# Computer Vision

# Assignment 9: Condensation Tracker

*Student: Ivan Alberico*

## 1. CONDENSATION Tracker based on color histograms

The first step in the implementation of the CONDENSATION tracker was about computing the function `color_histograms.m`, which returns the normalized color histograms of the pixels inside a specified bounding box (the bounding-boxes represent the objects to be tracked, and the initial one is drawn by the user). This was done through the Matlab function `imhist`, applied separately to the 3 channels of the bounding box (R, G and B values). The 3 obtained histograms are then stacked together in a single vector and then normalized with respect to their sum.

After having computed the color histograms, next step was deriving the matrix A of the system model, which is used in the prediction step of the algorithm. For this assignment, two prediction models are considered, namely one in which we do not have motion and one in which we assume constant velocity of the object. Being $s_t$ the state of the bounding-box at time t, $A$ the matrix defining the deterministic component of the model, $s_{t-1}$ and $w_{t-1}$ respectively the state of the bounding-box and the noise at the previous step, then the evolution of the model is represented by the following equation:

$$s_t^{(n)} = A\, s'^{(n)}_{t-1} + w^{(n)}_{t-1}$$

Let us consider first the case of the no-motion model. In this case, the state is given only by the position of the samples, namely $s = [x, y]^T$, and we take into account only the position noise $w = [\sigma_P, \sigma_P]^T$ on the two components x and y. Since we expect the particles not to move, matrix A is assumed to be the 2x2 identity matrix, and the system model is described as in the following:

$$A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \rightarrow \quad \begin{bmatrix} x_t^{(n)} \\ y_t^{(n)} \end{bmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{bmatrix} x'^{(n)}_{t-1} \\ y'^{(n)}_{t-1} \end{bmatrix} + \begin{bmatrix} \sigma_P{}^{(n)}_{t-1} \\ \sigma_P{}^{(n)}_{t-1} \end{bmatrix} \quad \rightarrow \quad \begin{cases} x_t^{(n)} = x'^{(n)}_{t-1} + \sigma_P{}^{(n)}_{t-1} \\ y_t^{(n)} = y'^{(n)}_{t-1} + \sigma_P{}^{(n)}_{t-1} \end{cases}$$

In the case of the constant velocity model, instead, the state of each bounding-box is expressed by 4 components, namely the 2 components of the samples' position and the 2 components of their respective velocities, as in the following $s = [x, y, \dot{x}, \dot{y}]^T$. For the noise, we take into account two different terms respectively for the position and the velocity, having the following expression $w = [\sigma_P, \sigma_P, \sigma_V, \sigma_V]^T$. In this case, since we expect the particles to move with constant velocity, we have to set their velocity not to change throughout the evolution of the system (since it is constant), while we have to set their position to change in the following way:

$$x_{t+1} = x_t + \dot{x}_t dt + noise, \qquad \text{with } \dot{x}_t = \frac{dx_t}{dt} = v$$

Hence, the system model was described in the following way:

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

$$\begin{bmatrix} x_t^{(n)} \\ y_t^{(n)} \\ \dot{x}_t^{(n)} \\ \dot{y}_t^{(n)} \end{bmatrix} = \begin{pmatrix} 1 & 0 & dt & 0 \\ 0 & 1 & 0 & dt \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{bmatrix} {x'}_{t-1}^{(n)} \\ {y'}_{t-1}^{(n)} \\ {\dot{x}'}_{t-1}^{(n)} \\ \dot{y}_{t-1}^{(n)} \end{bmatrix} + \begin{bmatrix} {\sigma_P}_{t-1}^{(n)} \\ {\sigma_P}_{t-1}^{(n)} \\ {\sigma_V}_{t-1}^{(n)} \\ {\sigma_V}_{t-1}^{(n)} \end{bmatrix} \quad \rightarrow \quad \begin{cases} x_t^{(n)} = {x'}_{t-1}^{(n)} + {\dot{x}'}_{t-1}^{(n)} dt + {\sigma_P}_{t-1}^{(n)} \\ y_t^{(n)} = {y'}_{t-1}^{(n)} + {\dot{y}'}_{t-1}^{(n)} dt + {\sigma_P}_{t-1}^{(n)} \\ \dot{x}_t^{(n)} = {\dot{x}'}_{t-1}^{(n)} + {\sigma_V}_{t-1}^{(n)} \\ \dot{y}_t^{(n)} = {\dot{y}'}_{t-1}^{(n)} + {\sigma_V}_{t-1}^{(n)} \end{cases}$$

In this specific case, I assumed $dt = 1$, since I consider a time step to be the time between two consecutive frames of the video. In this way, velocity is expressed in pixels per frame (PPF). Alternatively, I could also express the velocity in pixels per seconds by considering each time step to be equal to the total length of the video (expressed in seconds) divided by the total number of frames.

Once the system model is defined, the propagation of the particles was done through the function `propagate.m`, which computes the new state of the particles according to the model described before. At the beginning, the particles and their respective weights need to be initialized (this was already implemented in the code provided). Furthermore, a check on the particles' position was made in order to make sure that they lie inside the frame. In case they do not, I assigned to those particles the limit values of our frame, both in height and width.

After the prediction step there is the update step, in which new measurements are used to update the predictions by re-weighting each particle. The new observation are taken by computing the color histogram for all particles in the bounding-box and the weights are obtained from the chi-squared distance between the particle color histogram and the target color histogram (this latter is updated at each iteration as a convex combination of the old color histogram and the mean state color histogram, computed with the function `estimate.m`). The similarity between the two color histograms, and hence the weights of the particles, are assigned according to the following function:

$$\pi^{(n)} = \frac{1}{\sqrt{2\pi}\sigma} exp\left(-\frac{\chi^2\left(CH_{old}, CH_{target}\right)^2}{2\sigma^2}\right)$$

The particles are then resampled in the function `resample.m`, with the low-variance sampling method which was already implemented in Assignment 3 (particle filters).

## 2. Experiments

The second part of the assignment was about testing the previously implemented CONDENSATION tracker on three videos, provided in the task description, and see how the performance of the tracker was affected by changing the various parameters. Each time we run the `condensationTracker.m` function, we can see a blue line (with blue particles), which represents the mean state of the priori particles, and a red line (with red particles), which represents the mean state of the posterior particles. I will now analyse the three different videos separately.

### 2.1. Video 1

The first video is the simplest among the three, since it basically shows a hand which is moving on a uniformly colored background. The results I obtained from the default parameters already provided in the coded, are shown below:

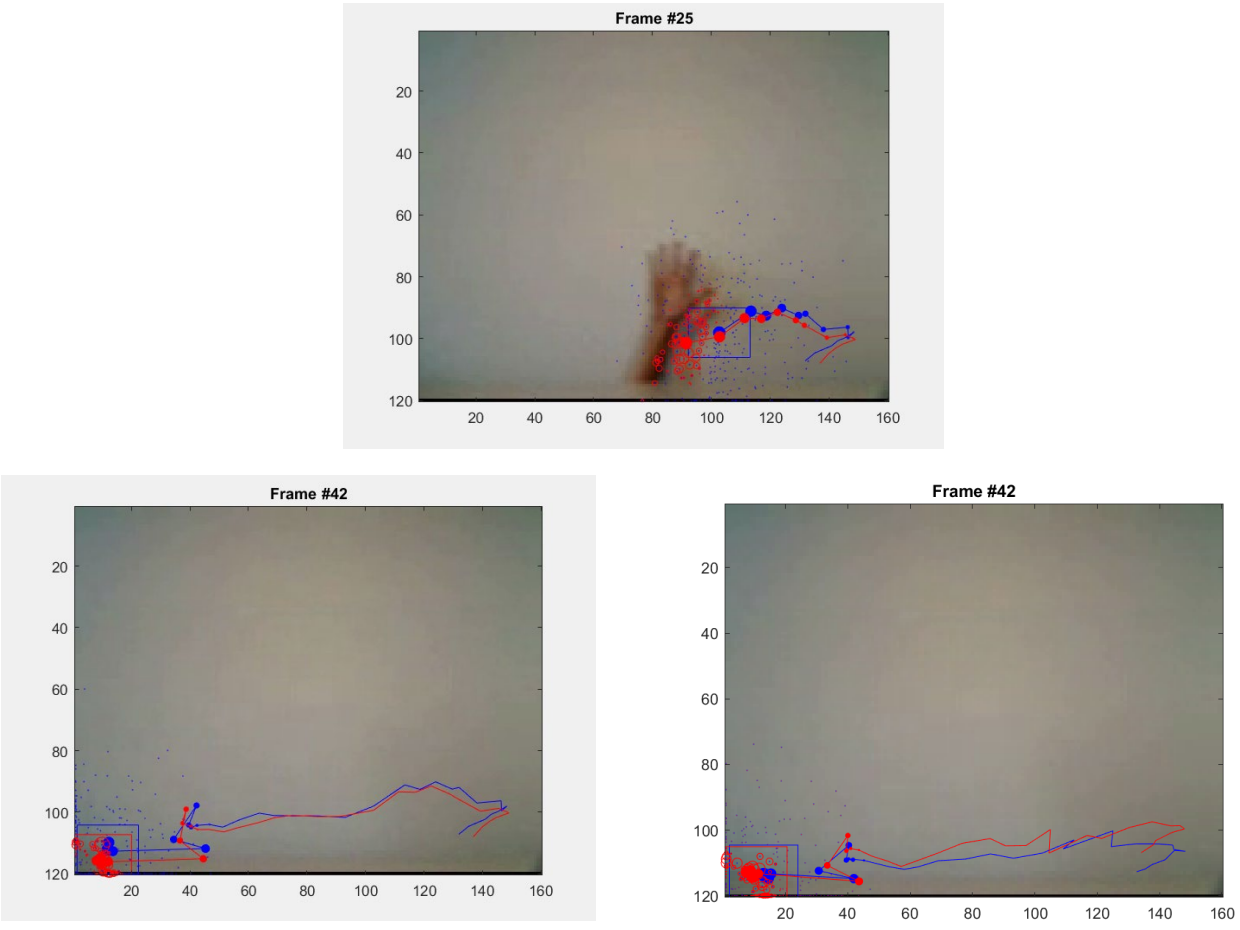


**Figure 1** — CONDENSATION tracker applied to Video1, with both no-motion model (on the left) and constant velocity model (on the right).

The results are satisfactory with both the no-motion model and the constant speed model. What is not totally accurate is that when the hand starts moving, the tracker follows the wrist rather than the hand itself or the fingers. This happens because at the beginning, when we draw the initial bounding-box around the fingers, the area is more shaded with respect to when the hand is moving, because the

lighting is changing throughout the motion. Instead, the lower part of the wrist has a shading which is very similar to that of the fingers in the initial frame.

A first way to solve this issue was increasing the value of $\alpha$ in the target histogram update, since it was initially set to 0. When $\alpha = 0$, it means that we are not updating the target histogram (we are setting it to the previous target histogram), as it can be seen from the formula $CH_{target} = (1 - \alpha)\, CH_{target} + \alpha\, CH_{E[s_t]}$. By increasing the value of $\alpha$ we are updating the target histogram at each iteration and so the model becomes more sensitive to the chromatic changes in the motion (such as different lightings or shadows).

Another way to overcome this issue was setting the initial velocity (hence, using the constant velocity model) in a way such that we push the particles to move in the direction of the hand motion. A very good result was obtained by setting the initial velocity to [-3, -15], since the hand moves from right to left and from bottom to top (the values are taken with respect to a reference frame positioned in the top left corner of the image, as shown in Figure 4).

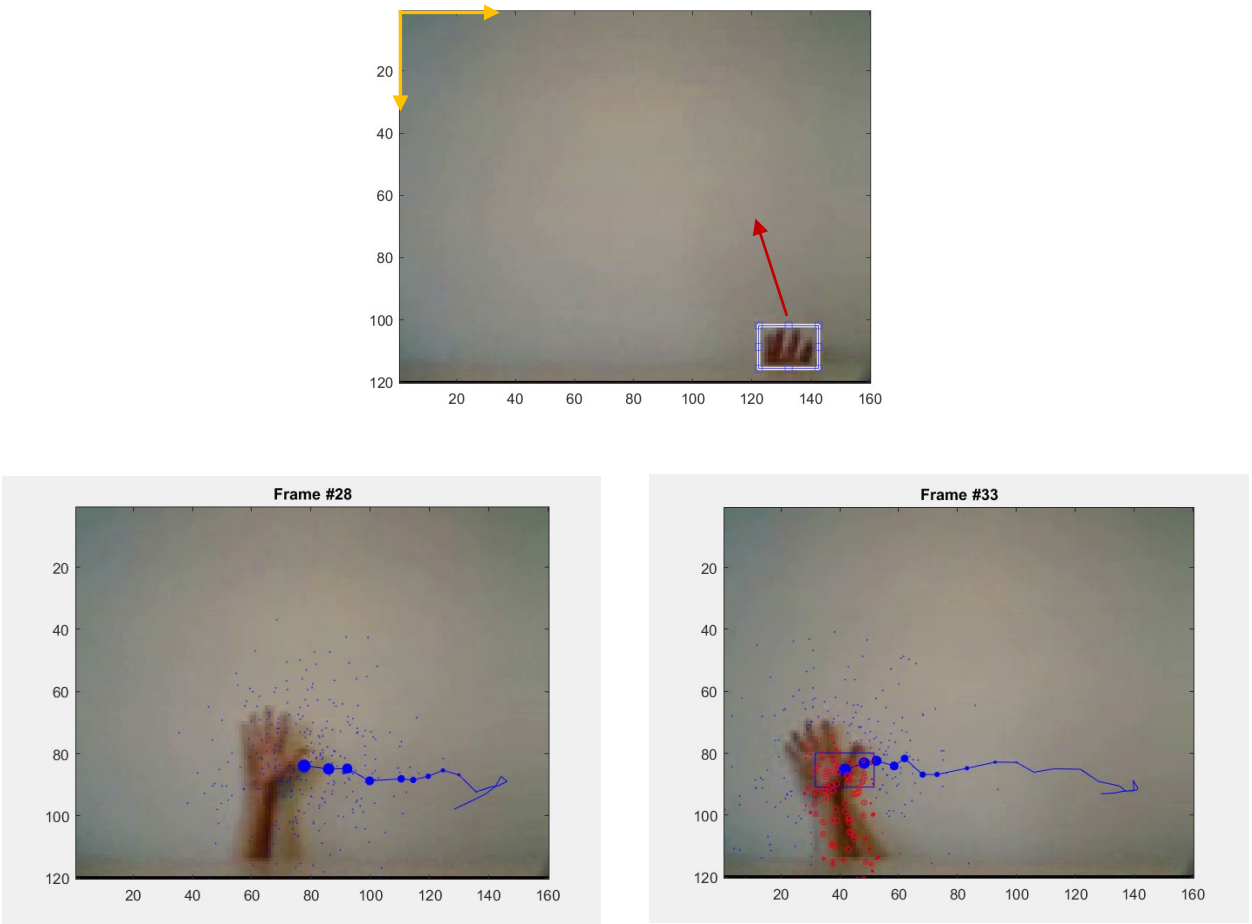


**Figure 2** — CONDENSATION tracker applied to Video1, with constant velocity model, $\alpha = 0.5$ and initial velocity = [-3,-15]

Another remark for this video could be made on the number of bins considered, which somehow tells us how the model is sensitive to the colour differences in the video. A high number of bins gives a more precise result, since it includes a wider variety of colours. On the other hand, a small number of bins could lead to having very similar histograms (since we are not able to differentiate well among the

different colours) but it is indeed more robust to noise. In our case, default value of the number of the bins was equal to 16. However, it can be seen that with this value, as the hand disappears, the particles then move to the area that is closer to the hand in terms of colour histogram (it is some darker region on the left part of the background). If, instead, I used a smaller number of bins, say 4, this did not happen. With 4 bins, as the hand disappears, the particles stay on the bottom of the frame (this because there is a tiny black strip at the bottom of the frame, so the particles instead of going somewhere on the white background, they just stay on that strip, since it has a RGB values closer to that of the hand).

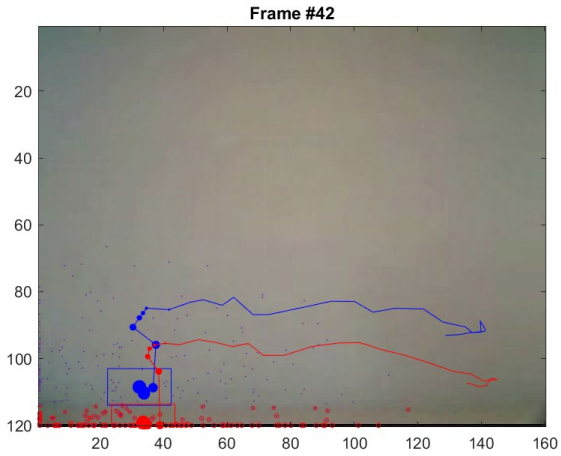


**Figure 3** — CONDENSATION tracker applied to Video1, with constant velocity model and number of bins = 3.

## 2.2. Video 2

The second video is slightly more complicated than the first one. We still have a moving hand, but now the there is some clutter and occlusion.

Both no-motion and constant velocity model worked well for this video and succeeded in tracking the hand.

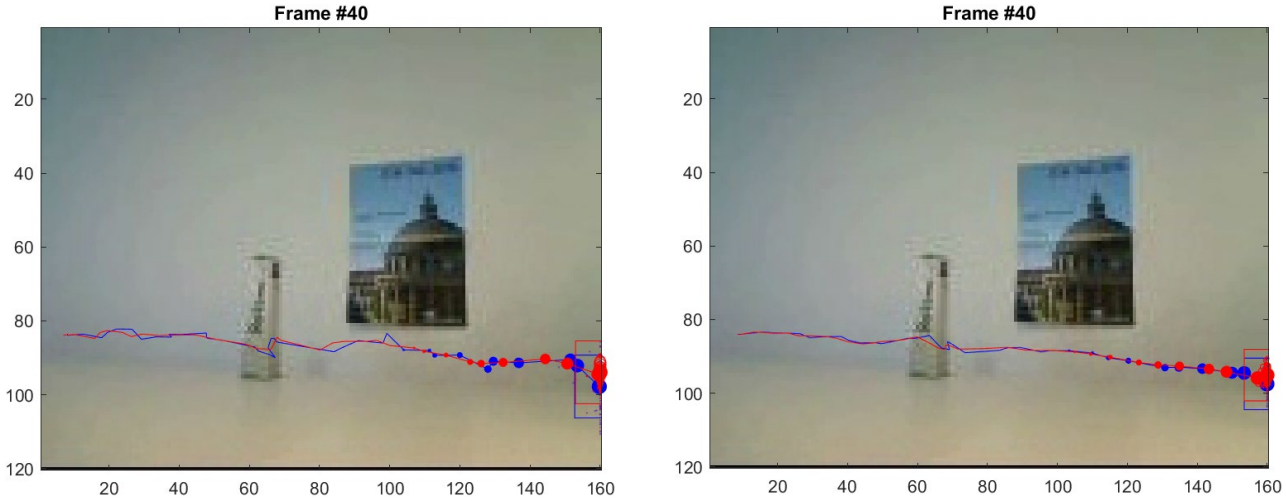


**Figure 4** — CONDENSATION tracker applied to Video2, with no-motion model (on the left) and constant velocity model (on the right)

However, what I was able to observe was that, while the constant velocity model was able to succeed also by varying some of the other parameters, the no-motion model failed when the noise term of the position was very low (for example $\sigma_{position} = 1$), with the particles being stuck on the occlusive object (as shown in *Figure 5*). For this reason, constant velocity model was preferred. The initial velocity was set to be equal to [1 0], since we are not having any vertical motion, and the horizontal motion goes from left to right, and it is quite slow.

Besides the choice of the model, the parameters that played a very important role in this tracking part, especially in the occluded region, were the number of particles and the noise over the position. This because, when the hand in passing through the occluded region, we want the area in which the particles are spread to be the largest as possible, so that when the hand is visible again, the tracker does not get stuck on the occlusive object but keeps following the hand. For small position noise (especially in the no-motion model) and number of particles the tracking fails, as it is shown below:

Increasing the number of particles increased the accuracy of our tracker, however the process became more computationally expensive. On the other hand, if we increase the position noise above a certain threshold, the algorithm could fail since we have the particles to be too much spread in the frame, and so they might follow other paths.

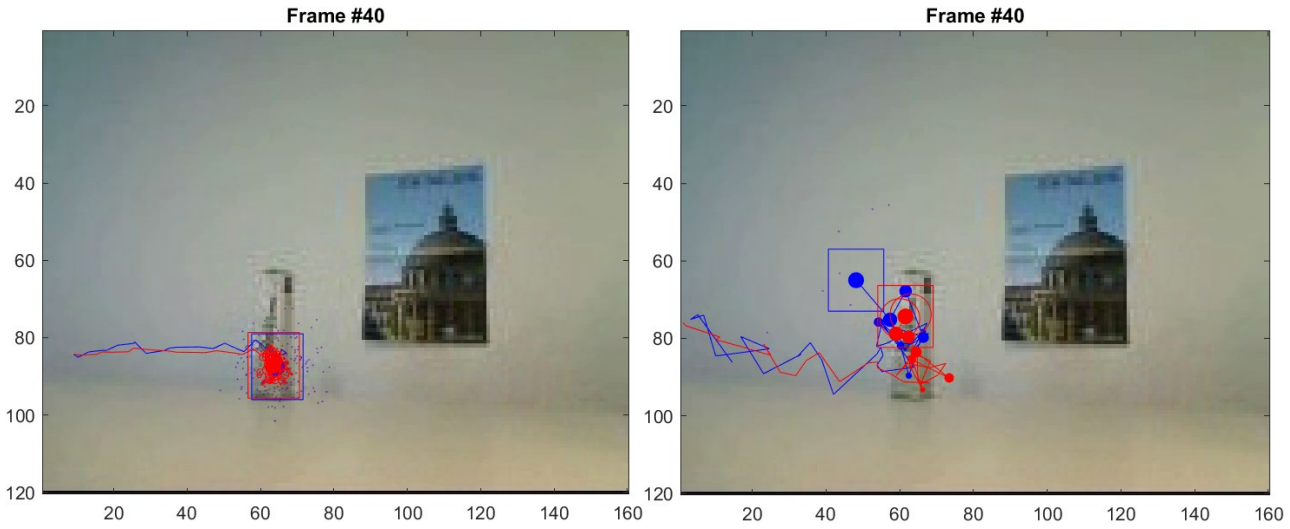


Figure 5 — CONDENSATION tracker applied to Video2, with too small value for the position noise, $\sigma_{position} = 1$, (on the left) and with only 30 particles (on the right), both cases leading to the tracker being stuck on the occlusive object.

Another important parameter to tune was the measurement noise, since it regualtes how the weights are related to the respective particles. For small values of the measurement noise, say $\sigma_{observe} = 0.01$, the particles are again stuck on the occlusive object and do not follow the hand until the end. By decreasing the value even further, for example $\sigma_{observe} = 0.001$, the tracking fails completely and the particles may also disappear or take very random paths. This happens because all the particles have very small weights, so basically none of them is a good candidate to follow.

On the other hand, high values of the measurement noise result in the opposite outcome. In this case, all the particles have high weights, so it is like all of them are good candidates to follow. This ends up in the tracker being completely stuck and not following the hand at all.

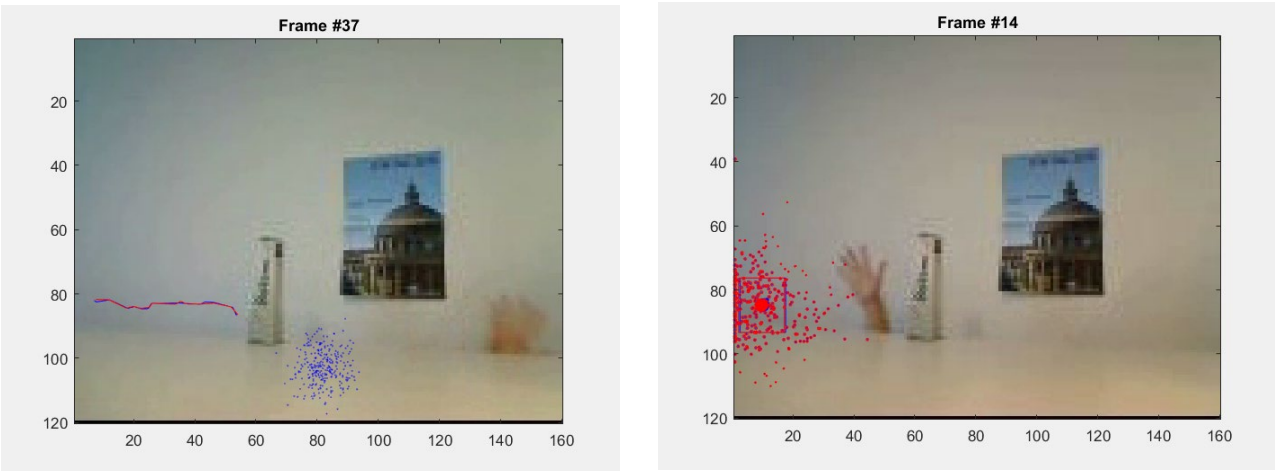


Figure 6 — CONDENSATION tracker applied to Video2, with too small value for the measurement noise, $\sigma_{observe} = 0.001$ (on the left) leading to the failure of the tracker, and with too high value for the measurement noise, $\sigma_{observe} = 1$ (on the right), leading to the particles being stuck at the beginning and not tracking the hand.

The tuning of the other parameters was imporant as well, but did not influence the performance of the algorithm as much as these previous ones did. In this case it was advisable to set the number of bins not too low, since in this setting we have more differences in terms of color, due to the poster on the background as well as the occlusive object. A small number of bins would not allow to differentiate between all these color differences and would end up in the tracker failing its task (it happened sometimes that the particles ended up in the poster).

While choosing the constant velocity model, it is also important (at least in this case) not to choose a very high value for the velocity noise, since this results in having a very non-linear path (as the particles want to explore more directions at every iteration), as this could have a detrimental under some circumstances. This result is shown in *Figure 7*.

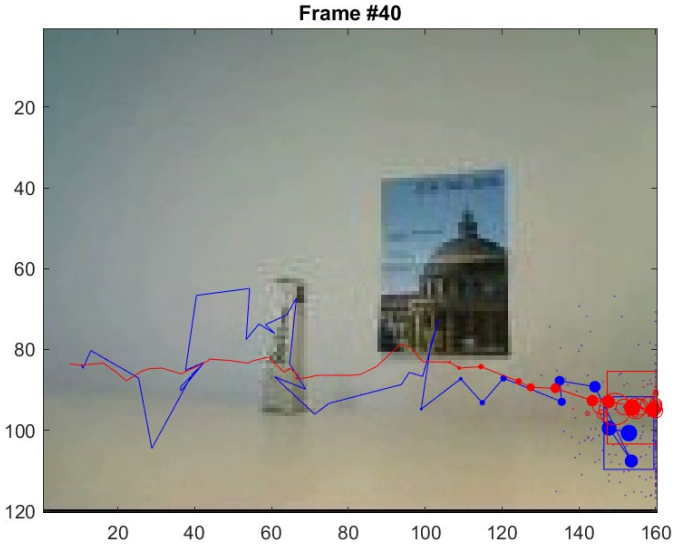


Figure 7 — CONDENSATION tracker applied to Video2, with too high value of the velocity noise ($\sigma_{velocity} = 10$), leading to a very sparse motion of the tracker, due to the particles having the tendency to explore different direction.

## 2.3. Video 3

The third video is different from the other two, since we no longer have a moving hand, but instead we have a ball that bounces on a wall and goes back. For this reason, applying the parameters used for the previous videos (say Video 2) is not completely accurate, since now we have a different setting of motion and colours. This is, in fact, one of the main disadvantage of the CONDENSATION tracker, such as it depends a lot on the scene we are considering, and it always needs to be tuned accordingly.

In fact, by using the best parameters chosen for Video2, I was not able to track the ball in the desired way. What happened was either that the particles stopped following the ball before reaching the wall, or that, after the impact with the wall, they kept being updated to the right and ended up in the bottom right corner.

In this case, the constant velocity model might not always be the optimal choice. This is because when the ball touches the wall, it changes direction and moreover it decreases its velocity after the impact (hence, velocity is no longer constant). In fact, in the moment of the impact, the particles kept being updated to the right, while the object was now going in the opposite direction. In general, for this third video, the no-motion model was preferred.

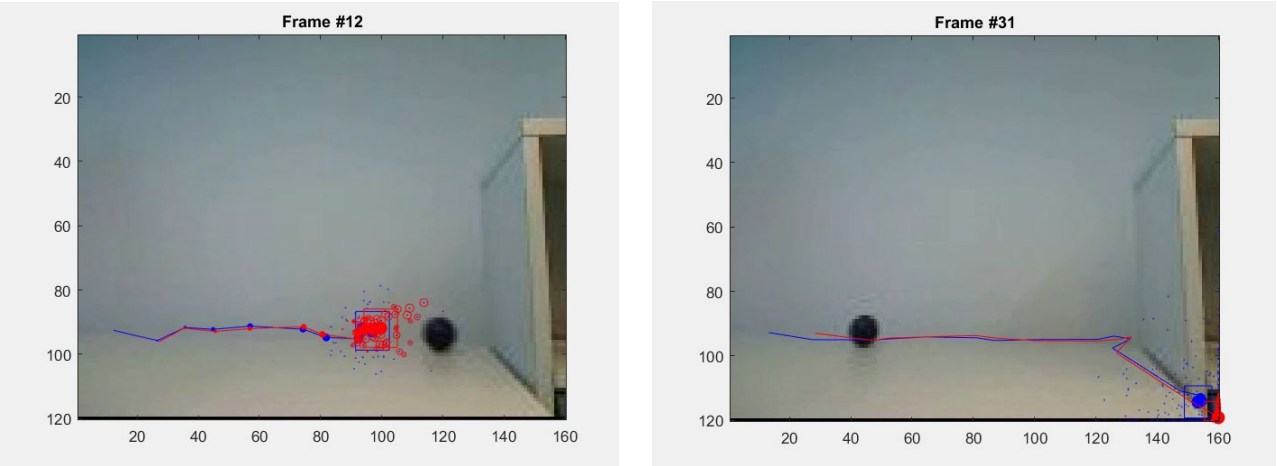


**Figure 8** — CONDENSATION tracker applied to Video3, with parameters of the previous videos. In the first image (on the left) the particles stop tracking the ball before hitting the wall, while in the second image (on the right), the end up in the bottom right corner.

In this case choosing a high value for the position noise solved the problem. In this way, being the particles spread on a wider area, they were able to trace back to the ball also after the collision with the wall. As far as the measurement noise is concerned, the same reasoning for Video 2 applies also now. At the same time, a high number of particles help also improving the accuracy of the tracker.

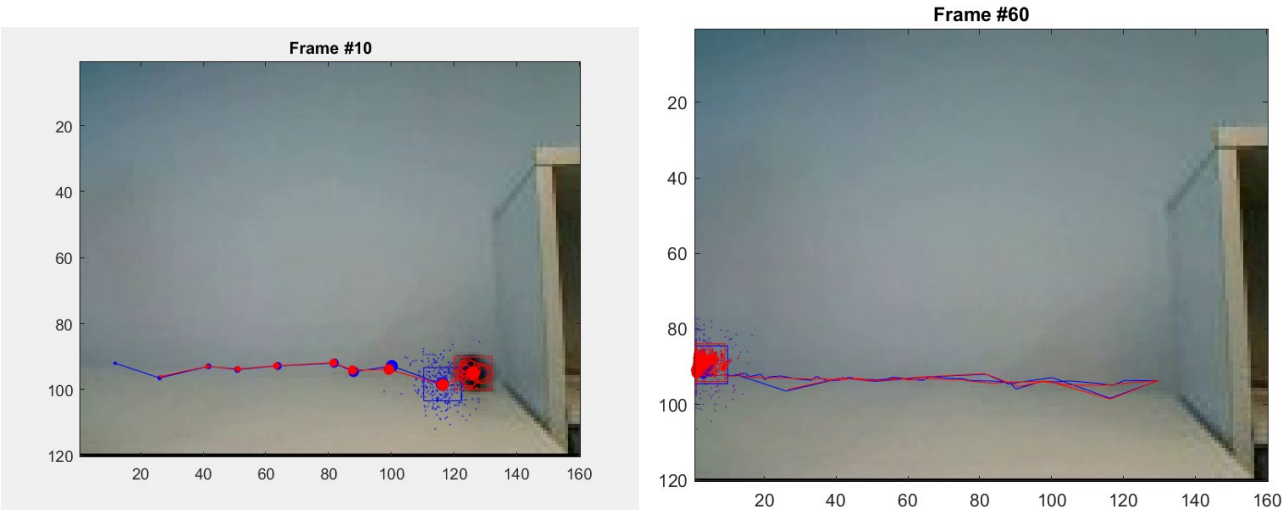


**Figure 9** — CONDENSATION tracker applied to Video3, with higher values of position noise and number of particles.

The number of bins also played an important role in tracking the ball. In fact, by choosing a small number of bins, the tracker is very likely to fail in its task, since it is very likely for the particles to end up in the bottom right corner. This happens due to the ball and the bottom right corner having similar RGB values (they are both black), and so not having too many bins for the histograms would not allow to properly differentiate between the two cases. A reasonable number of bins would be for example 15.

For all the 3 videos, a good compromise was setting $\alpha = 0.5$, as it happened to be a reasonable trade-off between updating the target histogram with the mean state histogram and not updating it at all. In this way, by allowing appearance model updating, the target histogram is not constant throughout the evolution of the model, but it adapts to the changes it encounters in the meanwhile.

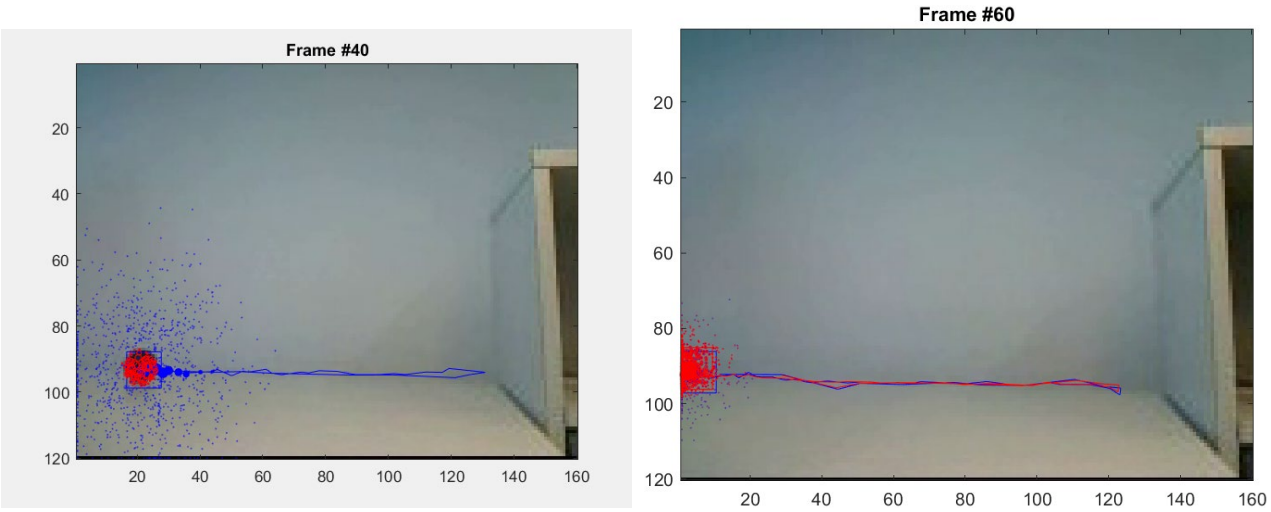


**Figure 10** — CONDENSATION tracker applied to Video3, with no-motion model and using 1000 particles.

## 3. Try your own video

In addition to the three videos already analysed before, I tested the implemented tracker also on other videos found on the Internet. I will now present the results of two videos in particular.

The first one shows a bird which is flying in the sky. At first glance, it may seem a very easy situation for the tracking algorithm, since the bird is flying over a blue background (hence they have very different RGB values and so colour histograms). However, problems arose from the fact that the bird and the stone from where it took off had very similar colours, as well as the blurred stones in the background. In fact, most of the times the tracker ended up having the particles stuck over the stones rather than the bird.

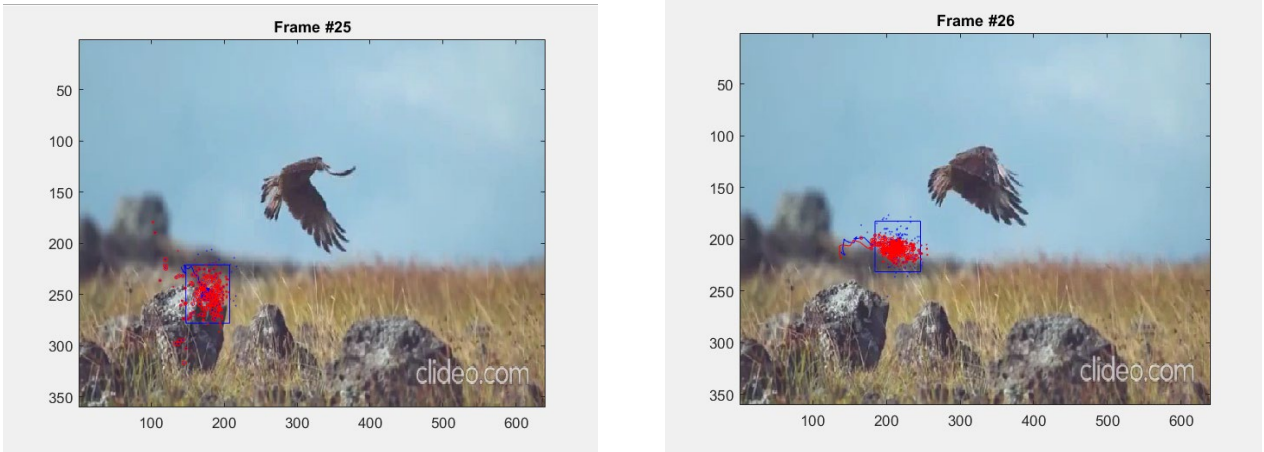


**Figure 11** — CONDENSATION tracker applied to myOwnVideo1, which fails to track the bird's motion

The problem was solved by increasing the number of bins for the histograms and using the constant velocity model with an initial velocity in the direction of the bird's flight.

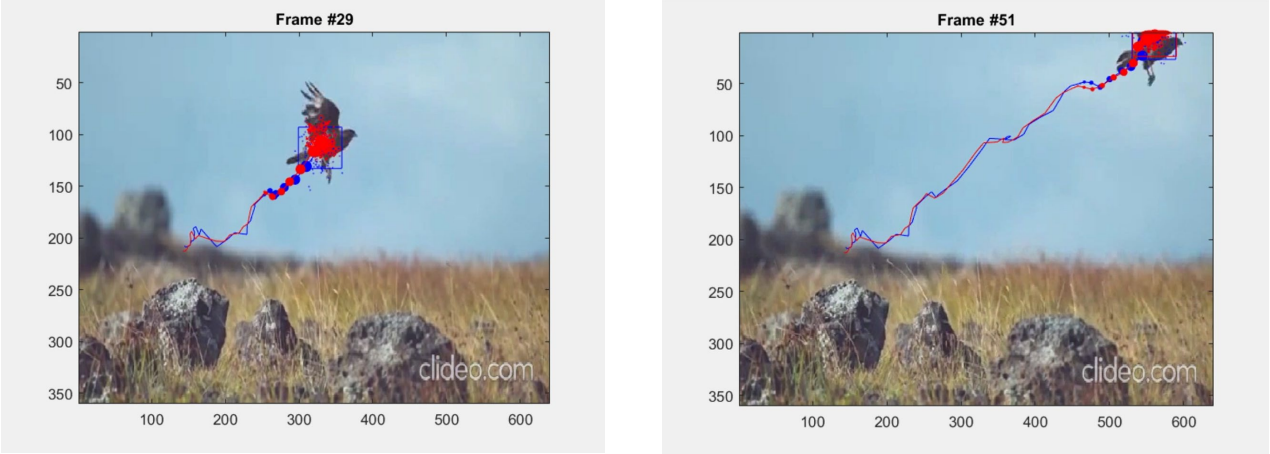


**Figure 12** — CONDENSATION tracker applied to myOwnVideo1, which correctly tracks the bird's flight.

The next video I am going to show was still taken from the Internet, and it shows a crowded scene of the traffic in Hanoi, Vietnam. The purpose was to test the tracker in a more crowded environment.


**Figure 13** — CONDENSATION tracker applied to myOwnVideo3, which correctly tracks black motorcycle in the crowd.

The results obtained were very satisfying. The tracker was able to follow the selected motorcycle even after a first occlusion while crossing with another motorcycle (in the second and third images), but also later on when it is surrounded by a lot more of them.

The overall performance of the CONDENSATION tracker achieves good results, however it has a lot of downsides as well. The biggest issue, in my opinion, is that it does not work in general but it needs to be tuned for the specific cases in order to work efficiently. Ways to improve the performance would be taking into account further motion models or consider other state variables other than the colour histograms (or maybe combining them together), like, for example, taking into account the shape of the object we want to track (shape context descriptors).