Computer Vision Assignment 6

Nicolas Wicki

December 2021

CONDENSATION Tracker Based On Color Histograms

1. Color histogram To compute the normalized RGB histogram, we slice the frame according to the provided x and y coordinates of the bounding box. Here, we need to make sure to switch the ordering of the coordinates as array indexing starts with the height (row), which is represented by the y coordinates, and follows with the width (column) represented by the x coordinates. We want the normalized histogram, which can be computed using the `numpy` function `histogram()`. We have to provide the amount of bins and afterwards need to divide each entry by the total sum of all histograms.

2. Derive matrix A Model 0 only uses the position of each particle to predict the motion (which is actually no motion at all). The matrix A is then given by the 2×2 identity matrix in two dimensions. Model 1 uses position and a constant velocity per particle, which is represented by a vector of length four: $(x, y, \dot{x}, \dot{y})$. To compute the prediction of each particle, the matrix A is of size 4×4:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix} \tag{1}$$

This matrix basically adds the displacement by the velocity to each coordinate and copies the velocity as it stays constant in our model.

3. Propagation To propagate our particles, we make the distinction between model 0 and 1. With model 0, we only add Gaussian noise scaled by the standard deviation of the particles. With model 1, we first update the particles by a displacement given by the velocity. Then, we also add Gaussian noise scaled by the standard deviation of the particle velocity to the current particle velocity. In the end, we need to make sure the particles stay inside the frame by clamping the values to the borders of the frame.

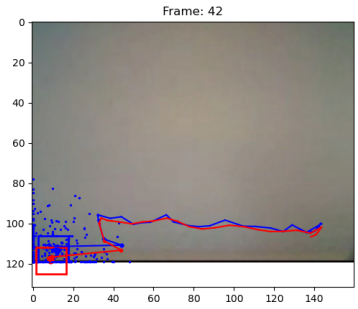

Figure 1: video1.avi, model 0

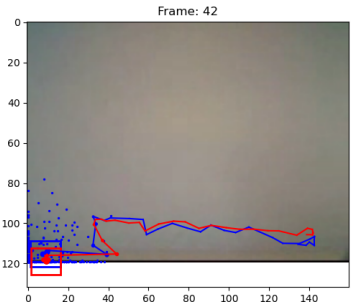

Figure 2: video1.avi, model 1

4. Observation For each particle, we compute the bounding box and then use our previously defined function to compute the color histogram. We compute the chi^2 cost by applying the corresponding provided function on the computed histogram and the target histogram. Putting the result into the probability density function of the Gaussian distribution gives us a probability for each particle. Of course, we have to normalize the probabilities to sum up to 1.

5. Estimation To compute the mean state of our set, we simply sum up over the weighted particles.

6. Resampling The resampling function uses the computed probabilities (the weights of the particles) as a distribution to sample particles from the current state. We return the newly sampled particles with the corresponding normalized probabilities.

Experiments

1. video1.wmv The tracking of the hand with the no motion model works quite good. However, instead of tracking the palm of the hand, it tracks the wrist at some points of the process. Once the hand disappears, the tracking diverges into the corner in the direction of the hand movement. For reference, see Figure 1 and Figure 2. For a failed tracking, see Figure 3 and Figure 4. For a successful tracking, see most of the other results.

2. video2.wmv Increase and decrease are by a factor of three each.

- When compared to the model with no motion, the constant velocity motion model exhibits a more jagged pattern. However, they are almost identical. See Figure 5 and Figure 6.

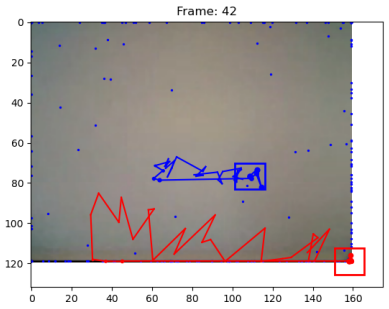

Figure 3: video1.avi, model 0, failed tracking, system noise increase by a factor of 10

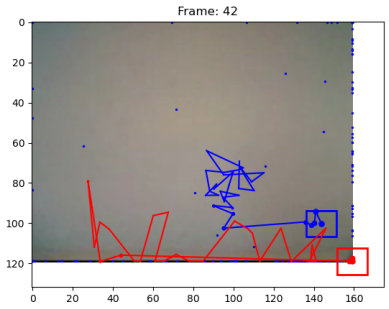

Figure 4: video1.avi, model 1, failed tracking, system noise increase by a factor of 10

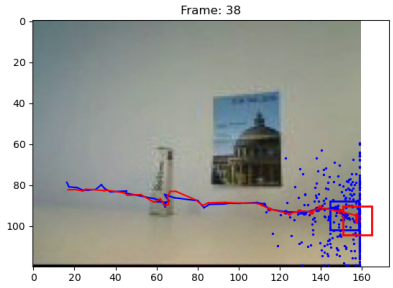

Figure 5: video2.avi, model 0

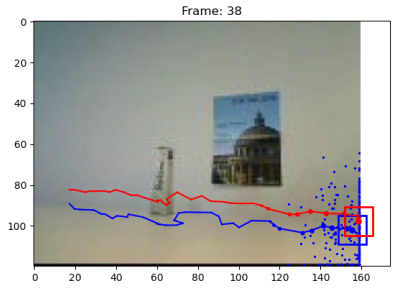

Figure 6: video2.avi, model 1

- Decreasing the system noise provides a stabler result. Increasing the system noise leads to a strongly jagged result, but still plausible tracking of the hand. See Figure 7 and Figure 8.
- Decreasing the measurement noise again stabilizes the tracking. Increasing the measurement noise leads to slightly more jagged tracking. See Figure 9 and Figure 10.

For a failed tracking, see Figure 11 and Figure 12. For a successful tracking, see most of the other results.

3. video3.wmv When using reduced noises and the model with constant velocity, it manages to track the ball at first, but vanishes into the ground shortly after. See Figure 13. However, trying to reproduce this result fails, and the result visible in Figure 14 is the most common result.

- The no motion model tracks the ball quite well until the ball disappears.

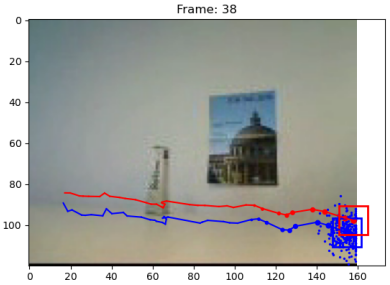

Figure 7: video2.avi, model 1, system noise decrease

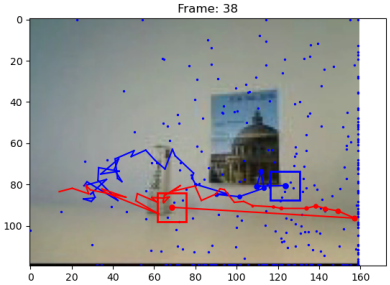

Figure 8: video2.avi, model 1, system noise increase

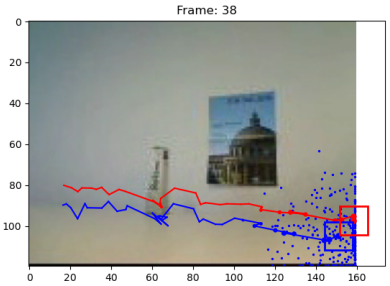

Figure 9: video2.avi, model 1, measure noise decrease

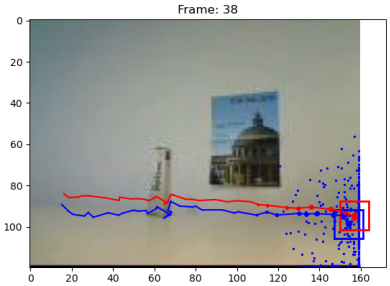

Figure 10: video2.avi, model 1, measure noise increase

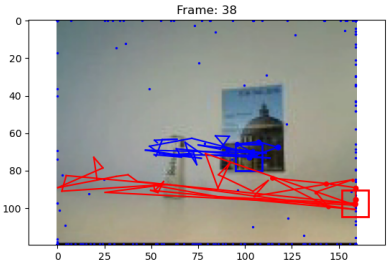

Figure 11: video2.avi, model 0, failed tracking, system noise increase by a factor of 10

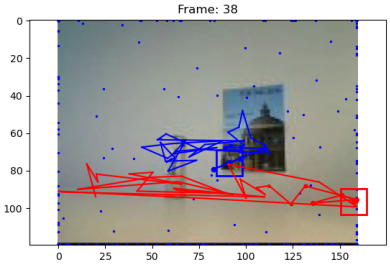

Figure 12: video2.avi, model 1, failed tracking, system noise increase by a factor of 10

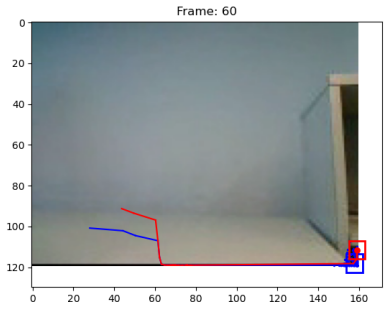

Figure 13: video3.avi, model 1, system and measure noise decreased, anomaly

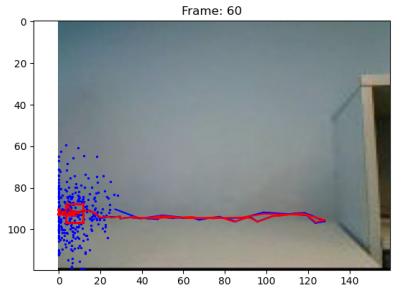

Figure 14: video3.avi, model 1, system and measure noise decreased, common

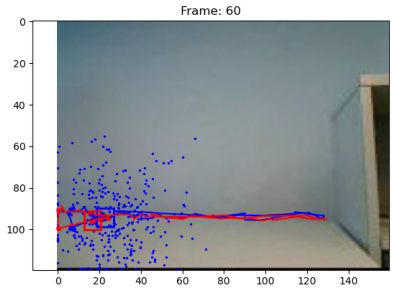

Figure 15: video3.avi, model 0

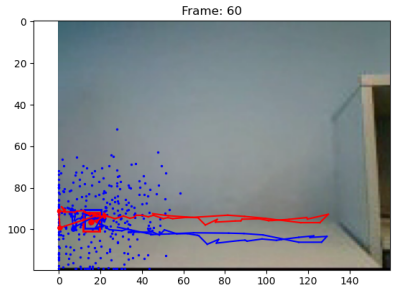

Figure 16: video3.avi, model 1

The constant velocity model performs quite similar, but slightly worse. See Figure 15 and Figure 16.

- Decreasing the system noise leads to slightly worse results than with the provided defaults. Increasing the system noise leads to tracking traces all over the frame and so far the worst performance encountered. See Figure 17 and Figure 18.
- Decreasing the measurement noise provides slightly worse tracking than with the defaults. Increasing the measurement noise works at first quite well, but once the ball changes directions, it ends up in a corner of the frame. See Figure 19 and Figure 20.

For a failed tracking, see Figure 21 and Figure 22. For a successful tracking, see most of the other results.

4. Hand in

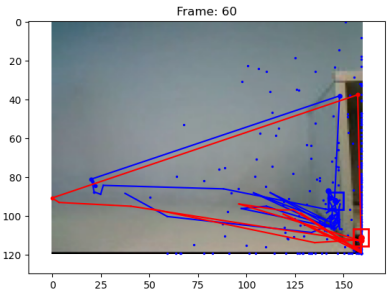

Figure 17: video3.avi, model 1, system noise decrease

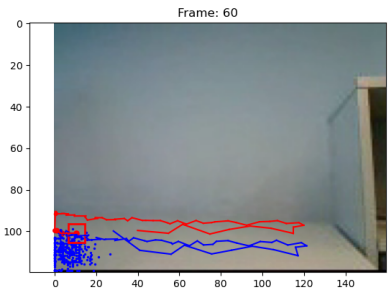

Figure 18: video3.avi, model 1, system noise increase

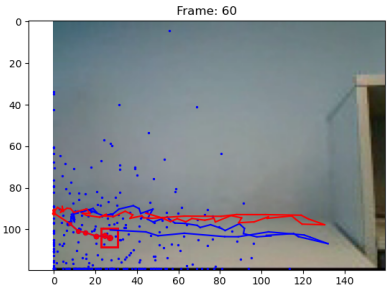

Figure 19: video3.avi, model 1, measure noise decrease

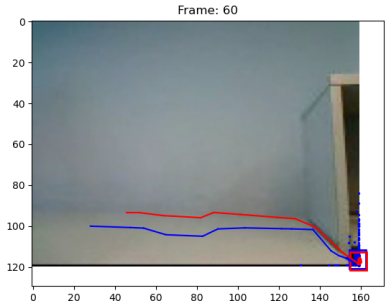

Figure 20: video3.avi, model 1, measure noise increase

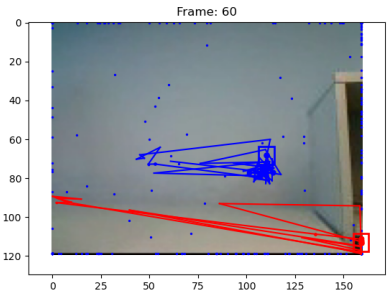

Figure 21: video3.avi, model 0, failed tracking, system noise increase by a factor of 10

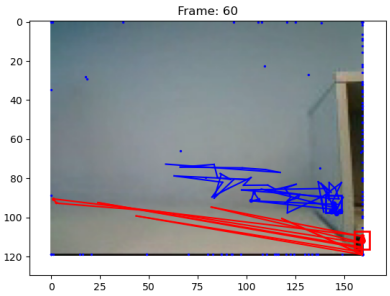

Figure 22: video3.avi, model 1, failed tracking, system noise increase by a factor of 10

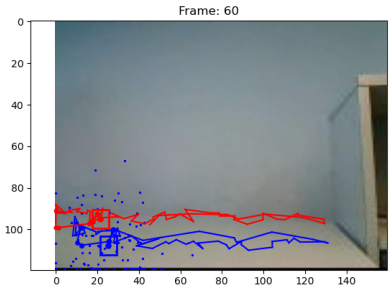

Figure 23: video3.avi, model 1, fewer particles

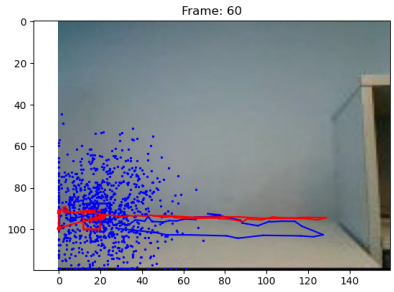

Figure 24: video3.avi, model 1, more particles

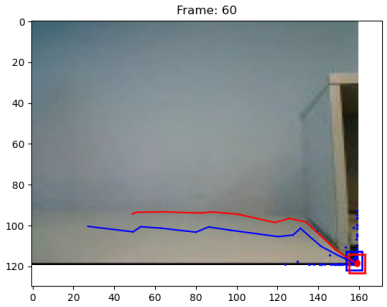

Figure 25: video3.avi, model 1, fewer bins

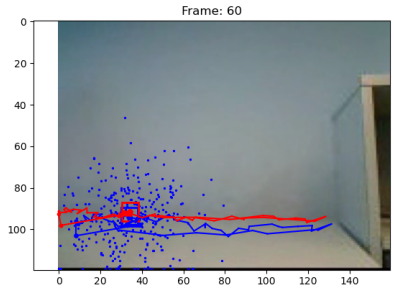

Figure 26: video3.avi, model 1, more bins

- Using fewer particles leads to a jagged pattern in tracking. Using more particles leads to smoother tracking. See Figure 23 and Figure 24.
- Using fewer bins leads to a divergent behavior in tracking, which ends in a corner of the frame. Using more bins has almost no impact. See Figure 25 and Figure 26.
- The difference in adjusting the alpha value is almost negligible. See Figure 27, Figure 28, Figure 29, and Figure 30.

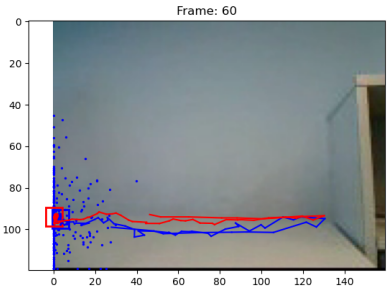

Figure 27: video3.avi, model 1, $\alpha = 1$

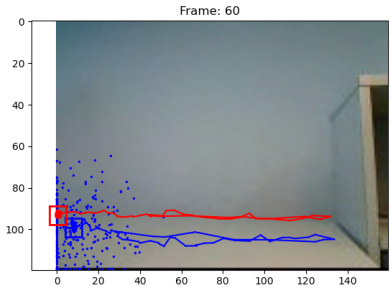

Figure 28: video3.avi, model 1, $\alpha = 0.5$

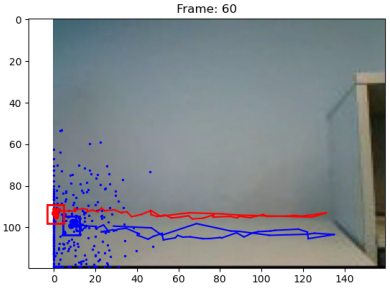

Figure 29: video3.avi, model 1, $\alpha = 0.25$

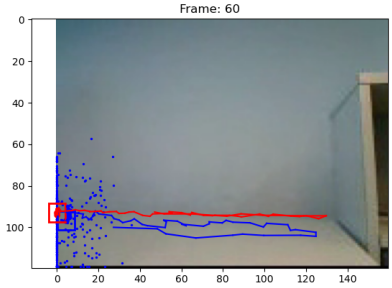

Figure 30: video3.avi, model 1, $\alpha = 0.1$