Exercise 8 - Condensation Tracker

Computer Vision

Alberto Montes (malberto@student.ethz.ch) December 2, 2016

Implementation

The task consist into implement the color histogram computation. The function needs to take into account the bounding box that overlap with the frame, compute the histogram, and then normalize it to sum up 1. The implementation is on Listing 1.

```
function [ hist ] = color_histogram( xMin, yMin, xMax, yMax, frame,
    hist_bin )
%COLOR_HISTOGRAM Perform a color histogram over the image given.
%   The image is given as a frame and the coordinates (x, y) of the left
%   bottom corner of the bounding box as well as its width and height

yMax = min(round(yMax), size(frame, 1));
yMin = max(round(yMin), 1);
xMax = min(round(xMax), size(frame, 2));
xMin = max(round(xMin), 1);

img = frame(yMin:yMax,xMin:xMax,:);
hist = zeros(hist_bin, size(frame, 3));
for i = 1:size(frame, 3)
    [hist(:,i), ~] = histcounts(img(:,:,i), hist_bin, 'BinLimits', [0,
    255],...
        'Normalization', 'probability');
end
hist = reshape(hist, [1, hist_bin * size(frame, 3)]);
```

Listing 1: color_histogram.m

Then to compute the propagation of particles, there are two ways of model it: no movement and constant velocity. For the first one, the particles are represented as x, y coordinates and the matrix A is the following:

$$\begin{bmatrix} x \\ y \end{bmatrix}_t = A \cdot \begin{bmatrix} x \\ y \end{bmatrix}_{t-1} + w_{t-1} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix}_{t-1} \tag{1}$$

In the case the model take into account constant movement, the particles are represented as coordinates and velocities for each coordinates, and the model matrix A is:

$$\begin{bmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{bmatrix}_t = A \cdot \begin{bmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{bmatrix}_{t-1} + w_{t-1} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{bmatrix}_{t-1} \tag{2}$$

This computation implemented on the propagate function is on Listing 2 where the propagation is computed and then a random noise is added.

```
function [ particles ] = propagate( particles, sizeFrame, params )
%PROPAGATE Summary of this function goes here
%   Detailed explanation goes here

nParticles = size(particles, 1);
```

```
if params.model == 0
    A = eye(2);
    noise = normrnd(zeros(nParticles, 2), params.sigma_position);
elseif params.model == 1
    A = eye(4);
    A(3) = 1; A(8) = 1;
    noise = [normrnd(zeros(nParticles, 2), params.sigma_position),...
            normrnd(zeros(nParticles, 2), params.sigma_velocity)];
end

particles = particles * A + noise;

% Correct the particles going out of the image frame
particles = max(particles, 0);
particles(:,1) = min(particles(:,1), sizeFrame(2));
particles(:,2) = min(particles(:,2), sizeFrame(1));
```

Listing 2: propagate.m

Then, the particles needs to be observed, where for each particle, the surrounding bounding box is extracted and its color histogram is computed. Then for each particle, a weight is computed following a Gaussian and the chi squared distance of the original histogram and the one computed for each particle. The implementation is on Listing 3.

```
function [ particles_w ] = observe( particles, frame, H, W, hist_bin,...
    hist_target, sigma_observe )
%OBSERVE Tihs function make observations
%   Compute for all particles its color histogram describing the
    bounding
%   box defined by the center of the particle and W and H.

numParticles = size(particles, 1);
particles_w = zeros(numParticles, 1);
for i = 1:numParticles
    xy = particles(i, 1:2);
    xMin = xy(1) - W/2;
    xMax = xy(1) + W/2;
    yMin = xy(2) - H/2;
    yMax = xy(2) + H/2;

    max(frame);
    min(frame);
    hist = color_histogram(xMin, yMin, xMax, yMax, frame, hist_bin);

    particles_w(i) = 1 / (sqrt(2*pi) * sigma_observe) * ...
        exp( -(chi2_cost(hist, hist_target))^2 / (2 * sigma_observe^2));

end

% Normalize particle weights
particles_w = particles_w / sum(particles_w);
```

Listing 3: observe.m

With all the particle and its respective weights, it must be estimated the position of the object to track computing a weighted mean along all the particles coordinates and velocities to obtain an estimation for each time step.

```
function [ meanState ] = estimate( particles, particles_w )
%ESTIMATE Estimate the mean state of the given particles and their
    weights.
```

```
%   Detailed explanation goes here
meanState = sum(particles .* repmat(particles_w, 1, size(particles, 2)),
    1);
```

Listing 4: estimate.m

Finally a resample is performed to keep the particles with higher weight and remove the ones less probable. The algorithm of resampling is the same used on Exercise 3 about the robot localization using particle filters too. The implementation of the resampling function is on Listing 5

```
function [ particles, particles_w ] = resample( particles, particles_w )
%RESAMPLE Resample the particles based on their weights
%   Return these new particles along with their corresponding weights
nParticles = size(particles, 1);
index = floor(random('unif', 0, 1) * nParticles) + 1;
beta = 0;
mw = max(particles_w);
updatedParticles = zeros(nParticles, size(particles, 2));
updatedParticles_w = zeros(nParticles, 1);
for i = 1:nParticles
    beta = beta + random('unif', 0, 1) * 2 * mw;
    while beta > particles_w(index)
        beta = beta - particles_w(index);
        index = mod(index, nParticles) + 1;
    end
    updatedParticles(i,:) = particles(index,:);
    updatedParticles_w(i) = particles_w(index);
end

particles = updatedParticles;
particles_w = updatedParticles_w;
particles_w = particles_w / (sum(particles_w) + eps);
```

Listing 5: resample.m

Experiments

Once the implementation is finished, is time to experiment with different videos, scenarios and parameters to see and understand the behavior of the Condensation Tracker.

`video1.wmv`

With the first video, where the background is uniform and there is a moving hand, it can be seen how the condensation tracker is capable of tracking the hand until this disappear (Figure 1).

The tracker is capable to follow the hand but with some limitations on the trajectory predictions. As it is initialize with only some fingers on the view, then the observations tent to give a high score at any part of the hand or arm so the trajectory predicted is not smooth. Also remark that when the hand completely disappear the frame, the position prediction tent to go to the bottom left corner, where the histogram is more similar to the initial one.

`video2.wmv`

First, with the default parameters, on Figure 1 there is the tracking for `vide2.wmv`. It can be seen how the tracker has been able to follow the object even whit an occlusion.

Effect of the model

In order to study the performance of the tracker depending on its parameters, first, the constant velocity model is tested. The tracking is on Figure 3 where it can be seen how, as the initial velocity was set wit

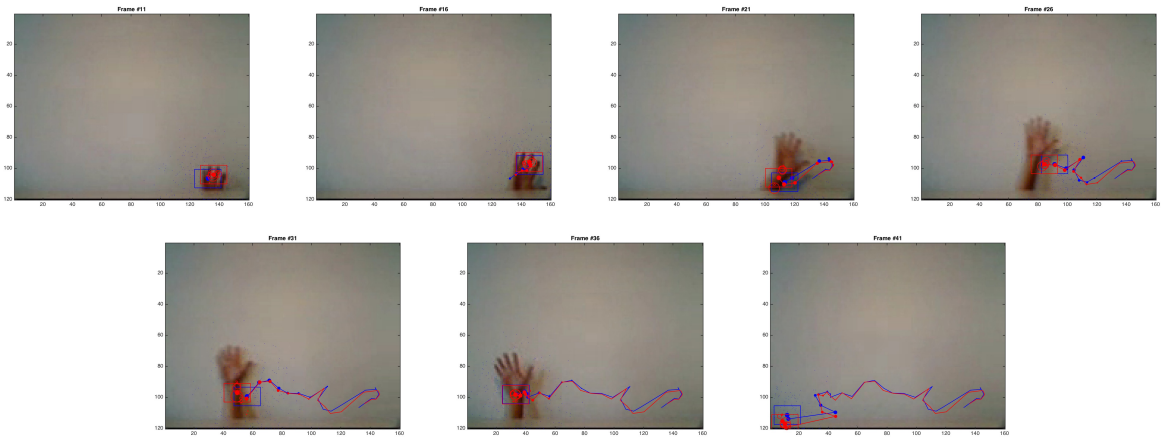

Figure 1: Tracking of the hand at the first video using the implemented Condensation Tracker

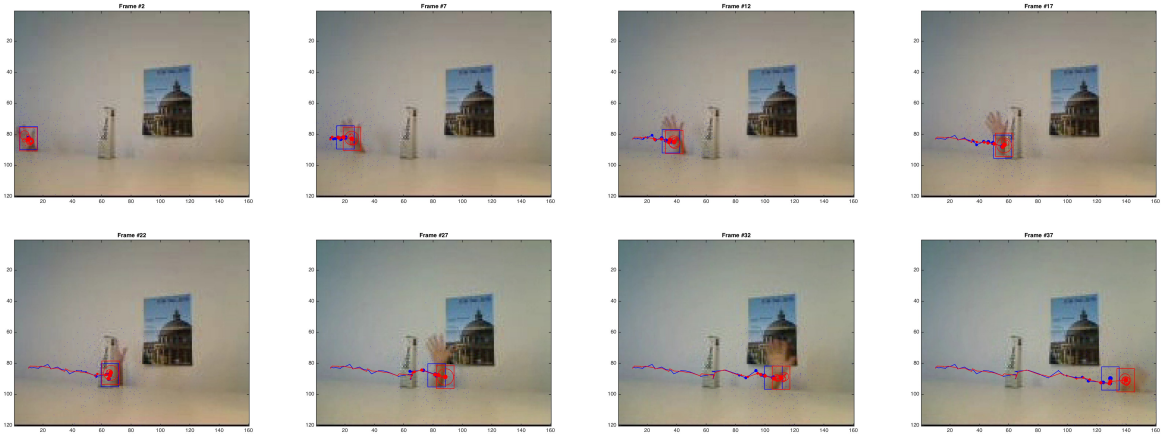

Figure 2: Tracking of the hand at `video2.wmv` using the implemented Condensation Tracker

$v_0 = [1, 10]^T$, the prior probability distribution tent to predict in a lower position in the vertical axis. But thanks to the observations, and the enough dispersion of the particles, the posterior probability perfectly tracks the object even when this is occluded.

Effect of the system noise

To study the effect of the system noise, with the system that models the constant position and the rest of default parameters, the $\sigma_{position}$ has been set to high value and into a lower one to observe the differences in the tracking.

For a highly noise system, with $\sigma_{position} = 50$, the tracking is represented in Figure 4 where it can be seen how the particles at each step spread along the image, leading to a prediction not very spatially precise as not many particles actually cover the object to track. This makes the tracking very imprecise along the movement. It could be solved adding more particles, but this will increase substantially the computations cost.

On the other hand, using $\sigma_{position} = 5$, the tracking is more smooth as all the particles concentrate around the previous position as it can be seen in Figure 5.

Effect of the measurement noise

Setting back the system noise to the default value ($\sigma_{position} = 15$), it has been tested the effects of the measurement noise into prediction of the Condensation Tracker. In Figure 6 the $\sigma_{observe}$ was set to 1, and it is observe that despite the tracking works well even with the occlusion, the effect is on the values of the particles weights. As $\sigma_{observe}$ had a high value ($\sigma_{observe} = 1$), the weight for the particles were more spread between them, taking into account more particles than in the default value. On the other

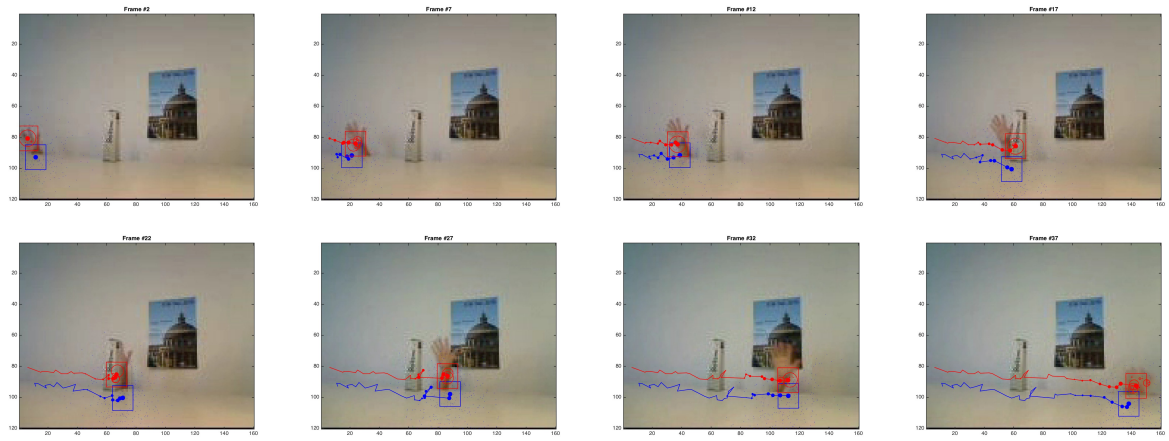

Figure 3: Tracking for `video2.wmv` with the constant velocity model.

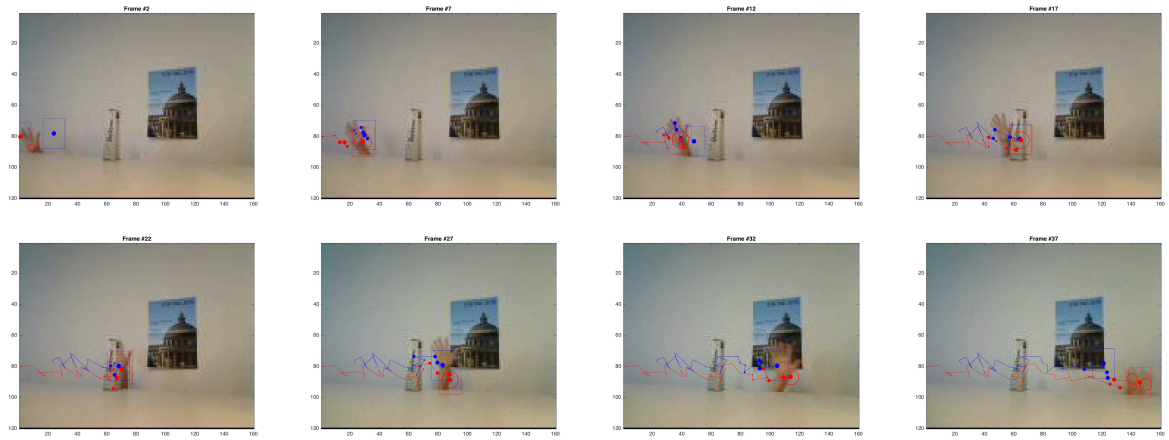

Figure 4: Tracking for `video2.wmv` with noise $\sigma_{position} = 50$.

hand, setting a small value ($\sigma_{observe} = 0.05$), it can be seen, how only the particle with highest score with its color histogram obtain most of the weight while the rest of the particles do not have any, and so, the posterior prediction of the position goes to this particle.

`video3.wmv`

For the third video, the same parameters are going to be used as in `video2.wmv` and only the remarkable facts are going to be commented.

Effect of the model

In Figure 9 can be seen how setting a constant velocity model, with $v_0 = [10, 1]^T$, the tracking works well but there is sometimes a lot of difference between the prior prediction of the position and the posterior. As it is assuming a constant velocity to the right, when the ball come to the left after touching the wall, the prior prediction starts to look far from the true position, so in this case, with an object moving in two different direction along the video, its not recommendable to use a model that assumes constant velocity for the object to track.

Effect of the system noise

Changing the system noise, it can be seen that setting a $\sigma_{position}$ a little bit higher than the default, happens that in a certain point of the video, some particle lay over the bottom right zone of the frame where the histogram obtain more match and the tracker loose the track of the ball (Figure 10). On the

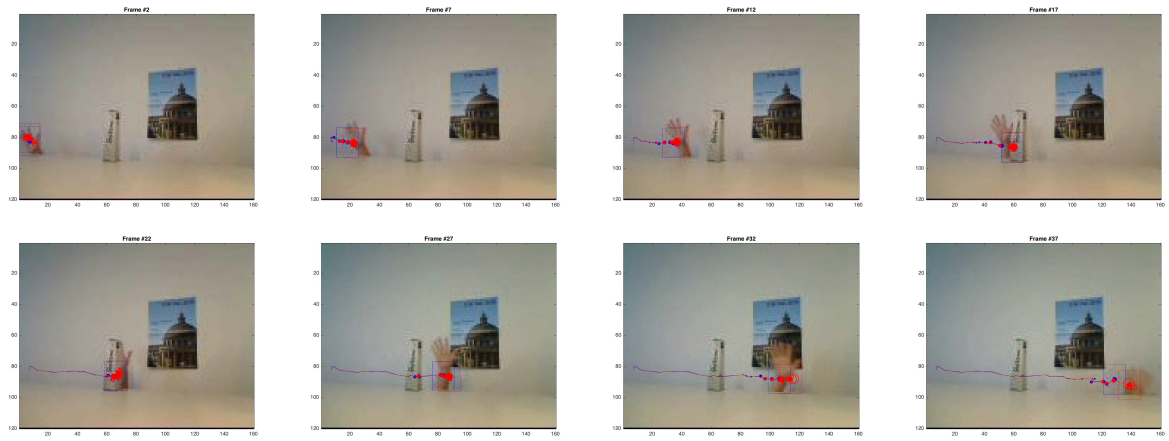

Figure 5: Tracking for `video2.wmv` with noise $\sigma_{position} = 5$.

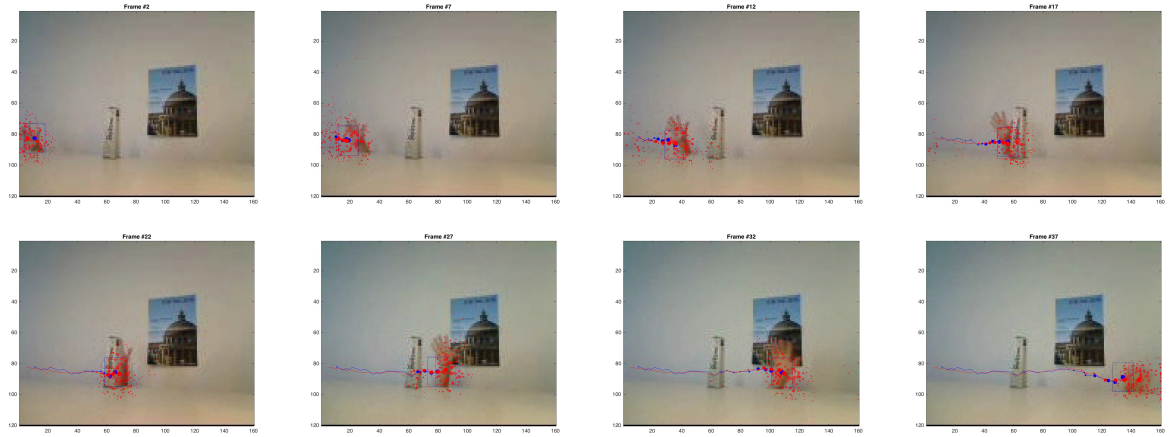

Figure 6: Tracking for `video2.wmv` with noise $\sigma_{observe} = 1$.

other hand, setting a smaller $\sigma_{position}$, the particles keep closer to the object to track (as it doesn't move so fast) and predict a smoother tracking of the ball (Figure 11).

Effect of the measurement noise

This parameter for this video, it has been observed to be the more sensible for the performance. In Figure 12, the $\sigma_{observe}$ was set high and so the weights of the particles were spread across them, triggering the same effect observed in Figure 10 where the predictions tent to detect the bottom right corner as the object and the track of the real object is loss. In Figure 13 where the $\sigma_{observe}$ was set low, it is observe the same phenomena that in `video2.wmv`, which only one particle tent to have a very high weight. For this video happened that may at some point do not have any particle matching the object, all the particles have a low weight and the track is lost, until any other particles lays over the object and the track continues.

Questions

Now I'm changing some other parameters of the tracker and see its effect.

Effect of the number of particles

To see the effect of the number of particles into the tracking, in Figure 14 there is the tracking using 600 particles and in Figure 15 using 100 particles. For both cases, the tracker is able to correctly track the ball during all the video. In the case with low number of particles, maybe the track is not as smooth as

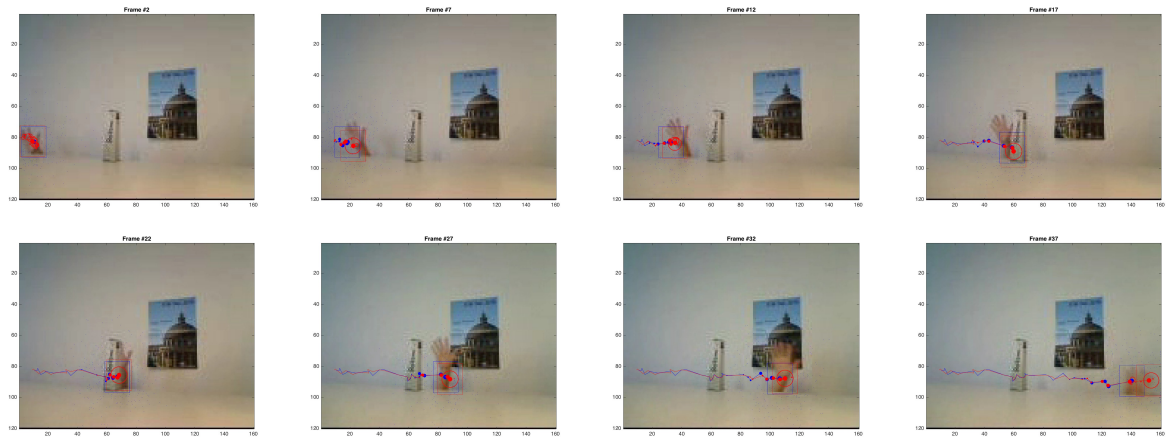

Figure 7: Tracking for `video2.wmv` with observe $\sigma_{observe} = 0.05$.

with high number due to the random positioning of fewer particles that do not offer too much precision. On the other hand, using a larger number of particles implies an increasing in terms of computational costs that in some applications are not permissive in the case of performing real-time tracking.

Effect of the number of bins

To see the effect of the number of particles into the tracking, in Figure 16 there is the tracking using 64 bins and in Figure 17 using 8 bins. For both cases, the tracker is able to correctly track the ball during all the video and no major difference is observed. Also comment that runs a little bit faster with a fewer number of bins as the computations are lighter.

Advantages and disadvantages of the type of model

Using one type of model or another one can give some improvement of performance for the Condensation Tracker. The selection of this model have to be done with a previous knowledge of the behavior of the object inside the video frame. Using the static position model with a good value of the system noise performs well, but in cases with an object with constant velocity, a model that takes into account this information can help to improve its performance. It has be seen on the experiments, that in the case where the ball moves with a decreasing velocity and a 180 change of direction, the prior estimation of the position didn't match at all the posterior as it predicted to be behind the ball all the time. The model must be chosen from the knowledge of the movement of the object and if a good model is used, then some parameters can be tune up to obtain a faster tracker with the same performance. If the model is not the correct one, more particles and computational resources will be required to perform the tracking. At the videos of the assignment, for `video2.wmv` the best model to do the tracking is a model that consider a constant velocity as the hand appearing does. But in contrast, for the `video3.wmv` where the velocity of the ball, at some point, completely change the sign of the velocity, may not be the best model, because at this point, the particles could loose the track of the ball.

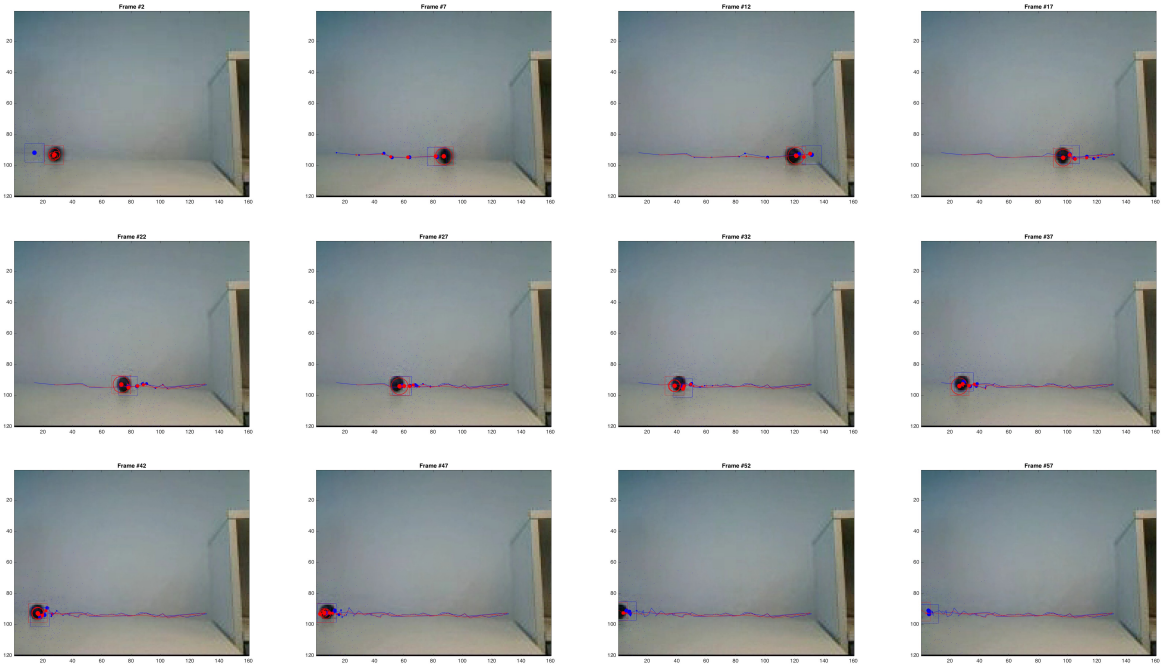

Figure 8: Tracking of the ball at `video3.wmv` using the implemented Condensation Tracker with default values

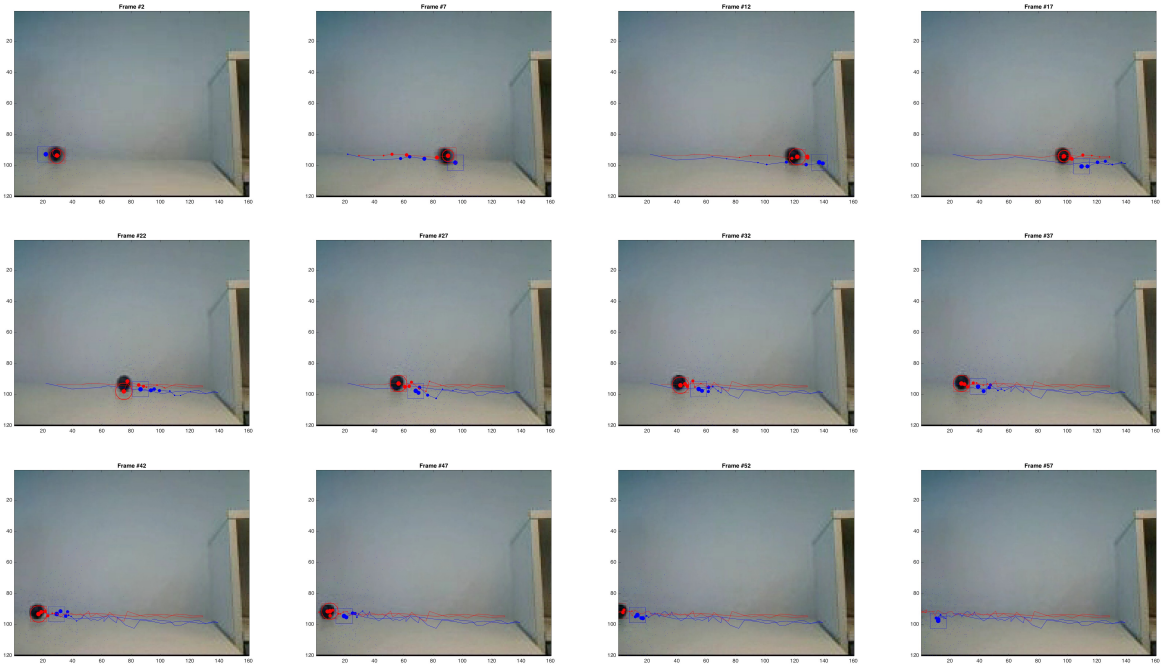

Figure 9: Tracking for `video3.wmv` with the constant velocity model.

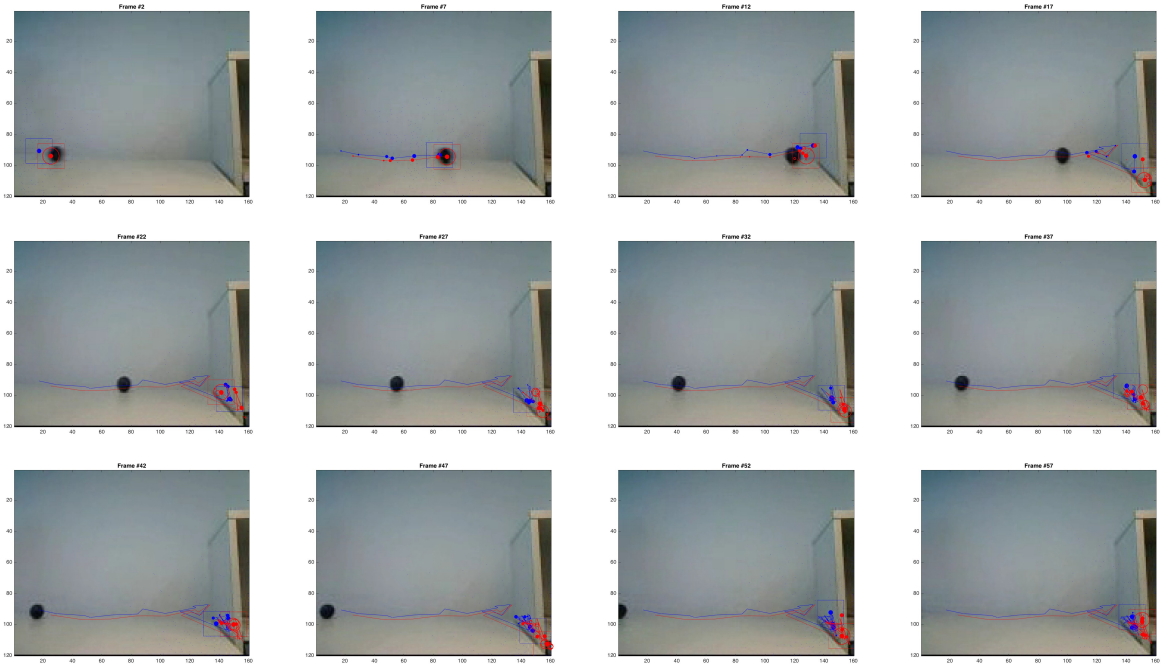

Figure 10: Tracking for `video3.wmv` with the $\sigma_{position} = 25$.

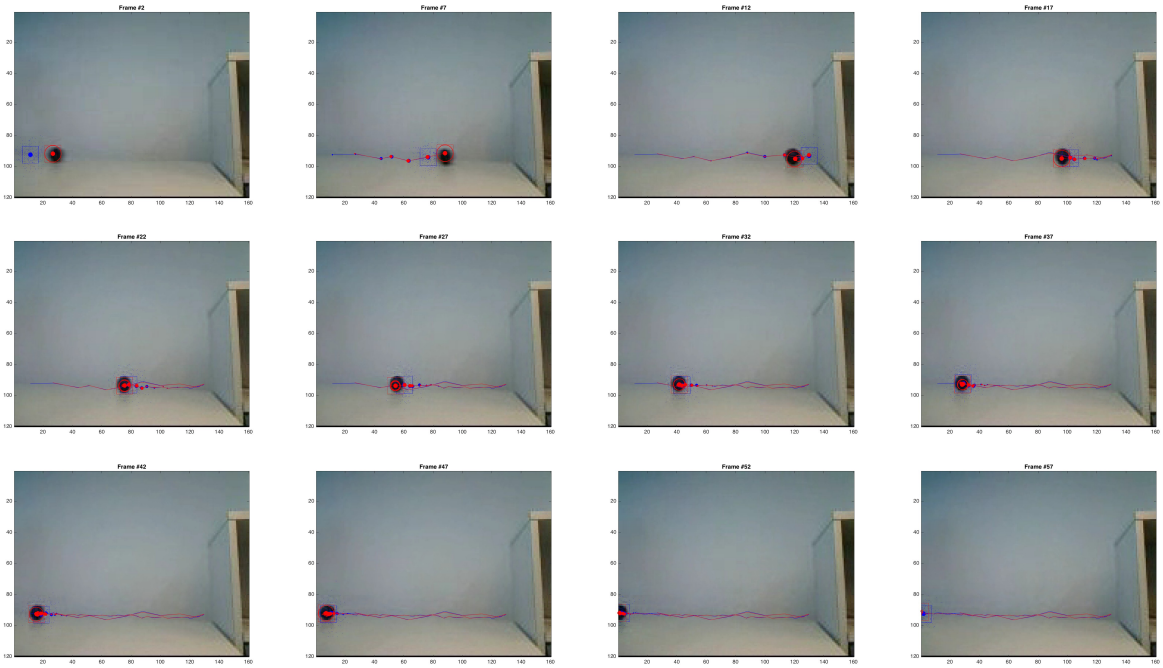

Figure 11: Tracking for `video3.wmv` with the $\sigma_{position} = 5$.

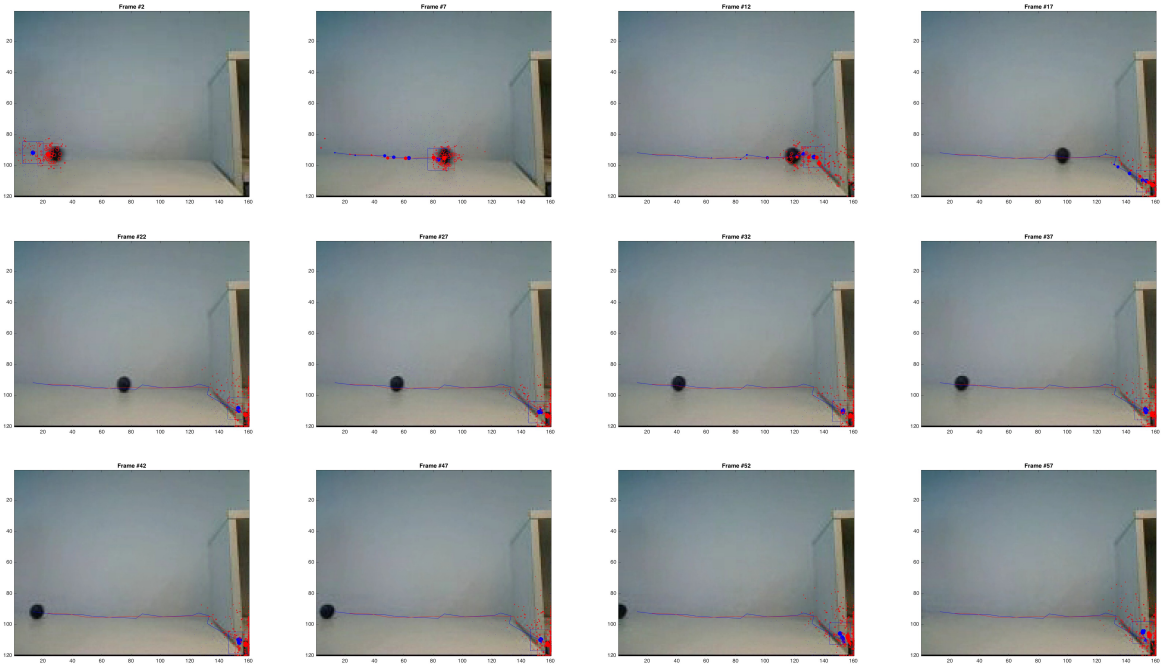

Figure 12: Tracking for `video3.wmv` with the $\sigma_{observe} = 1$.

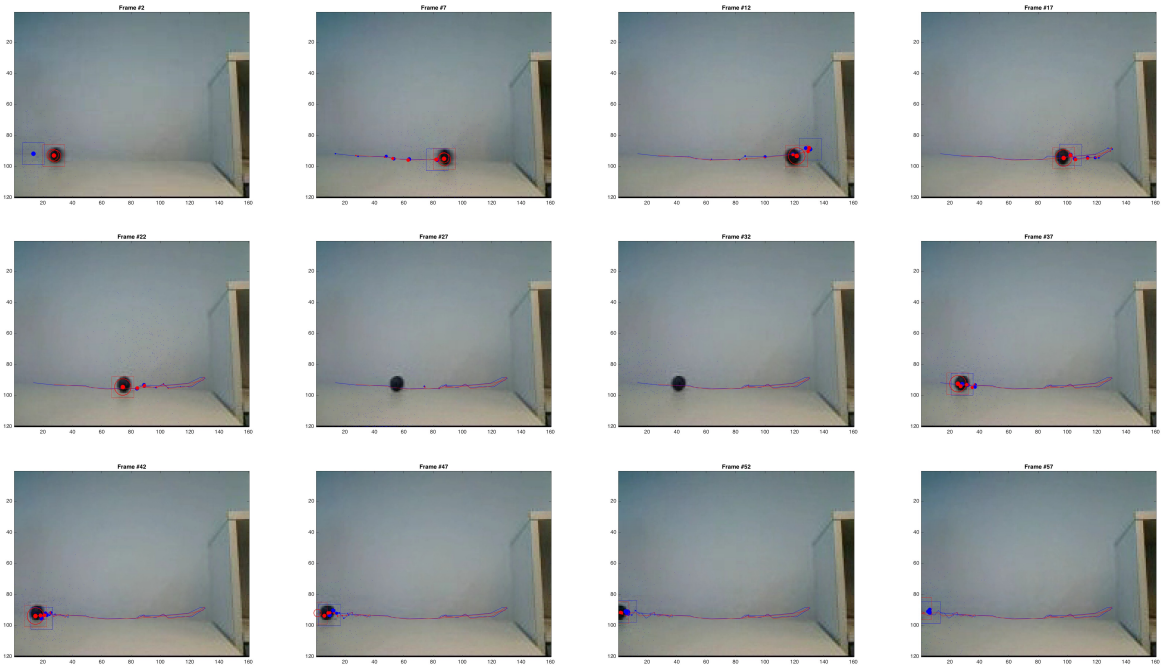

Figure 13: Tracking for `video3.wmv` with the $\sigma_{observe} = 0.05$.

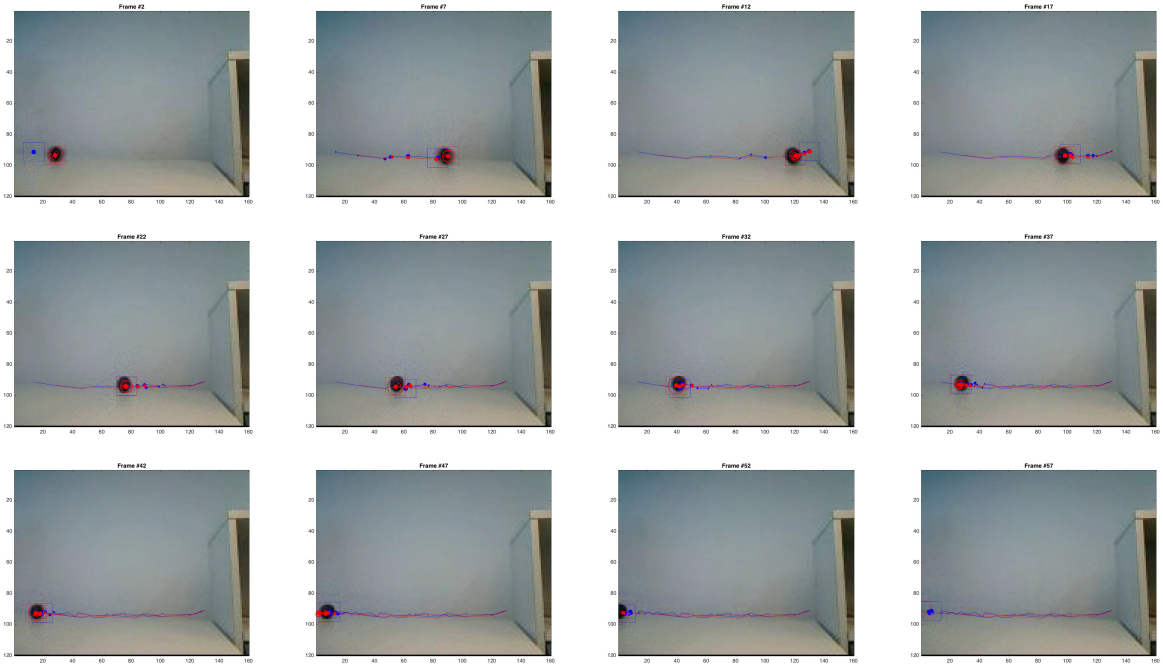

Figure 14: Tracking for `video3.wmv` with the 600 particles.

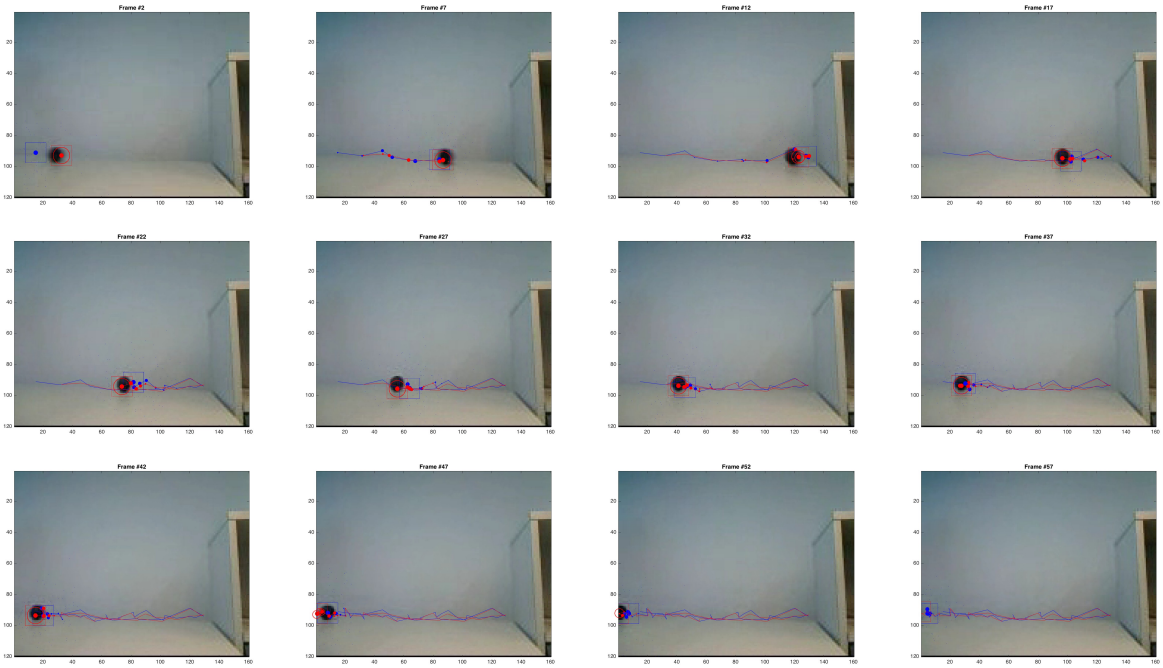

Figure 15: Tracking for `video3.wmv` with 100 particles.

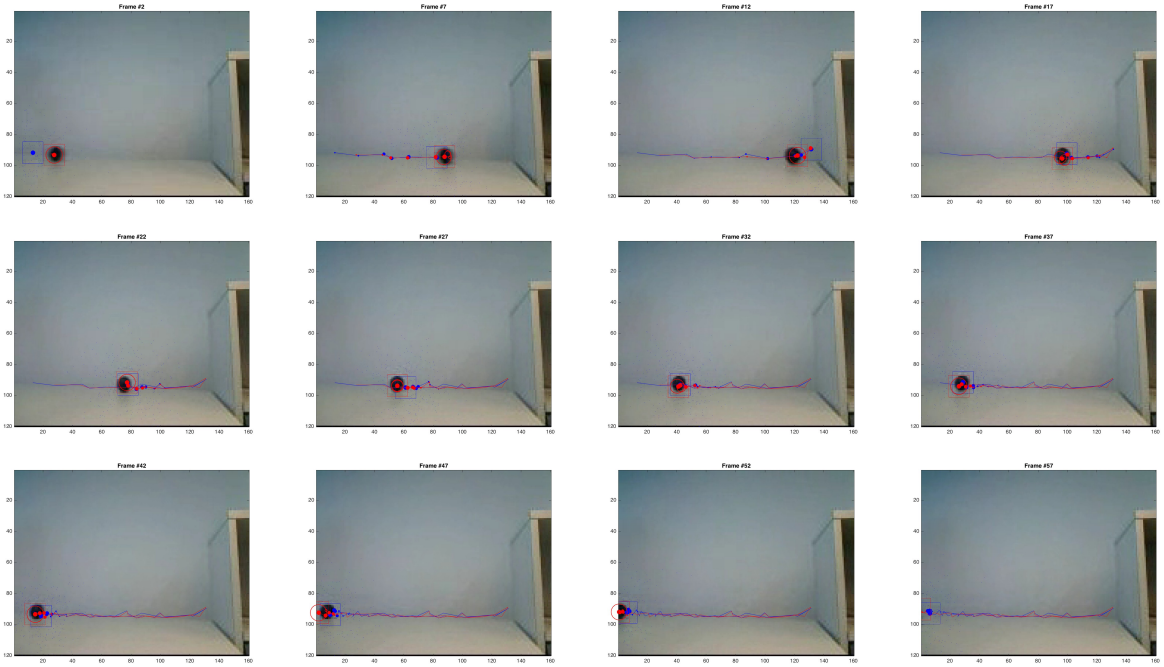

Figure 16: Tracking for `video3.wmv` with the 64 bins.

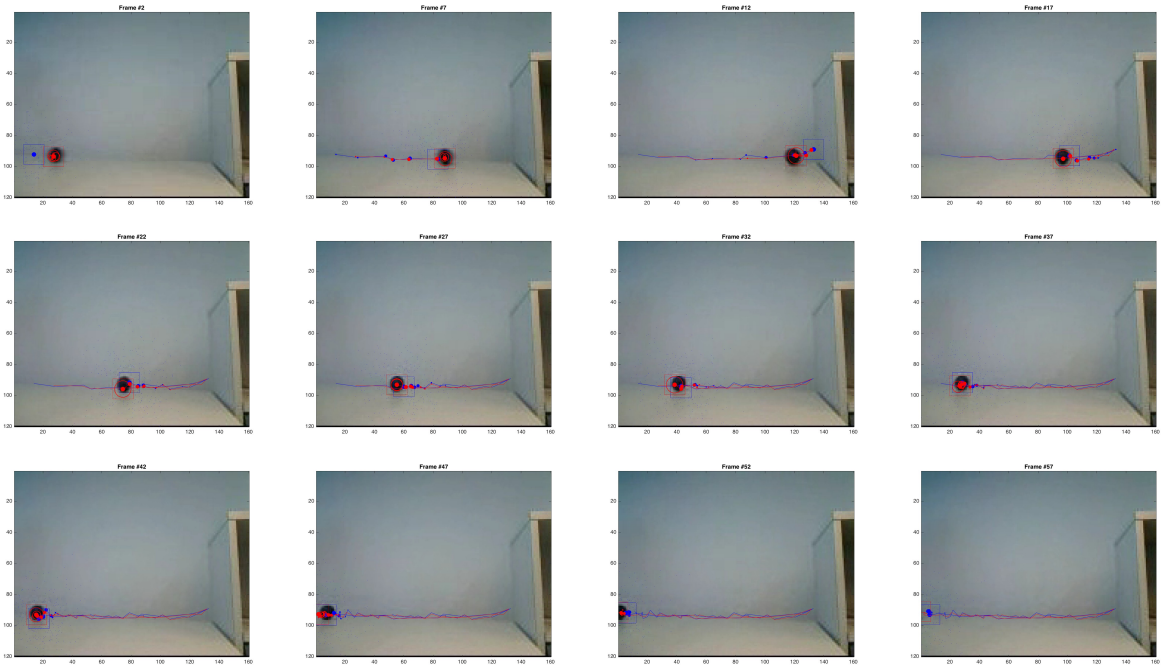

Figure 17: Tracking for `video3.wmv` with 8 bins.