact that $\sigma_{position}$ was too small and when the ball suddenly changed its directions near the right end, the degree of exploration was not enough to cover the "unexpected" new position of the ball and therefore, the model got stuck to the similar looking patch in the vicinity (the bottom right corner part).

Also, the initial velocity of $[1, 10]$ is not correct as the ball has no motion in y-direction. But this had little effect, because most of the y-values that exceeded y-max value due to this setting, were clipped to the lower edge of the frame and thus those particles stayed close to the target location anyways.

- ***What is the effect of using a constant velocity motion model?*** Comparing experiment 3.8 (No Motion Model, figure[8]) with 3.11 (Constant Velocity Model, figure [7]), both the red (posterior) trajectories look similar however the prior one (blue) for the experiment 3.11 is relatively unstable. The unstable part starts once the ball changes its direction, and it takes some time for the velocity to be re-adjusted (which happens only due to noise and probability adjustments after making the observations).
- ***What is the effect of assuming decreased/increased system noise?***

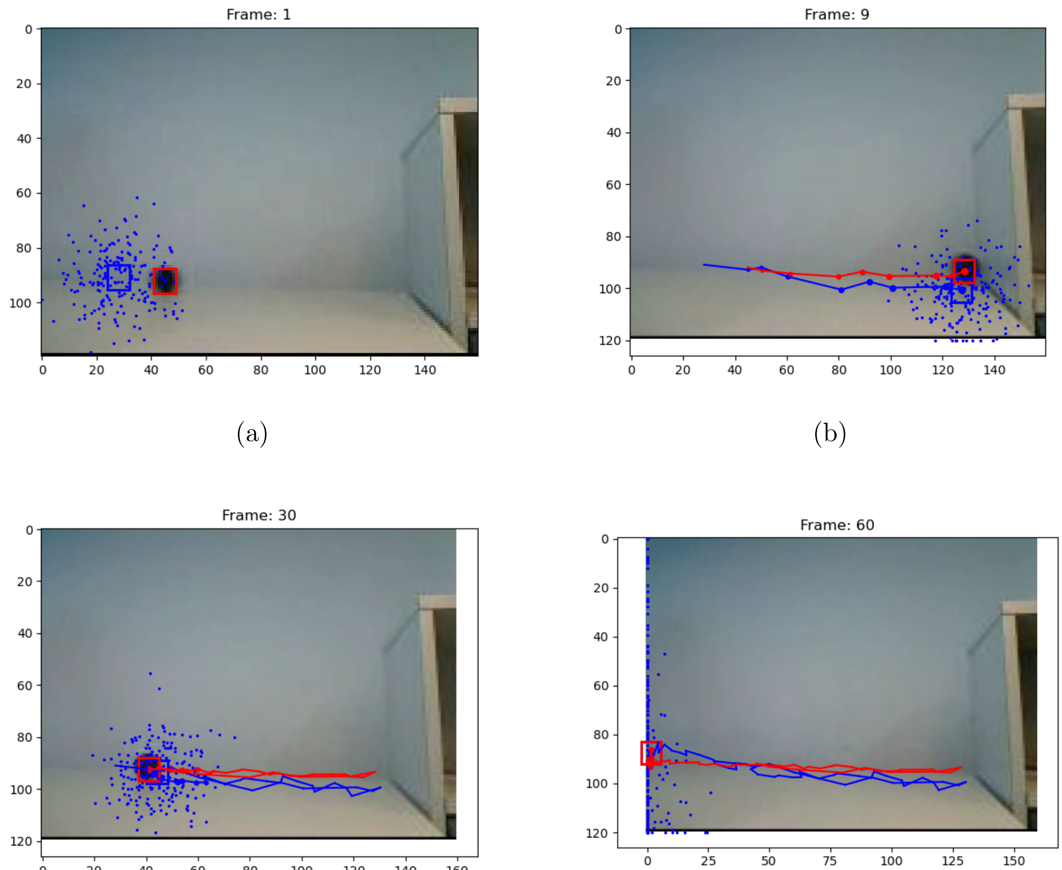
The effect is similar to as seen for video2, with increased noise (experiment 3.6) the blue trajectory was almost random and with reduced system noise (experiment 3.12) the model was not able to follow the ball and was drifting slowly in the initial direction about the initial position.

- ***What is the effect of assuming decreased/increased measurement noise?***
  Again, similar to video2, with increased system noise (experiment 3.3) and decreased system noise (experiment 3.13), the tracking was not successful. It was seemingly randomly evolving as the frames progressed.



(a) (b) (c) (d)

Figure 6: Experiment 3.1 Output (Please refer to table 3 for hyperparameters)



(a) (b) (c) (d)

Figure 7: Experiment 3.11 Output (Please refer to table 3 for hyperparameters)

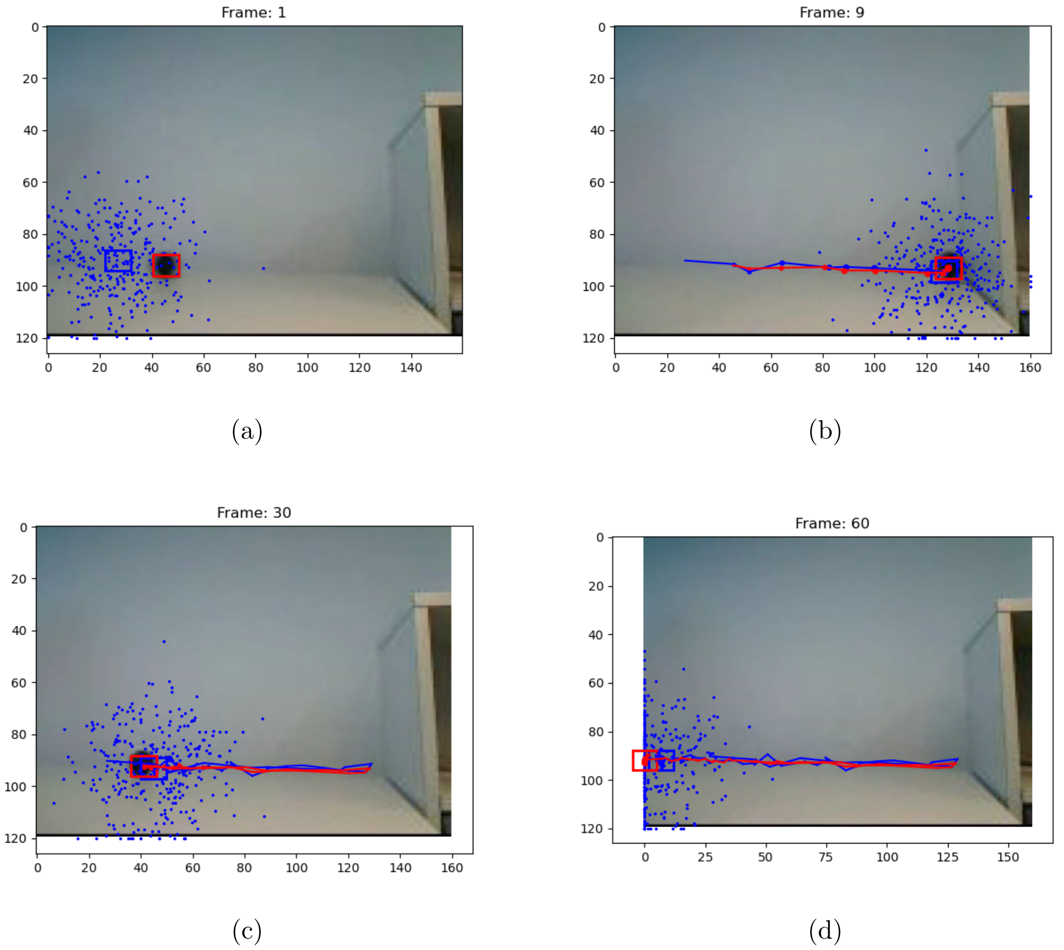


Figure 8: Experiment 3.8 Output (Please refer to table 3 for hyperparameters)

2.4 Effects of other hyperparameters

- ***What is the effect of using more or fewer particles?***
  Using fewer particles means that the sample particles drawn for exploration will not fully represent the underlying distribution, and thus the mean state will be prone to shifting randomly. Using more particles means the particles sampled will represent the underlying distribution and thus the tracking would be stable, however using more particles has an obvious increase in computational cost.
- ***What is the effect of using more or fewer bins in the histogram color model?***
  Using fewer bins means models reduced capacity to distinguish between different patches, because a lot of colour variation would map to a reduced set of features. Using too many bins would result in increased sensitivity to colour and intensity change, and it may hapen that the feature vectors thus formed have large distance for even similar looking patches with slight variation. Therefore, the number of bins should not be either too high or low.
- ***What is the advantage/disadvantage of allowing appearance model updating?***

  Model updating helps in setting the prior such that the exploration yields the actual state with high probability. With correct model configurations, the tracking can be very smooth and robust. However if the model is not correctly configured, it would result in poor performance.