KTH Computer Science
and Communication

Probabilistic Tracking of Multiple Rodent Whiskers in Monocular Video Sequences WORK IN PROGRESS

A small step towards cheap, reliable, and nonintrusive 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: Örjan Ekeberg
Examiner: ??? Örjan Ekeberg ???

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 when whiskers cross or overlap. A cheap, reliable solution to the tracking problem is needed.

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 searching a pretrained database, and filtering the results through the Particle Filter. The implementation is written in Python using NumPy and an SQLite3 database.

There are two main strengths of the proposed solution. First, it successfully tracks multiple whiskers at once, even when they cross or overlap. Second, being a standalone program operating on pre-recorded video, it does not notably restrict the experiment.

Referat

Statistisk Följning av Multipla Morrhår på Gnagare i Video.

Contents

Chapter 1

Introduction

1.0.1 Background

This section is WORK IN PROGRESS and is riddled with notes and whatnot.

- Video tracking (finding) is interesting - Biologists want to track the movements of animals or parts of animals - Most of the software available for this purpose is expensive - Most of the software available cannot handle too many whiskers at once (clutter) - Hedvig developed a way to track human motion in 3d using monocular video - We want to use the same method for tracking whiskers (different mechanical model) - What does Ekeberg want and why?

KEYWORDS - Spatial/temporal resolution

PROSE (kind of) BELOW THIS POINT

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 of animals or parts of animals. One such field aims to study the movements of rodent whiskers, including how they are used for perception and their correlation with neural signals. However, most whisker tracking software available today suffers a few fundamental flaws. First and foremost, they are often so expensive that not even well funded laboratories feel they can afford them. Cheaper solutions often have problems with 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 [1]. Such intrusive methods of tracking may cause systematic errors and not give a representative view of how the animal naturally uses its whiskers.

1.0.2 Difficulties

In general, the main difficulty in tracking and localization is to separate the tracked object from clutter such as similiar objects and occlusion. In the case of whisker tracking, the dominant problems are that whiskers often overlap and vary in size, and in low resolution images it is difficult to tell how far they extend.

In 2001, Hedvig Sidenbladh investigated probabilistic methods for tracking three-dimensional human motion in monocular video [2]. Many of the problems inherent in computer vision were regarded, and

1.0.3 The goal of the thesis

(Vad Hedvig introducerat som e najz) The goal of this thesis is to apply the same kind of method used by Sidenbladh to tracking whiskers. This thesis aims to investigate whether this is feasible, and to evaluate some tracking models if it is.

1.0.4 Contributions of this thesis

This thesis proposes a probabilistic tracking method using

Chapter 2

Related Work

Chapter 3

Definitions

3.0.5 Invariances

Section about invariances and why they are needed in localization.

The "filter" must give the same response invariant of the position in the image and its often wanted to have it rotational and scale invariant (if your not intending to measure things like angles or size that is)

3.0.6 Data

Image

In a computer an image is represented as an array of numbers, mostly 8bit numbers ranging between [0,255]

[should one use N or [0,255] instead of Z?]

RGB image is a function

image: ZxZ=pos->ZxZxZ=color (mapping each position into an RGBcolor) that is foreach pixel a value for Red,Green,Blue is defined. But one might as well see it as a list of pixels $Z^5 =< pos, color >$

Grayscale image is in the same way as an RGB image but without the RGB part

just a function; image: ZxZ->Z or alternatively list of $Z^3 =< pos, value >$

Video

Ordered list of images

Feature

Feature Space

One configuration of the object corresponds to one point in the feature space <example with a plot>

Hypothesis

Explain: Hypotesis (in the context of PF), sample, DOF?,

Prop. function as a collection of points with attached weight.

Dynamic system

It's a tuple <space,update,time> Any mechanical system following newton physics (or relativistic for that matter) can be considered to be an dynamic system in our case the tuple whould be <feature_space,update_rule (the one we are trying to preprocess data to approximate),time>

Chapter 4

Background

To track something in a dynamic system with uncertainties one needs a tracker and a localizer. The purpose of a tracker is to narrow the search range, see section **??**, so that only the currently relevant hypotheses are included in the search. For the tracker get some perception of the system one needs a localizer. $tracker_{localizer}(...)$

4.0.7 Tracking

Recursive bayesian estimator /Bayesian filter

A theoretical basis for the tracking filters, can't be realized in real life.

Kalman Filter

A more analytic version of the Bayesian filter

Assumptions:	Normal distributed data, linear process
Pros:	Fast
Cons:	Can't deal with multimodal distributions and may perform poorly with distributions that are not approximately normal distributed.

Particle Filter

A Monte Carlo version of the Bayesian filter which

Assumptions:	Not much (or?)
Pros:	Can deal with multimodal distributions.
Cons:	Slow

There are many variants of both kalman filter and particle filter but only the basic ones will be covered in this thesis.

4.0.8 Localization

Contour tracking(/based?)

Matching edges in the image to the edges in the thing you want to match

Blob tracking(/based?)

TODO

Visual feature matching

TODO

Kernel-based tracking

TODO

For this thesis, the particle filter was chosen. There are a number of reasons for this. First, the behavior of the feature space for a system of whiskers was not known, which eliminates parameterized filters like the Kalman filter. Second, there was no strict constraint on the running time of the program, meaning heavier computational tasks can be afforded.

Chapter 5

Theory

5.0.9 The particle filter

Introduction

The core of tracking engine used is a technique known as the *particle filter*. The particle filter is a kind of Bayesian filtering where one uses discrete hypotheses to approximate continuous probability distributions [3]. The main idea can be outlined as follows.

Suppose we have a system described by the state x_t at time t. x_t can be thought of as a vector of state parameters. Suppose that we know the previous state x_{t-1} and have an observation z_t of the system at the current time. In general, it is difficult to accurately determine x_t from the observation alone, and the observation may suffer from interference. Therefore, we cannot directly read x_t from z_t. However, we can estimate x_t if we know the following things about the system:

- A probability function p, where $p\left(x_t|x_{t-1}\right)$ is the probability that the current state is x_t if the previous state was x_{t-1}.
- A probability function q, where $q\left(z_t|x_t\right)$ is the probability that we observe z_t if the current state is x_t.

Using p and knowing x_{t-1}, we can generate a set of hypotheses $\{\bar{x}_t^m\}_{t=1}^N$ for the current state x_t. Using q and knowing z_t, we can evaluate how likely the hypotheses are. If it is likely to observe z_t if the current state is x_t, then x_t is probably a good estimate of the current state. With this information, we select the most probable hypotheses and let them be our estimate of the current system state.

The particle filter works recursively in two steps:

1. The *sampling step* is the generation of the hypotheses, known as *particles*, from the previous state x_{t-1}. The belief $bel\left(x_{t-1}\right)$, see below, is used as an estimate for x_{t-1}. What makes this recursive is the fact that $bel\left(x_t\right)$ is calculated using $bel\left(x_{t-1}\right)$.

2. The *resampling step* is the final selection of the most probable particles. After resampling the set $\bar{X}_t := \{\bar{x}_t^m\}_{t=1}^N$ we get the *belief* $bel\,(x_t)$, which often includes multiple copies of the most probable particles. This set is used as an estimate for x_t, and is used to estimate x_{t+1}.

In the next section, the particle filter algorithm will be stated. Implementing the filter in itself is only a matter of implementing the stated pseudocode, and is not difficult. The difficult part is designing the probability functions p and q for the given system. A probabilistic implementation of these functions is proposed in chapters TODO.

Formal description of the particle filter

Here the particle filter algorithm is stated. An elaboration on what this actually does and why is offered below.

DISTRIBUTION-SAMPLE(X_t, p)
1 **foreach** x_t **in** X_t
2 **do** *sample* $x_{t+1} \sim p\,(x_{t+1}|x_t)$
3 **return** X_{t+1}

IMPORTANCE(X, q, z)
1 **foreach** x **in** X
2 **do** $w \leftarrow q\,(z|x)$
3 **return** W

WEIGHTED-SAMPLE(X, W)
1 **foreach** x **in** X
2 **do** *sample* $x' \propto W$
3 **return** X'

PARTICLE-FILTER(X_{t-1}, z_t)
1 $X_t \leftarrow$ DISTRIBUTION-SAMPLE(X_{t-1}, p)
2 $W_t \leftarrow$ IMPORTANCE(X_{t-1}, q, z_t)
3 $X_t \leftarrow$ WEIGHTED-SAMPLE(X_t, W_t)
4 **return** X_t

PARTICLE-FILTER(X_{t-1}, z_t)

```
1  Let X̄_t = ∅.
2  For m := 1 to |X_{t-1}|:
3        do Draw x_t^m with probability p(x_t^m | x_{t-1}^m).
4           Let w_t^m := q(z_t | x_t^m).
5           Add (x_t^m, w_t^m) to X̄_t.
6  Let X_t := ∅.
7  For m := 1 to |X_{t-1}|:
8        do Draw x_t^m with probability proportional to w_t^m
9           Add x_t^m to X_t.
10 Return X_t.
```

We draw a set $\bar{X}_t = \{x_m^t\}_{m=1}^N$ of samples from p. These samples roughly represent a probability distribution for the current state x_t, but we have yet to consider our observation. Therefore, we will create a new probability distribution weighted by how probable the observation z_t is. For each x_t^m, we let $w_t^m := q(z_t|x_t)$. This defines a discrete probability distribution where x_t^m is assumed with a probability proportional to w_t^m. From this final distribution we again draw N samples X_t, which will be our estimate of the current state. The elements x_t^m are referred to as *particles* and the set X_t as the *belief at time t*.

In this thesis, x_{t-1} is not known. Rather x_{t-1} is estimated with a set X_{t-1} of N particles. In this case, x_t^m is sampled with probability $p(x_t^m|x_{t-1}^m)$.

Curse of dimensonality

from: "R. Bellman, Adaptive control Processes, p.94, Princeton University Press, NJ, 1961."

"In view of all that we have said in the foregoing sections, the many obstacles we appear to have surmounted. What casts the pall over our victory celebration? It is the curse of dimensionality, a malediction that has plagued the scientist from earliest days."

show curse of dimensionallity with a simple like in bik12.lecture7 1D->2D either maintain density samples increases drastically or maintain samples and density decreases drastically 2D->3D even worse

To have a sample density of C we need $O(C^n)$ samples for n-dimensional data, that is the number of samples grows exponential on the number of dimensions. And a consequence of this is that in order to approximate some function defined in higher dimensional space one needs many more samples.

One more thing with the high dimensional space is the large boundaries of the sample-set then from lower dimensional space which results in orders of magnitude higher chance for an point one want to approximate to fall outside the sample-set and needs to be extrapolated instead of interpolate which is much better.

The 8 DOF in our model produces vast amounts of space in our feature space and this has the consequence that we need proportional to n^8 datapoints to fill it

to an specified density c.

For instance, we would need $4^8 = 65536$ samples just to make a grid with 4 samples in each DOF which in many cases (aspecially for hand labeled data).

This phenomen of having hue searchspace in highdimensinonal space is fairly common and have the name "Curse of dimensionality".

One way to somewhat overcome this emptyness in space is to have a dynamic (active?) algorithm that adopts the sample density according to the models relative frequency in that area and this results in, for a given amount of samples, its more likely for an sampled model to occure in a more densly pre-sampled (pre-sampled?) area, in fact this method when doing it right (ideally) gives: given a set of samples the overall density for all the samples in the sampledatabase would be optimal) [proof for this will be given in <blabla>] ... (use the world hypotesis instead of sample, or perhaps sampled-hypotesis) (below is perhaps somewhat redudant, but one can pick bits and pieces from both) So having a bias towards trying out more plausible hypotesis for the models innerstate is better than doing a naive exhausted search tru the entire feature space, this could only be used if you have ≤ 3 DOF as in for example finding straight edges with houghtransformation[ref]. ... One method that uses prior knowledge of the models PDF and a pretrained database containing knowledge on how to approximate the models PDF in the next timestep is particle filter which is an (instance?) of the ideal bayesian filter.

5.0.10 Particle filtering whisker movements

Here we propose a way to model whiskers as a dynamic system.

5.0.11 Kinematic whisker models

In all our models we have separated the head from the whiskers since they have such different kinematic properties and actually are attached to each other.

==The first model Assumptions: >Material is linear elastic >Small deformations (probably the biggest assumption, which dont actually hold in our case)

The equation of elastic line ... [Grundläggande Hållfasthetslära - Hans Lundh p94 (7.6)]

The force that comes from the head moving on the base of the whisker is just sucked up by the Boundaryvalues and it will still be valid assumptions for the elasticline to hold.

Under just a few assumptions that the material is linear elastic and the deformations are small we have the ...

========== MODELS ==========

One possible model is to borrow the model for beam under small deformations from the theory of strength of materials, after all the whisker is a beam but we dont have small deformations at all but we assume that the model will approximatly hold ony way.

Chapter 6

Algorithms & Implementations

6.0.12 The particle filter

6.0.13 The state transition database

Chapter 7

Benchmarking & Results

Chapter 8

Analysis & Discussion

Bibliography

[1] 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.

[2] Hedvig Sidenbladh. *Probabilistic tracking of 3D human motion in monocular video sequences*. PhD thesis, Kungliga Tekniska Hogskolan, 2001.

[3] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics*. The MIT Press, 2006.