



**KTH Computer Science
and Communication**

Probabilistic Tracking of Multiple Rodent Whiskers in Monocular Video Sequences <FIRST DRAFT>

A small step towards cheap, reliable, and non-intrusive automatic tracking of whiskers.

JIM HOLMSTRÖM
JIMHO@KTH.SE
EMIL LUNDBERG
EMLUN@KTH.SE

SA104X Examensarbete inom teknisk fysik, grundnivå
Bachelor's Thesis at CSC, KTH
Supervisor: Prof. Örjan Ekeberg
Examiner: ???

TRITA xxx yyyy-nn

Abstract

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 in the presence of *clutter* or occlusion.

This thesis proposes a proof-of-concept implementation of a probabilistic tracking system. This solution uses 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 querying a pre-trained database and filtering the results through the Particle Filter. The implementation is written in Python using NumPy and an SQLite3 database.

Testing results indicate that the approach is feasible. Even using a rather crude database, the tracker manages to track multiple real whiskers at once, though only for short sequences at a time. Better training data, such as hand-labeled real data, might vastly improve the result.

Keywords: Tracking, Multiple, Whisker, Particle Filter, Transition Database, Model Evaluation, Proof-of-Concept

Referat

Statistisk Följning av Multipla Morrhår på Gnagare i Video.

Intresset för att studera morrhår på gnagare har på senare tid ökat, speciellt inom neurofysiologi. Som ett resultat finns det ett behov av automatisk följlning av morrhår. Nuvarande kommersiella lösningar är antingen extremt dyr, sätter begränsningar på experiment uppställningen eller misslyckas i stökiga miljöer eller under överlappning. Arbetets mål är en proof-of-concept implementering av ett statistiskt följlningssystem. Lösningen använder sig av en teknik känd som partikelfilter för att propagera en morrhårmodell mellan bildrutor på höghastighetsvideo. För varje bildruta förutspås nästa tillstånd genom att fråga en förtränad databas via partikelfiltret.

Nyckelord: Följlning, Multipla, Morrhår, Partikelfilter, Övergångsdatabas, Modeleveruering, Proof-of-Concept

Contents

1	Introduction	1
1.1	Background	1
1.2	Difficulties	1
1.3	Contributions of this thesis	1
2	Related Work	3
2.1	Examples of whisker tracking systems	3
2.1.1	Unsupervised Whisker Tracking in Unrestrained Behaving Animals[7]	3
2.1.2	High-Precision, Three-Dimensional Tracking of Mouse Whisker Movements with Optical Motion Capture Technology[3] . . .	3
2.2	Probabilistic Tracking and Reconstruction of 3D Human Motion in Monocular Video Sequences[5]	3
3	Definitions	5
3.1	Images and videos	5
3.1.1	Grayscale image	5
3.1.2	Video	5
3.2	Image Processing	6
3.3	States, hypotheses and estimates	7
4	Theory	9
4.1	Introduction	9
4.2	Markov processes	9
4.3	The Hidden Markov Model	10
4.4	The Curse of Dimensionality	10
4.4.1	Overcoming the Curse	11
4.5	The Particle Filter	12
4.5.1	The Particle Filter algorithm	14
4.6	Visual Cues	14
5	Algorithms and Implementations	17
5.1	The particle filter	17
5.1.1	Initilization x_0	17
5.2	The state transition database	18

5.2.1	Data format	18
5.2.2	Prediction $p(x_t x_{t-1})$	18
5.3	Tracker	18
5.3.1	Filtering $p(I_t x_t)$	18
5.4	Preprocessing of real images	19
6	Results	21
7	Analysis and Discussion	23
	Bibliography	25

Chapter 1

Introduction

1.1 Background

With the ever increasing power and mobility of computers, computer vision has recently seen a large increase in interest. Encompassing problems such as classification, recognition, perception and tracking, application of the discipline could make many tasks easier and more efficient.

In particular, biology and neuroscience researchers are interested in tracking the movements and posture <ref:(2)> of animals or parts of animals. One such field aims to study the movements of rodent whiskers. However, most whisker tracking software available today suffers a few fundamental flaws. They are either so expensive that not even well funded laboratories feel they can afford them or have problems tracking multiple whiskers at once, often requiring removal of almost all whiskers. Some higher precision systems impose other restrictions on the experiment, such as restraining the animal or attaching motion capture markers to the whiskers [3]. Such restrictions may cause systematic errors and not give a representative view of how the animal naturally uses its whiskers.

1.2 Difficulties

In general, the main difficulty in tracking and localization is to separate the tracked object from clutter and occlusion. In the case of whisker tracking, the dominant problems are that whiskers often overlap and vary in size, and the relatively low spatial and temporal resolution[2] in highspeed video results in subpixel whiskers and motion blur.

In 2001, Hedvig Sidenbladh investigated probabilistic methods for tracking three-dimensional human motion in monocular video [5]. Many of the problems inherent in computer vision were regarded, and ... «continue»

1.3 Contributions of this thesis

Chapter 2

Related Work

2.1 Examples of whisker tracking systems

2.1.1 Unsupervised Whisker Tracking in Unrestrained Behaving Animals[7]

Image flow, gabur

It works well under temporary occlusion but needs clipping of most whiskers and thereby introduce possible effects to the experimental results.

2.1.2 High-Precision, Three-Dimensional Tracking of Mouse Whisker Movements with Optical Motion Capture Technology[3]

Uses a 3D tracking system consisting of two high-speed cameras and markers mounted on the whiskers of the head-fixed mouse. The head-fixed is for the markers must be visible at all times to make the tracking work.

2.2 Probabilistic Tracking and Reconstruction of 3D Human Motion in Monocular Video Sequences[5]

Chapter 3

Definitions

3.1 Images and videos

3.1.1 Grayscale image

Definition 1. A grayscale *image* can be defined as a function

$$\begin{aligned} I : \mathbb{N}^2 &\rightarrow \mathbb{R}^+ \\ I : \text{position} &\rightarrow \text{intensity}. \end{aligned} \quad (3.1)$$

It can also be identified with $\mathbb{N}^2 \times \mathbb{R}^+$ as the tuple $\langle \text{position}, \text{intensity} \rangle$. The *image space* is denoted as \mathcal{I} in this thesis.

In a computer an image is represented as a integer matrix, often 8 bit integers.¹

Example 1.

$$\begin{pmatrix} 255 & 255 & 255 & 255 & 255 & 255 & 255 & 255 & 255 & 255 \\ 255 & 255 & 255 & 255 & 255 & 255 & 255 & 255 & 255 & 255 \\ 255 & 255 & 255 & 0 & 255 & 255 & 0 & 255 & 255 & 255 \\ 255 & 255 & 255 & 0 & 255 & 255 & 0 & 255 & 255 & 255 \\ 255 & 255 & 255 & 255 & 255 & 255 & 255 & 255 & 255 & 255 \\ 255 & 0 & 255 & 255 & 255 & 255 & 255 & 255 & 0 & 255 \\ 255 & 255 & 0 & 255 & 255 & 255 & 255 & 0 & 255 & 255 \\ 255 & 255 & 255 & 0 & 0 & 0 & 0 & 255 & 255 & 255 \\ 255 & 255 & 255 & 255 & 255 & 255 & 255 & 255 & 255 & 255 \\ 255 & 255 & 255 & 255 & 255 & 255 & 255 & 255 & 255 & 255 \end{pmatrix} \Rightarrow \begin{array}{c} \blacksquare \quad \blacksquare \\ \blacksquare \quad \blacksquare \\ \blacksquare \quad \blacksquare \\ \blacksquare \quad \blacksquare \\ \blacksquare \quad \blacksquare \\ \blacksquare \quad \blacksquare \\ \blacksquare \quad \blacksquare \\ \blacksquare \quad \blacksquare \\ \blacksquare \quad \blacksquare \\ \blacksquare \quad \blacksquare \end{array}$$

3.1.2 Video

Definition 2. A *video* is a function mapping an integer to an image:

$$\text{video} : \mathbb{N} \rightarrow \mathcal{I}. \quad (3.2)$$

¹Integers in the range $[0, 255]$.

3.2 Image Processing

Definition 3. The rendering function R takes a hypothesis x and renders an image with a resemblance of how a real whisker would have looked like having the same underlying model and parameters as x .

$$\begin{aligned} R : \mathcal{X} &\rightarrow \mathcal{I} \\ x &\mapsto \text{Render}(x) \end{aligned} \tag{3.3}$$

Definition 4. The addition operation on images is performed by elementwise addition.

$$\begin{aligned} + : \mathcal{I} \times \mathcal{I} &\rightarrow \mathcal{I} \\ (I_a, I_b) &\mapsto I_a + I_b \end{aligned} \tag{3.4}$$

Definition 5. The multiplication operation on images is performed by elementwise multiplication.

$$\begin{aligned} * : \mathcal{I} \times \mathcal{I} &\rightarrow \mathcal{I} \\ (I_a, I_b) &\mapsto I_a * I_b \end{aligned} \tag{3.5}$$

Definition 6. The image transformation ϕ takes an image I and returns a transformed image of the same size as I .

$$\begin{aligned} \phi : \mathcal{I} &\rightarrow \mathcal{I} \\ I &\mapsto \text{Transform}(I) \end{aligned} \tag{3.6}$$

The structure $\langle \mathcal{I}, +, * \rangle$ inherits the properties from $\langle \mathbb{R}^+, +, * \rangle$ by just being a vectorized version.

Definition 7. We will use subtraction loosely² as elementwise subtraction and then subtract the smallest element on all elements to make the operation closed.

$$\begin{aligned} - : \mathcal{I} \times \mathcal{I} &\rightarrow \mathcal{I} \\ (I_a, I_b) &\mapsto I_a - I_b \end{aligned} \tag{3.7}$$

Definition 8. The image transformation ϕ takes an image I and returns a transformed image.

$$\begin{aligned} \phi : \mathcal{I} &\rightarrow \mathcal{I} \\ I &\mapsto \text{Transform}(I) \end{aligned} \tag{3.8}$$

²Meaning, no analysis on the structure is done.

3.3. STATES, HYPOTHESES AND ESTIMATES

3.3 States, hypotheses and estimates

A system is said to have a *state*. The state is some quantity that defines the qualities of the system.

State The state of a system is denoted Z . When time is relevant, the state at time t is denoted Z_t .

State space The set \mathcal{Z} of all possible states, $Z \in \mathcal{Z}$.

Hypothesis x at the state Z of a system.

Hypothesis space \mathcal{X} of all possible hypotheses, $x \in \mathcal{X}$. In general, $\mathcal{X} \neq \mathcal{Z}$ since most models are simplifications of the system.

Estimate The hypothesis x^* we believe approximates Z best.

Observation In general, it is not possible to directly record the state Z of a system.³

We instead get an *observation* I of the state.

Degrees of freedom of adjustable parameters in a model, often abbreviated DOF.

Note that all of the above depend on the model used.

³If it were, there would be no need for tracking.

Chapter 4

Theory

4.1 Introduction

The core of the tracking engine is a technique known as the *particle filter*. The particle filter is a kind of Bayesian filtering where one uses discrete hypotheses, also known as *particles*, to approximate continuous PDF [6]. It builds upon the theory of *Markov processes* and the *hidden Markov model*.

4.2 Markov processes

A Markov process is a special case of a stochastic process. For a Markov process, the next state depends only on the present state and not on past states. For this

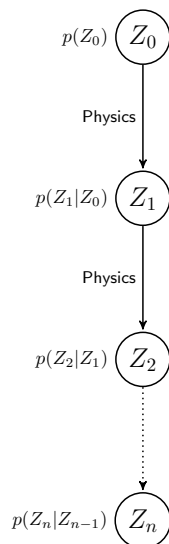


Figure 4.1. Schematic image of a Markov process.

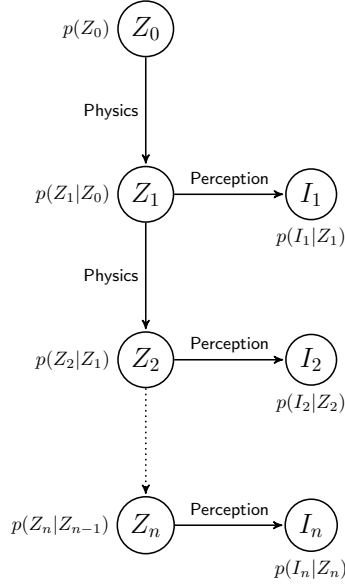


Figure 4.2. Schematic image of a hidden Markov model.

reason, a Markov process is often said to be “forgetful”.

In mathematical terms, a Markov process satisfies the following:

$$p(Z_t | Z_{t-1} \wedge Z_{t-2} \wedge \cdots \wedge Z_0) = p(Z_t | Z_{t-1}), \quad (4.1)$$

$p(Z_t | Z_{t-1} \wedge Z_{t-2} \wedge \cdots \wedge Z_0)$ is the probability that the system will have state Z_t at time t , given that the previous states were $Z_{t-1}, Z_{t-2}, \dots, Z_0$.

4.3 The Hidden Markov Model

The working principle of the particle filter is based on the *hidden Markov model* (HMM). A HMM describes a Markov process where we cannot measure the state directly - it is “hidden”[4]. Instead we obtain an *observation* I^1 of the state. This *perception* is generally non-deterministic, so we need to denote it as $p(I_t | Z_t)$ which is the probability that we will observe I_t if the state is Z_t .

4.4 The Curse of Dimensionality

A phenomenon that becomes apparent in high-dimensional spaces is the so-called “Curse of dimensionality” [4]. The problem is that the search volume grows exponentially with the number of dimensions. It originates from the fact that we need $\mathcal{O}(C^n)$ samples to obtain a sample density of C in a n -dimensional space.

¹In this thesis, the observation is always a grayscale *image*, therefore the observation is denoted I .

4.4. THE CURSE OF DIMENSIONALITY

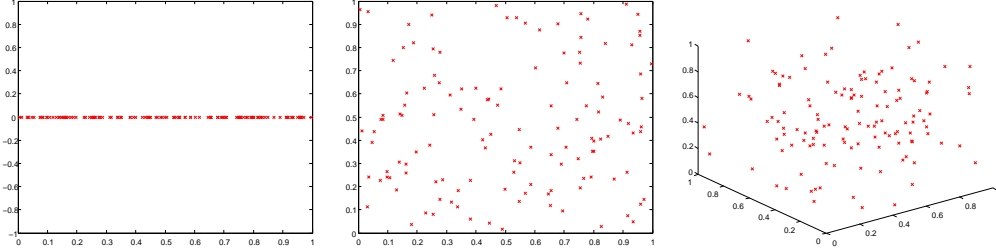


Figure 4.3. Plots of 128 scattered samples in 1, 2 and 3 dimensions, respectively.

The first consequence of this is that in order to approximate a high-dimensional function one needs orders of magnitude more samples.

The other drawback with high dimensional space is the large “borders” of the sample-set compared to lower dimensional space which results in orders of magnitude higher chance for a point one wants to approximate to fall outside the sample-set and needs to be extrapolated instead of the better alternative of interpolation.

Example 2. Figure 4.3 shows 128 randomly scattered points in 1, 2 and 3 dimensions. Notice how the density decreases with increasing dimension.

Example 3. For a 16 DOF model one needs $10^{16} = 10$ quadrillion datapoints to acquire a density of 10 samples per unit volume. Millions of gigabytes would be needed just to store the samples.

Example 4. In 2 dimensions it is sometimes feasible to use an exhaustive search. An example of this is the Hough transform [1], where the search is done through the $\rho\theta$ space of line responses on images.

4.4.1 Overcoming the Curse

One way to overcome the curse in the context of tracking is to perform a directed search. Let the search be in an n dimensional space with a grid of g grid lines in each direction.

1. Use the information about the most recent² location and assume that the tracked object cannot travel more than $R < g$ grid steps in one time step. This reduces the volume of the (discrete) search space from $\mathcal{O}(g^n)$ to $\mathcal{O}(R^n)$.

²In the Bayesian case, the most recent estimate

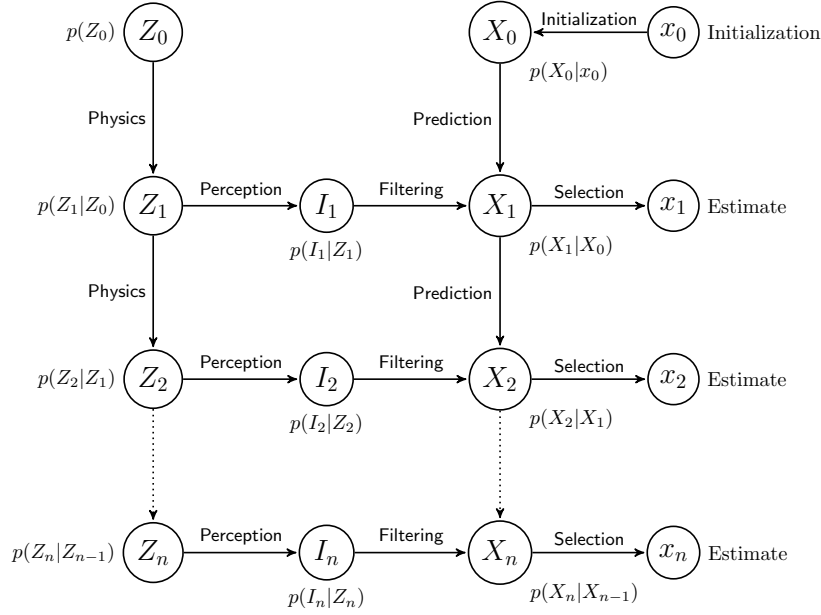


Figure 4.4. Schematic image of the particle filter alongside a HMM.

2. With prior knowledge of how the tracked objects move ³ we can direct our search to specific regions in the state space, depending on how probable it is for the tracked object to be located there. This reduces the size of the search space depending on how sure we are of the previous state.

4.5 The Particle Filter

One naive way to compute the state Z_t would be to perform an exhaustive search in the state space, and select the state for which $p(I_t|Z_t)$ is maximised. <ref:COD>

The particle filter is a technique for reducing the search space <ref:COD>. It uses a finite set X_t of hypotheses to approximate the PDF $p(Z_t|Z_{t-1})$ of a HMM. The hypotheses X_t are also referred to as *particles*, thereby the term “particle filter”.

Figure 4.4 shows the 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 PDF $p(Z_t|Z_{t-1})$ as a set X_t of discrete hypotheses.

More particles mean greater accuracy, since the PDF can then be approximated more closely. However, using many particles increases computational cost. Therefore the number of particles is an important trade-off. The particle filter employs a

³Such as the state transition probabilities $p(Z_t|Z_{t-1})$ in a HMM

4.5. THE PARTICLE FILTER

few tricks to *filter* the hypotheses, tending to keep probable ones and throwing improbable ones away, in order to intelligently reduce the number of particles needed for a good approximation. The filter works in four steps:

Prediction The hypotheses X_{t-1} are updated in the *prediction* step to an approximation \bar{X}_t of $p(Z_t|Z_{t-1})$. This is done by drawing new samples $p(x_t|x_{t-1})$, for each x_{t-1} in X_{t-1} .

Perception By measuring the state of the system, we gain an *observation* $I_t \sim p(I_t|Z_t)$ of the state Z_t .

Filtering The observation I_t of the system is then used for filtering bad hypotheses out of \bar{X}_t . We draw samples X_t from \bar{X}_t with probabilities given by $p(I_t|\bar{X}_t)$. The result will be a surjection, where X_t will be a subset of \bar{X}_t where more probable hypotheses appear multiple times. For this reason, this is also known as the *resampling* step. The set X_t is the *belief*, our approximation of $p(Z_t|Z_{t-1})$.

Selection Finally, we produce a single hypothesis x_t from X_t as our *estimate* of the state Z_t . Assuming X_t is a good approximation of $p(Z_t|Z_{t-1})$, and that $p(Z_t|Z_{t-1})$ is unimodal, the means value of X_t is a good estimate since it approximates the expectation of $p(Z_t|Z_{t-1})$. If $p(Z_t|Z_{t-1})$ is multimodal, however, the mean could be a bad estimate since the expectation may be very improbable.

The reason why this works is the following theorem, which is Theorem 3.1 in [5] with some modifications to notation:

Theorem 1. Given the PDFs $p(x_t|x_{t-1})$, $p(I_t|x_t)$ and $p(x_{t-1}|I_{t-1} \wedge \dots \wedge I_0)$, the PDF $p(x_t|I_t \wedge \dots \wedge I_0)$ can be expressed as

$$p(x_t|I_t \wedge \dots \wedge I_0) = \kappa p(I_t|x_t) \int p(x_t|x_{t-1}) p(x_{t-1}|I_{t-1} \wedge \dots \wedge I_0) dx_{t-1}, \quad (4.2)$$

where κ is a normalization constant

For the proof of this, see [5].

Let's sketch out the connection between this and what we've done so far. For a more rigorous derivation, see chapter 4 of [6].

In this thesis, $p(x_t|I_t \wedge \dots \wedge I_0)$ is approximated as the set X_t of hypotheses. Substituting this into (4.2) and replacing the integral with a sum yields

$$X_t = \kappa p(I_t|x_t) \sum p(x_t|x_{t-1}) X_{t-1}. \quad (4.3)$$

Note that this is horrible abuse of notation, but from here we can take the step to

$$X_t \sim p(I_t|x_t) p(x_t|x_{t-1}) \quad (4.4)$$

```

PARTICLE-FILTER( $X_{t-1}, I_t$ )
1   $\bar{X}_t \leftarrow \emptyset$ 
2  for each  $x_{t-1} \in X_{t-1}$ 
3      do
4           $x_t \leftarrow \text{PREDICT}(x_{t-1})$ 
5           $w \leftarrow \text{IMPORTANCE}(x_t, I_t)$ 
6          Append  $\langle x_t, w \rangle$  to  $\bar{X}_t$ 
7
8   $X_t \leftarrow \emptyset$ 
9  while  $|X_t| < |\bar{X}_t|$ 
10     do
11         Take  $\langle x_t, w \rangle$  from  $\bar{X}_t$  with probability  $\propto w$ 
12         Append  $x_t$  to  $X_t$ 
13 return  $X_t$ 

```

Table 4.1. The particle filter algorithm.

4.5.1 The Particle Filter algorithm

Table 4.1 shows the particle filter algorithm. Note that the functions PREDICT and IMPORTANCE are unspecified - they are problem specific. They correspond to the PDF $p(x_t|x_{t-1})$ and $p(I_t|x_t)$, respectively.

In this thesis, the real x_{t-1} is not known. Rather x_{t-1} is estimated with a set X_{t-1} of N particles.

4.6 Visual Cues

The biggest problem with computer vision is that computers do not have vision, only a data input device in the form of a camera.

A *visual cue* is an image transformation ϕ that extracts some property of the image, such as intensity⁴, edges or ridges[5].

<image showing the use of ϕ >

Theorem 2. Let f be a positive Riemann function with compact support, defined on a set Ω . Then

$$\operatorname{argmax}_{\bar{e}} \left(\int_{\Omega} f(\bar{x})f(\bar{x} - \bar{e}) \right) = 0 \quad (4.5)$$

Proof. Firstly define the window function as

⁴In our case this is possible since we have an homogenous object in the form of a backlit rodent

4.6. VISUAL CUES

$$\begin{aligned}
 W_a^b(x) &= \begin{cases} 0, & x < a \\ 1, & a \leq x \leq b \\ 0, & x > b \end{cases} \\
 \text{Multiplication:} \quad (W_a^b W_c^d)(x) &= W_{\max(a,c)}^{\min(b,d)}(x) \\
 \text{Translation:} \quad W_a^b(x-e) &= W_{a+e}^{b+e}(x) \\
 \text{Integration:} \quad \int_{\mathbb{R}} W_a^b(x) dx &= \Theta(b-a).
 \end{aligned}$$

$$\begin{aligned}
 \operatorname{argmax}_e \left(\int W_a^b(x) W_a^b(x-e) \right) &= \\
 \operatorname{argmax}_e \left(\int W_a^b(x) W_{a+e}^{b+e}(x) \right) &= \\
 \operatorname{argmax}_e \left(\int W_{\max(a,a+e)}^{\min(b,b+e)}(x) \right) &= \\
 \operatorname{argmax}_e (\Theta(\min(b, b+e) - \max(a, a+e))) &= 0.
 \end{aligned}$$

This trivially holds for superposition of windows, since all windows will scale and translate the same way. With a finite support, $e = 0$ is the only solution. Additionally, this also holds in higher finite dimensions since we can just repeat the process one dimension at a time.

All riemann functions can be written as a superposition of windows like this

$$f(x) = \sum c_i W_{a_i}^{b_i}(x),$$

and therefore (4.5) holds for any Riemann function f with finite support. \square

Chapter 5

Algorithms and Implementations

The testing implementation was developed with high modularity in mind, since it is meant to be a proof-of-concept implementation and not a production grade system. High modularity also makes development easier and the system more robust against changes, two very important qualities during this project.

The implementation is written in Python and consists of three main parts:

Particle Filter A direct implementation of the procedure in table 4.1.

Database A database with functions for extracting transition hypotheses. Provides the prediction PDF $p(x_t|x_{t-1})$ to the particle filter.

Tracker Manages the model and performs matching between hypotheses and images. Provides the filtering PDF $p(I_n|x_n)$ to the particle filter.

5.1 The particle filter

The particle filter implementation is a direct implementation of the procedure in table 4.1. It is implemented as a function that takes the parameters X_{t-1} , I_t , **importance_function** and **sampling_function**. The parameters are the hypotheses from the last time step, the current video frame and the functions to use as PREDICT and IMPORTANCE in 4.1, respectively. This means that the particle filter function is general and independent of the model used. The implementations of PREDICT and IMPORTANCE are provided by the database and tracker, respectively.

5.1.1 Initialization x_0

The test implementation needs to be manually initialized. When tracking generated whiskers, the states were always known and program could therefore be programmatically inserted. When testing on real whiskers, the start states were calculated by manually selecting five or six pixels along each whisker and using a MATLAB script to find the least squares solution for the coefficients $\langle a_1, a_2, a_3 \rangle$. The problem of automatic initialization is a difficult one [5], and is not covered in this thesis.

5.2 The state transition database

5.2.1 Data format

A *state transition* is a pair $\langle x_{\text{from}}, x_{\text{to}} \rangle$ that denotes we have observed a system go from state x_{from} to state x_{to} in one time step. Technically, the state transition database is implemented as an SQLite3 database. One transition is represented in the database as a row with the state parameters of the model before and after the transition.

5.2.2 Prediction $p(x_t|x_{t-1})$

```

DB-PREDICT( $x_{t-1}$ )
1   $x_t \leftarrow 0$ 
2   $W \leftarrow 0$ 
3  for each  $(x_{\text{from}}, x_{\text{to}}) \in \text{DATABASE}$ 
4      do
5           $w \leftarrow (||x_{\text{from}} - x_t||_{L^p})^{-a}$ 
6           $x_t \leftarrow x_t + w \cdot x_{\text{to}}$ 
7           $W \leftarrow W + w$ 
8  return  $x_t/W$ 
    
```

Table 5.1. Pseudocode for the prediction function, with the parameters a and p .

The PREDICT function in table 4.1 is implemented as a weighted mean of the state transitions in the database. The function is stated in table 5.1. Notice the parameters a and p . p is a positive integer that determines which L^p space to compute the norm in. a is a positive number, and determines how fast the weight w declines with the distance $||x_{\text{from}} - x_{t-1}||_{L^p}$. A high a means closer transitions get a much higher weight than ones far away, see figure 5.1.

5.3 Tracker

The tracker uses the whisker model described in chapter 4 and internally represents whiskers with the tuple $\langle a_3, a_2, a_1 \rangle$ of polynomial coefficients.

5.3.1 Filtering $p(I_t|x_t)$

The IMPORTANCE function in table 4.1 is implemented as follows. Given a hypothesis x_t , the tracker renders an image $I_{x_t} = \text{Render}(x_t)$ depicting the whisker shape represented by x_t . An example can be seen in figure 5.2. The image I_{x_t} is then multiplied with I_t . This has the result of “masking” the image I_t , extracting the

5.4. PREPROCESSING OF REAL IMAGES

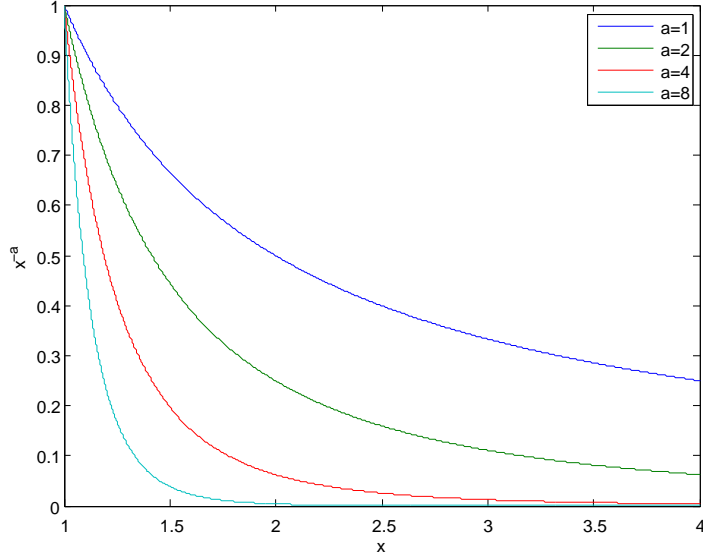


Figure 5.1. x^{-a} versus x .

pixels that the whisker would occupy if its shape were given by x_t . The importance is then calculated as the sum of the pixels of $I_{x_t} * I_t$, meaning the importance is high if the image pixels along x_t are bright.



Figure 5.2. An example I_{x_t} .

5.4 Preprocessing of real images

In the images of real rodents used for testing, the rodent was illuminated from below, meaning it and its whiskers were dark against a bright background. The filtering function described in the previous section expects whisker pixels to be bright, so the images had to be preprocessed.

Chapter 6

Results

<choose the word:(performance vs fitness)>

The bake-off <ref> setup for the experimental parameter evaluation<ref>:

Evaluated parameters:

- p The norm (where)
- a Penelty for the norm
- σ Resample blur (?)
- ϕ Transformation
- N Number of transitions
- n Particle count

will be tested on a small set of benchmarks consisting of 4 disperse(d?)/scatter generated whisker videos.

The evaluation tensor:

$$\Phi_{p,a,N,\dots}(\textit{benchmark}) \tag{6.1}$$

TODO (check the comments here)

The measures that will be used as fitness of the parameters:

$\int ||\epsilon(t)||_1 dt$ Integrating over time the the difference with the ground truth (do this for L1:10 and see if it correlates with the p choosed (to see how much it deviates from the ground truth)

$\int \sum \phi(B(t)) dt$ Integrating over time the response for the choosed hypothesis (to see how the different image transformation affects the results, that is if it only follows what it thinks is best (phi))

Subjective image samples

And all this are done for all 4 benchmark videos.

"There are many metrics by which a model may be assessed." - Encyclopedia

The fitness test was runned on different machines but this wont effect the result since we initially wont consider the running time.

The runtime for the algorithm is handled separately on one machine setup <...>

Tillvägagångsätt:

1. Since we have prior knowledge about the effect off varying the parameters n,N they will firstly be set to a sufficently large value.
2. A partion of the test-matrix

will then be evaluated by ... parameter group

Chapter 7

Analysis and Discussion

One must be careful doing this type of experimental analysis on generated whiskers and then applying the conclusions on real whiskers, but we could still use the best parameters as a starting point for further studies on real whiskers.

"However, much machine learning research includes experimental studies in which algorithms are compared using a set of data sets with little or no consideration given to what class of applications those data sets might represent. It is dangerous to draw general conclusions about relative performance on any application from relative performance on this sample of some unknown class of applications. Such experimental evaluation has become known disparagingly as a bake-off." - Encyclopedia (>algorithm evaluation)

"A learning algorithm must interpolate appropriate predictions for regions of the instance space that are not included in the training data." - Encyclopedia (>model evaluation)

Can a particle filter overfit? (should this perhaps be in theory?)

Bibliography

- [1] Rafael C. Gonzlez and Richard E. Woods. *Digital Image Processing, Third Edition*. Pearson Education, 2008.
- [2] Jason T. Ritt. High-speed videography of embodied active sensing in the rodent whisker system. In Tommaso Fellin and Michael Halassa, editors, *Neuronal Network Analysis*, volume 67 of *Neuromethods*, pages 283–302. Humana Press, 2012.
- [3] Snigdha Roy, Jerí L. Bryant, Ying Cao, and Detlef H. Heck. High-precision, three-dimensional tracking of mouse whisker movements with optical motion capture technology. *Frontiers in Behavioral Neuroscience*, 5(00027), 2011.
- [4] Claude Sammut and Geoffrey Webb. *Encyclopedia of Machine Learning*. Springer, 2010 (First edition).
- [5] Hedvig Sidenbladh. *Probabilistic Tracking and Reconstruction of 3D Human Motion in Monocular Video Sequences*. PhD thesis, Kungliga Tekniska Hogskolan, 2001.
- [6] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics*. The MIT Press, 2006.
- [7] Jakob Voigts, Bert Sakmann, and Tansu Celikel. Unsupervised whisker tracking in unrestrained behaving animals. *Journal of Neurophysiology*, 100:504–515, 2008.