Exercises for Probabilistic Graphical Models
Sheet No. 5

Bernt Schiele, Bjoern Andres, Eldar Insafutdinov, Evgeny Levinkov

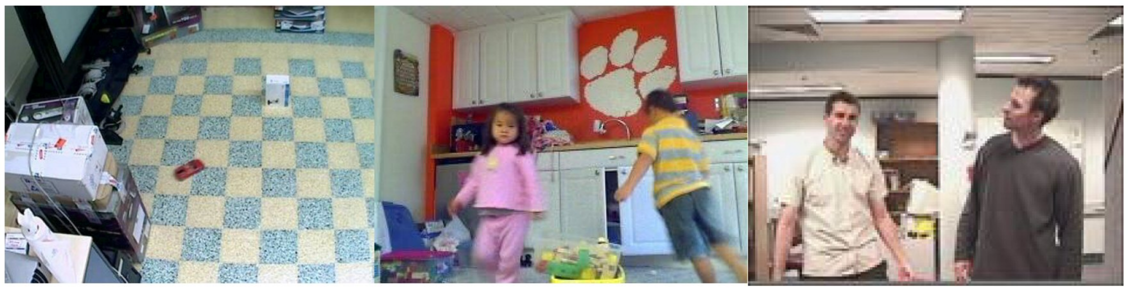
Due Date: 16 January

Hand in: by email as a zipped archive to `insafutdin@mpi-inf.mpg.de`. Please send your source code and your results on each of the video sequences.

The assigment is courtesy of Dr. Kevin Smith, with permission. Seeing his lecture `http://www.kev-smith.com/teaching/L7_ParticleFilters.pdf` (especially starting from slide #50) might help you understand this assignment better. If you still have doubts, you can look into the original paper of Isard and Blake: "CONDENSATION - Conditional Density Propagation for Visual Tracking" from 1998.

1 Particle Filters

The goal of this assignment is to implement a particle filter and apply it to track objects in three video sequences.



In sequence 1, the goal is to track the small red toy car. In sequence 2, the goal is to track the girl in pink. In sequence 3, the goal is to track the head of the person on the left. You will not lose points for tracking errors in your results sequences as long as it is evident that you have implemented the particle filter correctly.

Each of the sequences contains challenging problems such as occlusion, abrupt motion changes, and distracting backgrounds. Your particle filter should estimate the state of the object to be tracked over the entire sequence, $X = (x_1, x_2, \ldots x_T)$, where x_t is the state of the object at time t. The state space at a given time step parametrizes a bounding box, $x_t = (x, y, \dot{x}, \dot{y}, a, h)$, which includes the (x, y) location of the upper left corner of the bounding box in the image, the velocity vector $(\dot{x}, \dot{y})$, the aspect ratio $a = w/h$, and the height h. Your resulting video sequences should display the state of each particle (either as a point or as a bounding box), as well as the final estimated state, as shown below:

Matlab skeleton code and supplementary functions have been provided. Functions are provided to accomplish tasks such as reading files, displaying images, initializing the tracker, and writing results to file. Hints are also provided in the comments of the skeleton code. These materials can be downloaded at the course website.

1.1 Implementation

(17 points)

To implement a particle filter, you will need to accomplish the following steps (see also slide #72 in Kevin's lecture):

1. Represent the particle set. Each particle should consist of a state estimate and associated weight. The skeleton code represents particles using a Matlab structure.
2. Perform resampling. This step is necessary to prevent the degeneracy problem. Keep the number of samples to N in every iteration.
3. Model the dynamics. Use a linear dynamic model with additive Gaussian noise to predict the new state given the previous state. It will be necessary to tune the covariance matrix Q of the Gaussian to determine the "spread" of the particles.
4. Define an observation model. The observation model computes the likelihood that observed data from the image supports the estimated state hypothesized by a particular particle. In the provided skeleton code, this is done by comparing a color histogram extracted from the particle's bounding box to a known color model extracted beforehand. The included function `LABhistogram.m` is used to extract the LAB histogram. The skeleton code models the likelihood as $p(z_t^n|x_t^n) = exp(-\lambda dist(h, h^*))$, where $dist$ is the KL divergence of two histograms, h is a color histogram corresponding to the observation z_t^n associated to particle x_t^n, and h^* is a known color model (learned at the time you manually initialized the filter). The parameter λ adjusts the influence of the likelihood function.
5. Reweight the particle set using the likelihoods obtained from the observation model.
6. Infer a solution. After performing all of the particle filtering step, a final solution must be extracted from the approximate distribution represented by the particle set. You can use the mean state vector computed from all particles.

When implementing the particle filter, consider the following questions (for yourself): What happens to performance if the resampling step is removed? How does changing the parameter λ in the observation model change the performance? What happens if the velocity terms are removed from the state vector? How do the values of the covariance matrix Q influence the performance of the particle filter? What are some other ways to infer a solution from the particle set?

1.2 Tracking multiple objects

(3 points)

How would your particle filter perform in the case of tracking multiple targets and how can you improve the tracker in the case of multiple objects? Consider what happens if two visually similar targets interact. You can use sequence 4 to check your hypothesis.