Probabilistic Tracking of Multiple Rodent Whiskers In Monocular Video Sequences

Jim Holmström, Emil Lundberg
Bachelor's Thesis at CSC, KTH

Background

The Problem

The interest in studying rodent whiskers has recently seen a significant increase, particularly in the field of neurophysiology. As a result, there is a need for automatic tracking of whisker movements. Currently available commercial solutions either are extremely expensive, restrict the experiment setup, or fail when whiskers cross or overlap. A cheap, reliable solution to the tracking problem is needed.

A Probabilistic Approach

We propose solving the problem by a probabilistic approach. We use a technique known as the *Particle Filter* to propagate a whisker model between frames of high speed video. In each frame, the next state of the model is predicted by searching a pre-trained database, and filtering the results through the Particle Filter. The main difference between this and existing solutions is that it maintains a model of the whiskers. This makes it easier to keep track of them even when they cross or overlap.

The Probabilistic Framework

Our solution is based on discrete *Markov processes*, which are a special case of stochastic processes. For a Markov process, the next state depends only on the present state and not on past states.

An example of a discrete Markov process is that of throwing dice and summing the results: the throws and state space are discrete, and the possible states (sums) after the next throw depends only on the current state.

In mathematical terms, a discrete Markov process satisfies the following:

$$p\left(Z_{n+1} | Z_n \wedge Z_{n-1} \wedge Z_{n-2} \wedge \cdots \wedge Z_0\right) = p\left(Z_{n+1} | Z_n\right), \tag{1}$$

where Z_n is the system's *state* after step n and $p\left(Z_{n+1} | Z_n, Z_{n-1}, Z_{n-2}, \ldots, Z_0\right)$ is the probability that the system will have state Z_{n+1} in the next step, given that the previous states where $Z_n, Z_{n-1}, Z_{n-2}, \ldots, Z_0$.

A *hidden Markov model* (HMM) describes a Markov process where one cannot measure the state Z of the system directly - it is "hidden". Instead we obtain an *observation* I[a] of the state. This *perception* is generally non-deterministic, so we need to denote it as $p(I_n|Z_n)$ which is the probability that we will observe I_n if the current state of the system is Z_n.

The Particle Filter

The Particle Filter is a technique for simulating a process described by a HMM. It uses a finite set X_{n+1} of hypotheses to approximate the probability function $p\left(Z_{n+1}|Z_n\right)$. The hypotheses X_n are also known as *particles*, thereby the term "particle filter".

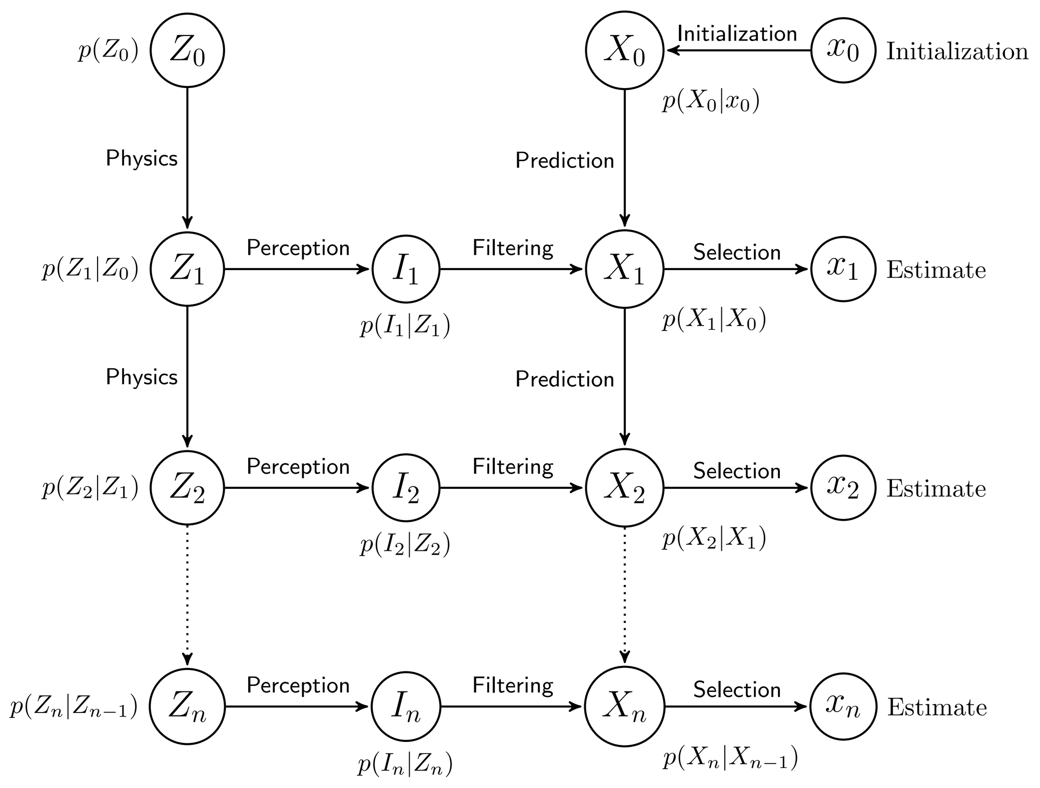

Figure 1: Schematic image of the Particle Filter working alongside a Hidden Markov Model.

Figure 1 shows the working principle of the Particle Filter working alongside a Hidden Markov Model. The following is the core function of the Particle Filter:

> The particle filter attempts to approximate the probability density function $p\left(Z_{n+1}|Z_n\right)$ as a set X_{n+1} of discrete hypotheses.

For brevity, we will commit some abuse of notation in the following discussion.

Initialization: Since the algorithm only does tracking, we need to initialize the algorithm with a start guess x_0. Using this we take a number of samples $X_0 \sim \mathcal{N}\left(x_0, \Sigma\right)^b$ and let the set X_0 be an approximation of $p\left(Z_0\right)$.

Prediction: The hypotheses X_n are updated in the *prediction* step to an approximation of $p\left(Z_{n+1}|Z_n\right)_{n+1}$. This is done by drawing new samples $\bar{X}_{n+1} \sim p\left(X_{n+1}|X_n\right)$.

Perception: By measuring the state of the system, we gain an observation $I_{n+1} \sim p\left(I_{n+1}|Z_{n+1}\right)$ of the state Z_{n+1}.

Filtering: The observation I_{n+1} of the system is then used for filtering bad hypotheses out of $\bar{X}_{n+1}$. We draw samples X_{n+1} from $\bar{X}_{n+1}$ with probabilities given by $p\left(I_{n+1}|\bar{X}_{n+1}\right)$. As a result, X_{n+1} will be a subset of $\bar{X}_{n+1}$ where more probable hypotheses appear multiple times. For this reason, this is also known as the *resampling* step. The set X_{n+1} is the *belief*, our approximation of $p\left(Z_{n+1}|Z_n\right)$.

Selection: Finally, we produce a single hypothesis x_{n+1} from X_{n+1} as our estimate of the state Z_{n+1}. Supposing X_{n+1} is a good approximation of $p\left(Z_{n+1}|Z_n\right)$, and that $p\left(Z_{n+1}|Z_n\right)$ is unimodal, the mean value of X_{n+1} is a good hypothesis since it approximates the expectation of $p\left(Z_{n+1}|Z_n\right)$.

Implementing of the Particle Filter

An implementation of the Particle Filter consists mainly of designing the probability functions $p\left(X_{n+1}|X_n\right)$ and $p\left(I_n|X_n\right)$, and providing the algorithm with a sensible initialization. This is what this project is all about. The rest just consists of taking samples from these functions.

Results

So far, we have run some tests on randomly generated video sequences of whisker-like objects. While the results are far from good enough for practical use, they are still quite promising.

[a] In this project, the observation is a grayscale *image*, which is why we use the symbol I.

[b] Σ is roughly estimated from the database