A System for Automated Hand Pose Estimation in Neurophysiological Research

Tobias Palmér

Master’s thesis
2011:E2
Abstract

The goal of this Master’s thesis has been to develop an algorithm that can track the spatial location of the rat forepaw as well as detailed movements around multiple joints when the animals engage in a skilled reaching task. The kinematic data obtained from the behavioural analysis is subsequently matched to neuronal activity patterns recorded in different motor related brain structures during reaching in order to investigate how the nervous system learns to control skilled movements. The method uses an articulated 3D model whose projection is fitted to the video by maximizing a function that estimates matching quality using segmented silhouettes and edges in the images. The matching function is maximized by a mixture of iterative methods and database search which take advantage of the stereotypical aspects of paw movements to build a computationally efficient method. Physical constraints are added to make sure the results are meaningful and to decrease the size of the pose space. Part of the work has also been devoted to adapting the experimental setup to allow for automatic tracking of specific motor behaviors. The result is an experimental setup that features two cameras acquiring images at 200 frames per second and, in addition to the cameras, three mirrors producing a total of six views used in tracking. Promising tracking results are shown as well as preliminary neurophysiological data recorded from the first animal.
I would like to thank my supervisors Olof Enqvist and Per Petersson for their great support and inspiration while keeping me on track with the project. I would also like to thank Martin Tamtè for helping me perform the experiments and Pär Halje for generating plots from the neural recordings. Lastly, I would like to thank the people at NRC, most importantly the people connected to Per Petersson’s research, for creating an inspiring environment.
Contents

1 Introduction 1

2 Background 3
  2.1 Neural Signaling 3
  2.2 Skilled Reaching 5
  2.3 Goals 5

3 Computer Vision and 3D Geometry 7
  3.1 Projections 7
  3.2 Triangulation 9
  3.3 Foreground Segmentation 9

4 Hand Pose Estimation 12
  4.1 The Hand Model 13
  4.2 Multiple Views for Hand Pose Estimation 13
  4.3 Matching Functions and Search Algorithms 15
  4.4 Database Search 17

5 Experiments 18
  5.1 Implementation Details 18
  5.2 Results and Discussion 19
  5.3 Neural Results 25

6 Future Work 27
  6.1 The Software 27
  6.2 The Experimental Setup 28

Appendices 30

A Using Point Features 31

B Some Practical Details 32
  B.1 Training Protocol 32
  B.2 Data Gathering Protocol 32
List of Figures

2.1 The components of a neuron and examples of nerve cell morphologies ........................................ 4
2.2 Extracellular recording, where the cone symbolizes an electrode, and an action potential. ..................... 4
2.3 The behavioural box with notations for some of its features. The shelf is 30 mm deep and placed 40 mm above the ground. The slit is 13 mm wide and the three sockets are located 15 mm from the wall and 6.5 mm apart. The obstacle is 40 mm high and placed at the center of the box. ........................................ 6

3.1 Projection from space onto an image plane. .................. 7
3.2 Observed image, background image, foreground image and edge image ........................................... 11

4.1 Examples of poses with different parameters. Poses in the first row have the same wrist parameters but different digit parameters (flexion/extension and adduction/abduction). Poses in the second row have the same finger parameters but different wrist parameters (supination/pronation). The red, green, blue and cyan lines make up the skeleton of the digits. ............... 14
4.2 Visualization of the visual hull for four silhouette images used for this project. The paw can be seen clearly and from some views also a piece of pasta the rat is reaching for. .................. 15

5.1 Estimated camera positions and directions. ................ 19
5.2 The calibration tool. The red, green and blue rectangles have been manually selected as bounds for three views that will be used for pose estimation. ........................................... 20
5.3 The pose fitting tool. The axes in the 3D plot are 50 mm long. Note that the 3D plot views the mesh of a model for the nose that is not yet automatically tracked. There are also 3D meshes for the pellet and forearm, none of which are automatically tracked. ........................................... 21
5.4 A typical pose estimation result. .................. 23
5.5 Pose estimation result. The estimation of the palm is a bit off and the fifth digit (cyan) is inaccurate. ............... 23
5.6 Pose estimation result. The food pellet is estimated as part of the second digit (red). .................. 24
5.7 Pose estimation result. Estimation of the fifth digit (cyan) is a bit off. ............... 24
5.8 LFP plots for selected electrode channels ........................................ 25
5.9 Spike plots for selected neurons ................................................. 26

6.1 The paw placed inside the paw. .................................................. 28
6.2 A lift-off sequence from the side view. Note that part of the left forelimb cannot be seen in the fourth image due to the long exposure time in this camera. .................................................. 29

B.1 A trial sequence. ................................................................. 33
Chapter 1

Introduction

The goal of this Master’s thesis has been to develop algorithms that can track position and articulation of a rat paw in a skilled reaching experiment. Most parts of the method used are general and can be used on basically any articulated object with only minor changes in the code created for the task, making extensions of the program to track other parts of the rat easy to implement. The method uses an articulated 3D model that can be projected on the video to evaluate how good it matches the video. The matching quality for a pose is a measure of how well silhouette images generated from the pose for each view matches foreground images computed from the observed images, where the foreground consists of the pixels that are not deemed to be part of the static background. Finding the best pose for a frame is done by minimizing the cost function (maximizing the matching quality), i.e. it can be seen as a regular optimization problem of some function. The methods implemented here uses physical constraints to efficiently find a plausible pose.

The goal of the experiments is to find out if specific kinds of reaching attempts are connected to the activity of specific nerve cells, i.e. to see how the activity of some specific parts of the brain can select and control behaviours. This means that the program needs to be able to classify different kinds of reaching attempts in a reliable way. There are many more topics that can be studied using the software provided; for example, by analyzing the skilled reaching behaviour of rats it is possible to detect minor motor disturbances such as those occurring in early stages of Parkinson’s disease. It is also possible to analyze how the skilled reaching behaviour changes as a result of learning, drugs, deep brain stimulation, different reaching targets, etc. There are aspects of the skilled reaching experiment that has not yet been explored and this project combines two of those areas: advanced grip analysis and neurophysiology.

Overview of the Thesis

Chapter 2 explains some of the ideas that motivates the experiment. Chapter 3 explains some of the background needed to understand the computer vision problem including the camera calibration used for this particular camera setup. Chapter 4 introduces hand pose estimation and explains the fundamental algorithms used in this project and Appendix A describes an alternative method that could be used to improve tracking. Chapter 5 provides an evaluation of the
pose estimation on real-world data and presents some preliminary neurophysiological results. And, finally, Chapter 6 discusses possible improvements in both software and experimental setup.
Chapter 2

Background

Neuronano Research Center (NRC) is an interdisciplinary research collaboration coordinated by the Section for Neurophysiology at the Biomedical Center (BMC) in Lund that combines the areas of neuroscience, nano- and microtechnology, and biotechnology. Within the area of neuroscience lies the subject of neurophysiology. In some dictionaries, physiology is defined as "the science of the normal functions and phenomena of living things" and, furthermore, neurophysiology is defined as "the branch of physiology that deals with the functions of the nervous system".

2.1 Neural Signaling

The nervous system consists of cells of two categories - nerve cells (neurons) and neuroglia (glia). Neurons are specialized in communication with other cells (other neurons, muscle cells, etc.). They transmit information by means of electrical signals and the characteristics of those signals can be measured by electrodes. In this signal transduction, particularly important parts of the anatomy of the neuron are the dendrites (input) and axons (output). Figure 2.1 presents the components of a neuron as well as examples of different structural arrangements of dendrites and axons.

A resting neuron has a voltage across its membrane, called the resting membrane potential, with a value that differs slightly between neurons. Measurement of neuronal electrical signals can be done in two ways: intracellular recordings (with the electrode placed inside the cell) and extracellular recordings (with the electrode placed near the cell). There are different kinds of neuronal electrical signals that are generated by changes in the membrane potential. Sensory neurons produce a receptor potential that changes their resting membrane potential, generating the electrical signals that carry information about a sensory stimulus. Synaptic potentials are generated when the synapses are activated, and these signals make up the communications between neurons. Action potentials that are sent out through the axons to distant cells in the body (e.g. muscle cells) in response to synaptic activation. Synaptic potentials and receptor potentials have longer durations and lower amplitude than action potentials, and can only be measured properly by intracellular recordings. Action potentials on the other hand can be measured by both extracellular and intracellular...
2.1. NEURAL SIGNALING

(a) Components of a neuron

(b) Pyramidal Cell

(c) Bipolar Cell

Figure 2.1: The components of a neuron (a) and examples of nerve cell morphologies (b), (c). Modified from [1].

Figure 2.2: Extracellular recording, where the cone symbolizes an electrode, and an action potential.

recordings. In the extracellular case, there is also the possibility of measuring action potentials from multiple neurons close enough to the electrode. Figure 2.2 visualizes how the membrane potential changes from the action potential.

The characteristics of the action potential are different between neurons in the sense of how often the cell can fire and to some extent the frequencies (waveform) contained in the signals, but the invoked amplitude is always the same for each neuron. This means that it is possible to distinguish the electrical signals sent out from different neurons - a procedure referred to as **spike sorting** or **spike tracking**.

Recordings of neuronal activity can be divided into two parts: spike trains (high frequency) and **local field potentials** (LFPs; low frequency, that result from the summed activity of a large population of neurons). After spike sorting (which usually consists of time consuming manual work) on recorded data, firing of each cell can be cross-correlated to the behavioural analysis, e.g. if a neuron in a motor related brain structure generates action potentials every time the subject turns right, and never otherwise, it can be assumed that this neuron is taking part in inducing the behaviour. Local field potentials can be seen
as a measure of how much a group of neurons are co-active, and as the LFP-
data do not need much pre-processing it is a simple first step of analysis of
behaviour-neuron correlation.

2.2 Skilled Reaching

The skilled reaching task used here was introduced by Ian Q. Whishaw in 1990
and the proposed methods have since proven useful in a number of studies.
Whishaw has primarily focused only on the behaviour of skilled reaching, in
particular its kinematic components \cite{2},\cite{3},\cite{4}. He has also studied skilled reaching
in humans and could establish great similarities \cite{5}, making the studies of
skilled reaching in rats relevant for studies on for example Parkinson’s disease.
He has also studied the behavioural effects of ischemia (stroke) in the rat brain
\cite{6}, \cite{7}, \cite{8}, \cite{9}. The kinematic extraction in all of the studies was done man-
ually and all except one is analyzed using LBA (Laban Movement Analysis)
or EWMN (Eshkol-Wachman Movement Notation) to enable comparisons be-
tween reaching attempts. For example, Alaverdashvili and Whishaw studied
how a motor cortex stroke impairs individual digit movements during a skilled
reaching experiment \cite{8}. By using both a frontal view and a side view, they
could manually measure angles between phalanges (giving crude estimations of
adduction/abduction and flexion-extension, respectively) and found that a rat
impaired with a motor cortex stroke moves its digits less independently than a
normal rat. The post-stroke reaching success ratio was initially decreased but
could be restored to normal levels after additional training, in which the animals
learned to compensate for the impaired movements.

Articles on skilled reaching commonly feature some measures of success rates
which makes them relevant to this project as well. An attempt is a movement
of the paw toward the food. A trial is a set of attempts to grasp food. A trial
begins when the rat approaches the food pellet for the first time and ends when
the rat either successfully picks up the food or gives up. A successful reach
is a reach where the rat grasps a food pellet and places it in its mouth. These
notations can be used to define some measures of success rates for the reaching
behaviours, and can be used as a measure of learning:

1. Total success is defined as the number of successful trials divided by the
total number of trials.

2. First trial success is defined as the number of trials where the first attempt
is successful divided by the total number of trials.

3. Total attempts is the total number of attempts, successful or not.

Appendix B.1 explains how the experiments are prepared and performed, and
Figure 2.3 shows the layout of the behavioural box.

2.3 Goals

The goal of the skilled reaching experiment is to find how the neural activity
diffs between attempts and, most importantly, to be able to predict suc-
cess/failure from neural data. The goal of this project is to provide an appli-
cation which can quantify differences between attempts.
Figure 2.3: The behavioural box with notations for some of its features. The shelf is 30 mm deep and placed 40 mm above the ground. The slit is 13 mm wide and the three sockets are located 15 mm from the wall and 6.5 mm apart. The obstacle is 40 mm high and placed at the center of the box.
Chapter 3

Computer Vision and 3D Geometry

The purpose of the tracking is to find the 3D hypothesis of the rat paw pose that best matches some images. The evaluation of the matching quality of a hypothesis is done by projecting it onto each of the images (i.e. cameras), evaluating how well it fits in each of the images and then combine the matching values of the different cameras to get a scalar matching score for the hypothesis. The first part of the chapter deals with projections - how to project a 3D point and how to estimate the projection matrices. The second part of the chapter explains how to extract moving objects from a video sequence, which is needed when estimating how well a hypothesis matches some images.

3.1 Projections

A camera matrix describes the transformation of 3D points to 2D points in a pinhole camera: $X \mapsto \bar{P}X$, where $\bar{P}$ is a $3 \times 4$ matrix and $X$ a 3D point represented in homogeneous coordinates. The pinhole model is visualized in Figure

![Figure 3.1: Projection from space onto an image plane.](image)
3.1 and similar triangles applied to the figure yields the following equations

\[
\begin{align*}
x/f &= X/Z \quad (3.1a) \\
y/f &= Y/Z \quad (3.1b)
\end{align*}
\]

which can be re-written on matrix form as

\[
\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}. \quad (3.2)
\]

If the object coordinate system is changed from \( \vec{I} \) to an arbitrary coordinate system, described by a rotation \( \vec{R} \) and a translation \( t \):

\[
\vec{X} = \vec{R} [\vec{I} - t]. \quad (3.3)
\]

This results in the following camera equation

\[
\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} [\vec{R} - \vec{R} t] \begin{bmatrix} X_E \\ Y_E \\ Z_E \\ 1 \end{bmatrix}, \quad (3.4)
\]

which can be written on a more compact form by introducing the calibration matrix

\[
\vec{K} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad (3.5)
\]

and the camera matrix

\[
\vec{P} = \vec{K} [\vec{R} - \vec{R} t]. \quad (3.6)
\]

**Definition 3.1 (Camera equation)** The relation between 3D points and their projections are described by the camera equation:

\[
\lambda \vec{x} = \vec{P} \vec{X}, \quad (3.7)
\]

where \( \lambda > 0 \) is the so called depth.

The calibration matrix \( \vec{K} \) described above is in its simplest form, modelling only the focal length of the camera. There are more internal parameters that can be modelled by modifying \( \vec{K} \):

\[
\vec{K} = \begin{bmatrix} \gamma f & s f & x_0 \\ 0 & f & y_0 \\ 0 & 0 & 1 \end{bmatrix}, \quad (3.8)
\]

where \( \gamma \) and \( s \) models the shape of the light sensor elements (length/width relation and skew), \((x_0, y_0)\) is the orthogonal projection of the focal point on the image plane and \( f \) is, as before, the focal length.
CHAPTER 3. COMPUTER VISION AND 3D GEOMETRY

The estimation of the camera matrix is called camera calibration (or sometimes camera resectioning) and can be divided into two parts: extrinsic calibration estimates the camera’s position in space ($\vec{R}$ and $t$) and intrinsic calibration estimates the camera’s internal parameters ($\hat{K}$). The simplest method is called Direct Linear Transformation (DLT) and estimates both the extrinsic and intrinsic parameters using a minimum of 8 equations.

3.2 Triangulation

Triangulation is the estimation of a 3D point from the corresponding image points. Given points in only one image plane triangulation will result in an infinite number of possible 3D points, making it impossible to estimate the shape of an object from only one image. The ambiguity can be resolved only when using multiple image planes (i.e. the object observed from multiple views).

Given the projection matrices $\mathcal{P} = \{\vec{P}_i|i = 1,\ldots,n\}$ for some cameras and one point in each image plane, $\mathcal{X} = \{x_i|i = 1,\ldots,n\}$, the 3D point can be estimated from the equation system containing the camera equations for all cameras. First, by noting that the camera equation implies

$$\vec{x} \times \vec{P}_i \vec{X} = 0,$$

the equation system can be written as

$$\begin{cases} 
\vec{x}_1 \times \vec{P}_1 \vec{X} = 0 \\
\vdots \\
\vec{x}_n \times \vec{P}_n \vec{X} = 0
\end{cases}$$

and using the matrix form of the vector product ($u \times v = [u]_x v$), the equations can be combined into one overconstrained matrix equation

$$\begin{pmatrix} 
[\vec{x}_1]_x \vec{P}_1 \\
\vdots \\
[\vec{x}_n]_x \vec{P}_n
\end{pmatrix} \vec{X} = \vec{A} \vec{X} = 0.$$  \hspace{1cm} (3.11)

The linear least squares solution $\vec{X}$ to 3.11 is given by the nullspace of $\vec{A}$. In practice, as the equation is overconstrained and the projection matrices contain errors, the nullspace does not exist but can be approximated by for example the right singular vector corresponding to the lowest singular value of $\vec{A}$ (the best approximation to the nullspace of $\vec{A}$ in the sense of least squares).

3.3 Foreground Segmentation

The aim of the pose estimation is to find the pose that best matches an image, where the matching quality uses foreground images and edge images from the videos. Foreground segmentation is used to separate the pixels that are projections of rat or reaching target from the pixels that are projections of the static part of the world (i.e. parts of the experimental setup, such as the shelf, the front wall of the behavioural box, etc.). The moving parts (e.g. rat, pellet,
3.3. FOREGROUND SEGMENTATION

pasta) are called foreground and the static parts are called background. In the
ideal case, all pixels corresponding to rat or reaching target are classified as fore-
ground and nothing except the rat or reaching target is classified as foreground.
This is almost never the case in practice due to image noise and slightly varying
backgrounds. A simple way to estimate the background image is to compute the
mean value of each pixel for a few selected frames. A binary foreground image
can be computed by subtracting the background from an image and then set
each pixel that is above some threshold to 1 and the rest to 0. A similar method
was introduced by Wren et al. and models each pixel of the background as a
Gaussian distribution [10]. A pixel is classified as background if its value is less
than some multiple of the standard deviation away from the expectation value.
Each pixel that is classified as background is used to update the expectation
value (background image) as well as the standard deviation (threshold).

The edge detector used in this project is a simple function designed for
quick extraction of edges and edge directions in silhouette images. The method
creates a binary edge image with ones for each pixel that corresponds to a 1 in
the foreground image and has at least one zero valued pixel in a surrounding
3x3 block. All other pixels in the binary edge image are set to zero. Directions
are estimated from the same 3x3 blocks.

The function is created specifically for the task of binary edge detection
because MATLAB’s built-in edge detectors (for example Canny edge detec-
tor; a multi-stage algorithm featuring Gaussian filters, intensity gradients, non-
maximum suppression and hysteresis thresholding) are not - they use the same
methods as for non-binary images. Comparisons with MATLAB’s Canny edge
detector shows that the proposed algorithm is approximately 4 times faster
when considering silhouette images (while it also extracts directions). Another
advantage of the method is that the estimated edge directions can be used to
further constrain the edge-matching functions.

Figure 3.2 illustrates an example of an image (a) with background image (b)
and the silhouette image (c) and edge images (d) computed from it.
Figure 3.2: Foreground image (c) and edges (d) (with edge directions plotted as blue lines) extracted from an observed image (a) with background (b).
Chapter 4

Hand Pose Estimation

The main idea of the hand pose estimation is to use some efficient way to find the specific pose that generates the foreground- and edge images that best matches the actual images extracted from the video. In other words, the pose estimation can be seen as a constrained optimization problem of an $n$-dimensional matching function, where $n$ is the number of degrees of freedom for a pose. In practice, this is done by creating and evaluating hypotheses for the hand parameter configuration and then assuming the best hypothesis as the hand pose for the frame. This chapter introduces the matching functions as well as algorithms for finding local maxima of the matching functions. A good starting guess is provided by the previous pose or from a database of well-matched poses, and is further improved by local optimization. Experience suggests that this often provides results close to what is believed to be the global maximum.

Related Work

The subject of hand pose estimation and tracking has gained a lot of attention the last 5-10 years as the computational power available is approaching what is needed for real-time applications. The main area of application is not studies on rat behaviour but rather to provide a method for natural Human-Computer Interaction (HCI) which is used in virtual environments, games, regular computer input, etc. As the main application of hand pose-trackers is HCI, most research is for worse conditions than considered in this project - for example with cluttered backgrounds and/or using a single camera. But there are also trackers using more information than in this project, for example depth images [11, 12, 13] together with usual video. There are a lot of articles on different methods, for example Erol and Bebis’s review [14] of hand pose estimation methods gives some insight into the main ideas of the field and explains some concepts such as particle filters, multiple hypotheses tracking etc. However, since the most recent article reviewed in this paper was published in 2005, there are many more recent methods not covered here. For example, by combining edge images with ordinary two-dimensional images, [11] reports of successful tracking of hands manipulating objects at a framerate of 6 fps. Wang and Popović used gloves colored in a pattern randomly generated by a computer algorithm. They could successfully track the articulation of the hand at 10 fps, but with some errors in the global pose as only one camera was used [15]. The problem of
finding the global pose in a robust and accurate way was studied by Usabiaga et al. who used 8 cameras and a glove with the fingers in different colors and an ellipse on the back of the palm [16]. Romero et al. used a database of synthetic hand images (each with a 31 parameter description attached to it) combined with conditions to ensure temporal consistency, and could present tracking of 10 fps [17]. Ho et al. used two normal cameras as well as one depth camera, training data from a DataGlove and presented a modification of particle filters, *separable state based particle filters*, better suited for hand pose tracking than standard particle filters [13]. Furthermore, in this setup the angle of the hand was computed by fitting an ellipse to silhouette images and then the camera positions relative to the normal of the palm were estimated using the relation between minor axis length and major axis length of the observed ellipse. From this information, the cameras were classified as either frontal or side, and the later applied probability functions depended on this classification.

### 4.1 The Hand Model

The interesting parts of the tracking for the skilled reaching experiment are the trajectory and the grip of the paw, which motivates the following model. The rat paw is modelled as a set of one palm and 12 phalanges, each with rotation, position and shape. Physical constraints, for example that the phalanges and the palm are connected and rigid parts, decrease the degrees of freedom to the following:

i) 3 parameters for wrist rotation

ii) 3 parameters for wrist position

iii) 1 parameter for deformation of the palm due to movement of the metacarpal bones

iv) 1 parameter for adduction/abduction for each digit

v) 1 parameter for flexion/extension for each of the first two joints of each digit. The third joint angle is too hard to track reliably because the distal phalanx is so small, and is assumed to have the same flexion/extension parameter value as the second joint.

The pose is defined by these 19 parameters and when combined with the constant parameters they can generate a mesh in the global coordinate system in which the wrist parameters are given. Furthermore, the mesh can generate the silhouette images that are used for computing matching quality. Examples of different parameter configurations are given in Figure 4.1. In later analysis, the wrist position gives the trajectory of the paw, while the wrist rotation along with flexion/extension and adduction/abduction gives the digit articulation.

### 4.2 Multiple Views for Hand Pose Estimation

When viewing an unknown object (i.e. a set of 2D points in an image) from only one view, it is not possible to do a 3D reconstruction as the depth parameter for each estimated 3D point is unknown but when adding more views
it becomes possible to estimate both size and position. Similarly, finding the optimal projection of some object to the silhouette image of a view suffers from instability in estimating the distance from the object to the image plane. The silhouette images of multiple views can be used to constrain the problem, decreasing the volume of the visual hull which is the set of all points in space that are projected inside the silhouette in all views. An example of a visual hull is presented Figure 4.2.

The idea behind the implemented pose estimator is to look for the optimal pose by iteratively varying the parameters of the 3D model. Different pose parameters generate different pose projections and the parameters that generate the projections that best match the video in all views are said to be the parameters of the optimal pose. How well a pose match the video is decided by two matching functions - one measuring the amount of silhouette area covered by the projections and one measuring the amount of silhouette edges covered by the edges of the projections. The area matching function constrains the pose to the silhouettes and the edge matching function makes sure that it follows the edges. Each iteration in the optimization procedure consists of generating hypotheses in the parameter space close to some starting pose, evaluating them using the matching functions and then assuming the best pose as the starting pose for the next iteration. Optimization stops when the matching quality is above some threshold (manually specified for each recording and depends on the video quality) or when the number of iterations has passed the maximal number of allowed iterations.
4.3 Matching Functions and Search Algorithms

Given the silhouette image $\mathbf{S}$ generated from the video of a view and a silhouette image $\mathbf{M}$ generated from the pose projection in the same view, the area matching quality is defined as

$$q_{\text{area}} = 1 - \frac{\sum_{i,j} \text{xor}(S_{i,j}, M_{i,j})}{N}$$  \hspace{1cm} (4.1)

where $\text{xor}(x, y)$ is the usual exclusive-or operator (with TRUE = 1 and FALSE = 0) and $N$ is the number of pixels in the image. In other words, the area matching quality gets minus points for each pixel that is classified as foreground in only one image. Furthermore, given the edge image $\mathbf{A}$ generated from the video of a view and an edge image $\mathbf{B}$ generated from the pose projection in the same view, the edge matching quality is defined as

$$q_{\text{edge}} = \frac{\sum_{i,j} A_{i,j} B_{i,j}}{\sum_{i,j} B_{i,j}}$$  \hspace{1cm} (4.2)

Conclusively, the area matching quality gains from the pose covering foreground and is punished for the pose covering background while the edge matching quality measures the quota of generated pose edges covering foreground edges. These two values are multiplied to get the matching quality for the view:

$$p_{\text{view}} = q_{\text{area}} q_{\text{edge}}$$  \hspace{1cm} (4.3)

This definition requires high matching quality for both edges and area to get a high matching quality for the view. Consequently, high area matching quality
can not compensate for lacking edge matching quality and vice versa, which is reasonable as the edge images are generated from the silhouette images. The total matching value for the pose is given by the weighted average of the probabilities for all views.

The goal of the searching algorithms is to optimize the matching functions using as few computations as possible. Four search algorithms have been implemented: one designed for searching for flexion parameters, one for adduction, one for wrist rotation and one for wrist position. As the computational complexity of the quality matching functions is quite high, heuristic functions have been implemented for each of the search algorithms to reduce the number of calls to the matching functions while testing all relevant hypotheses. The simplest heuristic function is a test where the skeleton of the digits is projected on the image planes to compute how much of the skeleton is covered by foreground. If the quota is too low it means that the digit cannot be accurate - the hypothesis is discarded. Another heuristic function used for digit adduction and flexion searching is similar to the full quality matching functions - it generates silhouette images for individual digits but computes only how much of the digit area and edges that are matched properly. The function is used to first find the $n$ best hypotheses for each digit and then all combinations thereof are evaluated using the full quality matching functions. Algorithm 4.1 describes the process of digit flexion/extension optimization in more detail. Adduction optimization is done similarly but with only one hypothesis for the second joint, namely zero as it cannot rotate along that axis. Algorithm 4.2 describes the process of wrist position optimization in more detail. Wrist rotation optimization is done the same way but with different numbers of hypotheses for each axis of rotation, as the rotations along certain axes are not as common as along others.

Algorithm 4.1 (Flexion/extension optimization) The algorithm searches for the best combination of flexion parameter values for a pose with their previous values as the starting point.

1. Create $n_1$ hypotheses for the flexion at the first joint of digit 2 in an interval centered at the current flexion value.
2. Similarly, create $n_2$ hypotheses for the second joint.
3. For each of the $n_1n_2$ hypotheses, compute the quota of the skeleton that covers foreground. If below some threshold, discard the hypothesis.
4. For the remaining hypotheses, generate the silhouette images and edge images for the digit and evaluate how much of the silhouette and edges correspond to observed silhouette and edges, respectively. Keep the $m$ best hypotheses.
5. Repeat step 1-4 for digits 3-5.
6. Evaluate all $m^4$ combinations of flexion parameters with the full matching quality function and assume the best combination as the result of optimization.

Algorithm 4.2 (Wrist position optimization) The algorithm searches for the best position of the wrist with the previous position as the starting point.

1. Create $n$ hypotheses for each spatial dimension in intervals centered at the current position values.
2. For each of the $n^3$ hypotheses, compute the quota of the skeleton that covers foreground. If below some threshold, discard the hypothesis.
3. Evaluate all remaining combinations of wrist position parameters with the full matching quality function and assume the best combination as the result of the optimization.
It is also possible to use SIFT (Scale Invariant Feature Transform)-points to track movement of parts of the paw, assuming that some of them represent the same points in space for a number of consecutive frames. Using SIFT-points for tracking is discussed in Appendix A.

### 4.4 Database Search

The number of possible poses is infinite due to space being continuous but as the cameras have limited resolutions, the number of different poses that can be told apart from each other has an upper limit. This means that it would be enough to create a database with a finite number of poses and simply test all of them to find the best pose for each frame. Furthermore, adding physical constraints and using that the reaching movements look almost the same every time, the size of the initially very large database can be decreased to a practical size.

The database used in this project is built by running pose tracking for a lot of trials and then manually submit poses that either by automatic, semi-automatic or manual fitting are of good matching quality. Tracking is usually done by starting at the best fitting database entry and then run local optimization, but sometimes the poses are too far from any pose in the database so an initial guess needs to be entered manually.

The easiest way to decrease and to maintain low database search times is to decrease the number of poses and use some measure of similarity of the poses to eliminate redundant poses. Furthermore, pre-rendering silhouette- and edge images is simple but effective as the generation of those is the most time-consuming part.

A more advanced search method is to try only a small number of appropriately chosen poses from the database, find the best matching entry and then test only the most similar poses (according to some metric) from the full database. It is also possible to search the database for poses close to an appropriate pose, for example the pose for the previous frame assuming that it is of sufficient quality. This search method guides the optimization algorithms to a maximum that provides temporal consistency (discussed further in Chapter 6).
Chapter 5

Experiments

This chapter contains descriptions of the experiments performed, results from the pose tracking and some preliminary electrophysiological results.

5.1 Implementation Details

The latest version of the experimental setup features two cameras and 3 mirrors for automatic tracking of the reaching attempts and a sideways camera for manual detection of lift-off (start of reaching trial). Image acquisition in all cameras is triggered externally by a pulse generator (Master-8) and timestamps are provided by the cameras internal clocks with 1 µs resolution. The paw tracking cameras uses the GigE interface and can record at 300 fps (640x480 monochrome pixels with 8 bits color depth, resulting in 90MB/s uncompressed), but due to hardware limitations the videos cannot be recorded to harddrives at higher framerate than approximately 200 fps (60MB/s uncompressed). The sideways camera uses a common firewire interface and can only record 60 fps, but is run at 50 fps as camera triggering is easier when the higher framerate is a multiple of the lower (1 pulse to the firewire camera triggers 4 pulses to the GigE cameras). Due to the fast movements of the rat, the exposure time for the cameras needs to be short to avoid motion blur. LED lamps are used to get good contrasts and low noise levels even when using short exposure times.

Camera and mirror positions were chosen to get as many distinct views as possible (minimize the volume of the visual hull) with as good focus as possible while still allowing the experimenter to place and remove food pellets and pasta with ease. Estimated camera positions and directions are shown in Figure 5.1.

After performing the experiments, the first step of analysis is to find the interesting intervals of the video and to do camera calibration for which a graphical user interface (GUI) was created in MATLAB, see Figure 5.2. The tool contains functions for manually splitting up each video in multiple views and to perform camera calibration on each of the views. It also features functions for efficiently looking through the videos and split them up in intervals to do pose tracking on. In Figure 5.2, the red, green and blue rectangles have been selected as bounds for three views that will be used for pose estimation. The yellow rectangle marks the area used for detecting when the nose enters through the slit and the area within the white rectangle helps detecting when the paw is
near the pellet. The area within the dashed blue rectangle is used for detecting when the experimenter manipulates the scene, for example by putting a food pellet on the shelf. The right plot shows the data used for extracting interesting intervals. The green graph represents the white rectangle, the red graph represents the yellow rectangle and the black graph represents the dashed blue rectangle. Finally, the red and green circles in the video images are manually marked corners of a calibration cube, and are used for camera calibration.

The second step of analysis is initialization and visualization of tracking, for which the GUI seen in Figure 5.3 was created. It gives the user the ability to choose the iterative optimization steps, choose thresholds for various functions and to do manual fine-tuning of the tracking results. Sometimes some views are not appropriate to use for tracking - the weights for those views are manually set to zero. The checkboxes in the corner of the plots in the figure indicates zero/nonzero weights of the views. The most common reasons not to use a view for automatic tracking is bad calibration or bad background estimation, but the view may still be used for manual pose estimation and are therefore presented in the GUI.

In order to handle the large amount of data without making unnecessary computations, the GUIs and methods are all using a class hierarchy and have the ability to easily connect to methods made by other researchers to produce LFP-plots, spike plots, etc. relating to the reaching behaviour.

5.2 Results and Discussion

The tracking works subjectively good for a lot of different poses, as seen in Figures 5.4, 5.5, 5.6 and 5.7. Execution times are slow compared to other hand
Figure 5.2: The calibration tool. The red, green and blue rectangles have been manually selected as bounds for three views that will be used for pose estimation.
Figure 5.3: The pose fitting tool. The axes in the 3D plot are 50 mm long. Note that the 3D plot views the mesh of a model for the nose that is not yet automatically tracked. There are also 3D meshes for the pellet and forearm, none of which are automatically tracked.
trackers (see Section 4) with approximately 60 seconds per frame, but it can be greatly speeded up when the database is big and diverse enough to run tracking solely from testing previously encountered poses. As of this moment, testing the 200 poses in the database takes approximately 1 second and can be improved using some of the more intelligent algorithms for searching the database discussed in section 4.4. Assuming that estimation of the pose in one frame takes one second, that 30 frames can be tracked in each reaching attempt and that one experiment consists of 30 reaching attempts, total execution time for the tracking part is 15 minutes. Even if using the current, slower algorithms, execution times (15 hours) are not impractically long.

One of the most difficult parts of the pose estimation is estimating the pose of the palm, due to the palm rotation and position deciding the axes of rotation for the digits, the low amount of edges surrounding it and its rather complex shape. The palm is estimated using the area it covers and the axes of rotation for the digits it creates, with the effect that the estimation for the palm itself can be bad while the digits fit nicely in the images. This could prove to be a problem in later steps of analysis and classification of different reaching movements and grips. Another difficult pose to estimate is when the rat makes a fist, decreasing the amount of edges and area it generates. Some problems with the axes of rotation has also been encountered while estimating these poses.

Another problem is that of estimating the position of the joints; the projected lines can fit nicely into the paw even when the position of the joints is a bit off, as seen in Figure 5.7. If the joint estimation is bad the following digit rotation estimations will also be bad. On the other hand the joints are hard to even see with the eye, motivating use of markers in future version should higher quality be required. As this is just the beginning of the experiments, it is not yet known how good pose estimation quality that will be required to reliably track e.g. gradual adaptation during learning or minor motor disturbances in parkinsonian animals.

The pellet sometimes cause bad pose estimations (see Figure 5.6) when in contact with the paw, because then the pellet is connected to the paw part of the visual hull of the foreground. The obvious solution to this problem is to track the pellet as well and has been tried on a smaller scale with promising results. Another solution is to add an upper threshold for the foreground estimation (the pellet is slightly brighter than the paw). The same problem appears when the reaching target is a spaghetti straw, but this is harder to solve since the pasta has slightly varying length, covers a larger part of the image, covers parts of the paw in some views and cannot be thresholded out of the foreground. One advantage, however, is that tracking the pasta is easier as a large part of it is visible even when the paw is grasping it, whereas the pellet at times can be almost completely occluded.

Estimation of the fifth digit (the cyan colored lines in Figures 5.5 and 5.7) is sometimes difficult; the digit is sometimes bent almost like a thumb along other axes of rotation than what would be predicted from the anatomy of the joints (probably due to tissue flexibility). The deformation parameter of the paw model seeks to solve this problem by modifying the axes of rotation and works decently in general, but a more advanced solution also modifying the positions of the knuckles might improve results further as well as allow more advanced motion analysis.
CHAPTER 5. EXPERIMENTS

Figure 5.4: A typical pose estimation result.

Figure 5.5: Pose estimation result. The estimation of the palm is a bit off and the fifth digit (cyan) is inaccurate.
5.2. RESULTS AND DISCUSSION

Figure 5.6: Pose estimation result. The food pellet is estimated as part of the second digit (red).

Figure 5.7: Pose estimation result. Estimation of the fifth digit (cyan) is a bit off.
5.3 Neural Results

In this early stage of neural analysis, one event of special interest was extracted from the tracking data and used to produce LFP-plots with low to moderate statistical reliability. The LFP-plots presented here shows the mean of the LFPs in a -3 to 3 seconds time window centered around the extracted timestamps for maximal extension of the left forelimb, see Figure 5.8 for LFP plots for selected electrode channels. As expected, there is more modulation visible in the right hemisphere than in the left during reaching with the left forepaw. In the right motor cortex, higher frequencies are first modulated, then lower frequencies (< 20Hz) increase for a short period and then again some higher frequencies reappear. However, only the low frequencies are passed on to right Striatum.

Figure 5.9 shows spiking activity for selected neurons. The plots clearly show that the firing rate of those neurons is significantly modulated relative to baseline (mean firing rate from -3 s to -1 s) over time during reaching. The red line marks significant (p < 0.01) suppression and the green line marks significant enhancement. When a sufficient number of trials has been analyzed with the pose tracking software, more detailed spike plots can be generated, for example for different components of the reaching behaviour, as the statistical reliability increases with the number of trials.

Figure 5.8: LFP plots for selected electrode channels. The horizontal axis shows time (s) with zero being the time of maximal extension of the forelimb and the vertical axis shows frequency (Hz). The color shows the difference in log power from the baseline (mean log power from -3 s to -1 s) for each frequency.
5.3. NEURAL RESULTS

Figure 5.9: Spike plots for selected neurons. The horizontal axis shows time (s) with zero being the time of maximal extension of the forelimb and the vertical axis shows the mean number of spikes for each bin. The red line marks significant suppression and the green line marks significant enhancement from the baseline (mean firing rate from -3 s to -1 s).
Chapter 6

Future Work

6.1 The Software

There are many parts of the software that can be improved. Some methods can be greatly speeded up by implementing them in C, while other algorithms are optimized for the data structures they are in and would need restructuring of parent algorithms to make improvement possible. Other parts seem to work well - for example the matching functions, which can be seen especially when considering pose estimation results when using only database search.

The conditions for the matching functions can be improved, however, to make the pose estimation more robust. In the current version, not all real-world objects that can generate foreground is being tracked by the pose, making it possible to classify for example pellet as digits (see Figure 5.6), forearm as palm or nose as digits (see Figure 6.1). Supposing that a reliable algorithm for tracking pasta, pellets and nose can be made, that mistake would not be possible anymore. The matching functions are defined in such a way that all of the foreground needs to be classified to get a full score, further motivating the implementation of a reaching target tracker and extending the model of the rat to cover forearm as well. A similar problem is occlusion by walls. Currently, the function that generates silhouette images does not account for the walls, but can generate foreground for parts of a pose that would be occluded in the real world. The extra foreground pixels can make an otherwise nicely fitting pose get a low score.

Another problem with the pose model is that the model for the palm is very simple (an ellipsoid) and misses some features of the real palm, for example the stump that corresponds to the thumb on a human hand. Also, the phalanges are modelled as cylinders whose projections are of non-regular shapes, resulting in time-consuming algorithms for generating silhouette images. Ellipsoids have nicer properties - their projections are always ellipses which could probably be used to create an efficient function for generating silhouette images.

A simple but not yet implemented idea is to use temporal consistency both as a constraint and as a predictor. For example, an initial guess for the next wrist position can be estimated using previous wrist positions and the parameters of the new pose can be constrained to intervals estimated by the previous poses. It can also be used to look through a tracking sequence, identify bad estimations
6.2. THE EXPERIMENTAL SETUP

Figure 6.1: The paw placed inside the paw.

and suggest new initial guesses to rerun optimization from.

Database search times can be improved by attaching estimated velocity vectors for all entries and then try all poses with velocities close to that of the previous pose or the estimated velocity of the new pose. This makes a natural segmentation of the database into poses encountered during advance and poses encountered during withdrawal. After implementing tracking of the reaching target, the database could be divided into one part for poses holding pasta, one part for poses holding pellets and one part for poses holding nothing. This should serve to further improve search times and increase robustness as the number of relevant poses decreases and only poses for the appropriate manipulation can be assumed.

A task that has not yet been addressed but is important for the experiments is to automatically find the time for paw lift-off in the side view, see Figure 6.2. Even better is automatic tracking of the whole reaching sequence starting from lift-off, which would enable more detailed neurophysiological studies. For that to be possible, however, better conditions (for example more uniform and darker background, less reflections, better focus, shorter exposure time, possibly paint on the inside of the opposing arm, etc.) is probably needed.

6.2 The Experimental Setup

There are also parts of the experimental setup that can be improved, for example the position and rotation of the cameras and mirrors could be moved so as to minimize the volume of the visual hull of a typical mid-reach pose.

The conditions for the foreground segmentation can be improved by making the background even darker and more matte. Also, the position and rotation of the LED lamps can be optimized to give maximal exposure on all of the paw while direct light from the lamps is not seen in any area of interest in the videos.

Conditions for the camera calibration could be improved by using "better" points of reference, for example by creating an object with as many points as possible that can be distinguished easily in all views and that fits perfectly into only one location on the shelf. Another solution would be to draw points on the scene at locations that does not interfere with the pose estimation.
Figure 6.2: A lift-off sequence from the side view. Note that part of the left forelimb cannot be seen in the fourth image due to the long exposure time in this camera.
Appendix A

Using Point Features

In 1999, David G. Lowe introduced what has come to be one of the most commonly used feature extraction techniques, SIFT, with the following properties

The features are invariant to image scaling, translation, and rotation, and partially invariant to illumination changes and affine or 3D projection. These features share similar properties with neurons in inferior temporal cortex that are used for object recognition in primate vision.

More practically, there are implementations of SIFT available which can easily be incorporated in MATLAB scripts to extract features. The implementation used in this project outputs interesting points and vectors with the corresponding feature values. Those features can be used to aid hand pose estimation by tracking some appropriate SIFT-points frame to frame and encouraging the projection of the hand pose to make the same movements, assuming that the pose in the previous frame is accurate. More specifically, the SIFT method extracts many keypoints of which the majority is usually not useful for the pose tracking (too far away from the pose projection) - those points are discarded. The remaining keypoints from the previous frame are matched to the remaining keypoints in the current frame and a list of point correspondences is constructed. Then the vector from the closest point on the pose mesh to each keypoint in both frames is computed as well as the distance between those vectors. For each vector distance that is small enough, the matching quality for the current pose is increased. Since there are cases where there are no useable SIFT-points and even in those cases there are problems controlling the results, the algorithm is here only used to strengthen the belief in a hypothesis.
Appendix B

Some Practical Details

B.1 Training Protocol

1. Every day for a few days prior to the experiments, the rats were accustomed to the pellet and pasta food reward after 20-24 hours food deprivation. After each training session the animals received free supply of their regular food for one hour.

2. The teaching of the reaching behaviour begins by making the rats aware that there will be food placed on the shelf (see Figure 2.3). This is done by placing multiple pellets/pasta on the shelf for the rats to reach for in any way it can.

3. The acquisition of food is subsequently made harder by moving the pellets further away from the opening slit. This forces the rats to use their forepaws since they can no longer reach using the tongue, thus gradually mimicking the final step.

4. Until now, food has been placed in all three sockets, but as the rats starts to develop a preference of paw, only one pellet is placed in the socket contralateral to their paw of choice, to promote the use of this paw.

5. The next training step is to teach the rat to move to the back of the cage every time it has made an attempt; this is achieved by simply not placing any food rewards on the shelf until it has reached the opposite end of the cage. This is needed to be able to properly distinguish between attempts.

6. The final step is to make sure that the rats only deliberately reach for food when there is actually food on the shelf, this is attained by not placing food pellets on the shelf on semi-random occasions and if the rat makes an attempt anyways, no pellet is placed there for the next trial. This forces the rats to identify the presence of food before attempting to reach for it.

B.2 Data Gathering Protocol

In preparation for each experiment, the experimenter places the calibration cube on the shelf in a few different locations while video recording is on. There is
APPENDIX B. SOME PRACTICAL DETAILS

Figure B.1: A trial sequence.

a switch controlling if the trigger signals get to the frontal cameras with the purpose of minimizing the amount of generated video data (60 MB/s/camera). During a trial, the switch is manually turned on when the rat approaches the shelf and manually turned off just after the rat has backed off from the shelf. This detail only slightly increases the amount of manual work, since there is still the need for manual placement of food but dramatically decreases the size of the generated videos and simplifies the extraction of trials (big gaps in timestamp data).

For each trial, the following protocol applies:

1. Food pellets or pasta are placed on the shelf as the rat approaches the opposite end of the cage. On a few occasions, no food is placed there.

2. The trigger switch is turned on as the rat approaches the shelf.

3. The trigger switch is turned off after the rat has withdrawn from the shelf.
Bibliography


