Tomas Bata University in Zlín
Faculty of Applied Informatics

Research of possibilities of using mouse-like input devices as a biometric identification system

Doctoral thesis

Author: **Martin Kolařík**

Course: Engineering informatics (P3902)
Selected field: Engineering informatics (3902V023)

Supervisor: doc. Mgr. Roman Jašek, PhD.

Zlín, July 2015

Abstract

Research into identifying people according to how they use mouse-like input devices, has so far only weakly explored presumptions of the methods used—for example environmental influences or influences of the source of original data. According to the author's knowledge, no work has yet tried to reproduce or enhance some predecessor's work. The results of existing works are promising, but only loosely connected.

In order to improve the above-mentioned situation, this doctoral thesis reviews existing works in the field, provides theoretical foundations to better understand and further evolve this identification method, and also explores modifications in feature selection algorithm.

Based on this theoretical summary, the experimental part of this dissertation focuses on improving feature selection and on comparing three different user environments and their data. It also enhanced selected former research on the use of unrestricted movements. Experiments designed by the author are carried out and their results are discussed for each mentioned experimental part.

Key words

biometric, identification, mouse, continuous identification, feature selection

CONTENTS

GLOSSARY AND ABBREVIATIONS

entity model of a *person*, set of *features*, *template* when in database

feature a random variable derived from *markers* of *sample* that characterizes a *person*

gap or *time gap*, the duration with no *movement* that delimits the *stroke* during stroke detection

input data the stream of *measurements*

marker the statistical property of a *stroke*, *markers* of *sample* are distilled into *features* when an *entity* is built, or matched with *feature* resulting in *similarity* when an *entity* is identified

measurement the raw value that samples a *person*'s *movement* in a single instant

movement mostly intentional action of moving mouse-like device by a *person*

person the subject that is identified, that performs *movements*

sample one or more consecutive *strokes*

similarity the measure of correspondence between *markers* of *sample* and *features* of *entity*

stroke a piece of cleaned and adjusted *input data* that models and corresponds to a single *movement*

template the database record storing data of single *entity*

API Application Program Interface (client interface of the OS)

EER Equal Error Rate, see chapter 5.2.4

FMR False Match Rate, see chapter 5.2.2

FNMR False Non-Match Rate, see chapter 5.2.1

GUI Graphic User Interface

HID Human Interface Device

OS Operating System

PIN Personal Identification Number

SFS Sequential Forward Selection, see chapter 5.3.2

SFFS Sequential Floating Forward Selection, see chapter 5.3.2

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

Measuring characteristics and properties of humans in order to distinguish one human from another human was first widely used by governments in the field of criminology. Governments tried to simplify and improve proof of guilt by linking certain measured properties to a particular person. In today's words, governments identified people using their biometric characteristics.

Biometric identification is fundamentally different from other identification methods. The key needed for identification is always at the right place, as the person themselves is the key. No cards, no chips, and no passwords are needed. Moreover, thanks to the key-person relationship being unbreakable by nature, the biometric identity practically cannot be stolen [2].

With current knowledge, it is still challenging to reliably detect differences between individuals using biometric identification. There are many factors that make biometric identification difficult—for instance, the time of measuring, how comparable results are, how unique properties are, the measuring precision, the variability of properties, or simply understanding the selected properties [1]. For all these reasons, all known methods are constantly evaluated and improved in the hope that better performance can be achieved.

The same motivation propels this dissertation. This dissertation targets biometric identification using computer mice because this method has been researched for more than ten years and results are still far from perfect. Searching for, investigating and evaluating some of these sources of imperfections is the main impulse behind the dissertation.

There are three principal areas of mouse-like device identification, which are not yet sufficiently covered by research, [3]: how much can results be reproduced (how the same person is recognized using different mice and computers), how should technical aspects of mouse movements be understood—which algorithm to use? how to tune the system? what is principal and common? what is needless and particular?, and the relationship between identifying features and existing neurological models of eye-hand coordination—is there any way to discover this? This dissertation primarily addresses the second area and partly the first area. In addition, the dissertation takes one former research, [4], enhances it and evaluates how much results are reproduced.

This dissertation is divided into three areas. Firstly, the current state of the art is described which includes brief notes about the origins and progress in the field. In the second part, basic theoretical terms and principles are explained, and lastly, experiments are described along with their arrangement, results, evaluation and a discussion.

2 STATE OF THE ART

Research into identifying people according to how they use mouse-like input devices has occurred in four stages over time:

The initial stage

The first, initial stage of research focused on testing whether information can generally be used that was gained from tracking mouse-like device movements. For example, [5] analyzed mouse movement dynamics in relation to Fitt's law [6], and [7] was thinking about measuring forces applied to mouse-like device during the movements.

This initial stage of exploration in this field is particularly important for my dissertation because these initial ideas were free—not intellectually restricted to any procedure or previous work.

The exploration stage

The second, exploration stage tried to find various ways of organizing mouse-like device input data into suitable identification features. Geometric models tracking positions on the screen were invented by [8] and [9], dynamics models measuring movement paths appeared in [10] and [4] together with motivation games. [11] and [12] used no motivation games because they tested unrestricted input data.

Different approaches to classification of the entities were tested. [4] used statistical models, [8] and [12] utilized a neural network, [13] explored decision trees. [14] and [11] started with classifying mouse actions into groups.

In this stage it is typical to have a gradual increase in the complexity of used models and a straightforward effort to obtain quality identification systems at the cost of reproducing results and a thorough understanding of the methods. Several groups of authors published papers containing improvements of the same work, like [10] and [4] or [8] and [15]. This dissertation benefits from such papers, because they clearly show what helped improve methods.

Papers published during this stage form the basis for these dissertation experiments. This dissertation will especially analyze and improve the *stroke*-based model introduced in [10], improved in [4] and re-introduced in [12]. This model was also analyzed and partially enhanced with unrestricted movements in [16].

An interesting comparison of different approaches is given in [15]: the author used a geometrical model and tried to compare results with [4], where a dynamic stroke model was used. Differences in the methods' performance were not fully explained in the work.

The separate branch of experiments evaluating a specially developed hardware appeared, like in [11], where a mouse with embedded fingerprint sensor was used.

Stabilizing stage

The third, stabilizing stage focused on cleaning and improving results achieved in the two previous phases. Various approaches appeared and the validity of features was first discussed.

For instance, [17] and [18] developed a survey of existing methods, [19] tested whether measured features improve using the K-nearest neighbor classifier.

A new approach to the research appeared in this stage—software models started to be constructed that should have helped understanding the mouse driven identification, for instance in [20].

Effort to find out the best identification features continued, usually based on statistical evaluation of identification templates, as in [21].

This dissertation refers to all the papers mentioned in this stage because the papers reveal that some types of features are more prominent than others.

The evaluation stage (currently)

Current stage. Existing approaches started to be critically reviewed, and enhanced variants of earlier works appeared. For example [22] tested yet another environment with restricted movements, [23] measured distances of templates with the help of Euclidean metric, [24] reevaluated [12] approach and extended it with random forest classification.

The further research into classifiers was presented in [25], and new classification methods were utilized, like learning vector quantization in [26].

A systematic efforts to reduce complexity of existing systems appeared. [27] recommended that only movements ended with click should be used, [28] suggested that the usage of mouse movements related to the file operations could lead to better results, and [29] developed an approach where only predefined mouse gestures are taken into account.

Papers attempting to critically identify systematic and methodological issues of all previous works started to appear, like [3].

This dissertation also has the intention to search for methodological correctness which is followed by using two possible sources of mouse information in three types of environments. These are compared and evaluated with the goal of exploring how they affect identification process.

3 DISSERTATION GOALS AND BENEFITS

This dissertation has the following preparation and study goals:

- to critically review previous research,
- to explore technical and behavioral variants of methods used.

This dissertation has the following practical goals:

- to deeply analyze and enhance feature selection methods and metrics,
- to enhance [4] for unrestricted movements,
- to compare two methods of obtaining mouse-like device data, and
- to explore the influences of various user environments.

When the goals of this dissertation are fulfilled, this will have the benefits:

- various features of mouse-like device movements will be evaluated according to how important they are,
- knowledge will be gained about interchanging and reusing of identification templates in different environments,
- the feature selection process will be generally improved.

THEORETICAL PART

The theoretical part is divided into three sections: the first part focuses on common terms and knowledge, the second part focuses on introduction to biometric identification, and the third part focuses on explaining and discussing theoretical fundamentals of mouse-like device identification methods.

4 PRINCIPLES, TERMS AND MATH USED

In this chapter, principal terms of identification will be briefly described and explained. The explanation starts with *identification* term itself, it continues with ideas about *key/token—person* relationship, then it discusses motivation and the meaning of the term *biometric* [30] and finally it ends with the specialized term *behaviometrics*.

In the second part of this chapter, an overview is given on the mathematical apparatus used in this dissertation, and particularly on the usage of the law of total probability and utilized random variables.

4.1 Identification

Identification of a person generally means looking for a person among an existing set of persons. If a particular person is missing in the set, identification should fail. If the particular person searched for is part of the set, identification should find the correct record. Use of the verb form *should* is fully intended: no identification is capable of being certain. Some uncertainty still remains simply because everything in the identification process is of probabilistic nature [2].

The key means allowing comparison and assessment of methods is the amount of uncertainty manifested by various methods. Methods having little uncertainty are considered to be high quality. The uncertainty in identification results is fairly complex, and there are standard ways of measuring uncertainty that are described in chapter 5.2.

Looking for a person's identity in the given set can be reduced to an abstract process (see also [1]): the identified person becomes the *key*, the given set becomes a *database* containing *templates*, and the matching phase searching for the key among templates in the database becomes an *algorithm*. These four terms are used throughout the whole text with these meanings.

The key can be anything suitable to prove a person's identity in order that it is believed that the key belongs to the person. It is crucial to believe in the validity of this person-key bond in order to validate the whole identification process.

4.1.1 Detached keys—tokens

If the key is detached from the person, the identity is then reduced to the key and the quality of identification is reduced to the quality of the key. The key is usually wrapped in some physical carrier called a *token*. Many different detached keys/tokens are used nowadays, for instance citizen identification cards, passports, chip-on cards (badges) and so on. Detached keys make it easy for the algorithm and database: the key is always constructed in a way that it allows algorithms to match with the database exactly, with no uncertainty.

There is a fundamental problem with detached keys/tokens. Because they are detached, they can certainly be used by improper people; this act is better known as identity theft. The possibility to steal tokens causes serious problems for identification, which is why tokens are frequently supplied with independent second-level sub-key(s) like a photograph of the owner, a PIN for cards, or a signature. The need for sub-key(s) is so strong that effectively only when the key and sub-key(s) are combined together it can be considered to be a complete key [2].

4.1.2 Biometric identification

If the key is attached to the person, the situation changes. There are different levels of attachment ranging from bangles for home prisoners, sub-tissue electronic devices and through to the person themselves being the key. When people themselves are the key, it is the most interesting. This part of identification making use of the fact that the person is exactly the key is called *biometric identification*. Having the key and the person as one entity is the fundamental advantage of biometric identification compared to identification using detached keys.

Using the whole person as the key offers a broader range of measurable and comparable characteristics. No abstraction is necessary, though reduction is still used to decrease computational complexity. The broad range of human characteristics has given rise to a broad range of biometric identification methods including fingerprinting and tracking a persons' daily movements/routines [30].

Biometric identification is never exact, which is directly contrary to detached key identification. There are uncertainties relating to the key (i.e. how precisely a person's characteristics are measured and how stable they are) and uncertainties related to the algorithm (the key can only match with the template approximately). As a result, biometric identification is able to identify a person only at some level of probability [2].

4.1.3 Behaviometrics

Biometric identification uses various personal characteristics. The presumption is that the personal characteristics used do not change over time [2]. However, because humans grow and age, this presumption is in principle invalid, and whenever possible, it should be replaced with a time interval when the particular characteristics are reasonably stable.

According to the nature of measured characteristics, two groups of biometric identification methods can be distinguished: the first group uses physiological characteristics and the second uses behavioral characteristics. The latter group using behavioral characteristics is also called *behaviometrics*.

Physiological characteristics are rather static and usually show dimension, space arrangement, chemical structure and so on. To the contrary, behavioral characteristics are rather dynamic and usually show change over time, applying of forces, or time arrangement. Obtaining physiological characteristics usually requires contact with the person and it can be obtrusive, but this contact and measuring usually gives repeatable and trustworthy values. Measuring behavioral characteristics, on the other hand, can be done remotely and thus almost unobtrusively, but the values read are more uncertain and noisy.

Behaviometrics also raises concerns about privacy, as characteristics can be measured without the person knowing it [30].

4.2 Probability distributions

This dissertation uses a probabilistic model of the identification. The principle and also the details are described later in chapter 6.4. Probability distributions and random variables play a central role in the identification system designed this way.

4.2.1 Distributions used in this dissertation

Two intentions have led to selecting which probability distributions to use:

- domain and variable type, where symmetrical unbounded distributions with real number domain are needed, as well as bounded exponential-like variables with their domain limited to positive numbers,
- simplicity and speed of estimating parameters of the distribution, only distributions with non-iterative estimators were chosen (more about estimates is written in chapter 4.2.2).

An overview of all used distributions is given in table 1. This table contains only a basic description; comprehensive information can be found, for instance, in [31].

Described probability density functions (pdf) do not directly allow computing probability of appearing a given value x. The probability can only be taken as an integral of near neighbourhood of x. To achieve this, Simpson's rule is used as a numerical integration method in this dissertation. The rule runs in ten steps over the interval $\left[x - \frac{0.5}{100}\sigma, x + \frac{0.5}{100}\sigma\right]$, where deviation σ is taken from feature (see chapter 9.3.2) and is learnt during training phase (see chapter 9).

4.2.2 Procedures for estimating parameters

Estimating parameters of probability distribution for given data can be done in many ways. Of these methods, let's remember the maximum likelihood estimators, and the momentum methods, [31]. Deriving these methods is beyond the scope of this dissertation, so only the methods' results will be described. For all used distributions, methods with low computational complexity were always selected (see chapter 4.2.1).

Table 1 Random variables overview

distribution	parameters	pdf	domain
Gaussian	μ, σ	$\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$	$\mathbb{R}$
logistic	μ, σ	$\frac{e^{-\frac{x-\mu}{\sigma}}}{\sigma\left(1+e^{-\frac{x-\mu}{\sigma}}\right)^2}$	$\mathbb{R}$
Rayleigh	σ	$\frac{x}{\sigma^2} e^{-\frac{1}{2}\left(\frac{x}{\sigma}\right)^2}$	$\mathbb{R}^+ + \{0\}$
lognormal	μ, σ	$\frac{1}{x\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{\ln x-\mu}{\sigma}\right)^2}$	$\mathbb{R}^+$
inverse Gaussian	μ, λ	$\sqrt{\frac{\lambda}{2\pi x^3}} e^{-\frac{\lambda}{2x}\left(\frac{x-\mu}{\mu}\right)^2}$	$\mathbb{R}^+$
Weibull	α, β	$\frac{\alpha}{\beta}\left(\frac{x}{\beta}\right)^{\alpha-1} e^{\left(\frac{x}{\beta}\right)^\alpha}$	$\mathbb{R}^+ + \{0\}$
gamma	α, θ	$\frac{1}{\Gamma(\alpha)\theta^\alpha} x^{\alpha-1} e^{-\frac{x}{\theta}}$	$\mathbb{R}^+$

At the beginning of work on this dissertation, it was uncertain which distributions to use and how their parameters could be estimated. For fast and parallel evaluation of many distributions, the EasyFit tool [32] was used. After realizing which probability distributions are useful, they were incorporated into the dissertation software. EasyFit was then used to validate the implemented code—this dissertation and EasyFit are both expected to produce, and indeed produce, the same estimates of parameters.

In equations that describe ways of estimating parameters, common quantities are used—sample mean (1), sample variance (2) and sample mean of squares (3):

$$\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{1}$$

$$\hat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{\mu})^2 \tag{2}$$

$$\hat{m}_2 = \frac{1}{n}\sum_{i=1}^{n} x_i^2 \tag{3}$$

The estimation procedures are as follows:

Gaussian distribution parameters μ and σ directly correspond to sample mean and variance. Simultaneously these estimators are maximum likelihood estimators:

$$\mu \approx \hat{\mu} \qquad \sigma^2 \approx \hat{\sigma}^2$$

Logistic distribution is similar to Gaussian distribution with an exception in logistic's σ:

$$\mu \approx \hat{\mu} \qquad \sigma^2 \approx \frac{3}{\pi^2}\hat{\sigma}^2$$

Rayleigh distribution is different. If the sample mean is taken as the distribution mean, the σ parameter can be computed:

$$\sigma = \sqrt{\frac{2}{\pi}}\mu \approx \sqrt{\frac{2}{\pi}}\hat{\mu}$$

Lognormal distribution is very similar to Gaussian distribution except that values are in logarithm form. Estimators are also maximum likelihood estimators:

$$\mu \approx \hat{\mu}(\ln x_i) \qquad \sigma^2 \approx \hat{\sigma}(\ln x_i)^2$$

Inverse Gaussian distribution uses the momentum method taken from [33]. The estimation of μ is maximum likelihood estimation:

$$\mu \approx \hat{\mu} \qquad \lambda \approx \frac{\hat{\mu}^3}{\hat{m}_2 - \hat{\mu}^2}$$

Weibull distribution's parameters cannot be correctly estimated without iteration (for instance look at maximum likelihood estimations of Weibull in [31]). The approximations used in this dissertation utilize quartiles and come from [34]; q_1 is the first quartile, q_2 is the second quartile (median) and q_3 is the third quartile:

$$c = \ln\frac{\ln 0.25}{\ln 0.75} \qquad \alpha \approx \frac{c}{\ln q_3 - \ln q_1} \qquad \beta \approx q_2(\ln 2)^{-\frac{1}{\alpha}}$$

Gamma distribution's parameters estimation method is taken from [35]. The method is also approximate:

$$\alpha \approx \frac{\hat{\mu}^2}{\hat{\sigma}^2} \qquad \theta \approx \frac{\hat{\sigma}^2}{\hat{\mu}}$$

The input data for estimation procedures was prepared in two ways: first, histograms were constructed and used, then whole data sets were processed. The first approach was convenient for comparing results to the EasyFit [32] tool mentioned above because the tool creates histograms prior to estimating the parameters. The second approach then replaced the histogram approach in the dissertation's software because constructing histograms is unnecessary.

4.3 Bayesian matching

The probabilistic model of the identification system developed for the purpose of this dissertation (see chapter 7.2) uses the Bayesian law of total probability. The law allows to guess which entity a sample belongs to.

Features trained using person's data, and represented with the random variables (for practice see chapter 9.3), are priors. Applying the single stroke of a sample to priors gives rise to the number of probabilities, each expressing a probability of belonging of one stroke's marker to a corresponding feature. This is described in chapter 4.3.1.

Once priors are used and prior probabilities are computed, posterior probability can be computed using the law of total probability. This shows how closely a particular entity belongs to the given sample. These computations are described in chapter 4.3.2.

The whole procedure was inspired by [4], though the concept is general.

4.3.1 Priors

Matching a stroke of an unknown sample with a known entity composed of features means computing probabilities, because features are represented with random variables. The stroke is represented with markers, which must be matched one-to-one with the corresponding features:

- matching of each marker results in single probability that the marker is of the corresponding feature,
- when all these single probabilities are combined, this results in the probability that the stroke is of the corresponding entity.

Expressed mathematically, the probability computed using the prior is a prior probability:

$$\vec{s} = (w_0, \ldots, w_{n-1}) \tag{4}$$

$$\varepsilon(w_i^e) = \left[w_i - \frac{0.5}{100}\sigma_i^e, w_i + \frac{0.5}{100}\sigma_i^e \right] \tag{5}$$

$$p\left(w_i|F_i^e\right) = \int_{\varepsilon(w_i^e)} \mathrm{D}_i^e \tag{6}$$

where (4) represents a stroke composed of markers w_i and n is a number of features F_i^e (corresponding to the number of markers) in tested entity E^e (e indexes entities). Prior probability for a single feature (6) is computed directly using corresponding random variable's density function (pdf) D_i^e. (5) is a neighborhood used for computing the integral of the D_i^e.

The concept was used and discussed in [4], where it was also shown, that individual features are close to statistical independence. This means that prior probability of the stroke can be obtained as a product of probabilities of individual markers:

$$p\left(\vec{s}|E^e\right) = \prod_{i=0}^{n-1} p\left(w_i|F_i^e\right) \tag{7}$$

The described approach can be extended for sequences and/or sets of strokes. Single stroke represents a person worse than a sample containing more strokes. Using complete sample (meaning using more strokes if they are available) is obviously beneficial.

Similar equations as for single stroke can be written for whole sample, when similar statistical independence is assumed. This independence is clear here because a stroke (single-intention user action) does not rely on previous strokes:

$$\mathcal{S} = (\vec{s}_0, \ldots, \vec{s}_{m-1}) \tag{8}$$

$$p\left(\mathcal{S}|E^e\right) = \prod_{j=0}^{m-1} p\left(\vec{s}_j|E^e\right) =$$

$$= p_e \tag{9}$$

where m is a number of strokes in the sample $\mathcal{S}$. It is evident that (9), product of matching of individual strokes $\vec{s}_j$ of the sample $\mathcal{S}$ with the entity E^e, uses (7).

(9) is the final result of matching of an unknown sample $\mathcal{S}$ with a particular entity E^e. It is the prior probability p_e that $\mathcal{S}$ belongs to E^e.

4.3.2 Posteriors

Computing prior probabilities for all of t entities E^e produces t prior probabilities p_e. Each p_e measures how tightly $\mathcal{S}$ belongs to E^e see (9).

The primary goal of identification is to determine the opposite relationship: if E^e belongs to $\mathcal{S}$; if $\mathcal{S}$ was produced by E^e. To compute this, the law of total probability can be used. The result is hereinafter called *similarity* s^e of E^e to $\mathcal{S}$:

$$s^e = p(E^e|\mathcal{S}) = \frac{p(\mathcal{S}|E^e)p(E^e)}{\sum_{f=1}^{t} p(\mathcal{S}|E^f)p(E^f)} \tag{10}$$

In (10), $p(E^e)$ and $p(E^f)$ is unknown. It depends on a real identification system and on its purpose, which values these probabilities have (i.e. how frequently each person appears). In large systems with many entities (persons) $p(E^e)$ and $p(E^f)$ will be getting close to $1/t$ (where t is the number of entities):

$$p(E^e) = p(E^f) \approx \frac{1}{t} \tag{11}$$

For the purpose of this dissertation (11) can be presumed because all experiments test all entities equally. With the help of (11), (10) reduces to:

$$s^e = p(E^e|\mathcal{S}) = \frac{p(\mathcal{S}|E^e)}{\sum_{f=1}^{t} p(\mathcal{S}|E^f)} \tag{12}$$

The similarity s^e (12), the probability that the unknown sample $\mathcal{S}$ is produced by known entity E^e, is the principal result used in this dissertation. s^e is used to compute the EER (see chapter 5.2.4), it is used in feature selection process (see chapter 5.3), and also in all experiments.

5 BIOMETRIC IDENTIFICATION SYSTEM

All systems operating data for many entities must in principle perform very similar tasks. The same is true for identification system, whose purpose basically predestines its structure.

This chapter is dedicated to this common structure and also to the particular tailoring used in this dissertation.

5.1 General model

When considering biometric identification, some facts emerge immediately:

- biometric → measuring → repeating, statistics,
- measuring → raw data → cleaned data → filtering → reduced data,
- data → representation, storage → storing, recalling,
- representation → selecting, looking for, matching → acceptance → threshold,
- acceptance or rejection → quality, reliability → quality metrics.

These thoughts have appeared in identification systems, in various forms, from the very beginning. It was soon possible to accept these thoughts as general ideas and use them to start to organize abstract models.

5.1.1 Structure of the general model

The generalized model of biometric identification system firstly appeared in [36] and later e.g. in [1], see figure 1. This generalized mode recognizes the following subsystems:

data collection subsystem (DCS)
: responsible for detecting presented biometrics (behavior) and converting this to computer-acceptable samples,

transmission subsystem (TS)
: transferring samples from the source to signal processing subsystem,

signal processing subsystem (SPS)
: responsible for cleaning data, feature extraction, selection and matching,

storage subsystem (SS)
: storing various data, for instance templates representing entities,

decision subsystem (DS)
: taking results from SPS, and deciding if sample is genuine or an impostor.

DCS and TS delivers data to three other subsystems, which then cooperate. There are two main types of cooperation between signal processing, storage and decision subsystems—the training and tuning mode, and the operation mode.

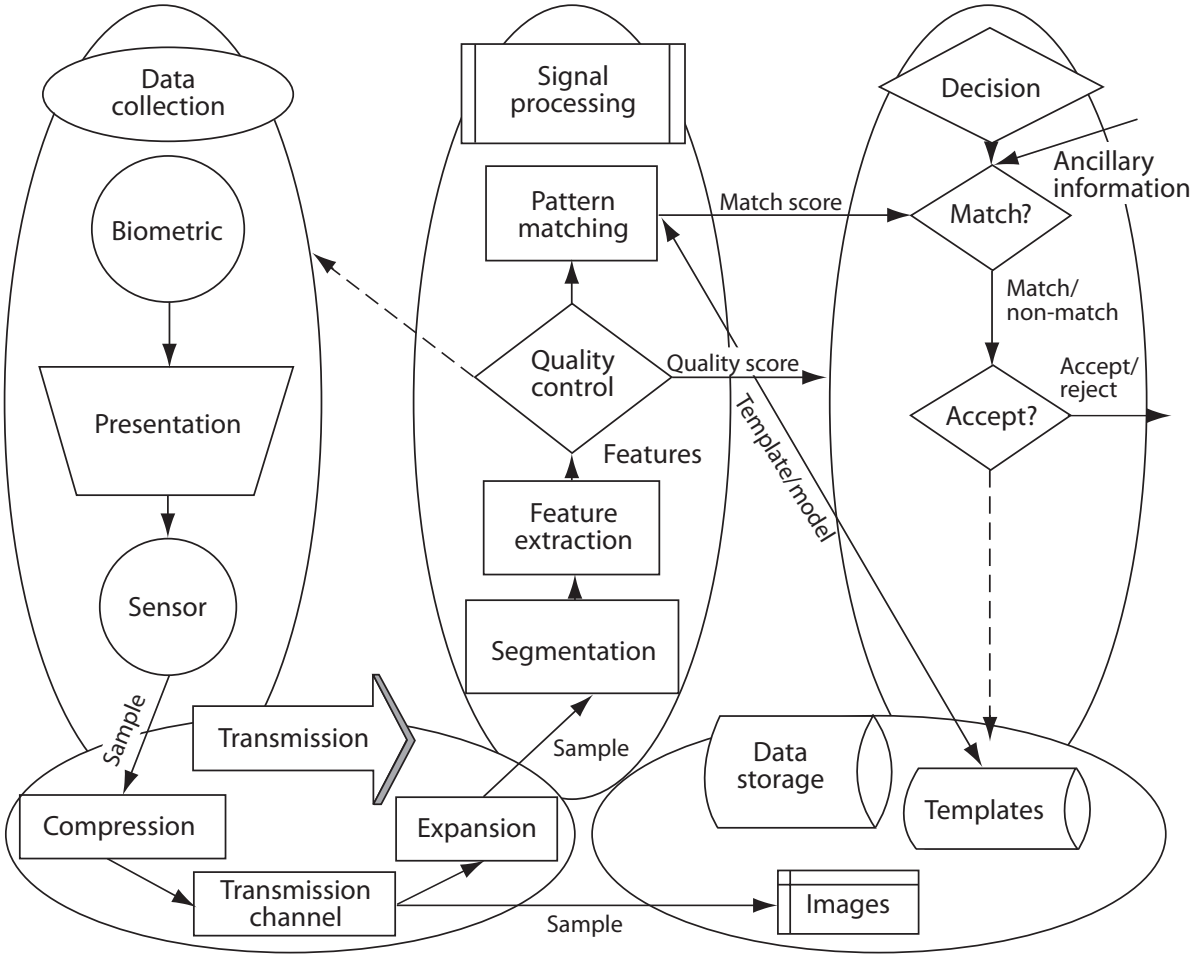

Figure 1 A generic biometric system—exact reprint from [1]

From the point of view of designing an identification system, decision and signal processing subsystems are the most interesting. Decision subsystem is responsible for training and also measuring the quality of the training. The SPS is responsible for extracting and selecting features and for giving the DS enough data.

5.1.2 Modes of operation

Operation mode

During operation, the interaction flow is simple: SPS prepares a sample to evaluate, matches it with templates from SS and passes this matching result to DS. DS then decides whether to accept or reject the resulting match.

Training and tuning mode

During training, decision subsystem is given a new part called *tuner* which is responsible for training. The tuner controls the whole training and tuning process (the tuner is missing in the original figure 1).

The tuner adjusts SPS parameters and asks it to repeatedly extract and select features. The features' quality is then evaluated using templates stored in DS. The entire process repeats many times until the system shows the desired level of

operational quality. It is typical at this tuning phase (see chapter 5.3) to change the SPS parameters and store and/or adjust the newly acquired templates.

The general biometric identification system model according to [36] also inherently contains primary metrics (needed by the DS) describing its operational state and quality—they are the FMR and the FNMR.

In order to have a better insight into these quality metrics and into general problems of feature selection (needed by the SPS) used in this dissertation, the chapters 5.2 and 5.3 are presented.

5.2 Quality metrics

It is important to thoroughly measure the quality of biometric identification systems and methods at least for the following reasons:

- All these methods use probability and are inexact.
- Being aware of exactly how much various methods can fail is the key to deciding how suitable each particular method is.
- Expressing the quality of results in standardized way allows comparison of quality among methods.

A good overview of quality and performance metrics of biometric methods is given in [37], it describes the FNMR, the FMR, the EER, the ROC or the FTA. The FAR and the FRR are, for instance, mentioned in [38].

Metrics can be classified by their nature to *rates*, *curves* and *points*:

Rate metric
: is a function that samples output of the identification system for more inputs with respect to the value of some driving parameter χ. Rate metric is usually expressed by a graph or with an equation to compute a single point of the metric. Examples of rate metrics are the FNMR or the FMR.

Curve metric
: is a function of the mutual relationship between some rate metrics. The function is usually presented as a graph. Example of curve metric is the ROC.

Point metric
: is a value with some special meaning, usually the point is taken from rate metric or curve metric. Examples of point metrics are the EER or the FNMR100. Some point metrics also denote operating points of the identification system.

5.2.1 FNMR

The FNMR is rate metric [37]. The exact meaning of the abbreviation is the False Non-Match Rate. The parameter χ driving the metric is similarity s (12). The

FNMR metric tracks results of identifying *genuine* samples that have a corresponding entity in the identification system. An ideal identification system would assign all genuine samples to a corresponding entity with 100-% probability, and the sample and the entity would be 100-% similar.

In real identification systems each sample, as well as the entity's features, contains imperfections, so the identification system always succeeds by recognizing samples only to some certain extent. Many genuine samples have their similarity close to 100 % (but not equal to 100 %) and some genuine samples have their similarity far from 100 %. The latter samples are recognized incorrectly by the system and are given a *false non-match*.

Incorporating s to count falsely unmatched samples results in the false non-match rate function. This function expresses how many similarities of genuine samples N_g lie below given s, in the area of unrecognized samples:

$$N_g(s) = \left|g_i \in \mathcal{G}; \xi(g_i) < s\right| \qquad \text{FNMR(s)} = \frac{N_g(s)}{\left|\mathcal{G}\right|} \tag{13}$$

where $\mathcal{G}$ is set on all genuine samples and ξ is a function of the identifying system returning similarity of a given sample, [37]. Codomain of the FNMR defined in this way is [0, 1]. An example of the FNMR is given in figure 2 and distribution $N_g(s)$ for the same FNMR is displayed in figure 3.

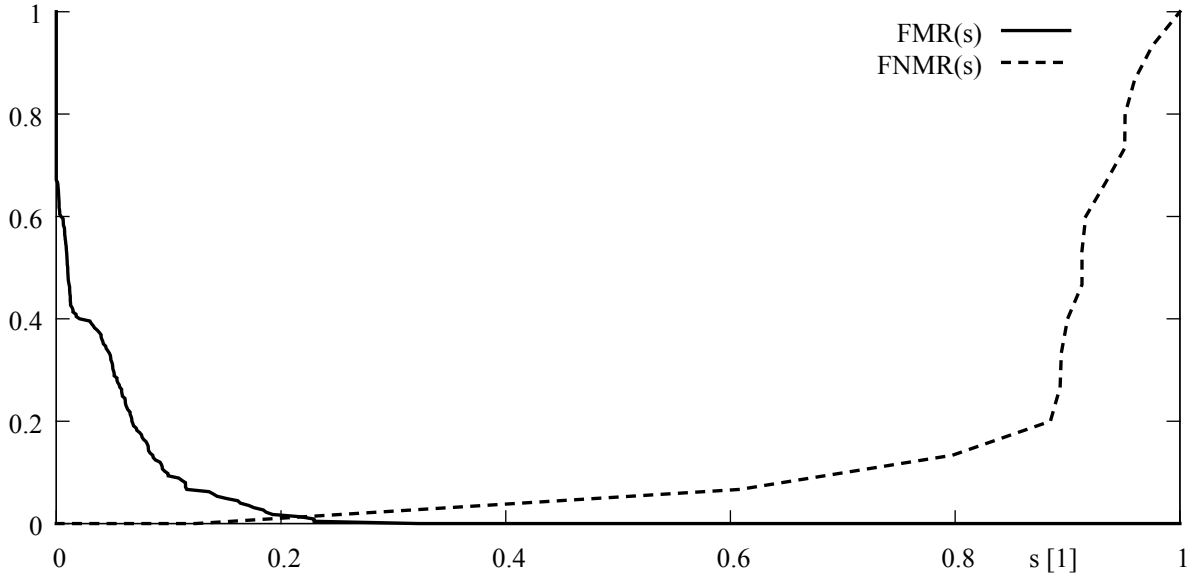

Figure 2 The False Non-Match Rate and the False Match Rate, an example

1 − FNMR(s) is a measure of accepting a genuine sample. For the given s it tells us how many genuine samples are properly accepted.

The FNMR is frequently used to require the behavior of identification systems. The FNMR is specified first and then the corresponding s is computed and taken

as the threshold for deciding about a match. Point metrics FNMR10, FNMR100, FNMR1000 and FNMR0 are used to specify that systems should operate at points where FNMR(s) = 0.1, 0.01, 0.001 and 0.0. The last point is nearly inaccessible in real systems, so it is rather used as a theoretical concept: s of FNMR0 is the threshold above which all genuine samples are correctly accepted.

Setting the FNMR(s) close to 0 increases the tolerance of the identification system—less entities are rejected. Relation to security for the FNMR is weak because the FNMR describes rejection of known, correct samples; and rejecting genuine sample is not a security issue.

5.2.2 FMR

The FMR is rate metric [37]. The exact meaning of the abbreviation is the False Match Rate. The parameter χ driving the metric is similarity s, (12), which is the same as for the FNMR in chapter 5.2.1. The FMR metric tracks results of identifying of *impostor* samples that have not a corresponding entity in the identification system. The ideal identification system would not match any impostor sample to any entity with 100-% probability, and the sample would be completely unlike all other entities.

In real identification systems each sample, as well as the entity's features, contains imperfections, so the identification system always succeeds with recognizing impostor samples only to some certain extent. Many impostor samples have similarity close to 0 % and some impostor samples have greater similarity which may even be close to 100 %. The latter samples are recognized incorrectly by the system and are given a *false match*.

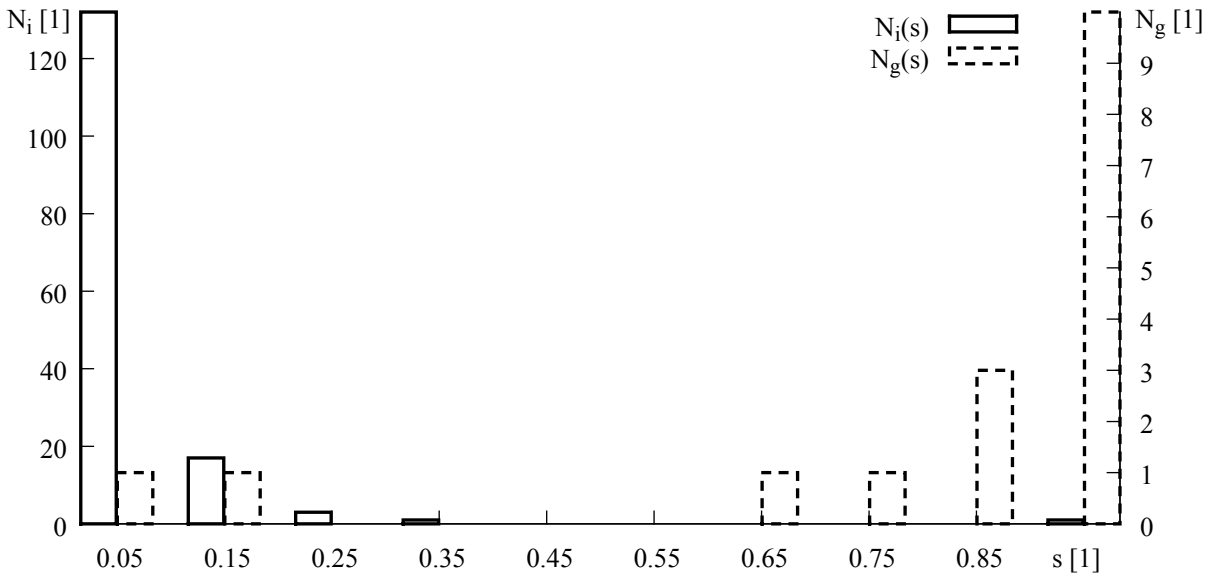

Figure 3 Distributions $N_i(s)$ and $N_g(s)$ of the FMR and the FNMR, an example

Incorporating s to count falsely matched samples results in the false match rate function. The function expresses how many similarities of impostor samples N_i lie above the given s, in the area of recognized samples:

$$N_i(s) = \left|i_i \in \mathcal{I}; \xi(i_i) > s\right| \qquad \text{FMR(s)} = \frac{N_i(s)}{\left|\mathcal{I}\right|} \tag{14}$$

where $\mathcal{I}$ is set of all impostor samples and ξ is a function of the identification system returning similarity of the given sample, [37]. Codomain of FMR defined in this way is $[0, 1]$. An example of an FMR is given in figure 2, distribution $N_i(s)$ for the same FMR is displayed in figure 3.

$1 - \text{FMR}(s)$ is a measure of rejecting an impostor sample. For the given s it tells us how many impostor samples are properly rejected.

The FMR is frequently used to require the behavior of identification systems. The FMR is specified first and then the corresponding s is computed and taken as the threshold for deciding about a match. Point metrics FMR10, FMR100, FMR1000 and FMR0 are used to specify that the system should operate at a point where $\text{FMR}(s) = 0.1, 0.01, 0.001$ and 0.0. The FMR0 is nearly inaccessible in real systems, so it is rather used as a theoretical concept: s of FMR0 is the threshold below which all impostor samples do not match with any entity.

Setting the $\text{FMR}(s)$ close to 0 increases the security of the identification system—less entities are improperly matched. Relation to security for the FMR is strong because it describes accepting the unknown, incorrect samples; and accepting of impostor sample is definitely a security issue.

5.2.3 FAR, FRR

The FNMR and the FMR described in previous chapters evaluate the technical ability of the matching algorithm to correctly link (or not to link) an entity to a sample. This matching algorithm is a key part of each identification system, but still it is only a part. Consequently the FNMR and the FMR measure only a part of the system.

There are more possible uncertainties and failures in each identification system. One example of this is the Failure To Acquire the sample (the FTA) which measures situations like when sensors fail, or there is not enough data. These failures add to the technical measures FMR and FNMR, and summed together it forms outer rates describing how the system behaves as a whole.

These outer i.e. complete rates, are called the False Acceptance Rate (the FAR) and the False Rejection Rate (the FRR). The FAR relates to the FMR and the FRR relates to the FNMR.

All settings and conclusions discussed in 5.2.1 and 5.2.2 are identically valid for the FRR and the FAR, for instance, that a higher FAR means less security [30].

Because the FAR and the FRR look at the system from the outside, they are used more frequently than the FMR and the FNMR. On the other hand, the FMR and the FNMR are more appropriate for quality comparison, when the identification algorithm itself is the subject of comparison.

5.2.4 EER and other operating points

The FNMR and the FMR are constructed using a priori information—during construction it is known whether a sample is genuine or an impostor. Identification systems in operation lack this information and must decide about a sample's genuineness only according to its similarity s. In order to make such decision the similarity itself is not enough, a second number to be compared with is required. This second number is an acceptance threshold T, and is always pre-selected as a parameter of the system. Then both values s and T are compared to obtain a decision. It is obvious that T is a special value of the similarity s.

Due to this, T has its corresponding $\mathrm{FNMR}_T = \mathrm{FNMR}(T)$ and $\mathrm{FMR}_T = \mathrm{FMR}(T)$. Moving T on the scale of s changes both FNMR_T and FMR_T, but the effect on both is different: a bigger T gives greater security (a lower FMR) and less tolerance (a higher FNMR) and vice versa [38].

Choosing the right T is one of the most difficult tasks in order to tune up the identification system. In principle there are two ways to set T up:

- the technical way whereby special points on the FNMR/FMR curve are chosen,
- the statistical way whereby special values of the FNMR/FMR are chosen.

In both cases, the FNMR or the FMR value is given, and T is computed accordingly.

The first way leads to metrics or operating points:

FNMR0
: The FNMR0 is the point on the FNMR curve where it touches zero. The point is usually inaccessible because the FNMR may touch the zero too close to $s = 0$. In this case, the FMR would be unacceptably high and the identification system would identify many entities wrongly.

FMR0
: The FMR0 is the point on the FMR curve where it touches zero. The point is usually inaccessible because the FMR may touch the zero too close to $s = 1$. In this case, the FNMR would be unacceptably high and the identification system would not recognize many entities that it could recognize.

EER
: The EER is the point on both the FMR and the FNMR curves where $\mathrm{FMR}(s) =$

FNMR(s) = Y. It is the intersection of the FMR and the FNMR. The EER as the chosen operating point does not prefer any of detection errors: false match of impostor sample or false non-match of genuine sample.

It depends on the particular paper or system, what EER exactly means: it may either mean complete 2D point (s, Y) or any of s or Y.

The second approach leads to metrics or operating points:

FMR10, FMR100, FMR1000
: These are points on the FMR curve, where FMR(s) is 0.1 (10 %, FMR10), 0.01 (1 %, FMR100) or 0.001 (0.1 %, FMR1000). The higher the number in the point name, the closer s is to 1.

FNMR10, FNMR100, FNMR1000
: These are points on the FNMR curve, where FNMR(s) is 0.1 (10 %, FNMR10), 0.01 (1 %, FNMR100) or 0.001 (0.1 %, FNMR1000). The higher the number in the point name, the closer s is to 0.

In the real world, FNMR0 and FMR0 are only rarely used because they are usually inaccessible (see chapters 5.2.1 and 5.2.2). Their s is usually too close to 0 (the FNMR) or 1 (the FMR), and this worsens the counterpart (the FMR or the FNMR) so that it is at unusable level.

Selecting the EER is a technical choice which does not presume any purpose. Due to this, it is rarely chosen in deployed systems because in the real world the purpose is the key factor. Instead, the EER is being frequently chosen in research and development where fair nature of the EER makes the system function more balanced. The EER is also the point which this dissertation uses in experiments.

The choice of FNMR10, FNMR100 or FNMR1000 is suitable for lookup systems, where a limited number of known entities are searched for within a huge amount of unknown entities. Typical applications are scanning for criminals in airports or public places where a low FNMR rate assures a higher percentage of detections. A simultaneous increase in the FMR is not a problem because identified persons are later inspected by personnel who filter out false matches.

For security targeted systems, the FMR100 or the FMR1000 are points of choice (FMR10 is too large). A low FMR assures a lower number of entities are incorrectly accepted. This behavior is expected in systems permitting entrance, or permitting access to restricted areas or data and so on. Allowing unauthorized persons to do functions requiring authorization is an unwanted situation. The FNMR, which increases accordingly, causes more frequent rejections of known persons, but for security systems, this is acceptable. Usually, these false non-matches are resolved by a security officer or with specific adjustments in identification systems.

5.2.5 Approaches to measure quality of identification

The Equal Error Rate—the generic metric

The ERR is defined as the point where the FMR(s) and the FNMR(s) equals. In graphical representations it is the crossing point of both rates. Because the FMR and the FNMR are almost always taken from measurements, no analytical functions exist that would describe them. Therefore, an analytical solution is typically impossible and computing the EER is almost always based on geometry.

Measuring (or simulating) results in pairs of points $[s, \text{FNMR}(s)]$ and $[s, \text{FMR}(s)]$ composing *polylines*, i.e. paths of concatenated line segments. The EER then can be determined as an intersection point of both the FNMR and FMR polylines.

Distance of the FNMR and the FMR—faster and more sensitive way

The generic EER is important, because it is used in all existing publication related to biometric identification. In this dissertation, a novel approach is used, that has been developed by the author and that is described in a separate publication [49].

The idea is straightforward: the EER's y value is bigger when more FNMR samples get low s (similarity) or when more FMR samples get high s. Consequently, if overall s for the FNMR is higher, or if overall s for FMR is lower, the EER's y value is lower (i.e. better).

The s of FNMR(s) gets its worst (i.e. lowest) value s_l in FNMR0 (see chapter 5.2.4). The s of FMR(s) gets its worst (i.e. highest) value s_h in FMR0, (15). In the words of probabilistic distribution, s_l and s_h are located in *tails* of corresponding distributions. An example of the FNMR ($N_g(s)$, (13)) and the FMR ($N_i(s)$, (14)) distributions is shown in figure 3.

$$\begin{aligned} \text{FNMR0} &= \text{FNMR}(s;\, s \leq s_l) = 0 \\ \text{FMR0} &= \text{FMR}(s;\, s \geq s_h) = 0 \end{aligned} \tag{15}$$

It is not known which random variable type corresponds to $N_g(s)$ and $N_i(s)$ in general, and it is beyond the scope of this dissertation to explore it. In such case, use of Gaussian distribution is the acceptable approximation [31].

Supposing $N_g(s)$ (13) and $N_i(s)$ (14) has Gaussian distribution, s_l will be located in the left tail of distribution $\mathcal{N}_g$ (16) and s_h will be located in the right tail of distribution $\mathcal{N}_i$ (17):

$$\mathcal{N}_g(\mu_g, \sigma_g) \approx N_g(s) \qquad s_l < \mu_g \tag{16}$$

$$\mathcal{N}_i(\mu_i, \sigma_i) \approx N_i(s) \qquad s_h > \mu_i \tag{17}$$

where μ_g, μ_i, σ_g and σ_i are estimated according to (1) and (2) from the groups of genuine and impostor similarities.

Means μ_g and μ_i represent $N_g(s)$ and $N_i(s)$ (and therefore the FNMR(s) and the FMR(s)) enough, but utilizing deviations helps further. Deviations represents spreads of values ranges and therefore incorporating deviations increases sensitivity to border values. The following tail points t_g and t_i represent this idea with adding/subtracting deviations to/from corresponding means:

$$t_g = \mu_g - \sigma_g \qquad t_i = \mu_i + \sigma_i \tag{18}$$

Tail points t_g and t_i are located on $\mathcal{N}_g$ and $\mathcal{N}_i$ similarly as the points s_l and s_h are located in $N_g(s)$ and $N_i(s)$. The relationship is direct because t_g and t_i are derived from approximations of $N_g(s)$ and $N_i(s)$. The idea is visible in figure 4, where a second example of $N_g(s)$ and $N_i(s)$ is given: look at t_g and t_i drawn nearby their $\mathcal{N}_g$ and $\mathcal{N}_i$ in gray.

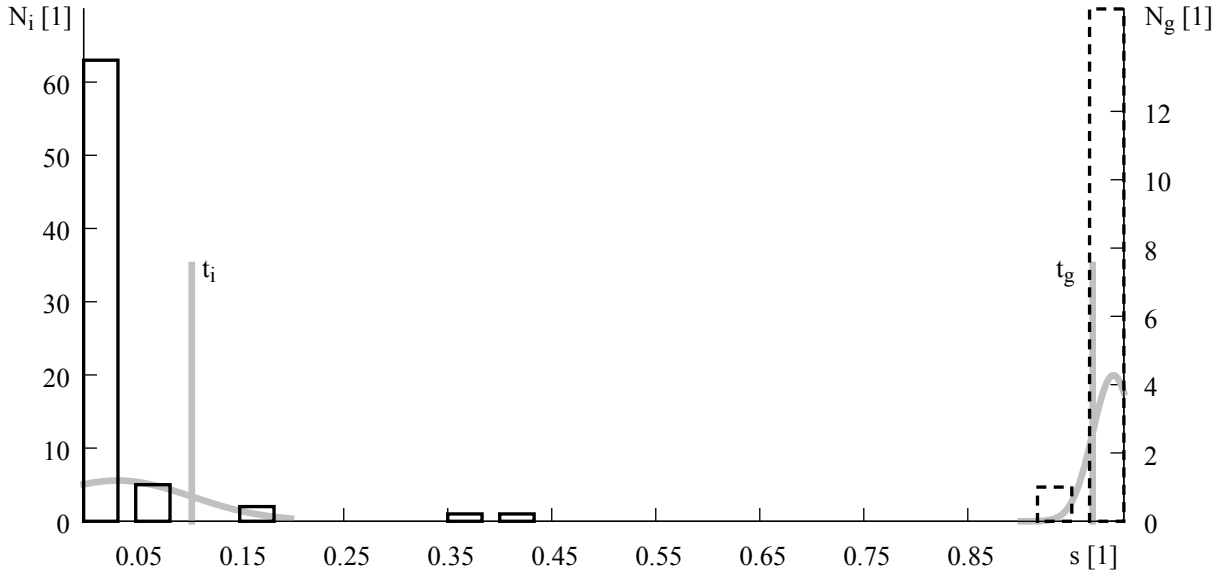

Figure 4 $N_g(s)$ and $N_i(s)$ distributions with Gaussian approximations and tail points

Having both points t_g and t_i, a distance of the FNMR and the FMR is:

$$d_{\mathrm{EER}} = 1 - (t_g - t_i) = 1 + t_i - t_g \approx \mathrm{EER} \tag{19}$$

This distance d_{EER} is the new metric that can be used for evaluating the quality of identification system.

The best value of this measure is $d_{\mathrm{EER}} = 0$ and this value corresponds to EER = 0. In this dissertation, d_{EER} is used in the experimental part for selecting features (see chapter 10.2.3).

5.3 Feature selection

Feature selection forms and important part of identification system in its design, development and tuning phases. According to general model of biometric system [36] (see chapter 5.1), the feature selection is used only in training and tuning mode (see chapter 5.1.2) where it follows the feature extraction step.

According to [39], the purposes of feature selection are:

- the primary purpose is the selection of relevant and informative features,
- general data reduction and feature set reduction,
- performance improvement, and
- understanding data.

The first two points relate to selection itself: selection goes through an initial amount of extracted features, not knowing which feature is relevant, nor how many features to select, evaluates many variants using evaluation criterion and finally outputs a set of selected features which fulfills the primary purpose.

The aim for feature selection in this dissertation is the same, i.e. to select relevant and informative features. An auxiliary aim is also to understand data. However, understanding data is of less importance because without selecting a feature no data can be understood.

For this dissertation, it is enough to describe the main issues of feature selection, including two applied algorithms. For thorough overview of the topic see [39].

5.3.1 General issues of feature selection

The main problems of feature selection are:

- computational complexity, because selecting the best set of features may require evaluating all combinations,
- hard predictability, helpful feature may be irrelevant by itself,
- definition of the evaluation criterion.

The second point is partially linked to the first one, as ability to predict usefulness may decrease computational complexity. A hidden relevance of features is related to the *nesting effect* [39] whereby bad decisions made at the beginning of the selection cannot be corrected later. The experimental phase shows that features extracted from experimental data manifest this nesting effect.

Tested algorithms, the Sequential Forward Selection and the Sequential Floating Forward Selection, are described in chapter 5.3.2. The first algorithm suffers from the nesting effect, the second one overcomes it.

Evaluating the performance of set of selected features may also help reduce computational complexity, because better criteria can remove irrelevant features faster. In this dissertation three criteria are used, all described in chapter 5.3.3.

5.3.2 Selection algorithms, SFS and SFFS

As already mentioned in previous chapters, Sequential Forward Selection (SFS) and Sequential Floating Forward Selection (SFFS), [39], were both tested and are used in this dissertation. Both algorithms have in common:

- Forward selection—the algorithm starts with an empty set of features and then fills the set. Opposite algorithms, the backward selection, shrinks the set.
- Sequential selection—the algorithm adds one feature in each step. Other algorithms may select features arbitrarily, e.g. as stochastic algorithms select.
- Evaluation criterion—evaluation criterion is an objective function that is minimized or maximized by the selection algorithm. In this dissertation it means a function combining similarities (posterior probabilities, see chapter 4.3.2) into EER or its substitute (see chapter 5.2.5). Because the lower the EER the better, the selection algorithm works as minimizer.
- Looking for the best match—for a given step and already built set of features, both algorithms test all remaining features to decide which new feature to add. The newly selected feature is the feature which improves evaluation criterion the most when it is added to the set of already selected features.

Sequential Forward Selection, SFS, [39]

All four described common properties merged together almost completely form the SFS. The only missing factor needed to know is when the SFS stops selecting. These following criteria stop the algorithm:

1. achieving the required number of selected features,
   this is fulfilled when there are enough items in the set of selected features.
2. when the EER, or its substitute, reaches the threshold,
   e.g. when the EER decreases below 0.01 %. It is the absolute criterion.
3. when the EER does not decrease during a certain number of steps,
   e.g. when three consecutive steps do not improve EER by more than 0.001 %. It is the relative criterion using two parameters (number of steps and limit).
4. when there is nothing to add,
   even if there are yet unselected features. It can happen when the evaluation criterion is worse for any added feature that is not yet selected. SFS cannot continue in this situation because it cannot improve the solution in any way.

Of the above criteria, only criteria 2, 3 and 4 were used in experiments in this dissertation. The fourth criterion effectively never stopped searching because the second and third criterion always had stopped searching earlier.

The SFS is a simple, fast and straightforward method. On the other hand, features extracted from mouse-like devices data manifest a nesting effect, which is an unsolvable problem for SFS.

Sequential Floating Forward Selection, SFFS, [39]

SFFS makes use of a backtracking step, which SFS does not have. Once a new best-improving feature is added, the SFFS tries to remove features from the selected set till the evaluation criterion improves. This removing phase is the advantage that helps SFFS overcome the nesting effect. This relates to the word *floating* in the algorithm name: a set of selected features floats, the size and the content of the set is not fixed in any algorithm step.

After the backtracking step a new forward step follows. Because of this the algorithm is liable to oscillations, when the same feature can be repeatedly added and removed. Defense against oscillations is simple: SFFS keeps track of the best result for each size of the set of selected features. It allows the SFFS to reject additions or removals of features if the evaluation criterion for the new set size would not change. As a beneficial side effect, maintaining performance results for each feature set size also allows deciding about the best number of features.

Criteria to stop the searching are:

1. achieving the required number of selected features,
   this is fulfilled when all variants are evaluated that have feature set size up to the required number.
2. when the EER, or its substitute, reaches the threshold,
   e.g. when the EER decreases below 0.01 %. It is the absolute criterion.
3. there is nothing to add or remove,
   either in the case of exhausting the available features or in cases when no addition or removing improves the evaluation criterion.

In the experiments, criterion 3 was never reached because searching always stopped due to the first or second condition.

The SFFS is more complex and is slower than the SFS. However, experiments showed (see chapter 10.2.1) that SFFS almost always found a better set of features.

5.3.3 Criterion functions

Criterion function drives feature selection algorithm in two ways:

- first, it simply compares which feature set gives better result,
- secondly, it can speed up selection if the criterion is more precise and stricter.

The first property is necessary, while the second property is just preferable.

In this dissertation, three variants of criterion function are explored: using only single posterior probability, the geometric approach computing the EER using the FMR and FNMR curves, and lastly the derived approach using the FMR and FNMR distributions (see chapter 5.2.5).

SPP, single posterior probability (12)

The measure of selection quality is similarity s_e, probability that an entity belongs to a given sample. It is computed from prior probabilities, so all entities are taken into account. To select the best set of features, (12) is expected to be as close to 1 as possible. SFS or SFFS using this criterion works as minimizer of $1 - s_e$ because $s_e = 1$ is the best and maximal value.

To be able to use (12) as an evaluation criterion, it must be related to the EER (see chapter 5.2.5, part about relative EER comparison). The relationship is straightforward: the bigger s_e is, the lower the number $(1 - s_e)$ is that must be divided into remaining entities. The remaining entity distribution is forced to be close to 0, and the genuine distribution represented with single s_e is close to 1.

The discussion in chapter 10.2.3 of experimental part shows that this approach works, but it gives results that are different from the other two methods. This method though, is very fast.

EER/polylines, EER obtained geometrically from polylines

This criterion directly uses the EER, so there is no need to consider an equivalence to it. The EER is computed as an intersection of two polylines, the FMR and the FNMR (see figure 2). To find the intersection, all segments of one polyline must be tested against all segments of the second polyline. The task is time consuming because the time complexity of the algorithm is $O(n^2)$ [49].

Constructing FMR and mainly FNMR requires many points; in experiments at least a number of entities is used. Each point on the FMR or FNMR requires individual computing (12), which slows computing down.

SFS or SFFS works as minimizer of the y-coordinate of the EER in this case.

d_{EER}, EER replaced with (19)

This approach is described in detail in chapter 5.2.5. In short, finding intersection of polylines is slow and not very precise. d_{EER} uses FMR and FNMR distributions (which are precursors for FMR and FNMR polylines) to produce equivalent measure faster and simpler.

SFS or SFFS using this criterion works as minimizer of d_{EER} because the best and minimal value of the d_{EER} is 0.

6 MOUSE-LIKE DEVICE IN BIOMETRICS

Mouse-like devices look like they are able to generate enough data for identification purposes. In order to discuss and hopefully prove this fact, principles and properties of the method need to be explored, explained and linked to previously described general terms.

In this chapter, *principles* and *links* to behaviometrics are firstly discussed. This is followed by some *questions*, and also the *pros and cons* are summarized. Finally, understanding the principles and problems, a *four-layer model* of mouse-like device identification is introduced. The model adheres to the general model described in chapter 5.1 and is structured according to the needs of this dissertation.

6.1 Principles

Moving and clicking mouse-like devices to operate the cursor in GUIs and/or to press some GUI control requires precise arm action as well as tactile and visual feedback. This complex interaction is fully controlled by the brain and its regulation loops.

When the brain processes input information and sends messages for the arm to move, at least the following factors are involved:

- The speed the brain can process images.
- The speed of transferring excitation along nerve fibres.
- The sensitivity of tactile sensors and the sensitivity of movement sensors in muscles, tendons and cartilages.
- The parameters of the brain's regulation loop: the speed, the smoothness of the movement and the quality of the loop. The regulation loop is not inborn, but develops during the first months of life.
- How sensitively the muscles react to stimulation.
- The speed at which muscles contract, and the amount muscles contract.
- The weight of all tissues in the arm.
- The geometry of the arm and distances regarding arm levers.
- The elasticity of tissues and the flexibility of the arm which limits movement.
- Trained movement patterns, e.g. how to move the arm by sight control. Some people move mouse-like devices quickly and precisely, while others quickly and imprecisely (then corrections are applied), and even others move mouse-like devices slowly due to fear of making mistakes.

As can be seen, many factors are involved and each factor is highly individual. It might be possible to measure factors one by one, but this is not easy. E.g., trained patterns and parameters of the regulation loop are almost immeasurable because they require studying the dynamics of the entire control loop and this is affected by all the other parameters.

In order to point a cursor using a mouse-like device, the entire arm is involved and it is affected by all the above-mentioned factors. Therefore the movement and its control is dynamic. This fact is a reason for classifying this method as behavioral and/or behaviometrics.

6.2 Questions about the method

The parameters affecting mouse-like device movements were discussed in a previous section, as was the fact that all factors are individual. This is the principal reason why the method can be used for identification purposes. However, many questions about the method need to be addressed:

- Are the physiological factors that control movements unique enough?
- How quickly can a person be identified using mouse-like devices?
- What happens when there is a change in devices used, such as using a different mouse or a different computer?

The answers to these questions are partially known because previous research has shed some light in these fields of study. This dissertation touches on all three points in the experimental part.

6.3 Pros and cons

Advantages of mouse-like device identification:

- Wide availability. Almost every PC today has a mouse-like device included.
- Fast data rate. Even after a short time working on a computer there are dozens of clicks and hundreds of positions related to mouse-like device moves.
- Reuse of information. Identifying the movement of mouse-like devices can be done without special equipment or special procedures bothering the user. It uses side channel information from mouse-like device communication.
- Simplicity.
- Continuity. Identification takes place all the time.

Disadvantages of mouse-like device identification are:

- Not immediate. Obtaining and evaluating information takes some time.
- Possible reliance on hardware. Identification using various mouse-computer pairs may not be possible using this method.
- Unknown robustness and reliability. Will the method work when a person changes their individual factors either intentionally (in an attempt to hide their identity), or unintentionally when tired, injured, or the like?

6.4 Four-layer model

All identification methods use different abstractions in their algorithms, and mouse-like behaviometrics is no exception. For better insight, a general model has been

outlined (see chapter 5.1), that has been fully applied in the four-layer model used in this dissertation. An overview of this model is displayed in figure 5.

6.4.1 Implicit layer

The implicit layer essentially allows identification to work. This layer contains the physiological and behavioral characteristics of the individual; these characteristics cannot be expressed in numbers and they form the foundation of this method. Mouse-like devices are moved by muscles. Feedback on the position and movement is given by muscles and also by tactile and visual sensors. All factors mentioned are highly individual. An overview of factors involved in moving mouse-like devices is given in chapter 6.1.

All these physiological and behavioral characteristics manifest as a complete, complex mixture. Detailed study from a physiological or psychological point of view is beyond the scope of this dissertation. However, an overall view is useful because it is primarily these characteristics that affect the identification:

- Knowledge of how the brain and body control mouse movements is important in order to understand the dynamics. For example, if the brain controls position, it makes sense to focus on positions, position corrections, or places of clicks; if the brain controls acceleration, it makes sense to focus on acceleration.
- The sensitivity and quality of coordination affects small unintended moves and how precise movements are. Therefore characteristics like the jitter, the frequency and size of backward movements or time before clicking (stabilization, settling time etc . . .) could be important.
- The speed of movements, coordination quality and general tendencies in movements first affect factors of time. Then characteristics like the settling time, times in multiple clicks or overall duration of an action may be significant.

Concerning this implicit layer, this dissertation uses all three groups of characteristics mentioned above in the model layer (the layer extracting movement markers and features, see chapter 6.4.3 below). Besides this, examining the features selected for each individual person gives information about how mouse-like device movements are controlled.

The layer corresponds to the data collection subsystem (DCS) of the general biometric model (see chapter 5.1).

6.4.2 Input layer

Windows operating system uses single abstraction for all mouse-like devices. This abstraction sends a continuous stream of measurements to API. These measurements contain position information and button activity, for each activity it is known which button is pressed and which are not.

The operating system usually divides the layer for its own purposes. Available data comes from the HID driver in raw form and the operating system modifies this data by a procedure called the *user experience filter*. The experience filter builds a modified stream of coordinates from the driver's data with dynamics and limits applied. A typical example of a user experience filter is the acceleration parameter of Windows mouse settings.

Here lies a question: which data, raw or filtered, is more suitable?

- The driver's data corresponds to real physical movements. This means that detecting dynamic processes should be easier because no filtering is applied to the data. On the other hand, the driver's coordinates are not seen by the user, so the brain's feedback is taken from different values.

  Let's call this *movement-measurement* approach.

- Data that is filtered by the user experience filter corresponds to what the user can see. It should mean that the brain controls the movement using these filtered coordinates. On the other hand, the filter transforms the data, so the original raw coordinates are lost and the algorithm can hardly detect if the user or the filter is accelerating.

  Let's call this *movement-eye* approach.

One of the goals of this dissertation is to decide which coordinates identify better.

The layer corresponds partially to data collecting subsystem and fully corresponds to transmission subsystem of the general biometric model (see chapter 5.1).

6.4.3 Model layer

In general, the model layer compacts and converts the input layer data into template data, it extracts markers and features that describes individuals. According to this, the model layer contains everything related to the particular identifying algorithm used and may vary heavily among various approaches.

Two groups of approaches exist in current research:

The time-geometric approach

This approach obtains input data, organizes it into geometric groups and then evaluates various characteristics of these groups. The time factor can be involved diversely, it can be used for example as grouping criteria, or as a matter of analysis after groups are established.

This approach does not attempt to guess what the user does, but rather focuses on coordinates and the relationship between time and position. Nothing more is available in input data, so the approach can be viable [8].

This approach is not used in this dissertation.

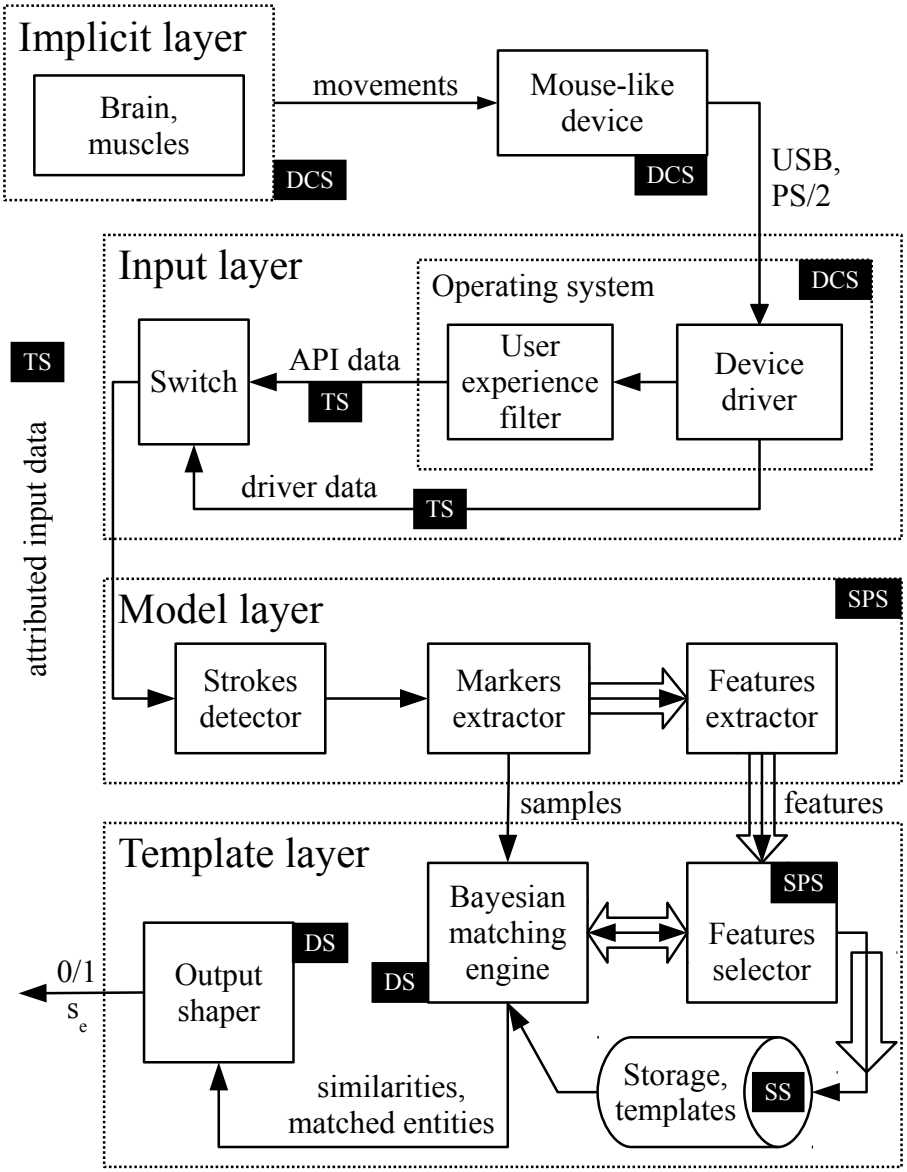

Figure 5 The four-layer model, a specialization of general model (see chapter 5.1). Black boxes refer to the general model.

The user action approach

This approach attempts to construct more abstract activity units from the input data. [4] for example calls these units *strokes* and this dissertation accepts this name. Information is added to input data on the presumption that these units used in particular GUI have a goal that the user wants to achieve. For each particular stroke, various quantities are derived like the length, variance from the middle path, curvature and so on, and then the corresponding markers are computed as statistical properties of the values of these quantities.

The user action approach guesses what the user does. Guessing is implicitly incorrect therefore this approach can be less precise. On the other hand, searching for identical user action can considerably improve statistical processing because more actions are available from which statistics are gathered.

The model layer computes either complete features or the computation ends only with obtaining the markers. It depends on the current operation mode of the identification system (see 5.1.2)—in the *operation* mode the final product of the model layer is the *markers*, whereas in the *training* mode it is the *features*.

The data and action flow in the training mode is displayed in the figure 5 with hollow arrows—what is connected using hollow arrows only works in the training mode.

The fact that *the single intention to perform some action* exists (i.e. moving the mouse to a web link or clicking a button to open a document) is the principal presumption that people can be identified using this user action approach.

Data processing in the model layer can be greatly adjusted and tweaked. However, if this tweaking is not done carefully in respect to statistical rules, this processing can bring artifacts which can manifest as noise, cross correlations and similar problems. Preparing and extracting data in this manner is a large part of the experimental part.

The layer corresponds to the signal processing subsystem of the general biometric model (see chapter 5.1).

6.4.4 Template layer

Results from the model layer, that are various extracted markers and/or features, can be directly used for matching by the algorithm or the database. Involving one more template layer compared with using just markers and features directly brings the following benefits:

- It separates outer usage from internals of the algorithm and the database. The model layer can easily change its interface in this arrangement without disturbing the clients who use it.
- It separates concerns. The model layer should apply the abstract model to input characteristics in order to obtain simplified characteristics expressed with markers and features; the model layer should create them. The template layer should only use results from the model layer. It should read markers and features, evaluate their matches and also select the best set of features during the tuning phase, but not create them.

The template layer contains two parts: matching with decision making, and training. Both parts are thoroughly explored in the experimental part of this dissertation.

The template layer corresponds partially to the signal processing subsystem and corresponds fully to the data storage and decision subsystems of the general biometric model (see chapter 5.1).

EXPERIMENTAL PART

The experimental part of this dissertation describes the process leading from raw mouse data to evaluating the ability to decide which mouse data belongs to whom. Because at the beginning of the process there is a mouse-like device with its simple coordinates, everything is described and explained in bottom-to-up manner—simple representations are built into more complex ones.

7 ABOUT THE EXPERIMENTAL PART

The experimental part consists of experiments and description of research. Research and experiments depend on each other, which is why descriptions of experiments always contain the corresponding part of research and also a discussion of results (or of a partial result). The top-level structure of the research and experiments is given in chapter 7.1.

All experiments explore real data taken from real users. In order to grab this data and in order to carry out all experiments in an intended way, a special software suite has been developed for the purpose of this dissertation. The suite is briefly described in chapter 7.2.

7.1 Top-level structure of the experimental part

The overall view of the experimental part is as follows (the whole experimental part corresponds to the four-layer model displayed in figure 5):

- The first experimental environments and type of obtained data are described in chapter 8. Here is a brief description of what is read from mouse-like devices.
- Mechanisms of cleaning and adjusting input data are then described in chapter 9, together with an explanation of *feature extraction* that reduces the input data to markers and features.
- Features are many and a procedure is needed to select only some of them. The corresponding *feature selection* is described in chapter 10, together with a discussion. The use of input data divided into training and tuning sets is also described in this chapter.
- The final section of the experimental part contains various validations—applying validation data sets to trained entities within a single environment and/or source, and also cross-validations—applying validation data sets to trained entities of different environments and/or sources.

7.2 The software developed for experiments

The software developed for the purpose of this dissertation consists of two parts, the *grabber* and the *identification system*, both programmed by the autor:

- The grabber is a standalone application running in the background. It was used on each computer where data for experiments was stored. The purpose of the grabber is to load data simultaneously from all chosen data sources and to store this data into a file.

  Sample output of the grabber can be seen in listing 1, the line format is:

  ```
  "trk/"<source>":" <dt> <flags> <dx>">"<x> <dy>">"<y> <wheel>
  ```

where <source> is R for HID record, or H for API record (see chapter 8.2); <flags> contains status of five mouse buttons; <dt>, <dx> and <dy> are differences to previous record from the same source; <x> and <y> are current absolute values and <wheel> contains a number of wheel ticks.

The grabber does not process data in any way leaving all processing to the identification system. Source codes for the grabber can be found on dissertation's CD in the folder <Software/Grabber>.

- The dissertation's identification system, i.e. program, realizes all steps and activities from the input data cleaning up to the comparison of environments and data sources. The program reads the file prepared by the grabber and calculates all information needed for this dissertation.

  The program contains all algorithms described in the dissertation, namely:

  — Catmull-Clark and spline smoothing,
  — estimating random variables parameters and calculating prior probabilities,
  — computing posterior probabilities using the law of total probability,
  — calculating EER and d_{EER}, SFS and SFFS,
  — input data cleaning, the detection of strokes and the feature extraction,
  — the feature selection and all experiments.

  The program also contains all code necessary for dumping out results of experiments. The format of all output data files is CSV.

  Source codes for the identification system can be found on dissertation's CD in the folder <Software/IdentificationSystem>.

```
trk/R: 7.9648 ----- 3>669 0>167 0
trk/H: 8.0173 ----- 1>852 0>186 0
trk/R: 8.0389 ----- 1>670 0>167 0
trk/H: 8.08 ----- 1>853 0>186 0
trk/R: 8.101 ----- 2>672 0>167 0
trk/H: 7.9825 ----- 1>854 0>186 0
trk/R: 7.999 D---- 2>674 1>168 0
trk/H: 0.1892 D---- 0>854 0>186 0
trk/H: 7.8075 ----- 0>854 0>186 0
trk/R: 8.0076 ----- 0>674 1>169 0
trk/H: 8.0087 ----- 2>856 1>187 0
```

Listing 1 Example output of the program grabbing user input

8 ENVIRONMENT AND DATA

Experiments need well-described environments (the question is *where*?), a clear procedure for obtaining data (the question is *how*?), and the specification of which data will be obtained (the questions here is *what*?). All three of these factors are discussed in the following three chapters.

8.1 The environment

Four types of environments are used:

Controlled environments E^- and E^+
: These environments have the stable combination of a mouse, a real computer (not a virtual computer machine) and user experience filter settings. The settings of the user experience filter remain the same for all users.

  To achieve identical conditions for each user, users were required to log in to a new session at the start of each obtaining of data. The login process assured that the same combination of processes still operated.

  Users were instructed to do whatever activity requiring a mouse for at least one hour. No activity was presumed and for users' convenience Machinarium [40] and Samorost games, Inkscape and OpenOffice Draw (both vector graphic editors) were installed.

Varying individual environments E^e
: These environments are varying, random environments of particular users. These environments are not controlled in any way, no applications are presumed, nor pre-installed. All E^e differ from both E^- and E^+.

  Users were given the same instructions as they were in the previous environments, to spend at least one hour doing mouse-requiring activities.

Synthetic environment E^s
: This environment is an artificial abstraction that encapsulates individual environments E^e. The environment E^s models possible future usage of the identification system when data for individual users are taken in various environments and during many sessions.

Controlled environments E^- and E^+ are intended to obtain comparable data in order to analyze how input layer variants depend on user and processing settings. Varying environments E^e and synthetic environment E^s, on the other hand, are intended to produce data to analyze effects depending on changes in environments.

All environments use Microsoft Windows 7. Possible differences in identification in other platforms were not explored.

8.1.1 Environments E^- and E^+

Both controlled environments E^- and E^+ have pre-defined mouse settings. Basic Windows control panel interface for adjusting mouse settings was used because this is common in all computers.

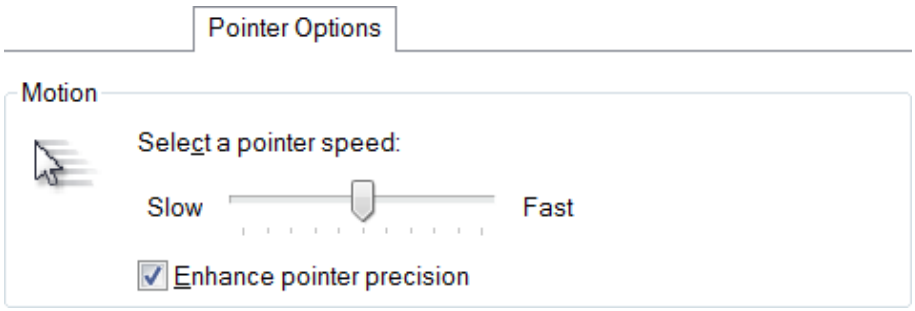

Figure 6 Windows settings for mouse pointer movements

The OS allows the changing of two independent parameters [41] (see figure 6):

Pointer speed

Pointer speed is a factor to multiply tick counts from mouse device in order to get a pixel count. This factor can be set in eleven degrees, where the sixth degree has the value = 1.0 [42][43] whereby the device tick = pixel.

Precision enhancement

Precision enhancement is a non-linear transformation that slows down small movements and speeds up long movements. Due to the fact that precision enhancement changes speed, it is known as an acceleration curve. In Windows, the acceleration curve is represented with five points and values are interpolated between these points.

Curve points are stored in the Windows registry and their settings is not available to the user. A few tools exist that can read and/or change the curve, e.g. CustomCurve [44].

Precision enhancement can only be switched on or turned off as a whole.

Environments E^- and E^+ have been pre-set to values displayed in table 2.

Table 2 Mouse settings in environments E^- and E^+

environment	enhancement	speed
E^-	off	6
E^+	on	6

An overview of environments and data sets arrangements (about data sets see chapter 8.3) is given in figure 7.

8.2 Data sources, obtaining data and data format

Operating system in principle has three places from where applications can read mouse data. The first place, the kernel mode device driver, is not used in this dissertation. The second place is the public interface of kernel at the top of the HID queue where the mouse data is presented in raw unprocessed form. The third place is the application's input queue where mouse data is filtered with the user experience filter and completely prepared for use in an application.

The second and third input places are used in this dissertation, both these sources are described in chapter 6.4.2 from a theoretical point of view. The data from these sources is read using the grabber (see chapter 7.2), technically:

- Raw input data is read from the top of HID queue. The coordinates are relative, and the grabber computes absolute ones. This data is further denoted as driver data D, because it is read from the HID *device* driver's output.
- The application's queue contains data after the user experience filter is applied. The coordinates are absolute and relative ones are computed. This data is further denoted as A, because it is read through the operating system *API*.

Obtained positions are marked with timestamps which have microsecond resolution and these are written to the output file as individual lines. Each line in the output file then corresponds to a single record representing a single mouse event.

8.3 Experimental data sets

From a statistical point of view, the more input data, the better. For the purposes of this dissertation, complete data sets from 16 people were taken. In total, it represents more than 100 hours of mouse movements and more than 1,500,000 tracked mouse events.

Data sets available from each person are:

D_e^- and A_e^- set
: Data taken from the person e in environment E^-, data set D_e^- contains data from a driver source and data set A_e^- contains API data.

D_e^+ and A_e^+ set
: Data taken from the person e in environment E^+, data set D_e^+ contains data from driver source and data set A_e^+ contains data from API.

D_e^e and A_e^e set
: Data taken from the person e in environment E^e, data set D_e^e contains driver data and data A_e^e contains API data.

Together, there are six independent data sets available per individual user, which allows mutual comparison of data sets between users, between data sets and also between environments.

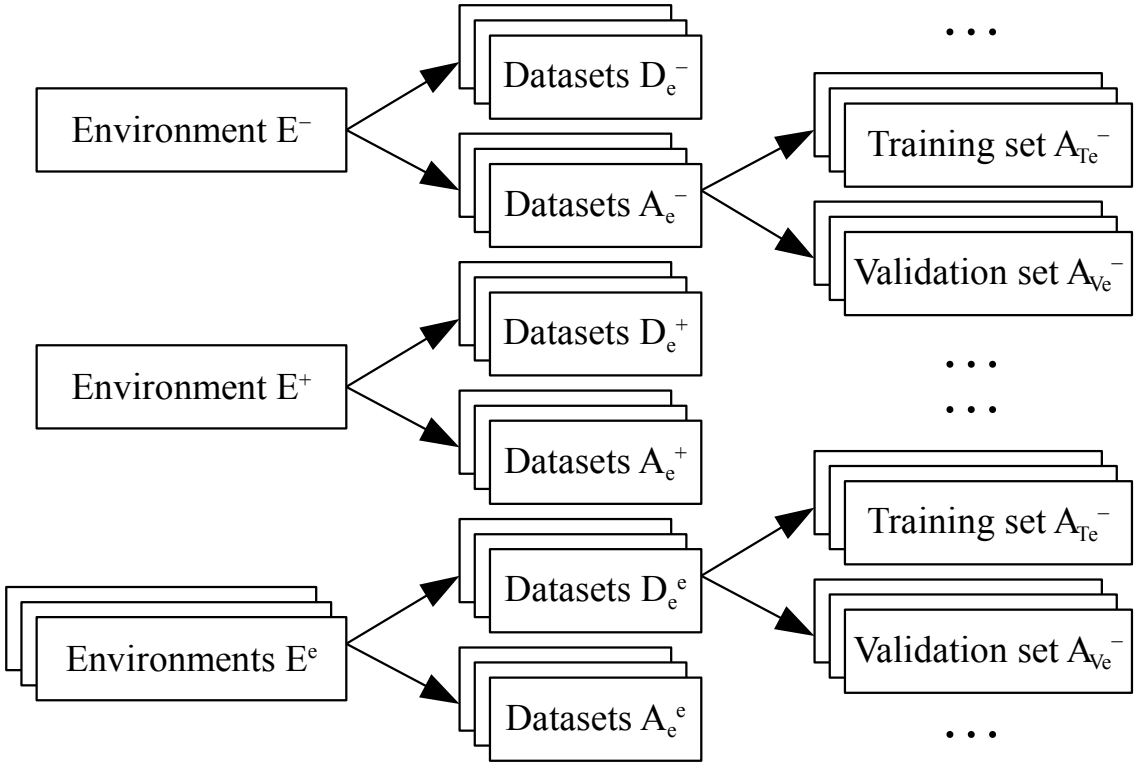

Figure 7 Overview of environments and their data sets

Arrangement of all data sets, together with their origin, is displayed in figure 7.

Note, that the synthetic environment E^s (see page 53) is composed of all data sets D_e^e and A_e^e, and that it has the same size as environments E^- and E^+.

8.4 Summary of environment and data

One of the goals of this dissertation is to analyze and compare various user environments. In order to accomplish this goal, three different environment types were prepared for involved users. The first and the second type of environments was controlled with a pre-defined and stable combination of the mouse and the computer, both these environments differed in the mouse settings. The third environment type was the proper environment of each particular user. All these proper environments were merged to a fourth mixed environment type in order to imitate the real usage of the identification system.

Mouse-like device data can be read from the operating system using two methods: from the HID queue and from the API. In order to compare which method is more convenient, data sets for the experiments were taken from both these sources.

For the each involved user, six data sets were recorded. These six data sets are the only input used in all experiments carried out in the experimental part.

9 FEATURE EXTRACTION

Data sets taken by the grabber are discrete and raw, intentionally unprocessed. To reduce them down to entities, three well-defined steps are carried out:

pre-processing
: which is aimed at fixing irregularities (see chapter 9.1),

building of strokes and their markers
: which divides input data stream into bundles of input events called *strokes*, and computes *markers* as characteristics of the strokes (see chapter 9.2),

extracting of features
: which creates high-level representation of input data based on random variables (see chapter 9.3).

All three of these steps are described in the following chapters.

The feature extraction can run in the two modes according to the running mode of the identification system (5.1.2). Both of these training and operational modes share the pre-processing phase, the detection of strokes and the building of markers. The training mode then continues with building features that finally form an entity. The process and data flow in the training phase is displayed in figure 8.

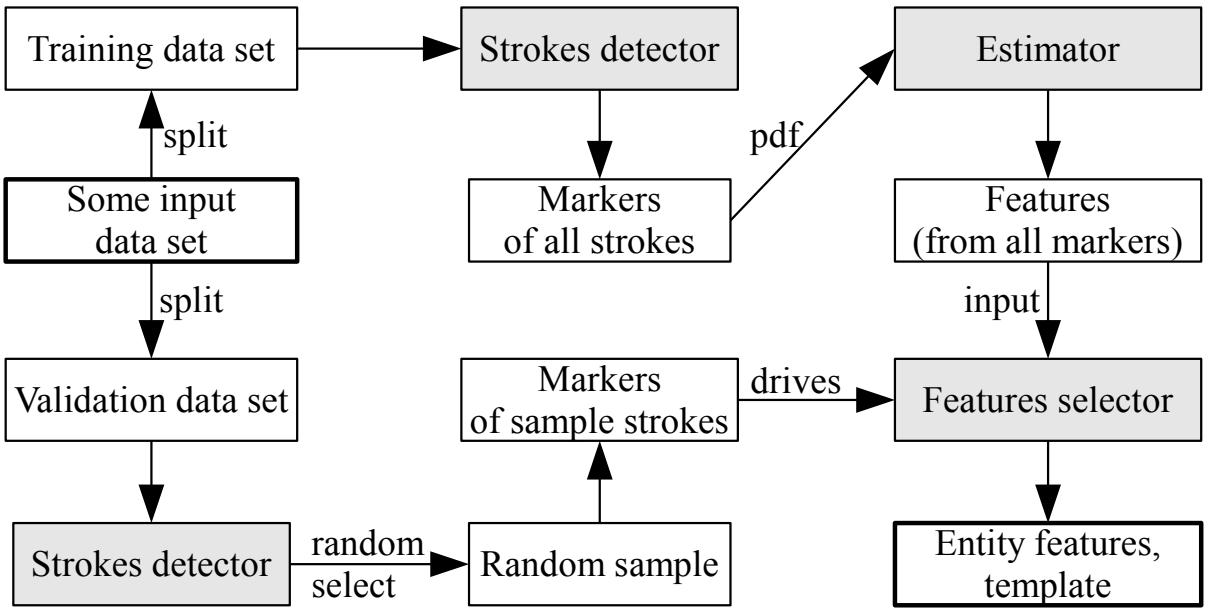

Figure 8 Data flow and data reduction steps in the training mode

The operational mode stops the feature extraction when markers are built. Nothing more is needed because markers are then only matched with entities, that are stored in templates from the preceding training run. Data flow for the operational mode is shown in figure 9.

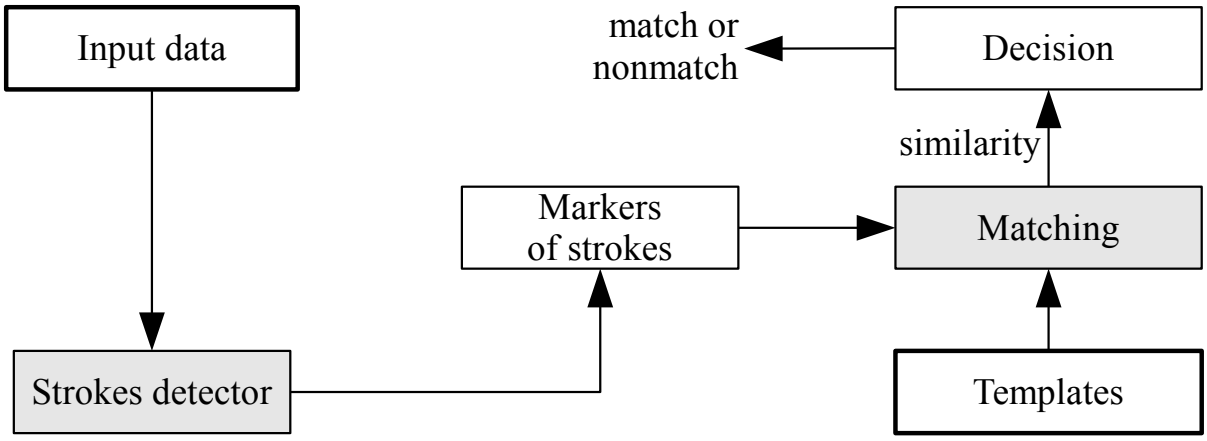

Figure 9 Data flow and data reduction steps in the operational mode

9.1 Pre-processing of input

Data sets produced by the grabber contain original data with no reductions or adjustments applied. This is intentional, see chapter 8.2. Before proceeding with further steps, feature extraction needs the data to be pre-processed.

The fact that movements are random, the limited spatial and temporal resolution of sampling, and also the effort to deliver mouse movements to applications as soon as possible, all lead to artifact. Removal of these artifacts and corresponding data adjustment is described in the following chapter 9.1.1.

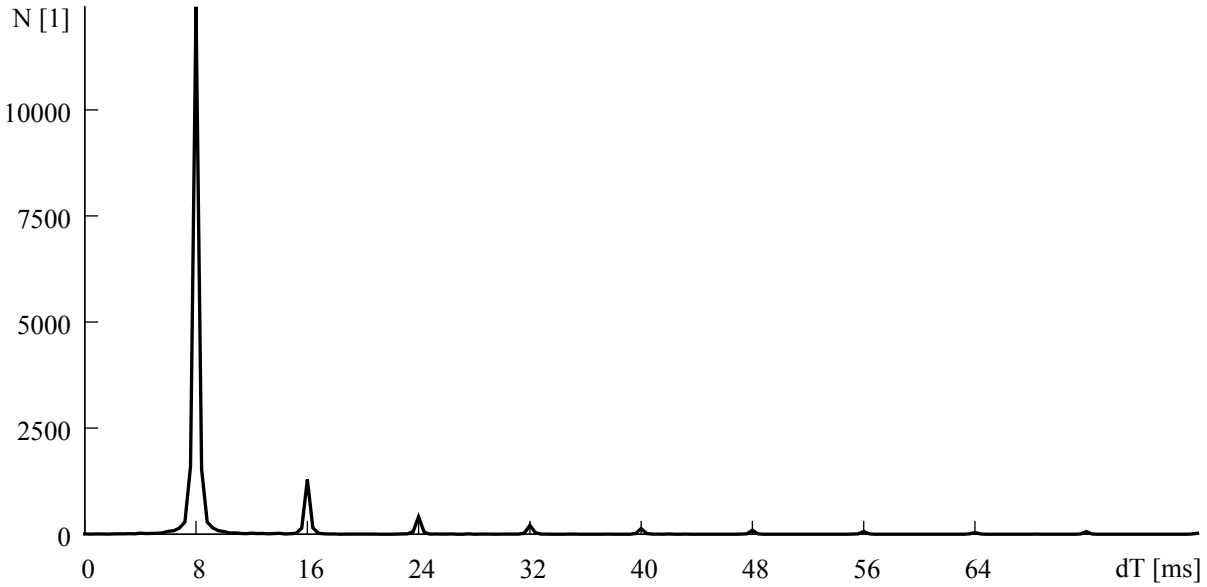

Figure 10 Histogram of time distance of grabbed mouse events, an example

9.1.1 Removal of quantization artifacts

It is clear from figure 10 that the sampling period of grabbed data is 8 milliseconds. The resolution is fairly limited and it causes two kinds of artifacts to appear in input data:

- events with zero time distance,
- events with zero coordinate distance.

Events with zero time distance would normally appear later, but due to the 8-ms sampling period, they merge with immediately preceding events. Zero-time events are inappropriate because, if used, they would lead to infinite movement velocity. To repair these zero-time events, their coordinates (x and y) and mouse button state changes are added to the closest preceding event that does not have zero time.

Events with zero coordinate distance are simply omitted because they do not carry any information about mouse movement. In order to preserve correct information, the time difference and button state changes of omitted events are again added to the closest preceding event.

9.2 Strokes and markers

The input data stream contains information about minimal changes in mouse movements. The reason is that the driver sends information about movements to the operating system as quickly as possible to give the user an impression of immediate responses. However, such an approach divides the single intention of a person performing some action (see chapter 6.4.3) into small unlinked pieces, that must be joined back in order to get the information about the whole intention.

The complete procedure of creating strokes and markers from the input stream is described in following chapters. An overview of the process is also given in the summary figure 16 at the end of this section.

9.2.1 Detecting strokes

The principle of using mouse-like devices in 2D GUI is that the device is a *pointing* device. Before the desired action takes place, the mouse-like device must be pointed or moved to the wanted target, where the action is finally performed. This simple description offers two ways of determining what the stroke is:

- Firstly, the stroke might be a continuous sequence of mouse events that ends when the movement stops, when no movement appears for some time.

  This time that delimits the end of the stroke is called a *time gap ending the stroke*, or simply a *gap* throughout the further text.

- Secondly, the stroke might be a sequence of events that ends when the mouse button is clicked. In other words it ends with performing the target action.

This dissertation uses the first way that utilizes the gap.

In the referred work [4], strokes are detected differently—users manipulated the mouse through a prepared path and then the whole track, including many moves and more clicks, was considered to be a single stroke.

9.2.2 Degraded strokes

Both methods of stroke detection may produce strokes having almost no movement. These can be, for example, delayed clicks (i.e. clicks without a preceding movement), or jitter movements when the mouse, in fact, stays in one place.

Such degraded strokes are unusable for purposes of identifying people because they do not contain information about controlled arm movements. Therefore, degraded strokes are discarded.

Stroke detecting algorithm accept only strokes that fulfil all the criteria:

- the stroke contains at least 5 mouse events,
- the stroke movement spread is bigger than 3 pixels,
- the stroke is not straight.

9.2.3 Smoothing

The limited sampling frequency of mouse inputs ($f_s = 125\,\text{Hz}$, see chapter 9.1.1) causes strokes not to be smooth. Because further processing uses differences as an approximation for a derivative, smoothness is important—insufficient smoothness produces false peaks and improperly extends the limits of measured quantities.

Although smoothing improves the quality of the stroke regarding differences, the stroke still may be considered of insufficient quality. Sometimes changes in mouse movements are sudden and rapid, and the used sample rate does not allow proper tracking. For example, when a stroke directs back and forth movement across a whole screen within 50 ms, it would approximately require about $t_{px} = 50\,\text{ms}/2000\,\text{px} = 2.5 \times 10^{-5}\,\text{s}$ to track a single pixel movement, that gives $f_s \geq \frac{2}{t_{px}} = 80\,\text{kHz}$ [45]. Mouse driver sampling frequency is far beyond this value. Refer to the discussion of the experiment 10.3.1 at page 88 to learn more.

Two methods of smoothing were evaluated:

- the 2D Catmull-Clark subdivision (with less computational complexity) and
- the smoothing spline (with greater computational complexity).

The 2D Catmull-Clark subdivision is a variant of 3D surfaces subdividing [46]. Its principle is displayed in figure 11. This method replaces each inner vertex of the path with three new vertices: two midpoints of path segments adjacent to the replaced vertex, and one barycenter of the original vertex and both midpoints. Looking at figure 11, the original vertex O2 is replaced with new midpoints N2, N4, and the barycenter N3, and the original vertex O3 is replaced with midpoints N4, N6 and the barycenter N5.

The 2D Catmull-Clark subdivision smooths only a very local part of the path. This characteristic is not convenient for smoothing strokes because local smoothing

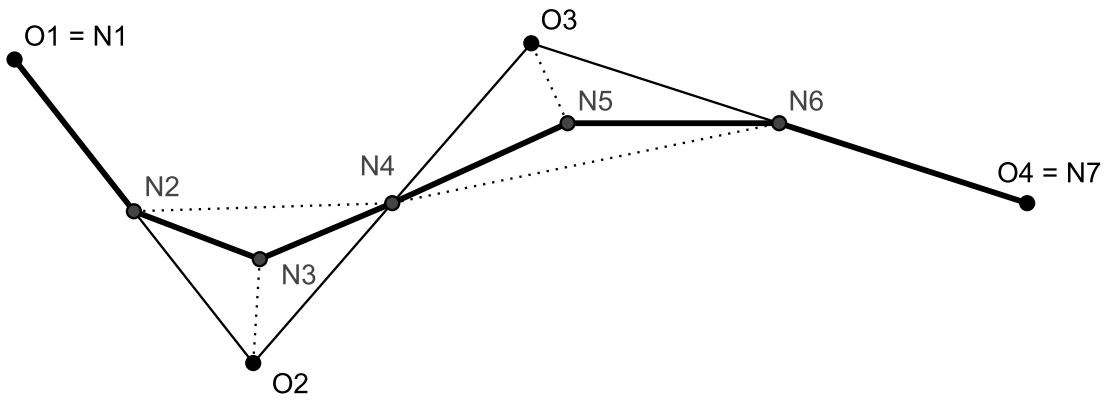

Figure 11 The 2D Catmull-Clark subdivision on a four-segment path

produces sub-pixel changes. It does not respect the path as a whole and it does not remove pixels that are evidently placed off the smooth movement curve.

The second method, the smoothing spline [47], is a method replacing each vertex V_i of the path with a spline S_i connected to the previous S_{i-1} keeping continuity properties up to the second derivative. This dissertation uses algorithms developed for the SSJ (Stochastic Simulation in Java, [48]) project; the algorithm is based on [47]. In a nutshell, smoothing spline is an interpolating spline that is allowed to by-pass interpolated points. The measure of passing by is expressed with parameter λ: $\lambda = 1.0$ means that the spline is interpolating (no passing by), $\lambda < 1.0$ mean that the spline is smoothing (some passing by). The smoothing spline is capable of leaving out points that are off smooth movement path, when $\lambda < 1.0$. Simultaneously the smoothing spline can preserve sharper tips when $\lambda \to 1.0$.

Comparing these two methods revealed that the smoothing spline gives more acceptable results than the 2D Catmull-Clark subdivision. This is likely because the smoothing spline is able to remove off-path points, and is able to respect points on a larger scale. As a result, the 2D Catmull-Clark subdivision is not used in this dissertation and preference is given to the smoothing spline with $\lambda = 0.999$.

Each input event contains three components—the x-coordinate, the y-coordinate, and the time. Therefore three independent smoothing splines must be constructed for each stroke. For each component x, y or t the corresponding spline is built with the help of the artificial parameter p_i^o which represents an independent variable of the spline. In the description that follows, upper index ^{o} means *original*:

$$\vec{x}^o = x_1^o, x_2^o, \ldots, x_n^o \tag{20}$$

$$\vec{y}^o = y_1^o, y_2^o, \ldots, y_n^o \tag{21}$$

$$\vec{t}^o = t_1^o, t_2^o, \ldots, t_n^o \tag{22}$$

$$\vec{p}^{\,o} = 0, \frac{1}{n-1}, \frac{2}{n-1}, \ldots, 1 \tag{23}$$

$$\mathcal{S}_x = C_{\mathcal{S}}\left((p_1^o, x_1^o), (p_2^o, x_2^o), \ldots, (p_n^o, x_n^o); \lambda = 0.999\right) \tag{24}$$

$$\mathcal{S}_y = C_{\mathcal{S}}\left((p_1^o, y_1^o), (p_2^o, y_2^o), \ldots, (p_n^o, y_n^o); \lambda = 0.999\right) \tag{25}$$

$$\mathcal{S}_t = C_{\mathcal{S}}\left((p_1^o, t_1^o), (p_2^o, t_2^o), \ldots, (p_n^o, t_n^o); \lambda = 0.999\right) \tag{26}$$

where n is the number of stroke items; (20), (21) and (22) are vectors of particular components; (23) is a vector of n values of the independent variable distributed evenly into [0, 1]; $C_{\mathcal{S}}$ is the constructor of the smoothing spline and (24), (25) and (26) are smoothing splines for $\vec{x}$, $\vec{y}$, $\vec{t}$. x_i^o, y_i^o or t_i^o are *dependent* variables of the each particular spline.

An example of (25) is given in figure 12.

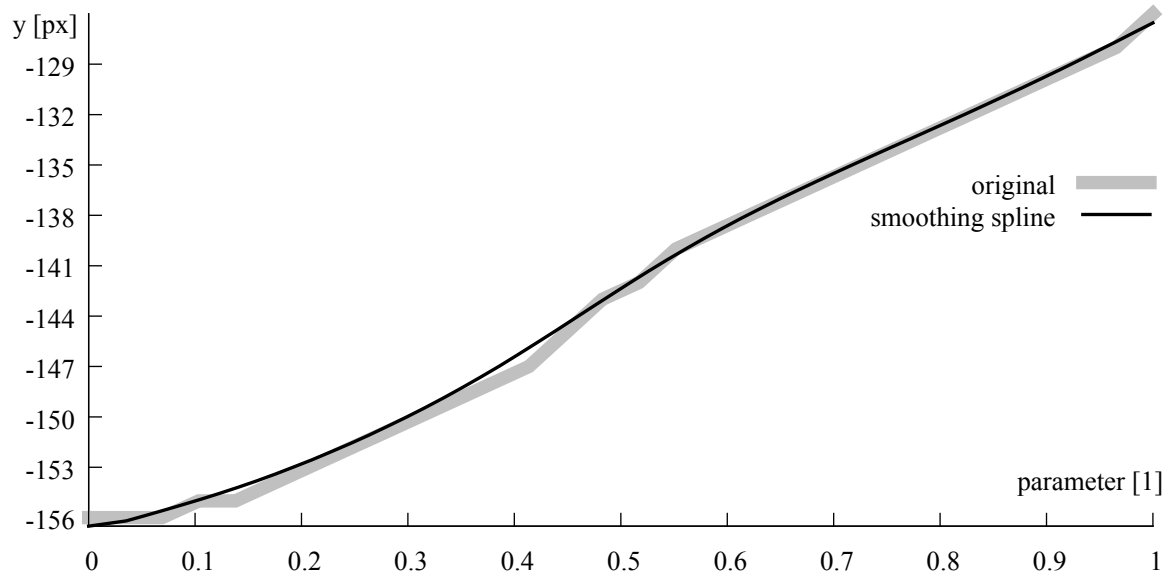

Figure 12 Stroke's *y*-coordinates smoothed with spline $\mathcal{S}_y$ (25), an example

9.2.4 Re-sampling

Strokes contain at least 5 input events. Considering second order differences (which need to be computed to obtain quantities), there is only space for two differences. This is a low number which complicates computing statistical properties of the quantities concerned.

Although algorithm extracting markers (see chapter 9.3) is designed to accept such short strokes, a side effect of smoothing actually helps with the problem. The 2D Catmull-Clark subdivision adds points by principle and the smoothing spline can be sampled as densely as needed, in as many points as needed.

For the purposes of this dissertation, smoothing splines that are constructed from the original vectors (20), (21) and (22) having n components are sampled twice as densely as the original. Therefore newly created re-sampled vectors (27), (28) and (29) have $2(n-1)+1 = 2n-1$ points:

$$\vec{p} = 0, \frac{1}{2n-2}, \frac{2}{2n-2}, \ldots, 1$$

$$\vec{x} = \mathcal{S}_x(p_0), \mathcal{S}_x(p_1), \ldots, \mathcal{S}_x(p_{2n-1}) \tag{27}$$

$$\vec{y} = \mathcal{S}_y(p_0), \mathcal{S}_y(p_1), \ldots, \mathcal{S}_y(p_{2n-1}) \tag{28}$$

$$\vec{t} = \mathcal{S}_t(p_0), \mathcal{S}_t(p_1), \ldots, \mathcal{S}_t(p_{2n-1}) \tag{29}$$

An example of the resulting re-sampled path is displayed in figure 13.

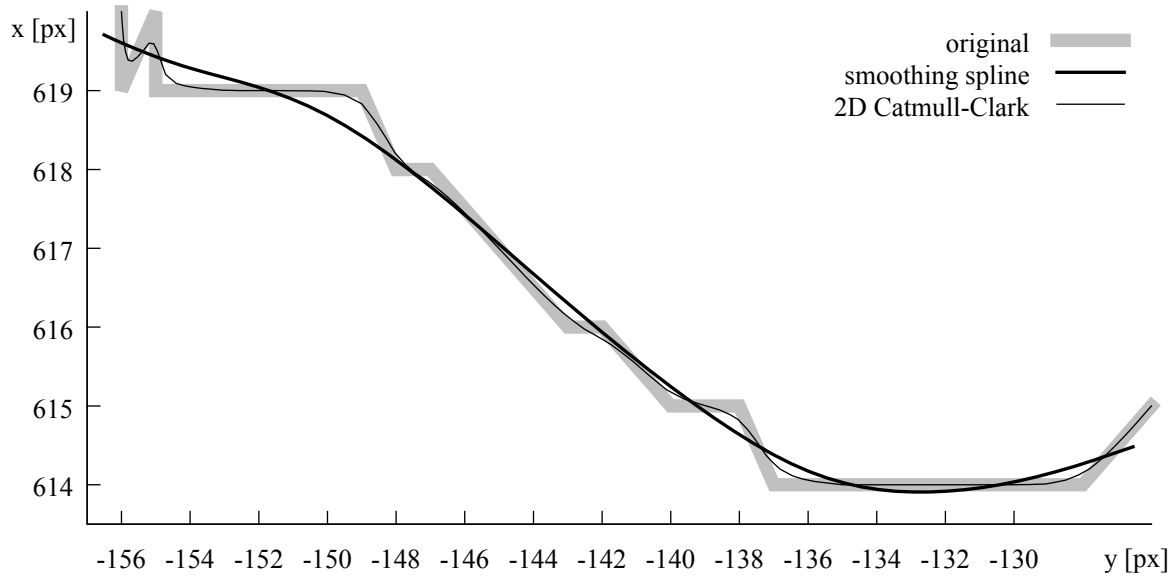

Figure 13 Comparison of smoothing using the 2D Catmull-Clark subdivision and the smoothing spline

9.2.5 Computing markers

Re-sampled strokes are composed of three vectors $\vec{x}$, $\vec{y}$ and $\vec{t}$. They certainly contain information that is specific to the corresponding person, but this information is hidden, sparse and still describes a movement rather than a person. Inspired by [4], this dissertation uses a procedure of reducing input positions to *statistical markers*.

In general, input positions are re-computed to many kinematic and dynamic quantities which are then statistically processed to obtain markers. This process is also shown in figures 14 and 15:

1. the computation of derived quantities
   All three vectors (27), (28) and (29) are processed in order to obtain new vectors of derived kinematic and dynamic quantities.

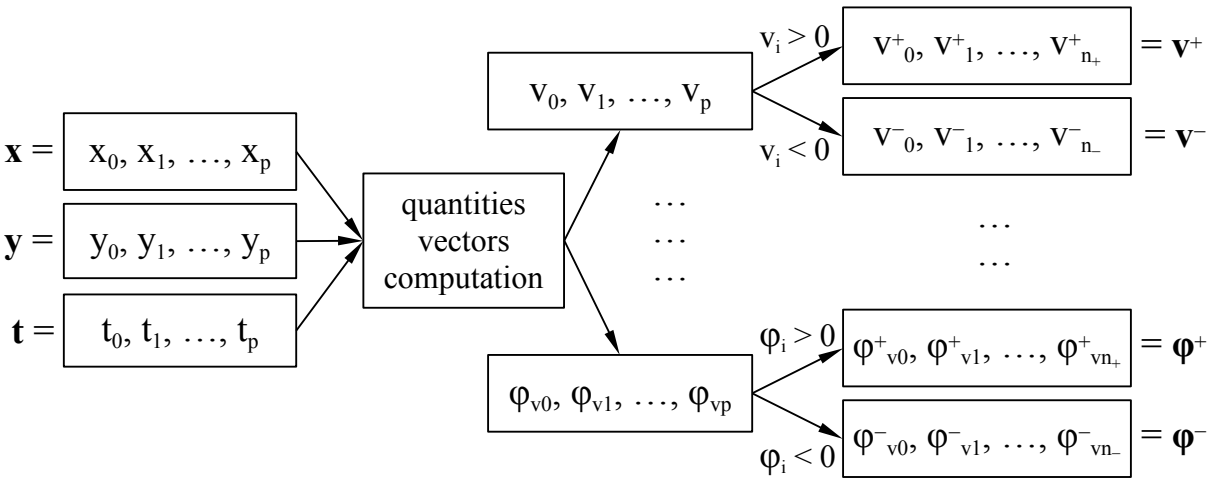

Figure 14 Division of vectors of quantities into negative and positive parts

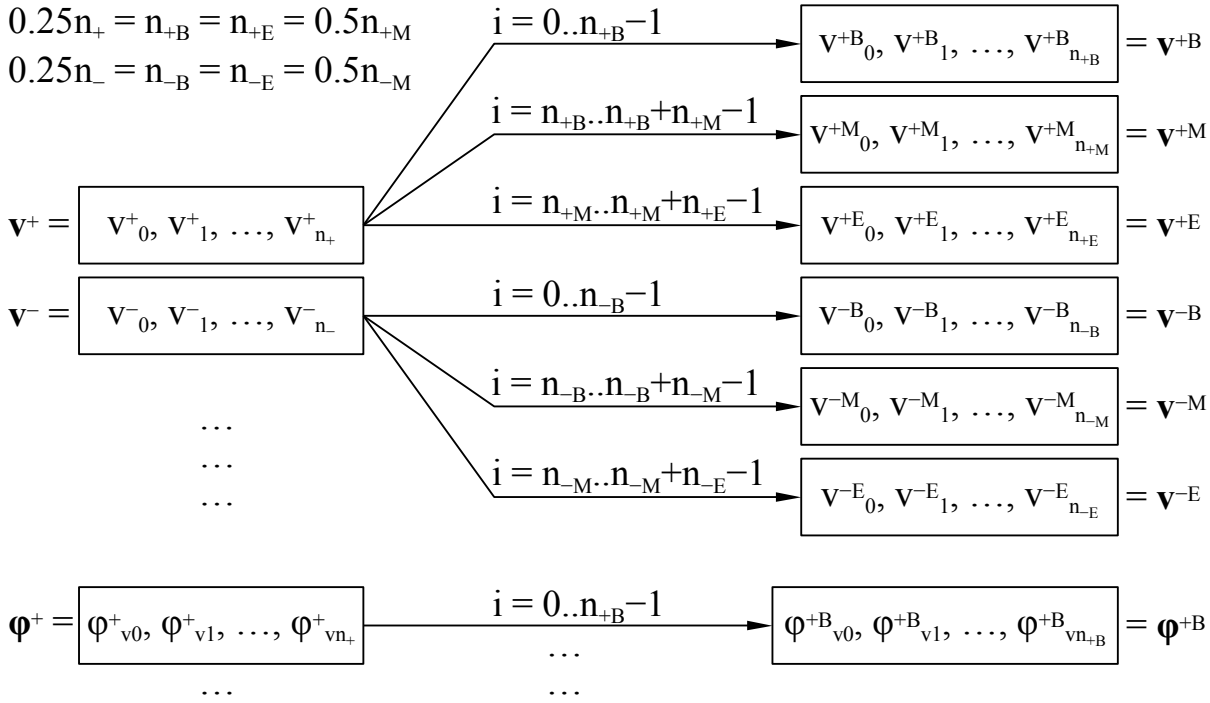

Figure 15 Division of negative and positive vectors of quantities into beginning, middle and end portions

For example, taking differences ${}_pdx = {}_{p+1}x - {}_px$ and ${}_pdt = {}_{p+1}t - {}_pt$, values of quantity v_x (x-coordinate velocity) ${}_pv_x = \frac{{}_pdx}{{}_pdt}$ are computed.

There are many other quantities q_i computed. Their list, along with their definition, is provided in chapter 9.2.6.

2. the division of vectors of quantities into negative, zero and positive parts
   $\vec{q}_i$ is split into negative $\vec{q}_i^-$, zero $\vec{q}_i^0$ and positive $\vec{q}_i^+$ parts (see figure 14). This is in order to help further compose features (see chapter 9.3).

3. the futher division of each part into beginning, middle and end portions
   The sub-vectors $\vec{q}_i^{-B}$, $\vec{q}_i^{-M}$, $\vec{q}_i^{-E}$, $\vec{q}_i^{+B}$, $\vec{q}_i^{+M}$ and $\vec{q}_i^{+E}$ are extracted as the beginning,

the middle and the end portion of complete $\vec{q}_i^{-}$ and $\vec{q}_i^{+}$. The beginning portion contain the first 25 % of vector items, the middle portion contains the next 50 % of vector items and the end portion contains the last 25 % of vector items. The process is displayed in figure 15.

The intention of using of four vectors (complete, beginning, middle and end vectors) for each quantity is to analyze the start, the middle and the end of the stroke separately. The idea is, for example, that at the beginning of the stroke the mouse must be accelerated, at the middle the acceleration should vary around zero, and at the stroke's end the movement must be decelerated. These changes in acceleration should appear in computed acceleration vectors. Splitting complete vectors into the beginning, the middle and end portions should help reveal these changes.

4. the computation of statistical properties of all vectors
   For each of these eight vectors $\vec{q}_i^{-}$, $\vec{q}_i^{-B}$, $\vec{q}_i^{-M}$, $\vec{q}_i^{-E}$, $\vec{q}_i^{+}$, $\vec{q}_i^{+B}$, $\vec{q}_i^{+M}$ and $\vec{q}_i^{+E}$ the following statistical properties are computed: the minimum ${}_0$, the arithmetic average ${}_2$, the maximum ${}_4$, the spread ${}_S$ (${}_S = {}_4 - {}_0$) and the deviation ${}_D$.

   Due to the fact that $q_{i0}^{-B} = q_{i0}^{-}$, $q_{i0}^{+B} = q_{i0}^{+}$, $q_{i4}^{-E} = q_{i4}^{-}$ and $q_{i4}^{+E} = q_{i4}^{+}$, the values q_{i0}^{-B}, q_{i0}^{+B}, q_{i4}^{-E} and q_{i4}^{-E} are dropped because they are duplicates.

   Together, there are 18 statistical properties for each quantity q_i used.

9.2.6 Quantities used for markers

This chapter contains a list of all utilized quantities, including the expressions used to compute them. Take notice of the following:

- indexes of vectors' components (like ${}_p$ or ${}_{p+1}$) are written to the left to the symbol so that indexes and quantity specification are not mistaken (like e.g. ${}_t$ means tangential),
- numbered equations represent utilized quantities, whereas unnumbered equations show only the process of computation.

$${}_p x \qquad x\text{-coordinate} \tag{30}$$

$${}_p y \qquad y\text{-coordinate} \tag{31}$$

$${}_p dx = {}_{p+1}x - {}_p x$$

$${}_p dy = {}_{p+1}y - {}_p y$$

$${}_p dt = {}_{p+1}t - {}_p t$$

$${}_p ds = \sqrt{{}_p dx^2 + {}_p dy^2} \qquad \text{length of smoothed stroke segment}$$

$$ {}_pv_x = \frac{{}_pdx}{{}_pdt} $$

$$ {}_pv_y = \frac{{}_pdy}{{}_pdt} $$

$$ {}_pv = \sqrt{{}_pv_x^2 + {}_pv_y^2} \quad \text{velocity} \quad (32) $$

$$ {}_pdv = \frac{{}_{p+1}v - {}_pv}{{}_pdt} \quad \text{time difference in velocity} \quad (33) $$

$$ {}_pd^2v = \frac{{}_{p+1}dv - {}_pdv}{{}_pdt} \quad \text{second time difference in velocity} \quad (34) $$

$$ {}_p\phi_{vxy} = \text{atan}({}_pv_x, {}_pv_y) \quad \text{(codomain of atan is } [0, 2\pi]) $$

$$ {}_p\phi_v = {}_{p+1}\phi_{vxy} - {}_p\phi_{vxy} \quad \text{velocity angle along the stroke} \quad (35) $$

$$ {}_pv_n = \sin {}_p\phi_v \quad \text{normal velocity} \quad (36) $$

$$ {}_pc_s = \frac{{}_{p+1}\phi_v - {}_p\phi_v}{{}_pds} \quad \text{curvature} \quad (37) $$

$$ {}_pdc_s = \frac{{}_{p+1}c_s - {}_pc_s}{{}_pds} \quad \text{difference in curvature} \quad (38) $$

$$ {}_p\omega = \frac{{}_p\phi_v}{{}_pdt} \quad \text{angular velocity} \quad (39) $$

$$ {}_pd\omega = \frac{{}_{p+1}\omega - {}_p\omega}{{}_pdt} \quad \text{difference in angular velocity} \quad (40) $$

$$ {}_pa_x = \frac{{}_pv_x}{{}_pdt} $$

$$ {}_pa_y = \frac{{}_pv_y}{{}_pdt} $$

$$ {}_pa = \sqrt{{}_pa_x^2 + {}_pa_y^2} \quad \text{acceleration} \quad (41) $$

$$ {}_p\phi_{axy} = \text{atan}({}_pa_x, {}_pa_y) \quad \text{(codomain of atan is } [0, 2\pi]) $$

$$ {}_p\phi_a = {}_{p+1}\phi_{axy} - {}_p\phi_{axy} \quad \text{acceleration angle along path} \quad (42) $$

$$ {}_pa_n = \sin {}_p\phi_a \quad \text{normal acceleration} \quad (43) $$

$$ {}_pc_a = \frac{{}_{p+1}\phi_a - {}_p\phi_a}{{}_pds} \quad \text{acceleration curvature} \quad (44) $$

$$ dx_d = {}_{\text{last}}x - {}_0x \quad \text{difference in } x \text{ of the last and the first stroke point} $$

$$ dy_d = {}_{\text{last}}y - {}_0y \quad \text{difference in } y \text{ of the last and the first stroke point} $$

$$s_d = \sqrt{dx_d^2 + dy_d^2} \quad \text{distance from the stroke begin to its end} \quad (45)$$

$$s_i = \sum_p {}_p ds \quad \text{integrated stroke length} \quad (46)$$

$$t_i = \sum_p {}_p dt \quad \text{integrated stroke duration} \quad (47)$$

$$c_{si} = \sum_p {}_p c_s \quad \text{integrated curvature} \quad (48)$$

$$v_d = \frac{s_d}{t_i} \quad \text{direct velocity} \quad (49)$$

$${}_p s_{ri} = \frac{1}{s_i} \sum_1^p {}_p ds \quad \text{ratio of developing of } s_i \quad (50)$$

$$r = 1 - \frac{s_d}{s_i} \quad \text{inverted straightness} \quad (51)$$

$${}_{p^o} ds^o = \sqrt{{}_{p^o} dx^{o2} + {}_{p^o} dy^{o2}} \quad \text{unsmoothed stroke segment length}$$

$$s_i^o = \sum_{p^o} {}_{p^o} ds^o \quad \text{integrated length of unsmoothed stroke}$$

$$j = \frac{s_i^o}{s_i} \quad \text{jitter, ratio of original to smoothed length} \quad (52)$$

In total there are 22 quantities computed for each stroke, which together with statistical properties computed gives 396 markers for each single stroke.

9.2.7 Summary and discussion of strokes and markers

Strokes are representations of single mouse movements that correspond to a single person's actions. Strokes are built from input stream containing a sequence of x and y coordinates of mouse-like device position and time instants t. The input stream is continuous whereas strokes are sequences clipped out of the stream.

The input stream is noisy and likely the primary source of the noise is insufficient sampling frequency. To overcome this, input data is smoothed using smoothing spline and then re-sampled to obtain twice as many samples.

The stroke's re-sampled data is passed to engine which computes 22 vectors of derived quantities (like curvature, velocity and so). Then, these vectors are statistically processed and 18 statistical markers are calculated for each of the 22 vectors of quantities. For example, from a vector, that contains curvature at each

point of the stroke, the minimum, the average, the maximum, the spread and the deviation of the curvature is computed.

There are many computed markers (396), compared with the minimum length of the stroke which is 5 input events. Computing many markers cannot add information to the strokes, therefore many markers are redundant. This redundancy is reduced during the feature selection (see chapter 10), where no more than 15 of the most significant features (built from markers) for each person is selected.

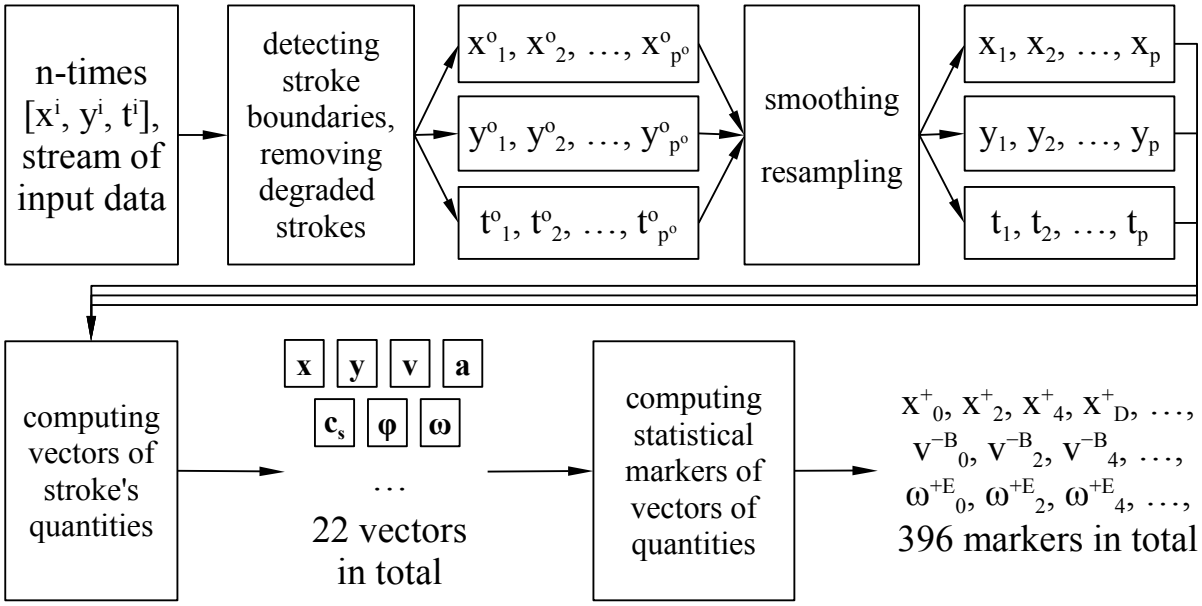

Figure 16 An overview of stroke processing, from input stream to markers

In short, the input data stream is split into strokes which are represented by statistical properties (i.e. markers) of their quantities. The complete process flow is reviewed in figure 16.

9.3 Features

Statistical markers computed for each stroke (that represents a single movement action) describe a given person, however, it is still not enough. The main reason for this is that single stroke does not account for all speeds, directions, strengths and sudden movements the person has—larger statistics are needed.

In [10] and [4], an approach has been developed whereby markers of many individual strokes are replaced with an estimated probability distribution. This approach reduces hundreds or thousands of individuals *markers* to a few numbers that parametrize chosen random variables called a *feature*; the theory for the reduction is given in chapter 4.3 and 6.4.3.

9.3.1 Acquiring more statistics with utilizing the whole data sets

As was described in chapter 8.3, six data source sets exist for each user that is involved in experiments. These data sets typically contain more than 2000 strokes in each data set. This amount is divided into two halves: the first half is used to extract/train features, and the second half is used to select/tune the best features for each person. The first half is a *training* set and the second half is a *tuning* or *validation* set.

Utilizing the full training set gives an amount of data that is large enough to find an appropriate random variable.

Data sets are split into training (^{T}) and validation (^{V}) parts (also see figure 7):

$$D_e^- \rightarrow D_e^{-T} + D_e^{-V} \qquad D_e^+ \rightarrow D_e^{+T} + D_e^{+V} \qquad D_e^e \rightarrow D_e^{eT} + D_e^{eT}$$
$$A_e^- \rightarrow A_e^{-T} + A_e^{-V} \qquad A_e^+ \rightarrow A_e^{+T} + A_e^{+V} \qquad A_e^e \rightarrow A_e^{eT} + A_e^{eV}$$

9.3.2 Determining probability distribution

There are three problems related to determining probability distributions:

Which probability distribution best matches with marker distribution?
: There are seven distributions available in the software model used in this dissertation. In [4], Weibull distribution is used for all markers because this distribution is able to emulate both exponential-type and Gaussian distributions. Data used in this dissertation shows that utilizing more different distributions is beneficial, because the observed data has only rarely clear Weibull distribution.

Data cannot fit into any available distribution.
: The solution to this problem is data transformation. Two techniques are used in this dissertation: splitting signs and logarithms.

Neither of the transformed data can fit into the available distributions.
: In this case the marker simply cannot be used because no random variable can be chosen and no feature can be derived from this marker.

Each of these problems will be discussed more thoroughly in the following text.

Which probability distribution best matches with marker distribution?

There are two ways of selecting the best-matching distribution: firstly, estimated distributions can be evaluated mathematically, and secondly, evaluation can be done using charts and eyes. This dissertation uses a combination of both ways.

All seven distributions available for describing the distribution of marker's values are described in chapter 4.2.1 in the theoretical part. These distributions are Gaussian, logistic, lognormal, inverse Gaussian, gamma, Weibull and Rayleigh.

In order to decide which distribution best describes the data, statistical tests are available. However, they were used only as a supporting criterion in this dissertation. The available distributions were firstly matched with data, secondly results of the matching were ordered according to Kolmogorov-Smirnov test [35] and at the last the best-matching distribution was chosen to be easy on the eye. If a priori knowledge of distribution had been available, it was also used (for example, the v should have Rayleigh distribution, because it is a vector sum of v_x and v_y).

The identification system developed for this dissertation is capable of estimating distribution parameters but it cannot visualize these distributions, and neither it can apply statistical tests to find the best one. For this purpose a standalone program called EasyFit [32] was used. Two examples of marker distributions in EasyFit are given in figures 17 and 18. The first figure shows the distribution of $d\omega_2^+$ (as in figure 22) with logistic distribution fitted. The second figure 18 shows the distribution of a_{nD}^{-B}, which does not fit any available distribution.

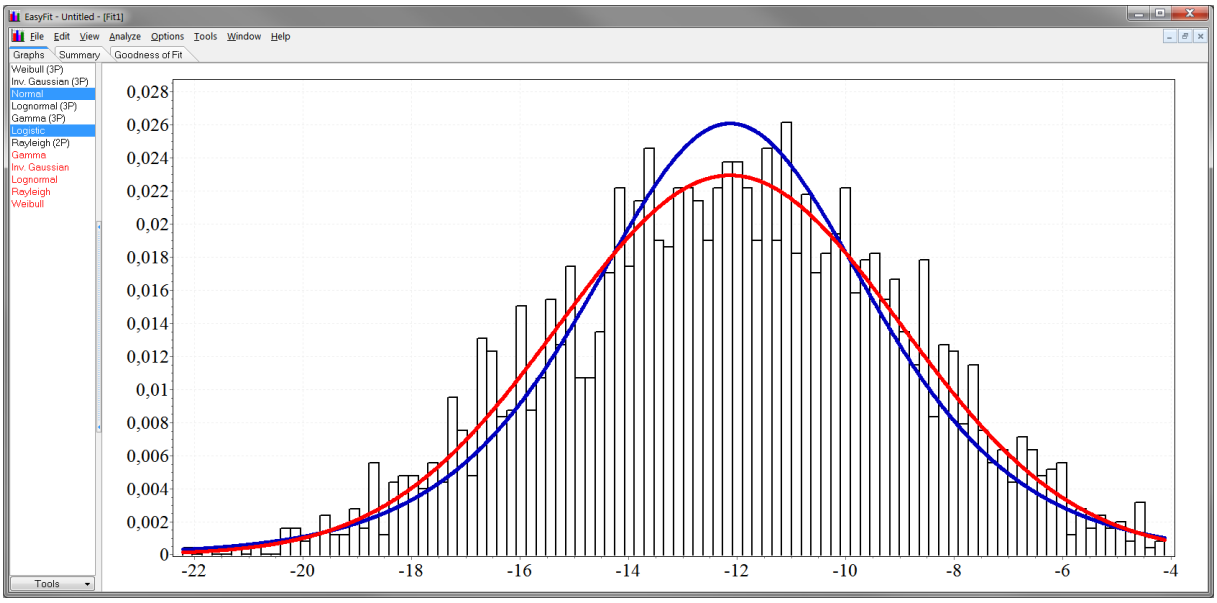

Figure 17 Histogram of $d\omega_2^+$ with estimated Gaussian and logistic distributions in EasyFit

Data cannot fit into any available distribution

The figures 19 and 20 show an example of histograms for two marker distributions, which cannot fit any distribution. The first histogram of $d\omega_0$ contains only negative values, which means it cannot fit any exponential distribution. Logistic distribution displayed in the chart is the best matching distribution of available Gaussian and logistic distributions.

The second histogram of $d\omega_2$ shows steep symmetrical distribution. Due to the steepness, the data can hardly match any distribution. As for the first case, the best matching distribution is also logistic.

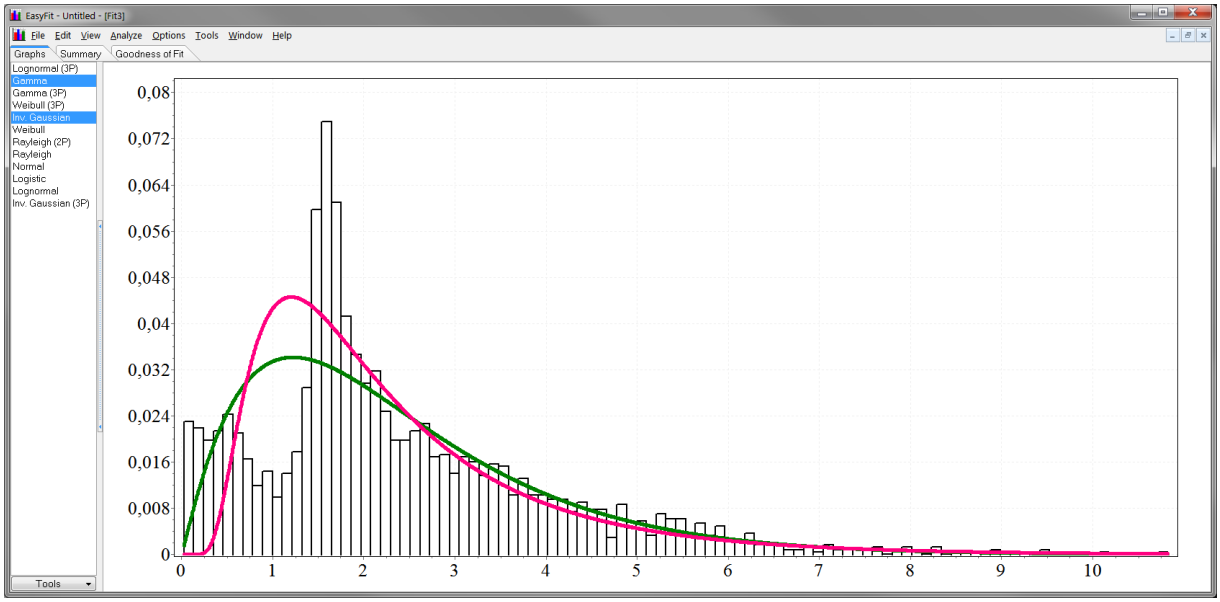

Figure 18 Histogram of a_{nD}^{-B} with estimated gamma and inverse Gaussian distributions in EasyFit

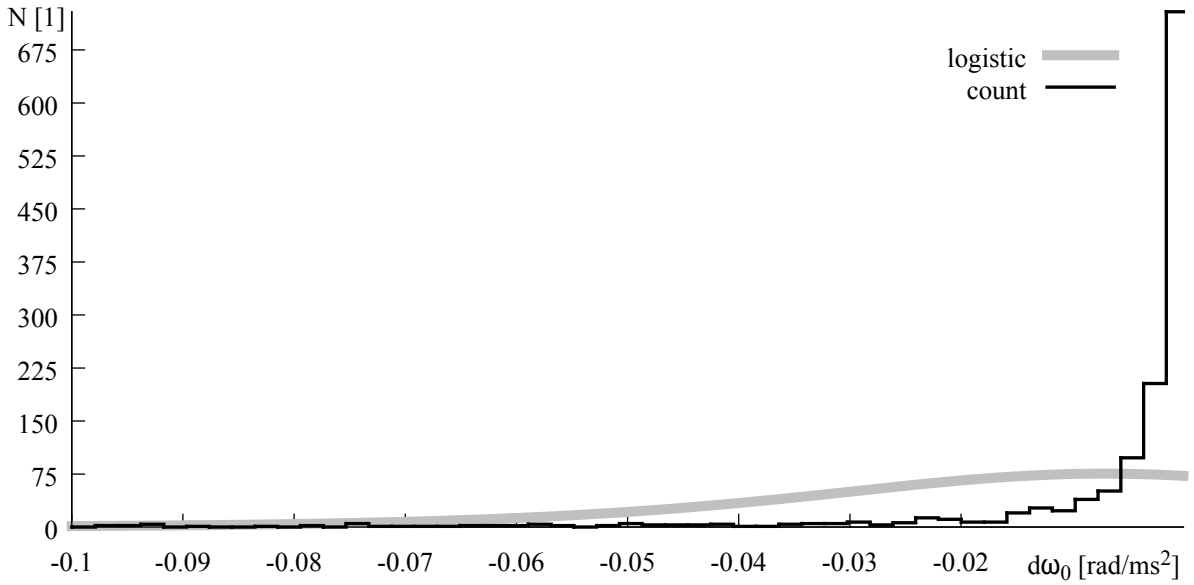

Figure 19 Histogram of $d\omega_0$ marker values with unusable best-fit distribution

Two ways of transforming data were tested. The first attempt used a simple subtraction of minimum (according to [4]), but the results were not satisfying. It moved data to positive values, but peaks (as for $d\omega_2$ in figure 20) or increasing exponentials (as for $d\omega_0$ in figure 19) were not resolved using this method.

The second attempt of transforming data was more complex and consisted of two steps. This attempt was also chosen to be used in this dissertation:

- In the first step, values of a particular marker are split into negative and positive parts. The process is explained in point 2 of chapter 9.2.5. The negative parts

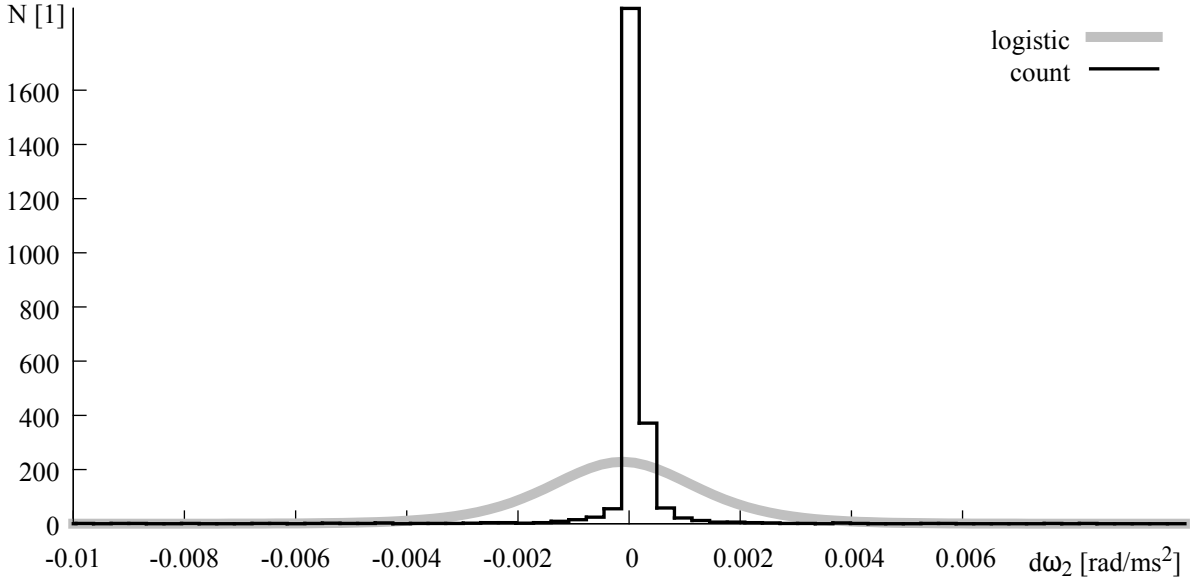

Figure 20 Histogram of $d\omega_2$ marker values with unusable best-fit distribution

are negated, which gives positive values. For instance, with respect to figure 20, $d\vec{\omega}_2$ is split and negated to $-d\vec{\omega}_2^-$ and to $d\vec{\omega}_2^+$.

The transformation allows separate analysis of negative and positive values, as well as separate matching with the negative and positive values of markers. The transformation also enables the use of bounded, non-negative distributions like lognormal, inverse Gaussian and so on. Separating positive and negative parts is applied to all markers with symmetrical distributions.

- In the second step, positive values resulting from the first transformation allows the use of logarithms. Logarithmic transformation decreases the value range and converts exponential-like distributions to other distributions. As a result, exponential-like distributions (which all look similar close to zero) are re-computed to distributions where differences can be better distinguished. Logarithmic transformation is applied only selectively to markers where applying logarithms improves matching with the probability distribution.

To demonstrate how $d\omega_2$ changes after applying both transformations, i.e. split and logarithm, figures 21 and 22 were prepared.

Comparing figure 20 to figures 21 and 22 shows that the probability distribution fitted to the transformed data is more precise.

Neither of the transformed data can fit into the available distributions

In order to keep only relevant data, all markers with unsuitable distributions (like in figure 18) were discarded. This removal decreased the number of markers from 396 (see chapter 9.2.5) to 228, which consequently means that the number of used features is also 228 (see the following chapter 9.3.3).

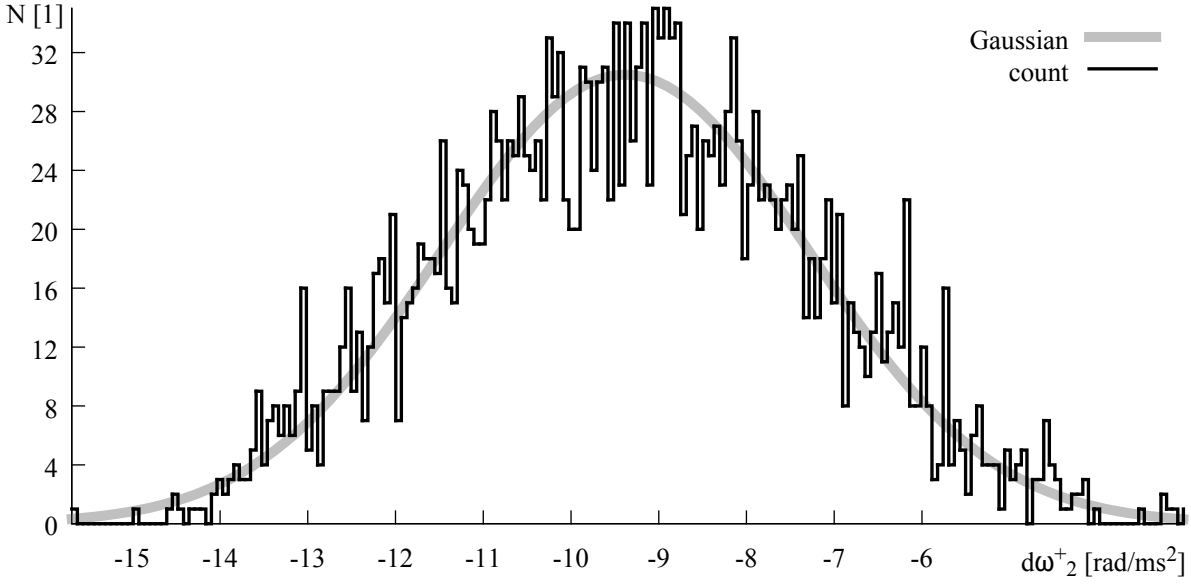

Figure 21 Histogram of $d\omega_2^-$ marker values with best-fit distribution

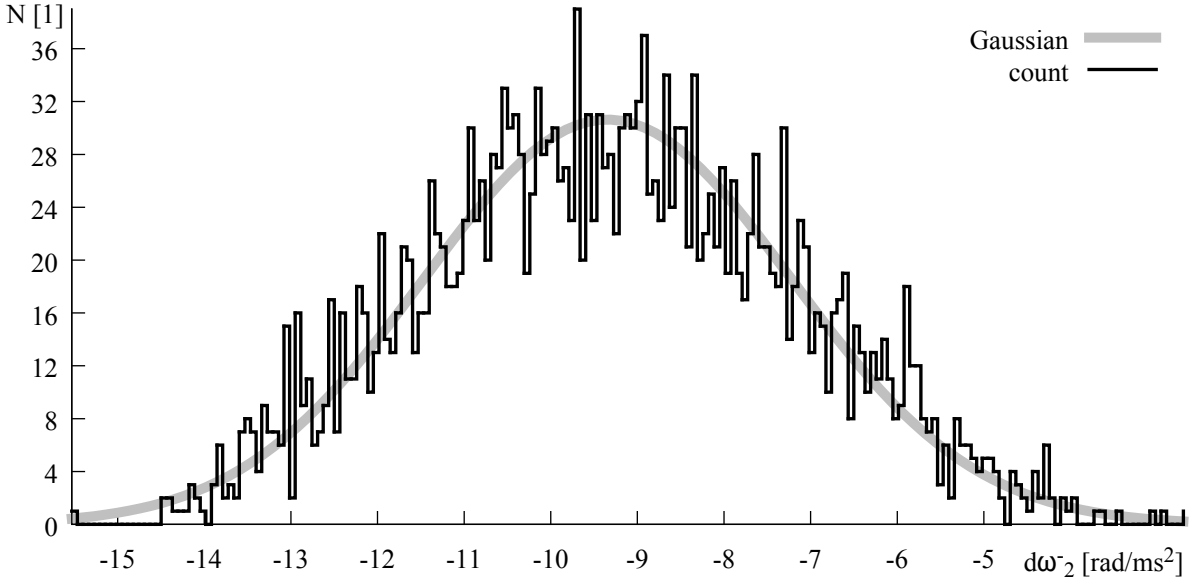

Figure 22 Histogram of $d\omega_2^+$ marker values with best-fit distribution

9.3.3 Overview of chosen features

The following overview lists all chosen features together with their corresponding best-fit probability distribution:

Gaussian s_{ri2}, s^M_{ri0}, s^M_{ri4}, s^M_{riD}, s^M_{riS}, s^E_{ri0}, t_i, s_d, v_d, r, x_2, x_S, y_2, y_4, y_S, c_{s0}, c_{s2}, c_{s4}, c^B_{s2}, c^B_{s4}, c^M_{s0}, c^M_{s2}, c^M_{s4}, c^E_{s0}, c^E_{s2}, dc_{s0}, dc_{s2}, dc_{s4}, dc^B_{s4}, dc^M_{s0}, dc^M_{s2}, dc^M_{s4}, dc^E_{s0}, dc^E_{s2}, v_0, v_2, v_4, v^B_2, v^B_4, v^M_0, v^M_2, v^M_4, v^E_2, dv^B_2, dv^B_4, dv^M_2, dv^E_0, dv^E_2, d^2v_0, d^2v_2, d^2v_4, $d^2v^B_2$, $d^2v^B_4$, $d^2v^M_0$, $d^2v^M_2$, $d^2v^M_4$, $d^2v^E_0$, ω_0, ω_2, ω_4, ω_S, ω^B_2, ω^B_4, ω^M_0, ω^M_4, ω^E_0, ω^E_2, $d\omega_0$, $d\omega_2$, $d\omega_4$, $d\omega^B_2$, $d\omega^M_0$, $d\omega^M_2$, $d\omega^M_4$, $d\omega^E_0$, $d\omega^E_2$, c_{a0}, c_{a2}, c_{a4}, c_{aS}, c^B_{a2}, c^B_{a4}, c^M_{a0}, c^M_{a2}, c^M_{a4}, c^E_{a0}, c^E_{a2}, a_0, a_2, a_4, a^B_2, a^B_4, a^M_0, a^M_2, a^M_4, a^E_0, a^E_2, ϕ_{a0}, ϕ_{a2}, ϕ_{a4}, ϕ_{aS}, ϕ^B_{a2}, ϕ^B_{a4}, ϕ^M_{a0}

logistic s_{ri0}, s^{M}_{ri2}, c_{si}, x_4, c_{sD}, c_{sS}, dc_{sS}, dc^{B}_{sS2}, v^{E}_{0}, dv_0, dv_2, dv_4, dv^{M}_{4}, d^2v_D, v_{n0}, v_{n2}, v_{n4}, v^{B}_{n2}, v^{B}_{n4}, v^{M}_{n0}, v^{M}_{n2}, v^{M}_{n4}, v^{E}_{n0}, v^{E}_{n2}, ϕ_{v0}, ϕ_{v2}, ϕ_{v4}, ϕ^{B}_{v2}, ϕ^{B}_{v4}, ϕ^{M}_{v0}, ϕ^{M}_{v2}, ϕ^{M}_{v4}, ϕ^{E}_{v0}, ϕ^{E}_{v2}, ω_D, $d\omega_D$, $d\omega_S$, c_{aD}, c^{M}_{aD}, a_{n0}, a_{n2}, a_{n4}, a^{B}_{n2}, a^{B}_{n4}, a^{M}_{n0}, a^{M}_{n2}, a^{M}_{n4}, a^{E}_{n0}, a^{E}_{n2}, a^{E}_{nD}, ϕ_{aD}, ϕ^{M}_{a2}, ϕ^{E}_{a0}, ϕ^{E}_{a2}

lognormal a_D

Weibull s_{riD}, s^{E}_{ri2}, s^{E}_{riS}, j, c^{B}_{sD}, c^{B}_{sS}, c^{M}_{sS}, c^{E}_{sS}, v^{B}_{D}, v^{E}_{D}, $d^2v^{B}_{D}$, $d^2v^{B}_{S}$, $d^2v^{E}_{D}$, v^{B}_{nS}, v^{M}_{nD}, v^{M}_{nS}, v^{E}_{nD}, v^{E}_{nS}, ϕ^{B}_{vD}, ϕ^{B}_{vS}, ϕ^{M}_{vD}, ϕ^{M}_{vS}, ϕ^{E}_{vD}, ϕ^{E}_{vS}, ω^{B}_{D}, ω^{B}_{S}, ω^{M}_{S}, ω^{E}_{D}, ω^{E}_{S}, a^{E}_{S}, a^{M}_{nS}, ϕ^{M}_{aS}

Gamma s^{B}_{ri2}, s^{B}_{ri4}, s^{B}_{riD}, s^{B}_{riS}, s^{E}_{riD}, c^{E}_{sD}, v_D, v_S, v^{B}_{S}, v^{M}_{D}, v^{M}_{S}, v^{E}_{S}, dv^{E}_{S}, d^2v_S, a_S, a^{M}_{D}, a^{M}_{S}, a_{nD}, a_{nS}

Rayleigh c^{M}_{sD}, dc^{M}_{sD}, dc^{M}_{sS}, $d^2v^{M}_{D}$, v_{nD}, v_{nS}, v^{B}_{nD}, ϕ_{vD}, ϕ_{vS}, ω^{M}_{D}, $d\omega^{B}_{D}$, $d\omega^{M}_{S}$, $d\omega^{E}_{S}$, c^{M}_{aS}, a^{M}_{nD}, ϕ^{M}_{aD}, ϕ^{E}_{aD}

9.3.4 Discussion of features

Extracted features can be classified into groups according to their physical meaning. There are, for example, static features (distance or coordinates), features related to dynamics of the movement (velocity or acceleration) and features related to turning (normal velocity or angular velocity). Similarly, groups can also be made from related statistical properties. For example, the spread is related to deviation and the overall average value might be related to the average of the middle portion. Features in each group, no matter how the group is created, might be interchangeable.

It is also likely that particular individual features are more prominent for some entities than for others.

It would be beneficial to know all mentioned relationships and meta-information, but due to the fact that the inner structure of used features is unknown, these relationships are difficult to uncover. The latter feature selection phase is partially utilized to get at least some of this information. The topic is further discussed in the summary related to feature selection (see chapter 10.7).

Selecting and estimating probability distributions could be further improved in other ways: more distributions could be used or bigger statistics could be obtained from longer input. It could be also possible to replace the method of how features are modeled— for instance, using random variables could be replaced with using histograms or interpolating curves. All these possibilities are not explored in this dissertation and need further research.

9.4 Summary of feature extraction

Extracting strokes and markers from raw mouse data is a preliminary phase for both operational modes—training and running. It is also a prerequisite for experiments with the identification system as a whole. Raw input is scattered and contains only sparse information, but by feature extraction process, it is converted to markers and features that represent condensed and abstracted information.

To convert raw data in the described way, it must first be cleaned, sliced to pieces corresponding to single-movement units i.e. strokes, then strokes must be smoothed and finally quantities can be computed from the stroke's input events. Two algorithms of smoothing were tested, 2D Catmull-Clark subdivision and the smoothing spline. The smoothing spline was chosen mainly due to its ability to respect the stroke path on larger scale.

When the system runs in its *operational* mode (see chapter 5.1.2), the *markers* are the final product of the extraction phase. The markers are statistical properties of quantities computed from stroke's path points. After extracting the markers, each stroke is represented with 18 statistical properties of 22 computed quantities.

If the identification system runs in its *training* mode, the markers are not final but they are further processed to obtain *features*. The features are random variables (of selected distributions) which parameters are estimated from the distributions of the corresponding markers. In order to obtain good statistics for estimating, markers of a half of all grabbed strokes were used. Not all markers were successfully matched with a probability distribution, and such markers were discarded. The number of available features then decreased from 396 to 228.

10 FEATURE SELECTION

Features derived from chosen markers are many, unspecific, and it is not known how well they describe the corresponding person. It is also not known how much information content they carry.

Feature selection is a process that helps resolve both these uncertainties by finding a set of the most relevant and informative features [39]. The theory of this process is described in chapter 5.3.

10.1 General mechanism

The general mechanism of feature selection uses a selection algorithm (either SFS or SFFS, see chapter 5.3.2) which compares candidate sets by using selection metric (see chapter 5.3.3). In particular, for each entity the entire amount of features is pruned to obtain the set which best describes the entity. This has the following significance:

- each entity gets its own set of best-describing features, therefore
- different entities are described with different features.

The *trained* features created by the feature extraction utilizing *training* data sets are evaluated using samples randomly selected from *tuning* data sets (see figure 7).

In order to determine how relevant particular features are for each particular person, the tuning samples must contain data of all persons. Therefore, in the experiments related to selection algorithm (see chapter 10.2), always a mixture $\mathcal{S}$ of samples of all t entities is used for this purpose: $t-1$ samples are taken from the tuned entity, and single sample is taken from other $t-1$ entities. In total, the mixture $\mathcal{S}$ contains $2(t-1)$ samples. An overview of constructing $\mathcal{S}$ is shown in figure 23.

All three used metrics (SPP, EER/polylines and d_{EER}, see chapter 5.3.3) require multiple samples, what is in concordance with the previous paragraph. For each particular metric, suitability of $\mathcal{S}$ is computed as follows (see also figure 23):

SPP, single posterior probability, (12)
: Single s^e (12) is used to measure suitability, and effectively only t samples of $\mathcal{S}$ are used. The set $\mathcal{S}$ is considered better when s^e is closer to 1.

  Using SPP, the measure of suitability of $\mathcal{S}$ is computed from a single value:

$$\left|\mathcal{M}_{\mathrm{SPP}}\right| = 1 \tag{53}$$

EER/polylines
: Multiple s^e values of genuine samples and multiple s^f values of impostor samples are computed, see (12). s^e values are $t-1$ and they are used to build FNMR polyline, s^f values are $(t-1)^2$ and they are used to build FMR polyline.

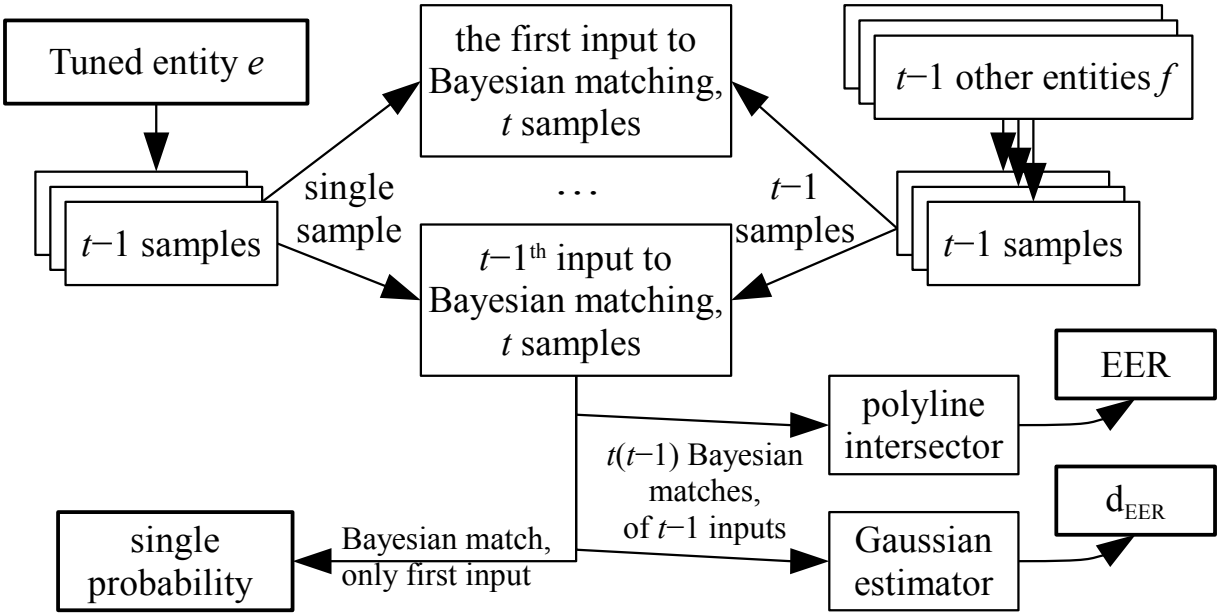

Figure 23 Constructing sets of tuning samples, computing selection metrics

Following this, the EER is computed geometrically as the intersection point of both FNMR and FMR polylines. $\mathcal{S}$ is considered better when the y-coordinate of the EER is closer to 0.

Using ERR/polylines, the measure of suitability of $\mathcal{S}$ is literally computed from four values, from the first and the second point of the intersecting segments of both polylines. Moreover, the position of the intersection point is indirectly determined by all points in the FMR and the FNMR polylines:

$$4 \leq \left|\mathcal{M}_{\mathrm{EER/p}}\right| \leq (t-1)^2 + (t-1) = t(t-1) \tag{54}$$

d_{EER}, (19)

Multiple s^e values of genuine samples and multiple s^f values of impostor samples are computed, see (12). s^e values are $t - 1$ and s^f values are $(t - 1)^2$. According to (18) and (19) d_{EER} is computed statistically.

The set $\mathcal{S}$ is considered better when d_{EER} is closer to 0.

Using d_{EER}, the measure of suitability of $\mathcal{S}$ is computed from $t(t - 1)$ values:

$$\left|\mathcal{M}_{\mathrm{dEER}}\right| = t(t-1) \tag{55}$$

Chapter 5.3.2 explains the criteria needed to stop the selection algorithm. The experiments described below use the following common settings for thresholds, s indexes steps (denote w = EER or $1 - s_e$ or d_{EER}):

absolute threshold

${}_s w \leq 1 \times 10^{-6}$,

relative threshold

$$|_{s}w - {}_{s-1}w| \leq |_{s-1}w - {}_{s-2}w| \leq 1 \times 10^{-6},$$

maximum number of searched features for SFFS

$$k = 15.$$

10.2 Experiments with feature selection algorithms

Feature selection can run two algorithms (SFS and SFFS) and it can use three variants of metrics. The success of selection also depends on the size of tuning samples. The experiments described in the following chapters were designed to explore all these aspects.

10.2.1 Comparison of SFS and SFFS

To compare the abilities of SFS and SFFS in finding the most relevant set of features, the experiment was carried out with the following setup:

- training sets A_e^{+T} were used to create entities and their features,
- tuning samples containing 20 consecutive strokes were selected once and were fixed for the entire operation of the experiment,
- the gap used in detection of strokes was 100 ms,
- using these samples, both algorithms were run with all available metrics for all entities; this, in total, gave 96 ($2 \times 3 \times 16$) runs, i.e. six for each entity,
- six runs of each entity were compared regarding their selected features to see how many features found by SFS and SFFS actually corresponded.

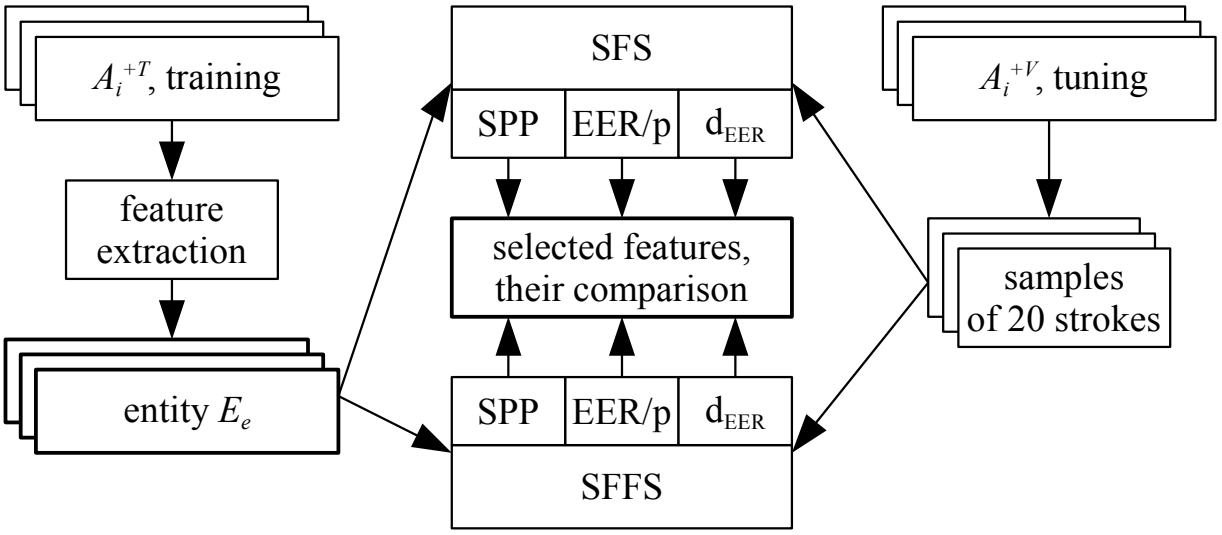

Figure 24 Scheme of experiment comparing SFS and SFFS

The scheme of the experiment is displayed in figure 24 and the results of the experiment are displayed in table 3. The features in each determined set are ordered according to decreasing contribution.

The table contains two more pieces of information:

- if SFS and SFFS found results that differ from each other, they are marked with a frame surrounding the feature sets,
- the feature sets for which the selection algorithm did not achieve absolute ending criterion (it roughly means that the selection algorithm did not find a good solution, see chapter 10.1) are preceded with a cross mark ×.

Table 3 Features found by SFS and SFFS for each entity using all metrics

e	metric	SFS features	SFFS features
1	SPP	ϕ^E_{a0}, s^E_{riD}, ϕ^B_{a2}, s^E_{riS}	ϕ^E_{a0}, s^E_{riD}, ϕ^B_{a2}, s^E_{riS}
	EER/p	$d^2v^B_4$, ϕ^M_{a0}, a^M_{nD}, v^B_2, ϕ^E_{a2}, ϕ^E_{aD}	$d^2v^B_4$, a^M_{nD}, v^B_2, ϕ^E_{a2}, ϕ^E_{aD}
	d_{EER}	d^2v_2, a^M_{nD}, ϕ_{aS}, ϕ^B_{a2}, s^E_{ri2}, ϕ_{aD}, ϕ^E_{a2}, ϕ^E_{aD}	a^M_{nD}, ϕ_{aD}, ϕ^E_{a2}, ϕ^E_{aD}
2	SPP	$d\omega^B_D$, ϕ_{a0}, c^E_{sD}	$d\omega^B_D$, ϕ_{a0}, c^E_{sD}
	EER/p	s^M_{ri2}	s^M_{ri2}
	d_{EER}	s^M_{ri2}	s^M_{ri2}
3	SPP	j	j
	EER/p	j	j
	d_{EER}	j	j
4	SPP	dv^B_2, c_{a4}, ϕ^E_{aD}, $d^2v^B_D$	c_{a4}, ϕ^E_{aD}, $d^2v^B_D$
	EER/p	v_d, v^E_0, v^M_{nD}, s^M_{ri4}, a^M_D	v^E_0, v^M_{nD}, s^M_{ri4}
	d_{EER}	c_{a4}, v^M_{nD}, v^E_0, $d\omega^B_D$	c_{a4}, v^M_{nD}, v^E_0, $d\omega^B_D$
5	SPP	ω_D, c^E_{sD}, v^B_2, c_{sD}, c^E_{sS}, a^M_{nS}, v^B_S, v^E_D	ω_D, c^E_{sD}, c_{sD}, c^E_{sS}, v^B_S, $d^2v^B_S$, ϕ_{a4}, ϕ^E_{aD}, a^B_4
	EER/p	a^B_4, a^E_{nD}, ϕ_{a4}, c_{sD}	a^B_4, a^E_{nD}, ϕ_{a4}, c_{sD}
	d_{EER}	a^E_{nD}, ϕ_{a4}, a^B_4, c_{sD}	a^E_{nD}, ϕ_{a4}, a^B_4, c_{sD}
6	SPP	v^B_{n4}, ϕ_{a4}, ϕ^E_{v2}	v^B_{n4}, ϕ_{a4}, ϕ^E_{v2}
	EER/p	v_0, ϕ_{aD}, ϕ^E_{aD}, d^2v_0	ϕ_{aD}, ϕ^E_{aD}, d^2v_0
	d_{EER}	$\times\, a^M_D$	$\times\, a^M_D$
7	SPP	$d^2v^B_D$, dc^B_{s2}, v^E_D	$d^2v^B_D$, dc^B_{s2}
	EER/p	$\times\, s^M_{ri0}$, a^B_{n4}, s^M_{ri2}, y_4	s^M_{ri0}, a^B_{n4}, s^M_{ri2}, y_4, y_S, a^M_0, ϕ_{a4}
	d_{EER}	$\times\, y_2$, $d\omega^B_D$, y_S, s^M_{ri2}	y_2, $d\omega^B_D$, y_S, a^B_{n4}, ϕ_{a4}

	SPP	a_0^M, c_{si}, s_{riD}	c_{si}, s_{riD}
8	EER/p	$d^2v_0^M$	$d^2v_0^M$
	d_{EER}	$d^2v_0^M$	$d^2v_0^M$
	SPP	c_{sD}^B, ϕ_{a4}, ω_D^B	c_{sD}^B, ϕ_{a4}, ω_D^B
9	EER/p	c_{sD}	c_{sD}
	d_{EER}	c_{sD}	c_{sD}
	SPP	dc_{s0}^E, s_{riD}, a_{nS}^M	dc_{s0}^E, s_{riD}, a_{nS}^M
10	EER/p	$\times$ ϕ_{aS}^M, s_{riD}^M, v_2^M	ϕ_{aS}^M, s_{riD}^M, ϕ_{aD}, x_4
	d_{EER}	$\times$ ϕ_{aD}^M	v_{nD}^M, ϕ_{aD}, x_4
	SPP	c_{a4}^M, a_D^M, $d^2v_D^M$, c_{sD}^E, s_{ri2}^E	c_{a4}^M, a_D^M, $d^2v_D^M$, c_{sD}^E, s_{ri2}^E
11	EER/p	c_{a4}^M, a_D^M	c_{a4}^M, a_D^M
	d_{EER}	c_{a4}^M, a_D^M	c_{a4}^M, a_D^M
	SPP	ϕ_{v4}^B, r, ω_D^E, $d\omega_D^E$, c_{aS}, $d\omega_D^B$	ϕ_{v4}^B, ω_D^E, $d\omega_D^E$, c_{aS}, $d\omega_D^B$
12	EER/p	$\times$ $d\omega_D^B$, $d^2v_D^M$	$d\omega_D^B$, $d^2v_D^M$, j, ϕ_{a4}, ω_D^E
	d_{EER}	$\times$ v_S^M, c_{sD}^E, c_{sD}^M, a_{nD}^E, a_D	$\times$ c_{sD}^E, , a_{nD}^E, s_{riS}^M, c_{a4}, $d\omega_D^B$
	SPP	ϕ_{v0}^M	ϕ_{v0}^M
13	EER/p	d^2v_0	d^2v_0
	d_{EER}	d^2v_0	d^2v_0
	SPP	c_{aD}, ϕ_{vD}^E, ϕ_{a4}^B	ϕ_{vD}^E, ϕ_{a4}^B
14	EER/p	v_{n4}, a_{n4}^M, $d\omega_D^B$	v_{n4}, a_{n4}^M, $d\omega_D^B$
	d_{EER}	a_{n4}^B, $d\omega_D^B$, ϕ_{vD}^E	a_{n4}^B, $d\omega_D^B$, ϕ_{vD}^E
	SPP	v_{nS}^B	v_{nS}^B
15	EER/p	v_{nS}^B	v_{nS}^B
	d_{EER}	v_{nS}^M	v_{nS}^M
	SPP	ϕ_{a2}^E, a_{nS}^M, ϕ_{v4}^M, $d\omega_D^B$	ϕ_{a2}^E, a_{nS}^M, ϕ_{v4}^M, $d\omega_D^B$
16	EER/p	a_{n2}^M, a_{n4}^M, ϕ_{a4}, a_0^M	a_{n2}^M, a_{n4}^M, ϕ_{a4}, a_0^M
	d_{EER}	a_{n2}^M, a_{n4}^M, ϕ_{a4}, a_{nS}^M, a_{n4}	a_{n2}^M, a_{n4}^M, ϕ_{a4}, a_{nS}^M, a_{n4}

These results contain 15 differences between SFS and SFFS, which is 31 % of the total. SFFS was also more successful with achieving the absolute criterion—5 out

of 7 (71 %) attempts were improved. The number of differences between SFS and SFFS also shows that features are sensitive to the nesting effect (see chapter 5.3.1).

There are evident differences in sets found by SPP criterion and found by two other criteria. This can be explained with the already mentioned number of values used to compute the criterion, see (53), (54) and (55). More points mean better separation between genuine and impostor distributions, and consequently both the criteria using more points have their results more similar.

Differences produced with SFS are not further analyzed because SFS is a single-point criterion, see (53), which cannot be used as a quality measure of the identification system (see chapter 5.2).

The overall results of the experiment are that:

- the used features manifest the nesting effect,
- SFFS outperforms SFS, mainly in terms of its ability to find better solutions and to overcome the nesting effect,
- SPP gives different results,
- EER/polylines gives similar results to d_{EER}, which validates (19). It also means that d_{EER} can replace the EER computed in a traditional way.

10.2.2 Comparison of the computational complexity of metrics

The computational complexity of feature selection algorithms is of medium importance. Feature selection does not run frequently, it needs to be run only when a new entity for a person is created. The process is time consuming: it takes around 4 hours on a single core computer to explore the space of a maximum 15 out of 200 features for 16 entities using SFFS.

The experiment intended for the comparison of the computational complexity of the metrics was set up and carried out in an identical fashion to previous experiment:

- training sets A_e^{+T} were used to create entities and their features,
- tuning samples containing 20 consecutive strokes were selected once and were fixed for all runs of the experiment,
- the gap used in detection of strokes was 100 ms,
- the SFS and SFFS were run with all available metrics for all entities, in total this was 96 ($2 \times 3 \times 16$) runs,
- the complexity was compared according to the number of *steps* the SFS or SFFS performed. One *step* represents a single matching a 20-stroke sample with all entities using the current feature set. All s_e and all s_f values (see chapter 10.1) are computed in this one step in the amount that the chosen criterion needs.

The results are displayed in table 4.

Table 4 The computational complexity of SFS and SFFS for all metrics

		SFS			SFFS		
entity	metric	steps	features	η	steps	features	η
	SPP	906	4	1.00	909	4	1.00
1	EER/p	1353	6	1.00	1365	5	0.99
	d_{EER}	1796	8	1.00	2049	4	0.44
	SPP	681	3	1.00	681	3	1.00
2	EER/p	228	1	1.00	228	1	1.00
	d_{EER}	228	1	1.00	228	1	1.00
	SPP	228	1	1.00	228	1	1.00
3	EER/p	228	1	1.00	228	1	1.00
	d_{EER}	228	1	1.00	228	1	1.00
	SPP	906	4	1.00	909	3	1.00
4	EER/p	1130	5	1.00	913	3	0.99
	d_{EER}	906	4	1.00	909	4	1.00
	SPP	1796	8	1.00	2763	9	0.73
5	EER/p	906	4	1.00	1135	4	0.80
	d_{EER}	906	4	1.00	909	4	1.00
	SPP	681	3	1.00	681	3	1.00
6	EER/p	906	4	1.00	909	3	1.00
	d_{EER}	×	×	0.00	×	×	0.00
	SPP	681	3	1.00	681	2	1.00
7	EER/p	×	×	0.00	1829	7	0.86
	d_{EER}	×	×	0.00	1365	5	0.83
	SPP	681	3	1.00	681	2	1.00
8	EER/p	228	1	1.00	228	1	1.00
	d_{EER}	228	1	1.00	228	1	1.00

	SPP	681	3	1.00	681	3	1.00
9	EER/p	228	1	1.00	228	1	1.00
	d_{EER}	228	1	1.00	228	1	1.00
	SPP	681	3	1.00	681	3	1.00
10	EER/p	×	×	0.00	1137	4	0.80
	d_{EER}	×	×	0.00	910	3	0.75
	SPP	1130	5	1.00	1141	5	0.99
11	EER/p	455	2	1.00	455	2	1.00
	d_{EER}	455	2	1.00	455	2	1.00
	SPP	1353	6	1.00	1593	5	0.71
12	EER/p	×	×	0.00	1374	5	0.82
	d_{EER}	×	×	0.00	×	×	0.00
	SPP	228	1	1.00	228	1	1.00
13	EER/p	228	1	1.00	228	1	1.00
	d_{EER}	228	1	1.00	228	1	1.00
	SPP	681	3	1.00	681	2	0.59
14	EER/p	681	3	1.00	681	3	1.00
	d_{EER}	681	3	1.00	681	3	1.00
	SPP	228	1	1.00	228	1	1.00
15	EER/p	228	1	1.00	228	1	1.00
	d_{EER}	228	1	1.00	228	1	1.00
	SPP	906	4	1.00	909	4	1.00
16	EER/p	906	4	1.00	909	4	1.00
	d_{EER}	1130	5	1.00	1137	5	0.99

η displayed in the last column of the table is computed as a ratio of the optimal *minimum* number of steps (needed for the particular resulting number of features) to *achieved* number of steps.

SFFS always requires at least the same number of steps as SFS (which is by design). SFFS is always worse than SFS, when features must be skipped in order to overcome the nesting effect.

Regarding the complexity of the algorithms, SFS is only capable of adding features and its speed is related to n^2, where n is the number of features. SFFS, on the other hand, selects from all possible combinations of features, and its speed relates to 2^n. Regarding the metrics, SPP is always faster because it needs t (where t is the number of entities) matching samples with entities, but EER/polylines and d_{EER} require $t(t-1)$ matching operations.

10.2.3 Discussion of the feature selection algorithms

Both experiments that compared SFS to SFFS, including all three available metrics, revealed that both algorithms, to some extent, were capable of selecting descriptive features for each entity. The comparison of the algorithms is as follows:

- The extracted features are sensitive to the nesting effect (5.3.1) which cannot be resolved by SFS. Of the result sets computed, 31 % were better computed with SFFS.
- SFFS has substantially bigger time complexity. For SFS the time complexity is $O_{\mathrm{SFS}}(n^2)$ and for SFFS it is $O_{\mathrm{SFFS}}(2^n)$, where n is the number of features.
- Because SFFS is an extension of SFS, SFFS can find the solution in the same time as SFS when the nesting effect does not happen.

Three compared metrics, SPP, EER/polylines and d_{EER} behave in this way:

- EER/polylines and d_{EER} use all points of FNMR and FMR so they drive the selection algorithm better than SPP. The difference is visible in the resulting sets of features: SPP sets differ from sets found with EER/polylines and/or d_{EER}. Due to less used points, feature sets obtained with SPP are more likely to be improper.
- The number of used points also affects the time complexity. The complexity of SPP and d_{EER} is $O_{\mathrm{SPP}}(t)$, while the complexity of the EER is $O_{\mathrm{EER}}(t^2)$.

There is a high number of nesting effect occurrences (approximately one third), and this rules out SFS. Also, because usage of SPP is likely to lead to improper sets, SPP is not good metric for selecting utilized mouse-related features. As a result, the most suitable combination for this dissertation's feature selection is the SFFS together with ERR/polylines or d_{EER}.

The overall time complexity of the SFFS/EER variant is $O(2^n \cdot t^2)$, and the overall time complexity of the SFFS/d_{EER} is $O(2^n \cdot t)$. The features part (2^n) is quite manageable because the number of features is pre-defined and it does not increase during the operational phase. The entities part for the EER (t^2) may cause problems when the number of entities increases (the entity number equals the number of people). The feature selection can run offline, so time need not be an issue, however online usage, or usage for a large number of entities (approximately more than 100) would be difficult. In such cases, feature selection would require a completely

different algorithm, for example stochastic (like [50]). The other way is not to use the ERR and replace it with d_{EER}, as it is done in this dissertation.

More about the performance of the EER and d_{EER} can be found in [49].

10.3 Exploration of selection stability

The feature selection process, which searches for the most descriptive features, is driven by random tuning samples containing a pre-defined number of strokes m. Random sampling poses one question and choosing m poses two more questions:

1. Are feature sets found using different samples the same or not? The question is analyzed in experiments 10.3.1, 10.3.3, 10.3.2 and 10.3.4.
2. Is feature set found for some m_i the same as feature set found for different m_j, and if not, then how much do they differ? An analysis of this problem is given in experiments 10.3.5.
3. How does the system that is tuned for samples having m_t strokes accept samples containing more or less strokes m_s? Two experiments relate to this question: experiment 10.5.1 and 10.5.2. Both these three experiments are explained in chapter 10.5.

Answering these questions allows better decisions about:

- The usability of the identification model used in this dissertation (see chapter 7.2).
- Optimal m. The optimality criterion may, for example, be the number of selected features, or the EER.
- The relationship between m_s and m_t. Should be $m_t = m_s$, or less, or bigger?

10.3.1 Repeatability of selection (all features used)

The following experiment was designed and carried out to determine how tuning samples affect the selection process:

- training sets A_e^{+T} were used to create entities and their features,
- tuning samples containing 40 consecutive strokes were randomly chosen in each of the five runs of feature selection,
- the stroke detector delimited strokes with the gap of 64-ms duration,
- the configuration for the selection process was SFFS/d_{EER},
- the selection process selected from all 228 features available,
- five runs of feature selection for all of 16 entities produced 80 feature sets in total,
- the content of feature sets for each entity was compared—the experiment produced 5 different attempts for each entity.

Table 5 The comparison of feature sets size, 64-ms gap, all features used

entity	run 1	∩	run 2	∩	run 3	∩	run 4	∩	run 5	∩ of all
1	×1	1	×2	0	×1	0	×4	0	×1	0
2	6	4	×8	2	×6	4	×11	4	×5	1
3	4	1	5	2	6	0	×2	0	3	0
4	4	1	×5	0	×5	0	×6	0	×13	0
5	×13	0	3	0	×6	0	5	0	4	0
6	×12	1	5	2	6	3	9	2	×6	0
7	3	1	3	1	3	0	×2	0	3	0
8	3	1	4	0	3	0	2	1	3	0
9	4	0	2	0	3	1	7	1	5	0
10	4	1	5	0	×6	1	3	1	×5	0
11	×6	1	×8	2	×10	1	×6	0	×5	0
12	4	1	4	2	3	1	5	0	×7	0
13	5	1	7	1	3	1	2	0	3	0
14	×7	4	×4	1	×13	1	×10	0	6	0
15	1	0	1	0	1	1	1	0	1	0
16	1	0	1	1	2	2	2	1	2	0

The results are displayed in table 5. If the selection algorithm did not find the optimal set (the absolute ending criterion was not satisfied, see chapter 10.1) a × mark is put into the cell.

The table 5 contains the numbers of features selected in each run. To measure the similarity of sets, the number of shared features is computed between set 1 and 2, and also between 2 and 3, between 3 and 4 and also between 4 and 5, and then placed into a column between corresponding runs. In the last column there is the count of features shared among all five runs.

Results clearly show that repeating feature selection gives results that are unrelated. Each feature selection run constructs its own feature set that has minimal overlap with other sets. The feature set is properly chosen, but it works only for the given tuning samples; the feature set depends on tuning samples.

The result, in principle, means that the used feature extraction and feature selection processes could not satisfy requirements to the identification system, and consequently that the identification system cannot be constructed this way. There is the possibility though, that unrelated feature sets can still be related due to similarities in features. This surmise will be explored in further experiments.

Why do not feature sets correspond? Here are some hypotheses:

- H1. Input data is too irregular or undersampled (see chapter 9.2.3). Smoothing that is applied helps remove artifacts but cannot add any information.

  To overcome this problem, a lot more statistics, different cleaning, smoothing or re-sampling procedures might help.

- H2. The pieces of input data sliced into strokes are too short. The problem might be that the gap (64 ms) used to delimit the strokes is too short.

  To overcome the problem, another setup of the selection algorithm would help.

- H3. Features modeled as random variables may be inadequate or incorrectly chosen. There might be other quantities that would better describe movements. Also, features created from the beginning, the middle and the end portions might just produce artifacts and noise, than bring relevant information.

  To overcome this problem, deep analysis of quantities and their relationships would help. Better statistics would be also beneficial.

- H4. There may be too many features. This fact is partially linked to previous hypothesis, but there is one more aspect: some features might be internally statistically linked, so statistically they describe the same quantity.

  To overcome this problem, the internal structure of features/markers needs to be explored and understood.

Regarding the first hypothesis (H1), [4] uses a different smoothing algorithm. In [4], before the smoothing spline is applied, the stroke's input events are geometrically interpolated to make points of equal unit distance. The interpolation was not explored in this dissertation.

Regarding the second hypothesis (H2), strokes of [4] are longer and they finish after the user goes through a whole predetermined path. Here, there is no ending gap. If strokes were lengthened by accepting longer gaps, the approach used in this dissertation would be closer to the one in [4]. This possibility is explored in the experiment 10.3.4.

Regarding the third hypothesis (H3), results of the discussed experiment 10.3.1 contradict [4]. Features used in [4] and in this dissertation are similar, the majority of features was adopted. A significant difference may be the usage of beginning,

middle and end portions. Removing features of these portions is explored and discussed in the experiment 10.3.3.

Regarding the fourth hypothesis (H4), the possible linkage in statistical properties could be taken from the results of the entity 7 (see table 5). Here are all five feature sets found by the experiment for this entity:

$$v^E_{n2}, \omega^M_4, \omega_2 \quad v_{n4}, v^E_{n2}, a_{nS} \quad a_{nD}, d\omega^E_0, v^E_{n2} \quad v_{n2}, v^B_{n2} \quad d\omega^M_0, v_{n4}, a^M_{nS}$$

Features a_{nS} and a_{nD} might be related, because spread (see chapter 9.2.5) describes distance from the minimum to the maximum, as also the deviation similarly does. Let us remember that feature is a random variable, so the relation between spread and deviation can be seen as similarity in probability distribution in both markers. The relation between a_{nS} and a^M_{nS} resembles the previous situation, and maybe more clearly. The first feature is the spread of all a_n values. The second feature is the spread of the middle part of a_n values. If the distribution is symmetrical, which might be a valid presumption for a_n, the features should have related distributions. Removing possibly linked features is explored and discussed in the experiment 10.3.2.

10.3.2 Repeatability of selection (similar features omitted)

This experiment is a restricted modification of the previous experiment 10.3.1. The following features were removed from the previous experiment and were not used to describe entities:

- All spread features q_{iS}, q^B_{iS}, q^M_{iS} and q^E_{iS}, because they represent similar properties as deviations q_{iD}, q^B_{iD}, q^M_{iD} and q^E_{iD}.
- Features q^M_{i0} and q^M_{i4}, because they are similar to q^B_{i4} and q^E_{i0}.
- Features q^M_{i2} and q^M_{iD}, because they are similar to q_{i2} and q_{iD}.

The number of features used in this experiment decreased to 136 after this reduction. This amount of features N_{FL} is called a *limited number of features* or shortly *limited features* in the rest of the dissertation text.

The experiment was run with the following parameters:

- training sets A^{+T}_e were used to create entities and their features,
- tuning samples containing 40 consecutive strokes were randomly chosen in each of the five runs of feature selection,
- strokes were ended if no movement occurred for at least 64 ms,
- the configuration for the selection process was SFFS/d_{EER},
- five runs of feature selection for all entities produced 80 feature sets in total,
- the content of feature sets for each entity was compared—the experiment produced 5 different attempts for each entity.

Results are displayed in table 6. If the selection algorithm did not find the optimal set i.e. the absolute ending criterion was not satisfied (see chapter 10.1), a × mark was put into the cell.

Table 6 The comparison of feature sets size, 64-ms gap, limited features

entity	run 1	∩	run 2	∩	run 3	∩	run 4	∩	run 5	∩ of all
1	×3	0	×1	0	×1	0	×2	0	6	0
2	×2	0	×3	0	×1	0	×1	0	×3	0
3	×5	0	×1	0	6	2	6	1	×1	0
4	7	0	×7	1	10	0	6	0	×11	0
5	4	0	×1	0	5	2	4	0	×4	0
6	5	1	×3	0	×2	0	×6	0	×4	0
7	6	0	5	0	×4	1	7	1	5	0
8	4	1	6	1	4	2	7	1	3	0
9	3	0	4	0	3	0	4	0	3	0
10	5	0	4	1	×4	0	6	2	9	0
11	×5	1	×2	0	×10	1	×6	1	×9	0
12	3	1	4	0	×5	0	5	1	4	0
13	8	1	5	1	4	0	6	0	5	0
14	×9	1	×3	1	×6	1	6	0	×5	0
15	1	0	1	1	1	0	1	0	1	0
16	1	0	2	1	2	1	2	1	1	0

The results in this table are comparable to the results of experiment 10.3.1. The conclusion is that omission of similar features did not improve the repeatability of feature selection and therefore the corresponding tested hypothesis H4 (see page 88) is falsified by the result.

10.3.3 Repeatability of selection (reduced features used)

This experiment is an even more restricted modification of the previous experiment 10.3.2. All features related to the beginning, the middle and the end portions were removed and not used to describe entities, in addition to all features removed in the previous experiment. These are: q^B_{i2}, q^B_{i4}, q^B_{iD}, q^M_{i0}, q^M_{i2}, q^M_{i4}, q^M_{iD}, q^E_{i0}, q^E_{i2}, q^E_{iD}.

The number of features used in this experiment decreased to 63 after this reduction. This amount of features N_{FR} is called a *reduced number of features* or shortly *reduced features* in the rest of the dissertation text.

The experiment was run with these parameters:

- training sets A_e^{+T} were used to create entities and their features,
- tuning samples containing 40 consecutive strokes were randomly chosen in each of the five runs of feature selection,
- strokes were ended when no movement occurred for at least 64 ms,
- the configuration for the selection process was SFFS/d_{EER},
- five runs of feature selection for all entities produced 80 feature sets in total,
- the content of feature sets for each entity was compared—the experiment produced 5 different attempts for each entity.

Results are displayed in table 7. If the selection algorithm did not find the optimal set i.e. the absolute ending criterion was not satisfied (see chapter 10.1), a × mark was put into the cell.

The results in this table are comparable to results of experiments 10.3.1 and 10.3.2.The conclusion is that omission of features computed from beginning, middle and end portions of vector quantities did not improve the repeatability of feature selection, and therefore hypothesis H3 (see page 88) is not valid.

10.3.4 Repeatability of selection (long strokes used)

This experiment is identical to the experiment 10.3.3 except that the stroke detector was configured to use a 500-ms or 1000-ms gap. The experiment was run with these parameters:

- training sets A_e^{+T} were used to create entities and their features,
- tuning samples containing 40 consecutive strokes were randomly chosen in each of the five runs of feature selection,
- strokes were ended when no movement occurred for at least 500 ms or 1000 ms,
- the configuration for the selection process was SFFS/d_{EER},
- the selection process selected from $N_{FR} = 63$ features,
- five runs of feature selection for all 16 entities and each gap produced 160 feature sets in total,
- the content of feature sets for each entity was compared—the experiment produced 10 different attempts per entity sorted by their gap size into two groups.

The results are displayed in tables 8 (500-ms gap) and 9 (1000-ms gap). If the selection algorithm did not find the optimal set i.e. the absolute ending criterion was not satisfied (see chapter 10.1) a × mark was put into the cell.

Table 7 The comparison of feature sets size, 64-ms gap, reduced features

entity	run 1	∩	run 2	∩	run 3	∩	run 4	∩	run 5	∩ of all
1	×1	0	×1	0	×1	1	×2	1	×1	0
2	×1	0	×2	2	×5	0	×1	1	×1	0
3	×1	1	×3	2	7	1	×1	1	6	1
4	1	0	×4	0	×3	0	×4	1	×8	0
5	×6	2	×5	0	×1	0	4	3	×11	0
6	×6	0	×1	0	×6	0	×1	0	×2	0
7	4	3	6	2	4	1	3	1	7	1
8	×3	0	3	0	2	1	2	0	2	0
9	×4	0	3	0	5	2	×6	1	6	0
10	8	1	×3	1	5	1	×4	2	×5	1
11	7	2	5	2	3	1	×3	0	×2	0
12	5	2	3	1	4	2	×5	2	×5	0
13	×8	0	×5	0	×8	0	×4	1	5	0
14	9	1	×4	0	×1	0	×4	0	×6	0
15	1	0	1	0	1	0	2	1	2	0
16	2	1	2	1	2	1	3	1	2	1

The results in these tables are again comparable to all the related experiments 10.3.1, 10.3.2 and 10.3.3.

The first group of results of runs with 500-ms gap is similar to previous experiments. There is no visible improvement when the results are compared to the initial experiment 10.3.1. The second group of results of runs with 1000-ms gap is different. Table 9 contains many fewer zeros, and more runs ended with finding a good solution (which is indicated with no cross in the cell).

To further compare all five groups of repeated runs, the total size of all feature sets $f_a = \sum \text{run1} + \sum \text{run2} + \sum \text{run3} + \sum \text{run4} + \sum \text{run5}$, the total size of all intersections $f_i = \sum \text{run1} \cap \text{run2} + \ldots + \sum \text{run4} \cap \text{run5}$, total size of common intersection $f_c = \sum \cap$ of all and total count of selections ended without achieving absolute threshold $f_\times = \sum \times$ sign was computed for all runs and everything was put into table 10. This table also contains the number of used features N_F and two ratios that express relationships between sizes sums: f_i/f_a and f_c/f_a.

Table 8 The comparison of feature sets size, 500-ms gap, reduced features

entity	run 1	∩	run 2	∩	run 3	∩	run 4	∩	run 5	∩ of all
1	5	0	×1	0	×1	0	×4	1	×5	0
2	3	0	2	2	7	4	7	0	2	0
3	×4	3	×5	2	3	1	×3	2	×7	0
4	×1	0	×1	0	×1	0	3	0	×2	0
5	×5	1	×6	1	×7	1	4	2	6	1
6	×1	0	×1	0	×1	0	3	0	×6	0
7	2	1	3	0	3	1	2	0	4	0
8	3	2	3	0	2	0	2	0	3	0
9	6	1	5	0	3	1	4	2	5	0
10	×5	0	×4	0	×4	3	×5	2	×10	0
11	7	4	×10	1	×1	0	×4	0	×1	0
12	3	2	5	1	3	1	3	1	3	1
13	4	0	3	2	3	2	4	2	3	0
14	×3	2	×5	2	×6	2	×2	1	×6	0
15	1	1	1	1	1	0	1	0	1	0
16	×1	0	×1	0	×1	0	×1	0	×2	0

This table reveals the following:

1. The total number of selected features f_a decreases. The speed of this decrease does not correspond to the speed of decrease in the number of features N_F used in the particular experiment.

   The simplest interpretation is that the number of selected features likely depends on something other than the number of used features, though some small dependence is still visible. The last three lines use the same number of used features and their f_a is close.

2. The number of runs where a good solution was not found may be affected by two trends: the first trend makes good selection difficult as the number of features decreases. This trend corresponds to $f_\times = 31, 32, 44$ (38) for the number of features $N_F = 228, 136, 63$ (63).

Table 9 The comparison of feature sets size, 1000-ms gap, reduced features

entity	run 1	∩	run 2	∩	run 3	∩	run 4	∩	run 5	∩ of all
1	×4	1	4	3	6	3	4	0	4	0
2	3	1	2	0	4	1	2	1	2	0
3	×3	1	3	1	3	2	×6	3	×6	1
4	6	3	4	2	4	3	7	2	×2	1
5	5	1	3	0	4	3	4	3	6	0
6	×5	4	×6	5	×6	2	×2	0	×1	0
7	2	2	2	1	2	1	2	1	2	1
8	3	3	3	2	4	2	3	1	3	0
9	4	1	3	1	3	1	3	0	3	0
10	2	2	2	2	4	2	3	2	4	1
11	×4	2	×9	3	4	2	×6	4	7	1
12	2	0	2	0	2	1	5	1	2	0
13	2	1	2	2	2	1	2	0	2	0
14	3	1	4	1	3	1	3	1	5	1
15	1	1	1	1	1	1	1	1	1	1
16	×1	0	×1	0	×1	0	×1	0	×12	0

Table 10 General comparison of repeated feature selection

experiment	N_F	f_a	f_i	f_c	$\frac{f_i}{f_a}$ [%]	$\frac{f_c}{f_a}$ [%]	$f_\times$
10.3.1	288	371	58	1	16	0.27	31
10.3.2	136	340	31	0	9.1	0.72	32
10.3.3	63	296	51	4	17	1.4	44
10.3.4, 500-ms gap	63	275	58	2	21	0.72	38
10.3.4, 1000-ms gap	63	273	94	7	34	2.6	18

The second trend may be that prolongation of the ending gap makes selection easier and more runs end with finding good solutions. This trend could be visible in $f_\times$ = (44) 38, 18 for gaps of (64) 500 and 1000 milliseconds.

3. Runs using the longest gap have the biggest intersections. In both absolute and relative numbers, this is almost twice as good as runs using shorter gaps

(64 ms or 500 ms): e.g. 94 vs. 58, 34 % vs. 17 % or 2.6 % vs. 1.4 %. It means that selections, that used longer gap, found similar feature sets more frequently than in other cases.

4. The difference between 10.3.2 and 10.3.1 is small. Also 10.3.3 is close to 10.3.4, when f_a (and partially f_c) is taken into account.

The second and the third points are in concordance i.e. longer gaps more likely lead to finding good and repeatable solutions. This result also conforms to H2 (see page 88): [4] was able to achieve stable results when long strokes (longer than a second) were used, and this experiment 10.3.4 gives comparable results when the end gap is 1000 ms.

The outcome is important: short strokes, together with used feature extraction and selection, are not sufficient to achieve results similar to [4] (see also the discussion of results in the first related experiment 10.3.1), contrary to longer strokes whose results are closer to what [4] researched.

The second outcome of these experiments concerns the number of features and the gap length to use in further experiments. According to the fourth step above:

- the settings used in experiments with limited features can be omitted because the result 10.3.2 is close to the experiment 10.3.1,
- the settings used in experiments with reduced features (i.e. a maximally reduced number of features) must be preserved because they are different to 10.3.1,
- 10.3.4 points out various results for 1000-ms gap, it also shows that 500-ms gap do not differ at all from 64-ms gap (used in 10.3.1 and 10.3.3). Therefore, only 500-ms and 1000-ms gaps will be further used,
- summed up, further experiments will explore four variants: all and reduced features with gaps of 500 ms and 1000 ms lengths.

A relation between the gap duration and the length and of strokes (N and t) is plotted in figure 25. Values N and t are computed as arithmetic averages from all available strokes of all entities. The figure shows that for the gap within the 500 ms–1500 ms interval N increases more slowly than t. This observation could be interpreted as that the improvement in results having 1000-ms gap is rather linked to the duration of the stroke than to the number of stroke's items. On the other hand, the difference between N and t still looks too small to account for the observed improvement. This topic is not further analyzed in this dissertation.

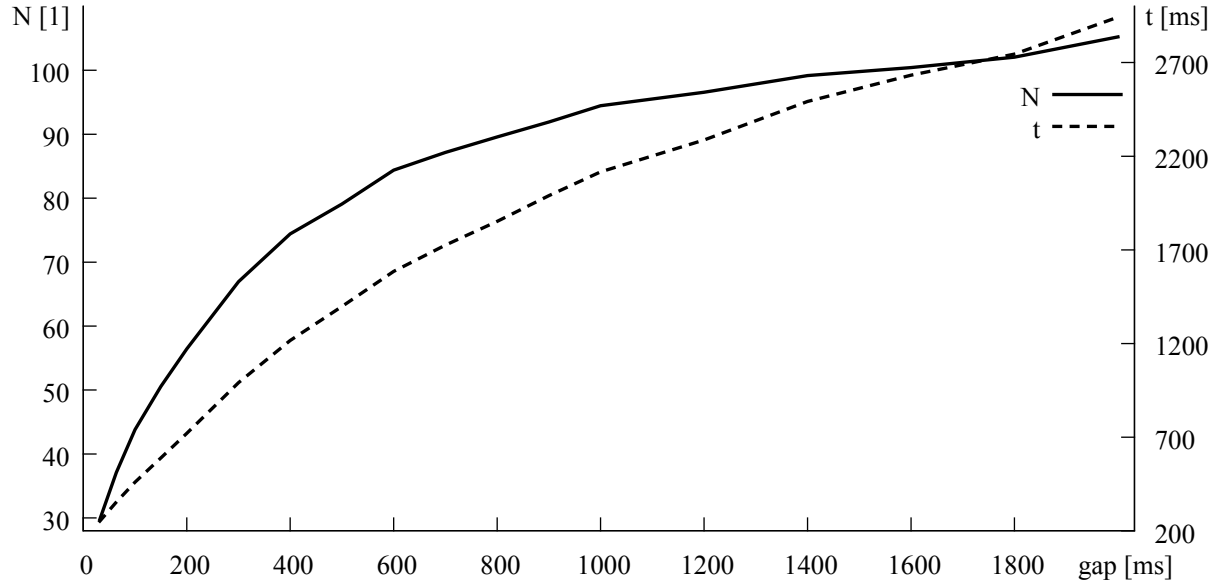

Figure 25 Dependence of the stroke length (items N and duration t) on the gap duration

10.3.5 The influence of the sample length on the selected features

This experiment was aimed at determining how the content of feature set develops when the number of strokes m_t changes in the tuning sample. The setup for the experiment was:

- training sets A_e^{+T} were used to create entities and their features,
- the experiment ran all 16 entities,
- tuning samples contained m_t = 1, 5, 10, 20, 30, 40, 50, 60, 70, 80, 90 and 100 consecutive strokes,
- the configuration for the selection process was SFFS/d_{EER},
- strokes were ended when no movement occurred for at least 500 ms or 1000 ms; these times were taken from the results of experiment 10.3.4,
- variants using all features (N_{FA} = 228) and reduced features (N_{FR} = 63) were tested,
- in total the experiment executed 1536 runs in groups of the gap (two gaps), entities (16 entities) and m_t (12 lengths of samples). Half the runs used random sample for each m_t and the other half used the same, not random sample that was gradually lengthened,
- from this amount of runs, results of entities 1 and 15 were chosen to present their results.

These results are displayed in eight pairs of figures, each pair contains the same entity. The left figure in the row contains results for samples that are not random, while the right figure contains results for samples that are random in each run. Columns of the figures represent features and rows correspond to m_t. If a particular feature was selected in a particular run, the box in the figure is painted in black.

Each figure contains a legend inside it: m is the number of strokes in the tuning sample (m_t), e is *entity*, 'p' means features of the beginning/the middle/the end *portions*, the letter 's' means *similar* features, 'r' means *random* and 'g' means a *short* ('s' = 500 ms = 0.5 s) or *long* ('l', 1000 ms = 1 s) *gap*.

The figures also contain information about the most frequent features:

- the column with the most frequent feature has a gray background (if the box is not already painted in black), and the feature is given a number,
- the same number of the same particular feature is displayed in all other figures that belong to the same entity.

If the column contains a number with × sign, it simply means that the corresponding number of columns was shrunk.

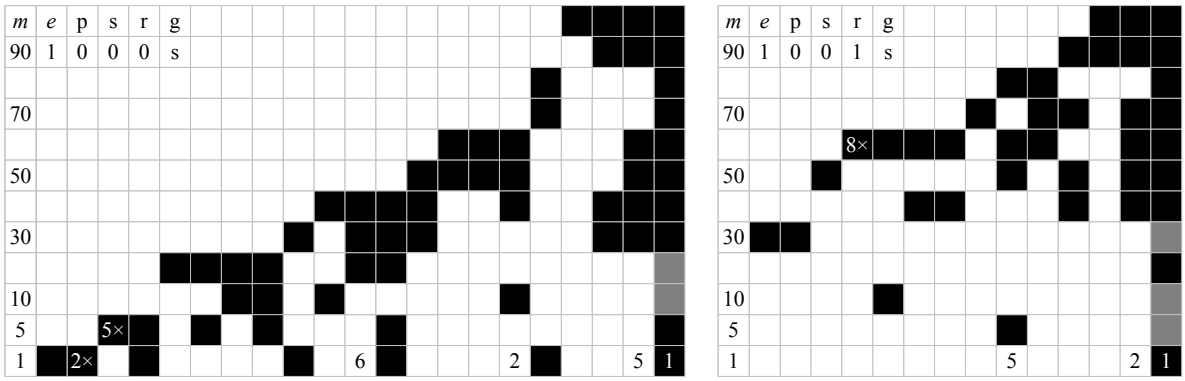

Figure 26 Dependence of the feature set on m, entity 1, reduced features, 0.5 s

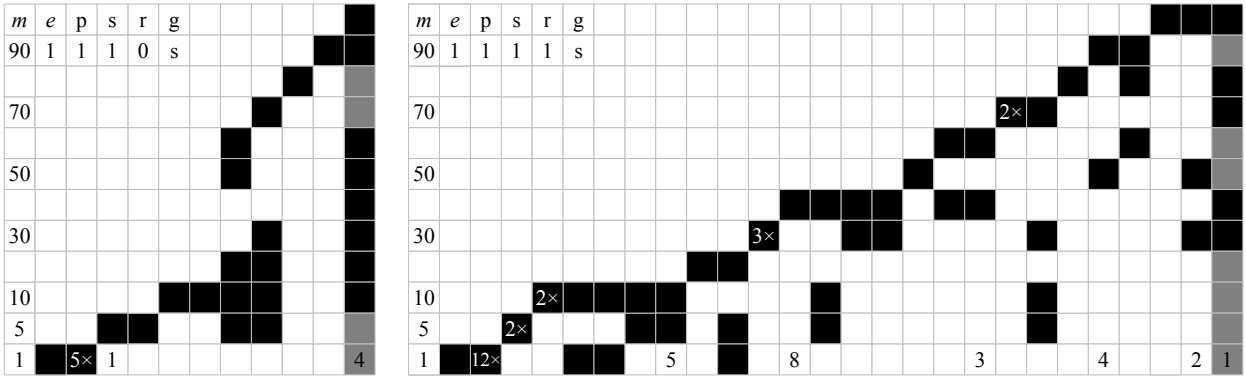

Figure 27 Dependence of the feature set on m, entity 1, all features, 0.5 s

The following findings can be found in the results shown:

- The number of features in a set continuously evolves and decreases as m_t increases. This trend is visible as the diagram increases in steepness.
- Somewhere between m_x = 10 and 30, the diagrams change. Below this m_x, feature sets usually contain many features, for example as shown in figure 31. Above this m_x, feature sets usually contain between one and four features.
- Random and non-random runs look similar and their results do not reveal any substantial differences. The same applies to utilizing all or reduced features.

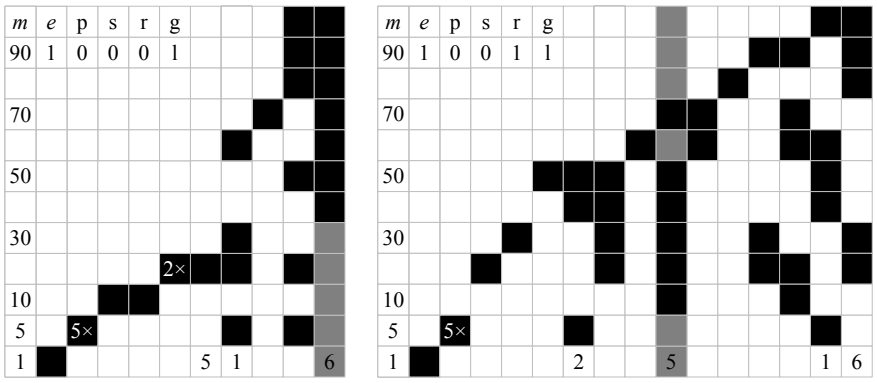

Figure 28 Dependence of the feature set on *m*, entity 1, reduced features, 1 s

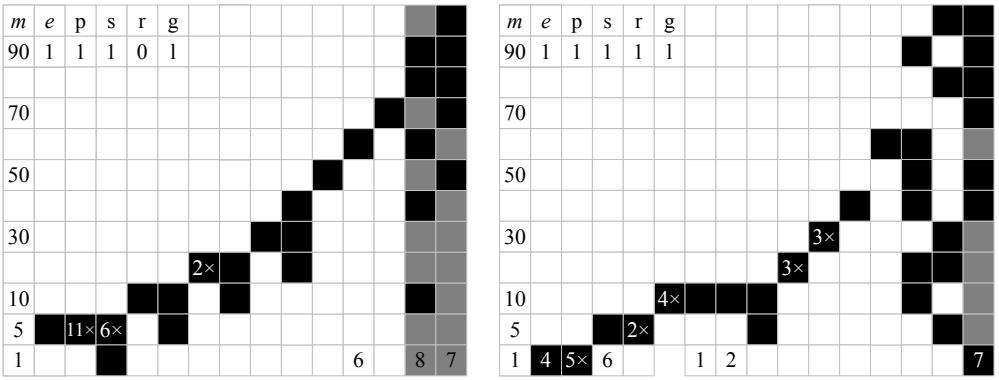

Figure 29 Dependence of the feature set on *m*, entity 1, all features, 1 s

- Different runs frequently produce the same most frequent feature.
- It is not clear whether the 500-ms gap or the 1000-ms gap is better. For entity 1, runs with a 500-ms gap (figures 26 and 27) found the feature 1 three times, whereas runs with a 1000-ms gap (figures 28 and 29) found the only common feature 7 only twice. For the entity 15, the results are the opposite: runs with shorter gap never picked out the same feature, whereas runs with longer gap succeeded in doing this in all four cases.
- In most cases $m_t = 60$ means stabilizing the feature set. When m_t is further increased, the feature selection only rarely replaces features in the set.

The conclusions of the findings are:

- $m_t < 20$ produces results with many different features. These results are not repeatable and samples of these lengths cannot be used in feature selection.
- $m_t > 50$ produces mostly repeatable results. The smallest value $m_t = 60$ is chosen in further experiments because it loads computing the least.
- The values $20 \leq m_t \leq 50$ sometimes works and sometimes does not. Whether the descriptive features are more prominent and more easily detectable depends on the entity. Due to this, values of m_t from the range $[20, 50]$ cannot be used for feature selection either.

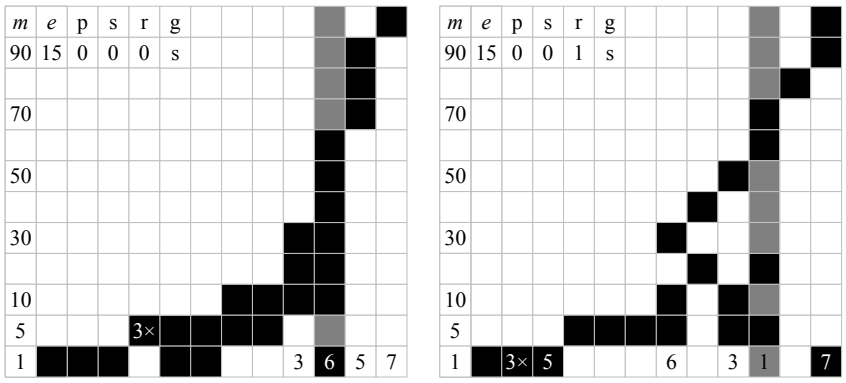

Figure 30 Dependence of the feature set on *m*, entity 15, reduced features, 0.5 s

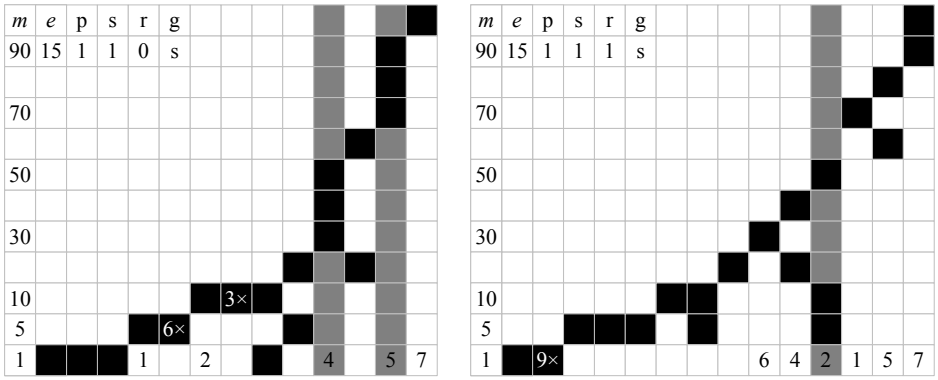

Figure 31 Dependence of the feature set on *m*, entity 15, all features, 0.5 s

- The difference between random and non-random samples is small. It means that random tuning works just as well as pre-determined tuning and thus that random tuning is capable of tuning up the identification system.
- The difference between runs utilizing all features and reduced features is also small. The result confirms previous experiments, as shown in the discussion related to the experiment 10.3.4.
- The experiment 10.3.5 partially supersedes experiments 10.3.1, 10.3.2, 10.3.3 and 10.3.4. It also confirms these experiments because this experiment also shows that repeated runs may end with selecting identical sets of features.

10.3.6 Discussion and summary of feature selection stability

Experiments that tested feature selection focused on two main problems: if the feature selection is able to repeatedly find informative and relevant features, and how the number of strokes in the sample affects the selected feature set.

Regarding repeatedly finding relevant features, experiments revealed:

- The number of statistical quantities computed for markers is of low importance. The experiment 10.3.1 using all available features produced similar results as experiments 10.3.2 and 10.3.3, which used reduced features. This finding was also confirmed with experiment 10.3.5.

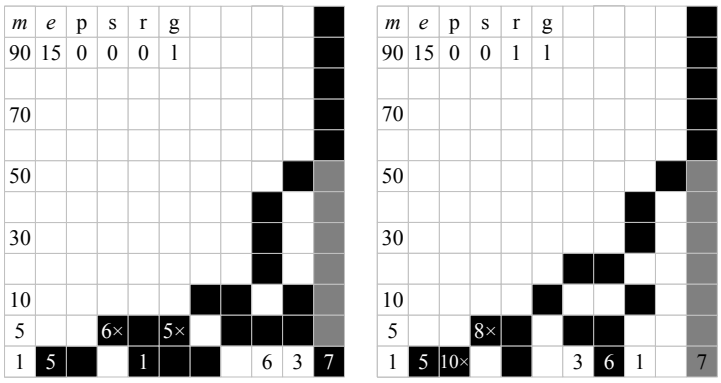

Figure 32 Dependence of the feature set on *m*, entity 15, reduced features, 1 s

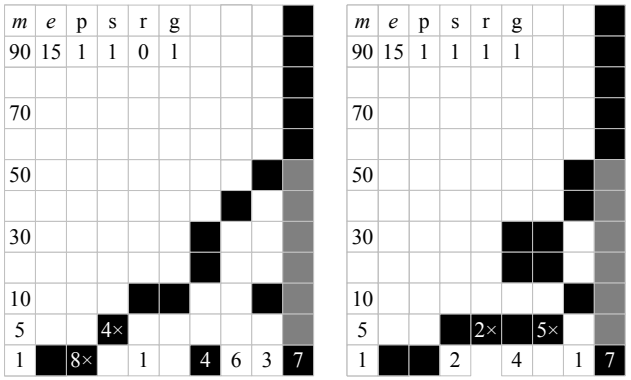

Figure 33 Dependence of the feature set on *m*, entity 15, all features, 1 s

The conclusion is that adding markers related to the beginning, the middle and the end portions of vector quantities did not help. This conclusion is in concordance with [4], where only basic markers were used and it worked.

- The length of the stroke is of big importance. Prolonging the ending gap greatly improved feature selection. The experiment 10.3.4 discovered this as fact, although it did not explain why.

  This outcome is also in concordance with [4]. According to this dissertation, feature selection gives good results when the average stroke duration is approximately two seconds. This duration is nearly the same as [4] discovered.

Regarding how the m_t affects the selected feature set, experiment 10.3.5 revealed:

- Single strokes cannot be used to select features. At least 20 strokes in a sample are needed when cases are favourable and at least 60 strokes are needed generally in order to make feature selection work.

  Taking into account the discovered average stroke duration (2 s), the absolute theoretical nett minimum time needed to build an entity is at least 2 minutes of tracking. This time is surprisingly short, but it is simply theoretical: data used in this dissertation shows that suitable strokes only appear a few times a minute when the person is instructed to continuously use the mouse. General

computer work produces suitable strokes with even less frequency so it may take considerably longer to grab enough data.

This outcome is in partial concordance with [4], because [4] succeeded when using samples containing only 10 strokes. A possible explanation could be that all [4]'s strokes are long (at least a second), whereas strokes in this dissertation are often shorter.

- Randomness of the tuning sample is not important, which is why the identification system can be successfully tuned with randomly selected samples.

The overall conclusion of experiments relating to feature selection is that the identification system used in this dissertation (including data grabbing, feature extraction and feature selection) is sufficiently capable of building entities representing identified persons.

10.4 The most often selected features

The feature selection selects features that are relevant and informative for the particular person. It means that each person is described with their own set of features that are likely not shared with other people. On the other hand, the movements needed to control the mouse-like device are similar for all people (because intentions to control something inside a GUI are similar) and the selected feature sets could therefore be more common.

The previous experiments ran a sufficient number of feature selections to prepare enough data to analyze the features that were selected. Runs with these particular parameters were chosen:

- training sets A_e^{+T} were used to create entities and their features,
- the experiment ran all 16 entities,
- tuning samples contained 60 consecutive strokes,
- the configuration for the selection process was SFFS/d_{EER},
- the gap was set to 1000 ms,
- in total 640 runs (40 runs for all 16 people) were used.

The results are displayed in table 11.

The contribution represents the average appearance of a feature of the particular quantity in the result of the single experiment run. This single experiment run carries out 16 individual feature selections, one selection for each individual entity.

Examples:

- the contribution 1.7 for j means that the marker is likely to appear once or twice in each run (one or two of 16 people is likely to use this marker),
- the contribution 0.73 for a_n means that 11 out of 15 features linked to a_n is likely to appear in each run.

Table 11 The contribution of features, frequency of selecting the features

quantity	s_{ri}	t_i	c_{si}	s_d	v_d	r	j	x
contribution	0.43	**0.85**	**1.4**	**1.2**	0.38	**1.5**	**1.7**	0.63
quantity	y	c_s	dc_s	v	dv	d^2v	v_n	ϕ_v
contribution	0.50	0.28	**0.73**	0.50	0.38	**0.68**	0.55	0.25
quantity	ω	$d\omega$	c_a	a	a_n	ϕ_a		
contribution	0.18	0.55	0.10	0.25	**0.73**	0.56		

Contributions larger than 0.65 (what is an average contribution) are printed in bold in the table. Quantities related to these contributions are considered typical. The evaluation of the results is:

- typical features represent either overall scalar quantities related to straightness (like integral curvature c_{si} or inverted straightness r) or vector quantities related to turning (like normal acceleration a_n or jerk d^2v),
- the opposite, i.e. less frequent quantities are curvatures and velocities (both angular and translational).

10.5 Validation of feature selection

Previous experiments revealed that feature selection can work successfully and can produce repeatable results. This chapter goes further. First of all, this chapter aims to answer the third question of chapter 10.3 (see page 86): how does the identification system, tuned for some sample length, work with samples of various lengths? Next, the proposed experiments should check the identification system in numerous runs with the goal of proving that previous findings are valid.

All proposed experiments measure the identification system as a whole. It is the first time this dissertation uses such an entire measurements whereas previous experiments were targeted on particular problems of the system. The metric used to measure the system is the EER (see chapter 5.2.4) because it is the general metric of first choice when measuring a quality of the identification system.

The common settings for all validation experiments is as follows:

- the length of the sample (the number of strokes in the sample) used to tune the system is 60 strokes (see chapter 10.3.6),
- the duration of the gap is 500 ms or 1000 ms,
- the length m_s of test samples changes from 1 to 200, and all lengths are tested 100 times; each sample is selected randomly,
- variants using all and reduced features are explored,
- the EER of the entire system is measured, as well as the EER of each entity.

10.5.1 Dependence of the EER on sample length

Two figures displaying development of the EER in dependence of m_s are presented, figure 34 shows the result for all features and figure 35 show the result for reduced featues. In both figures gray lines delimit $\pm\sigma$ of the EER value.

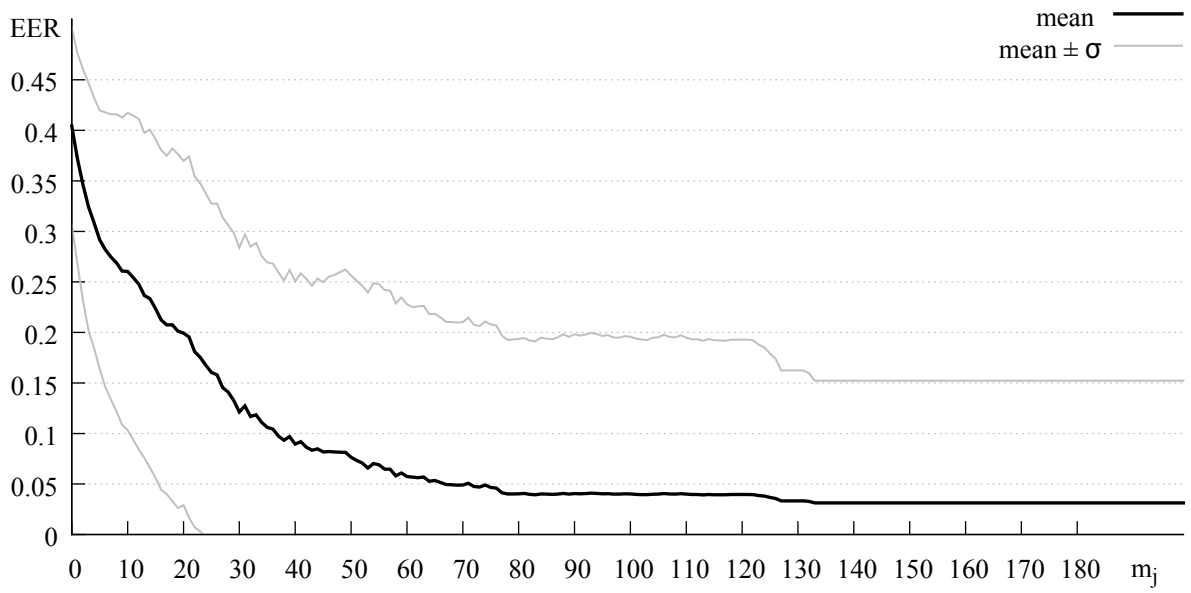

Figure 34 Development of the EER, all features, all entities

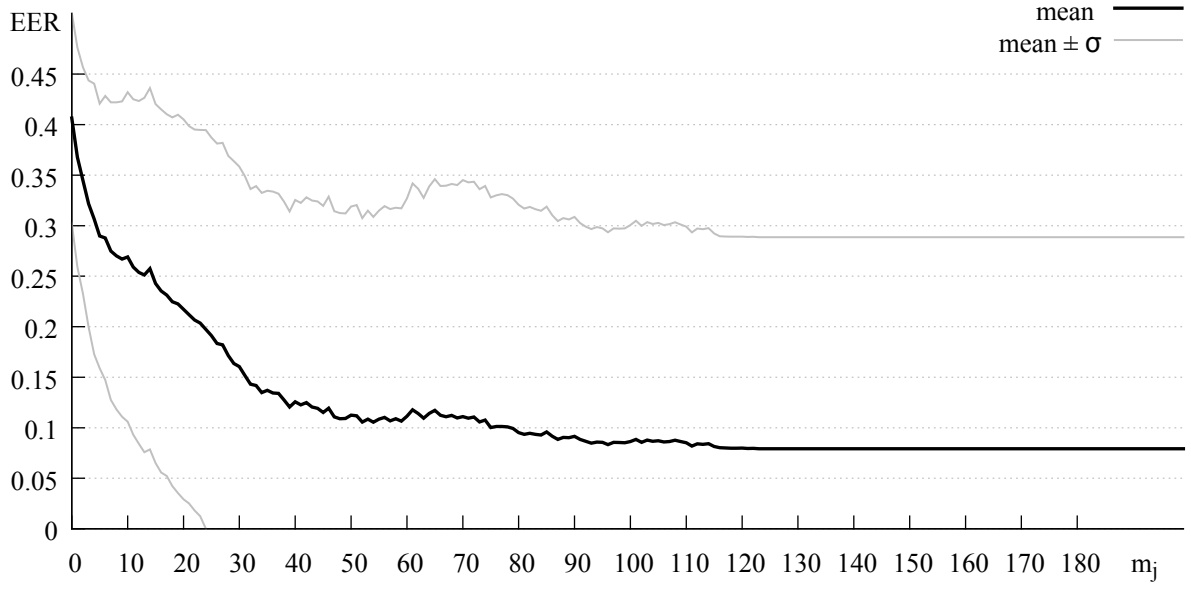

Figure 35 Development of the EER, reduced features, all entities

The difference between figure 34 and figure 35 is not in concordance with findings of previous experiments (see the discussion in chapter 10.3.6). It is visible that runs using all features (the figure 34) have results that are twice as good as runs using reduced features (the figure 35).

Two possible explanations for this observation are formulated in the following hypotheses:

- H5. The difference could be caused simply by the size of the set of features, and previous experiments did no reveal this fact.
- H6. The second explanation could be that the system failed to identify some entities, and the corresponding error was so big that it significantly altered the overall mean value (H6). This second hypothesis may be supported with a large σ visible in both figures—effectively the uncertainty of the computed EER is more than 300 % in both cases.

The experiment 10.5.2 was designed in order to realize which hypothesis is valid.

In order to demonstrate and compare the worst and the best entity of figure 34, the best entity 15 and the worst entity 16 are extracted to standalone figure 36. Note that the figure has a *y*-axis scale that is different to other figures in this experiment. The principal outcome of these results is that the identification system did not properly tune some entities. Previous experiments may have indicated this behavior (e.g. the experiment 10.3.4 for entity 16), especially in the number of runs that did not satisfy the absolute ending criterion.

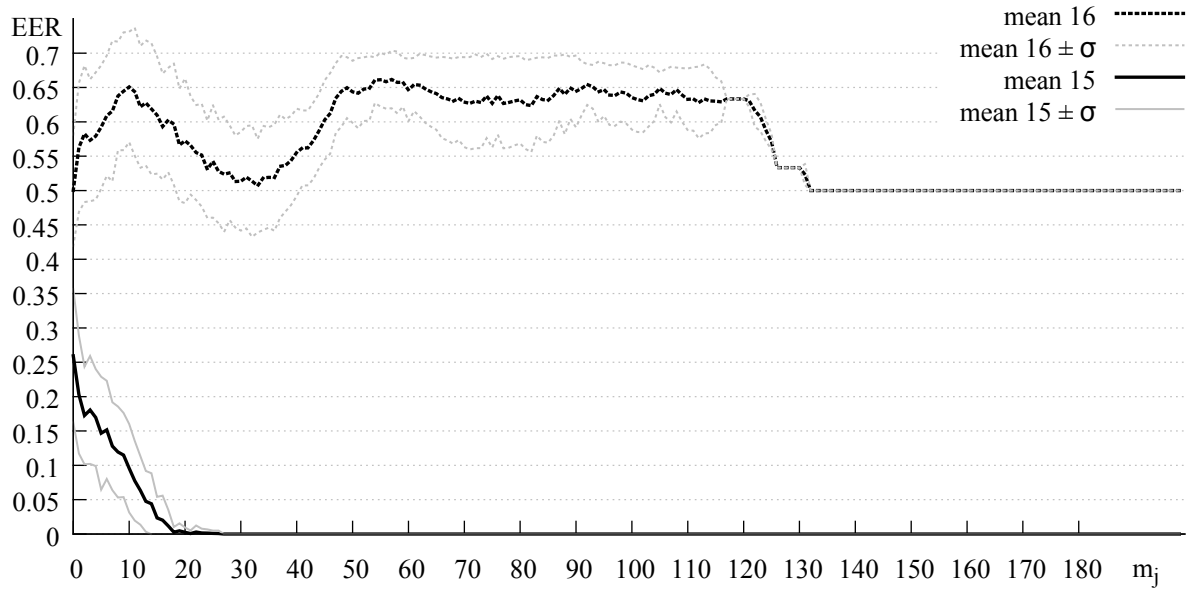

Figure 36 Development of the EER, all features, entities 15 and 16

Figure 35 also shows a small bump for m_s around 60–80. This could be explained as oscillation after exceeding the number of strokes (60) that the system has been tuned for.

The improvement of identification that appeared after prolonging the stroke-end gap has not been explained. Because it will not be further analyzed in this dissertation, the runs with shorter gaps were additionally carried out as the last insight into the problem. The result is displayed in figure 37.

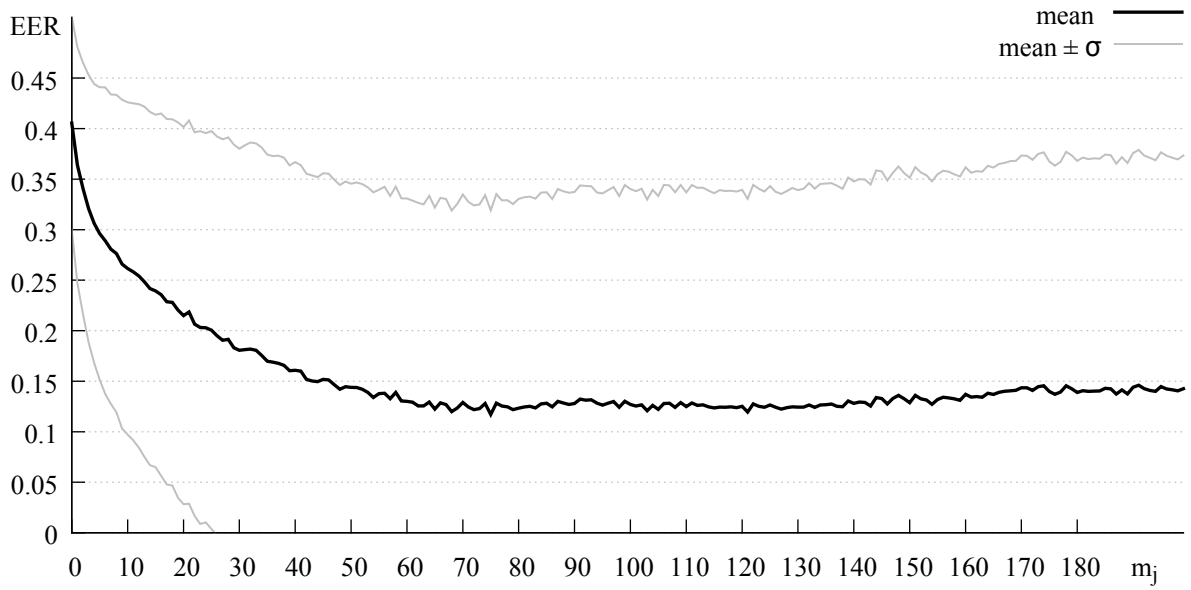

Figure 37 Development of the EER, all features, all entities, 500-ms gap

The comparison of the figure 37 with figures 34 and 35 reveals that if the system is tuned with shorter strokes, then it has worse performance. This supports findings of the previous experiments (e.g. 10.3.4). Also, the same trend of worsening the ERR for $m_s > m_t$ as in figure 35 (the bump after m_t), is visible in figure 37.

The overall conclusion of this experiment is that the identification system is able to build entities and use them to identify unknown samples, though the measured standard deviation of results is unacceptably high.

10.5.2 Dependence of the EER on sample length for selected entities

The experiment 10.5.1 revealed that the used identification system does not create suitable entities for all people, and consequently, that the system is not able to identify all people.

In order to learn more, an analysis was carried out using 25 independent tuning runs with various settings, grabbed from previous experiments. In total, 400 tuned entities were analyzed, and the analysis tracked if the ERR for particular entity decreased to 0. This result, i.e. the count N_0 of zero ERRs for all entities, is shown in table 12.

Table 12 The count of runs achieving the ERR = 0, all entities compared

e	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
N_0	3	13	3	0	8	2	16	20	20	11	4	16	25	3	25	0
×		×					×	×	×			×	×		×	

The last row of the table contains × signs for entities that achieved EER = 0 in at least 50 % of cases. These entities were selected and used for feature selection in this experiment.

There are six entities (1, 3, 4, 6, 14 and 16) that did not achieve EER = 0. This does not immediately mean that their results were bad (except for entity 16 whose results are truly bad, see figure 36), because it is rare to have EER = 0 in the identification system. However, the system is capable of tuning properly for some entities and therefore there is a chance it could be tuned for all entities. Analyzing the causes of this fact is not carried out in this dissertation.

Results from experiment runs with a reduced number of entities are displayed in figures 38 and 39. Because the reducing the number of entities also changed their mutual relations, the system was completely tuned from the beginning using only the selected entities.

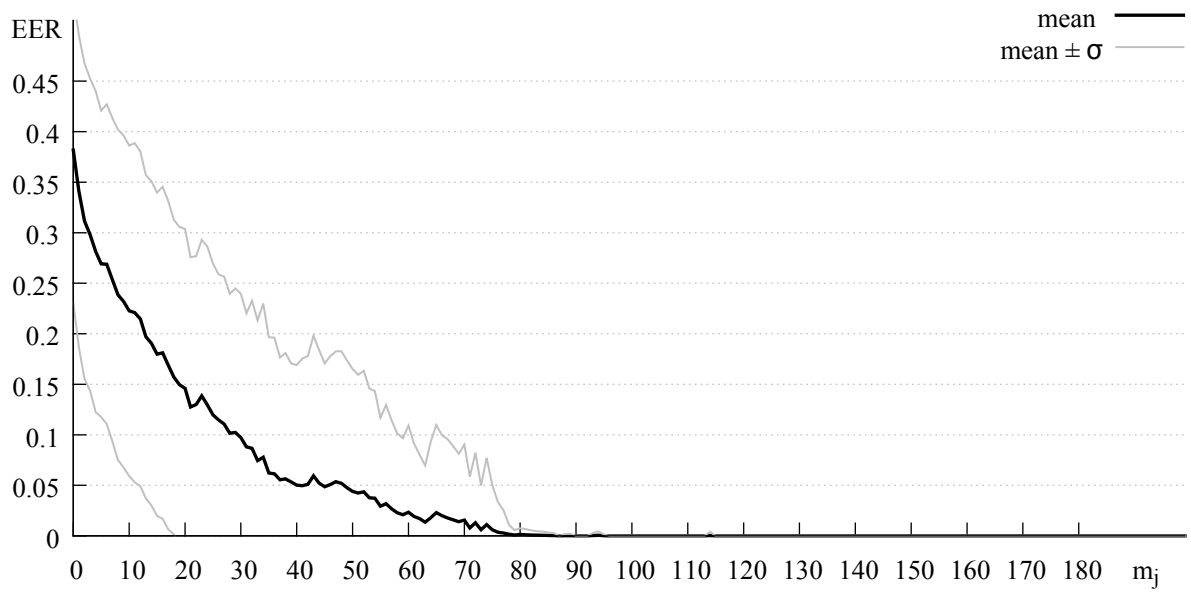

Figure 38 Development of the EER, all features, selected entities

Both of these results are better than results in figures 34 and 35, both in terms of standard deviation σ and convergence towards zero. σ is approximately 0.15, and all selected entities in both runs achieved EER = 0.

Both variants converged to zero around $m_s = 80$. This is about twenty strokes more than the system was tuned for ($= m_t$).

The overall conclusion is that we can consider the identification system to be functional in the expected way, if only correctly tuned entities are used. The result falsifies H5 and supports H6 (for both H5 and H6, refer to page 104). The convergence to minimal EER appears when samples are longer than the length of tuning samples ($m_s > m_t$).

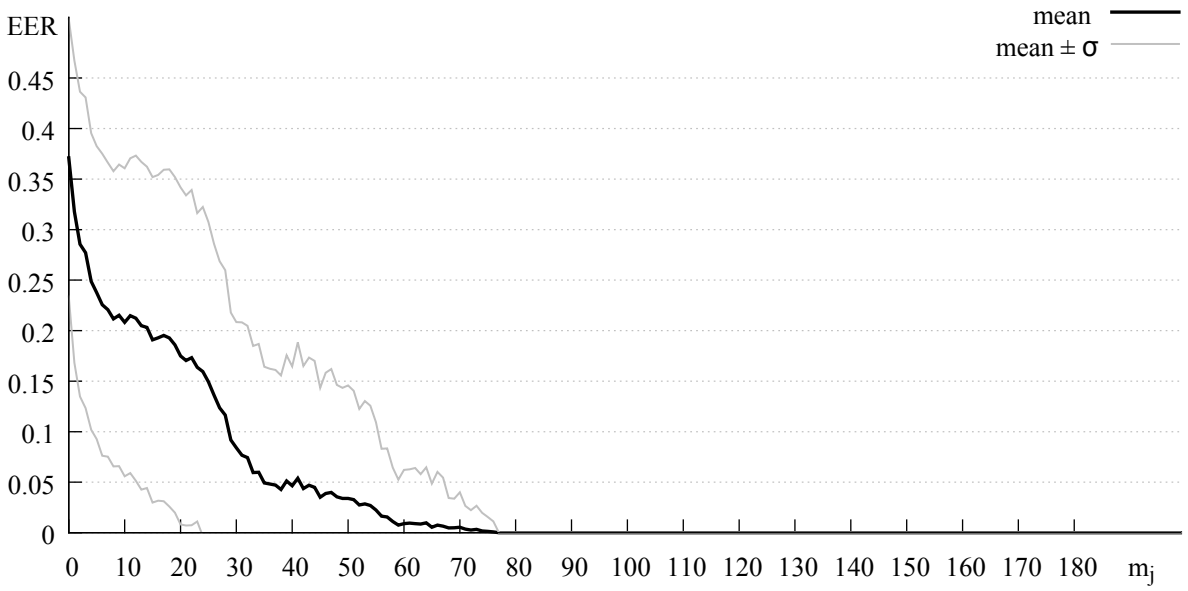

Figure 39 Development of the EER, reduced features, selected entities

10.5.3 Discussion of feature selection validation

The results of feature extraction and feature selection was validated by measuring the ERR of the whole identification system. Measurements of the EER were repeated 200 times for each entity, with a continuously increasing number of strokes in the test sample.

The validation results are:

- The identification system in this dissertation in principle works, including feature extraction and also feature selection. The best EER the system achieved was approximately 0.03, which is comparable to [4], although this dissertation achieves this EER level in twice as long time.
- The identification system is not able to prepare equal quality entities, approximately half the entities are suboptimal. The reason for this is unknown and it needs further research which is beyond the scope of this dissertation.

  The problems with tuning entities manifest mostly in a big standard deviation of the measured results.
- In order to test the identification system in theoretically ideal conditions, the badly tuned entities were removed, and the system was again tuned and then validated. The resulting behavior corresponded as expected and validated the identification system—the decreasing EER converged to a minimal zero value when the number of strokes in the test sample exceeded the number of strokes in tuning sample.
- The observation further confirmed that longer strokes give better results.

- It was also revealed that in real case (when all entities were used), the number of used features does matter: the results displayed in figure 34 are almost twice as good as the results displayed in figure 35. This observation is contrary to previous results discussed in chapter 10.3.6 at page 99.

The overall conclusion from this validation is that the used identification system works, that mouse-like devices can be used to identify people, and that there are problems with describing some people. These problems effectively prevent the system from being used. Further research into this field is needed.

10.6 Comparison of data sets

As described in chapter 8.3, six data sets were grabbed from each entity. This chapter and the experiment it contains, compares these data sets using the same method as in previous experiment 10.5.

Note that data sets of user's individual environments E^e cannot be compared to one another. This is because these data sets always only contain information related to a single user, and therefore how this data set accepts or rejects other users cannot be evaluated. Due to this fact, the synthetic environment E^s (composed of data sets D^e and A^e) is used in place of environments E^e.

10.6.1 The experiment setup and its results

The experiment ran consecutive feature selections for each data set and for each entity and used the following settings:

- the number of strokes in the sample used to tune the system was 60 strokes (see chapter 10.3.6),
- the duration of the gap ending the strokes was 1000 ms,
- the length m_s of the test samples changed from 1 to 200, and all lengths were tested 100 times; each sample was selected randomly,
- all available features were utilized,
- the EER of the whole system was measured.

The results are displayed in three figures according to table 13. The left part of each figure contains the result for the driver data source (D^-, D^+ or D^e), and the right part contains the result for the API data source (A^-, A^+ or A^e).

Table 13 Mapping of data sets to figures, comparison of data sets

data set	D^-	A^-	D^+	A^+	D^e	A^e
figure	40 L	40 R	41 L	41 R	42 L	42 R

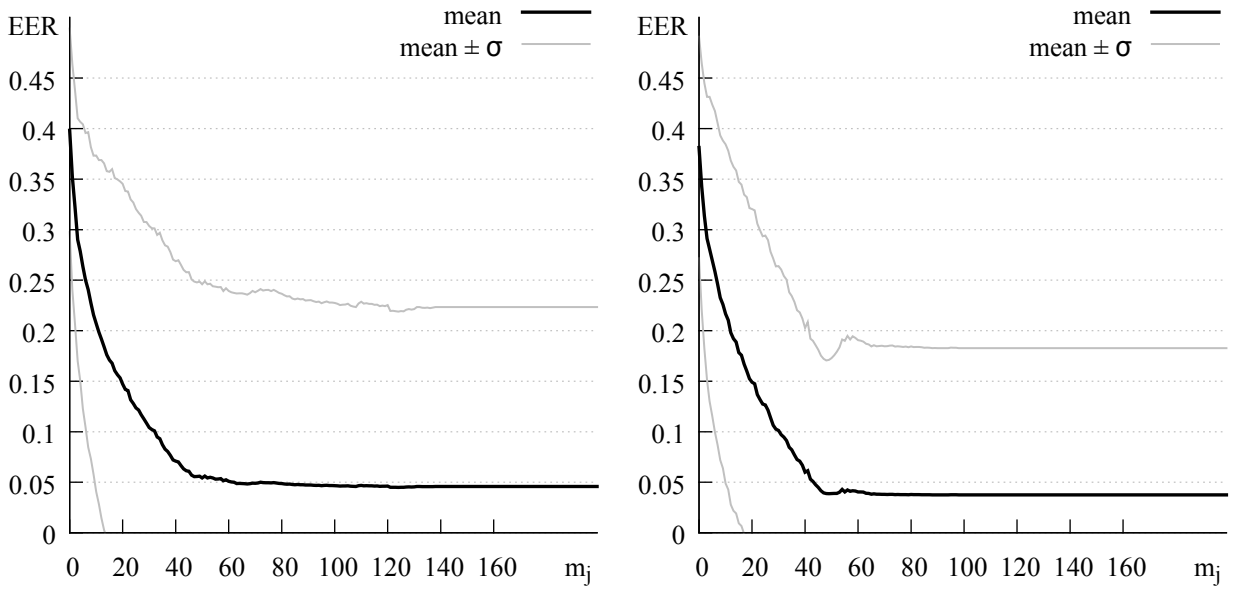

Figure 40 Development of the EER, D^- on the left, A^- on the right

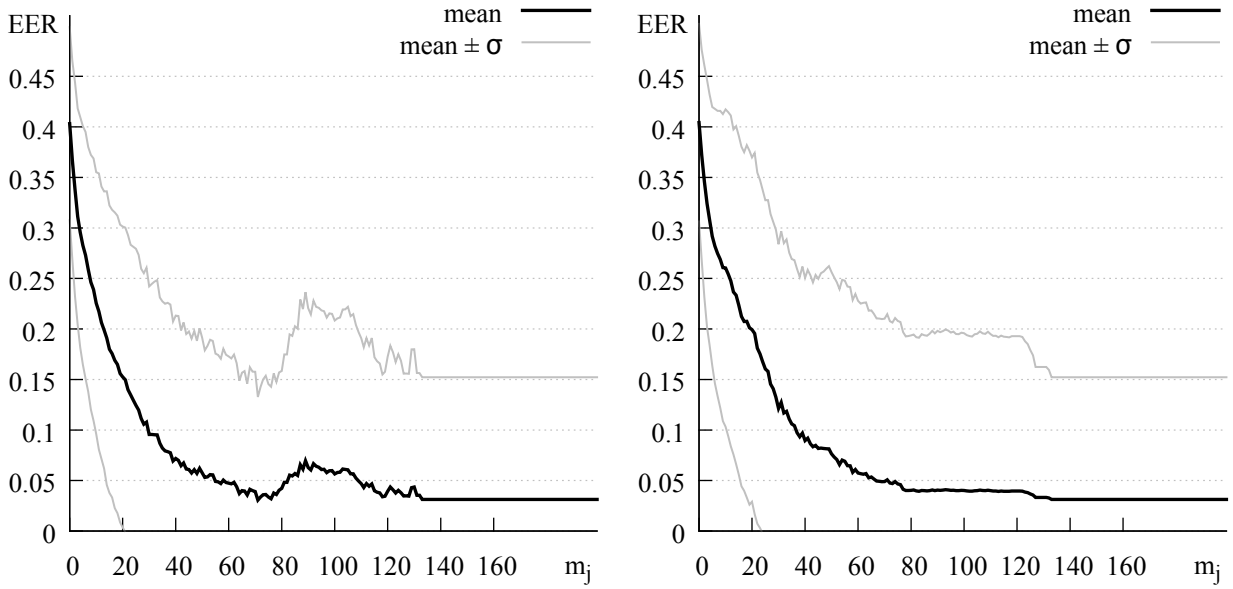

Figure 41 Development of the EER, D^+ on the left, A^+ on the right

10.6.2 Discussion of differences in data sets

The experiment results displayed in all three figures 40, 41 and 42 can be summed up as follows:

- API data sets (figures 40 R, 41 R, 42 R) always achieved better results than driver data sets (figures 40 L, 41 L, 42 L) though differences for both controlled environments E^- and E^+ were small.

  A possible explanation for this could be that the brain controls the position it can see, and that it cannot see the modifications of coordinates made by the user experience filter.

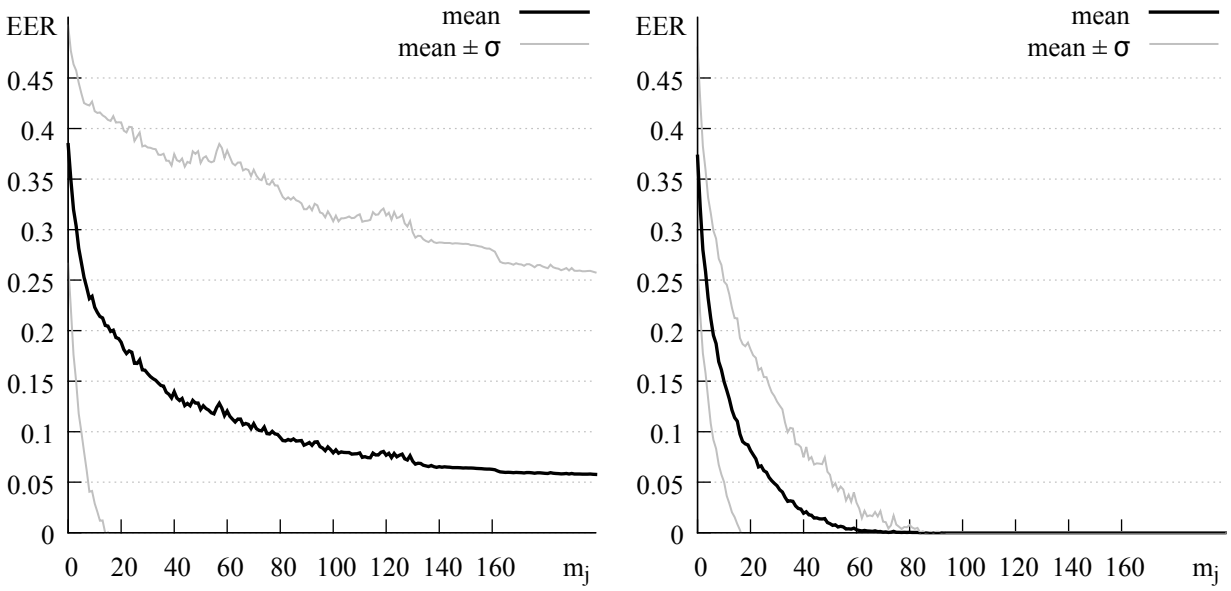

Figure 42 Development of the EER, D^e on the left, A^e on the right

This fact also replies to the question in chapter 6.4.2. API data sets have better results which emphasizes that the *movement-eye* approach is preferred to the *movement-measurement* approach (see page 46).

- The results from the controlled non-accelerated environment E^- displayed in figure 40 looks smoother than other results.

  This fact could be explained with the absence of the user experience filter: if the user experience filter is not used, the brain sees the exact position of the hand. The mouse cursor accelerates according to the hand, errors in hand positioning correspond to errors in mouse cursor positioning, and the brain has everything in concordance.

  On the other hand, if the filter is used, it can produce large and sudden changes to the coordinates. The filter amplifies acceleration so that a small error in hand positioning leads to a larger error in mouse cursor positioning. This amplified error could then be the source for the noise and bumps in the results.

  Note that better mouse placement precision, resulting in a smoother EER curve, does not necessarily mean better distinguishing between entities.

- The best result of all is displayed in figure 42 R which corresponds to the data sets A^e. There is also a big difference between results displayed in figures 42 L (datasets D^e) and 42 R (datasets A^e).

  A likely explanation of this observation is that users use mouse devices most naturally in their own environments; that they are accustomed to their own combination of the mouse, the computer and the mouse settings. In such case, controlling movements is learnt with long-term training, done best by the brain.

The bad result of the data sets D^e shown in figure 42 L supports this idea. When the brain is well-trained to a familiar environment, it fully overcomes the user experience filter and therefore the driver data corresponds worse to regulation, and consequently the driver data corresponds worse to the features.

The overall conclusion of data set comparison is that what the brain sees is more important than the real measured coordinates. This is because results are better from data sets using the user experience filter. However, the identification system is also able to tune entities to driver data sets.

10.7 Summary of feature selection

Feature selection is the second step to building entities that represent identified people. The primary goal of feature selection is to find the combination of features that best describes a particular person.

The feature selection algorithms were explored first. Two variants using three metrics were compared, and the combination of SFFS and d_{EER} was chosen. SFFS helps with overcoming the nesting effect that appears in measured data, and d_{EER} utilizes available information better than EER/polyline metric.

Next, analyses of variants using various numbers of features, using various lengths of strokes, and using various numbers of strokes in the sample were carried out. The following was discovered:

- more features does not mean better results (the finding is later overcome),
- longer strokes give better results than shorter strokes, the length of strokes was expressed as the time gap separating one stroke from the following stroke,
- sets of selected features start becoming stable when the sample contains approximately 20 strokes, and can be considered stable when the sample contains more than 60 strokes,
- the used method is capable of finding features that describe individuals well.

The next research phase into feature selection focused on the use of entities in the context of the complete identification system. Experiments carried out revealed that the identification system is principally capable of identifying people. The following is important to point out:

- using more features gives better results, contrary to the result mentioned above,
- in some runs the identification system was not capable of creating proper entities for all people. This fact prevents the system from being functional and this problem needs be further explored.

The last part of researching feature selection focused on comparing available data sets. These experiments discovered that data sets using displayed coordinates lead to better results than data sets using measured coordinates.

11 REUSABILITY OF DATA SETS

All previous experiments focused on forming representations of people, or focused on analyzing results achieved with a single data source. These experiments always used entities tuned for some particular data source and compared them with samples created from the same source.

The next two chapters will explore the behavior of the used identification system when the tuning and testing data sets differ. The goal of such cross-data set comparison is to learn more about reusing tuned data in various environments. Reusability is the key parameter affecting identification system operation scenarios.

Experiments carried out in this chapter copy arrangements used in other validation experiments, as in experiments 10.5 and 10.6.

11.1 Reusing data sets of different data sources

This experiment analyzes how the identification system can identify an entity when the sample is computed with data from a different data source of the same environment. This scenario is unlikely because both data sources are available in both environments and the selection algorithm can always select the correct source. Notwithstanding this fact, the experiment is still useful because it can shed a light on how similar data from different data sources are.

As in the experiment 10.6, the synthetic environment E^s is used in addition to controlled environments E^- and E^+.

The experiment used the following settings:

- there were 60 strokes in the sample used to tune the system,
- the duration of the gap was 1000 ms,
- the length m_s of test samples changed from 1 to 200, all lengths were tested 100 times; each sample was selected randomly,
- all features were utilized,
- the EER of the whole system was measured.

Results from this experiment are displayed in three figures according to table 14. The left part of each figure contains results of matching the driver data source (D^-, D^+ or D^e) with the corresponding API data source (A^-, A^+ or A^e). The right part of the figure contains the opposite, i.e. matching the API data source with the samples of the driver data source.

The results show that entities do not match well with samples from different data sources. This conclusion is also supported by the results of both non-accelerated variants $A^- \overset{?}{\leftarrow} D^-$ and $D^- \overset{?}{\leftarrow} A^-$ (see figure 43): convergencies of the driver data and of the API data in the environment E^- are almost identical because there is no user experience filter in the data flow path. Therefore tuning with both A^- and D^-

Table 14 Reusing data of different data sources, mapping data sets to figures

tuning data set	D^-	A^-	D^+	A^+	D^e	A^e
testing data set	A^-	D^-	A^+	D^+	A^e	D^e
figure	43 L	43 R	44 L	44 R	45 L	45 R

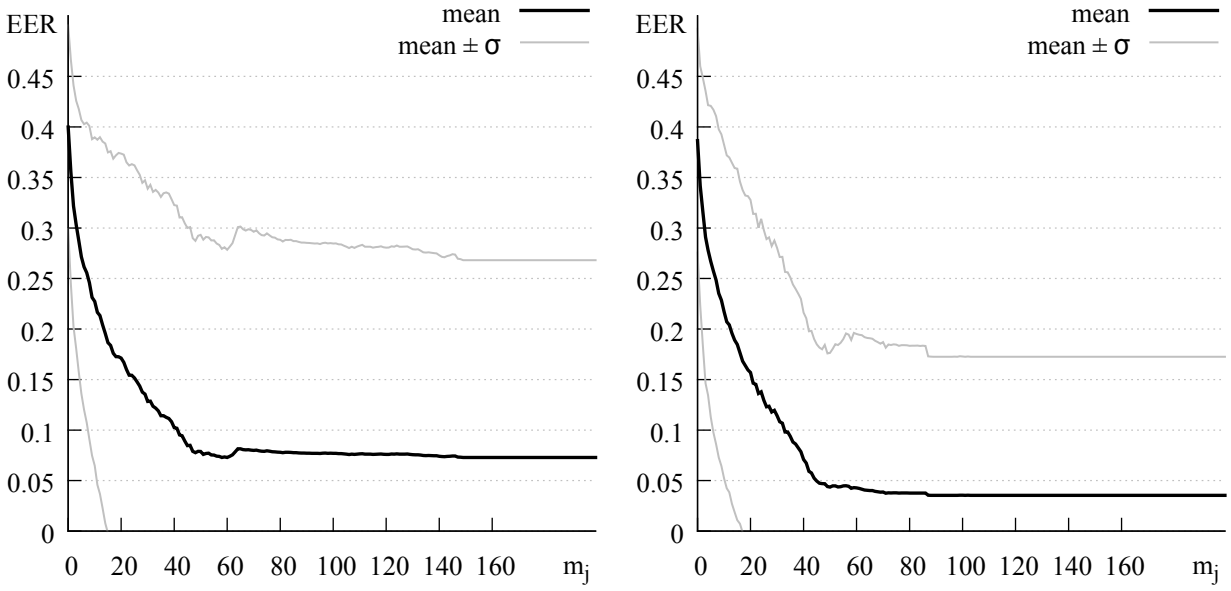

Figure 43 Development of the EER, $D^- \overset{?}{\leftarrow} A^-$ on the left, $A^- \overset{?}{\leftarrow} D^-$ on the right

should produce almost very similar entities that should match with data from both sources, just as can be seen in figure 43.

11.2 Reusing data sets of different environments

This experiment analyzes how the identification system is capable of identifying an entity when the sample is taken from different environment. The data source type (API or driver) is preserved.

This scenario is typical for identification systems whereby they use various input devices and/or their settings. The identification system can only work if tuned entities can successfully match with any sample from any input device.

As in the previous experiments 10.6 and 11.1, the synthetic environment E^s is used in addition to environments E^- and E^+. The environment E^s best represents mixture of devices and samples that can be expected in the real system.

The experiment used the following settings:

- there were 60 strokes in the samples used to tune the system,
- the duration of the gap was 1000 ms,
- The length m_s of test samples changed from 1 to 200, all lengths were tested 100 times, each sample was selected randomly,

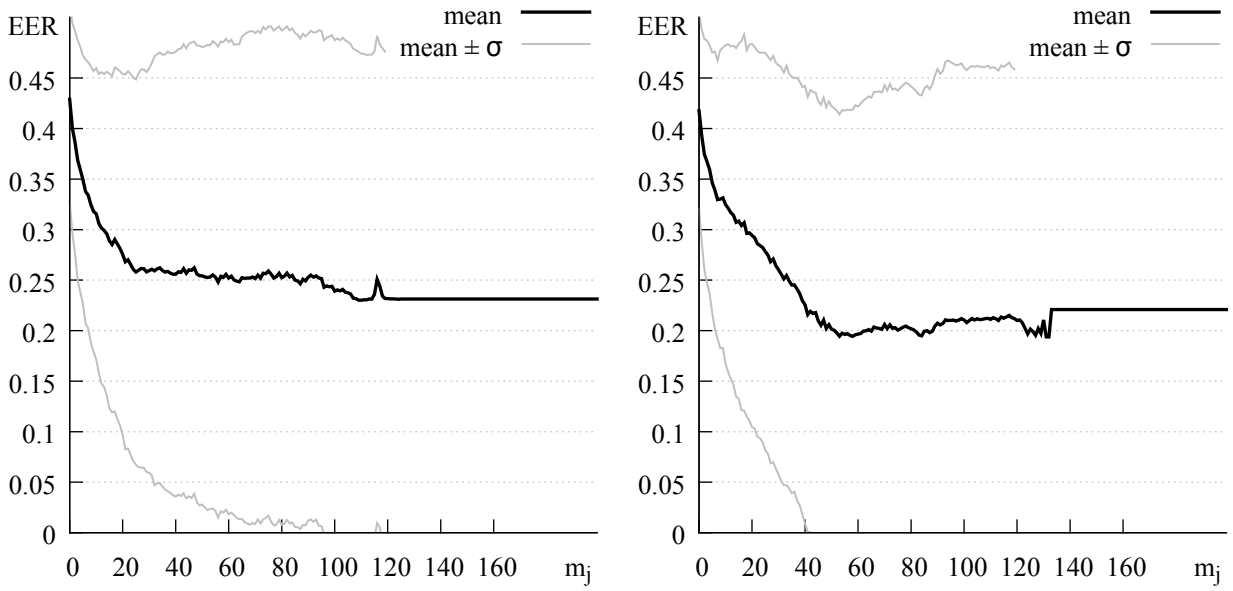

Figure 44 Development of the EER, $D^{+} \overset{?}{\leftarrow} A^{+}$ on the left, $A^{+} \overset{?}{\leftarrow} D^{+}$ on the right

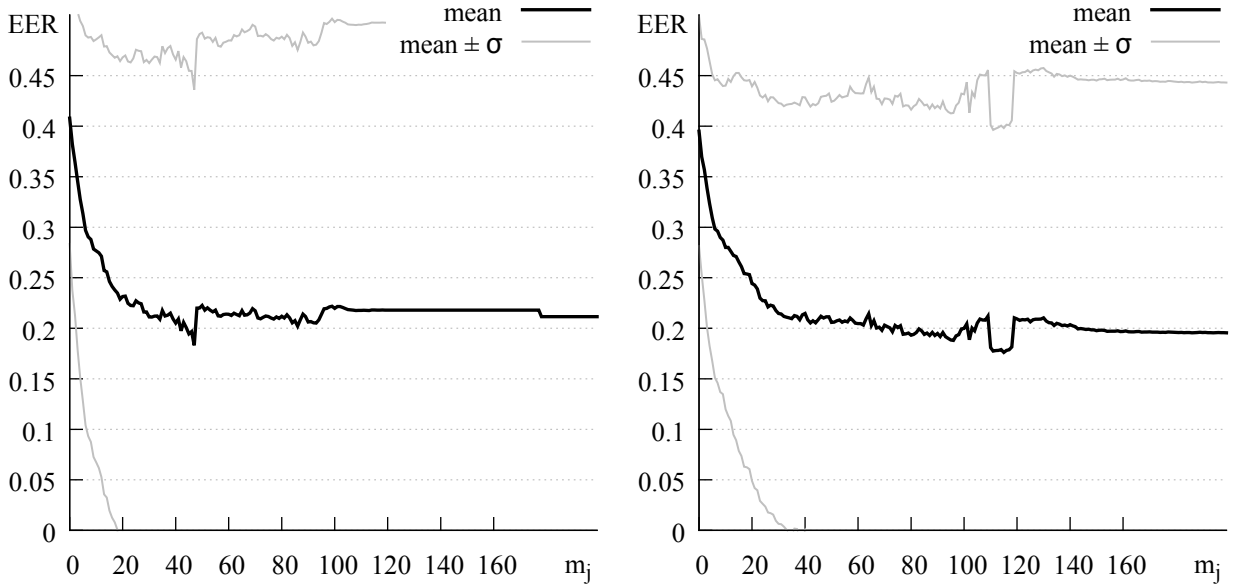

Figure 45 Development of the EER, $D^{e} \overset{?}{\leftarrow} A^{e}$ on the left, $A^{e} \overset{?}{\leftarrow} D^{e}$ on the right

- all features were utilized,
- the EER of the whole system was measured.

Results are displayed in three figures according to table 15. The left part of each figure contains the result of matching the driver data sources (D^{-}, D^{+} or D^{e}), and the right part of the figure contains the result of matching API data sources (A^{-}, A^{+} or A^{e}).

The results displayed in all these figures show that entities tuned in some environment do not at all match data from other environments. The best results have the ERR close to 0.35 and this value is unacceptably high. The standard deviation σ

Table 15 Reusing data of different environments, mapping data sets to figures

training environment	E^-	E^-	E^+	E^+	E^s	E^s
testing environment	E^+	E^s	E^-	E^s	E^-	E^+
figure	46	47	48	49	50	51

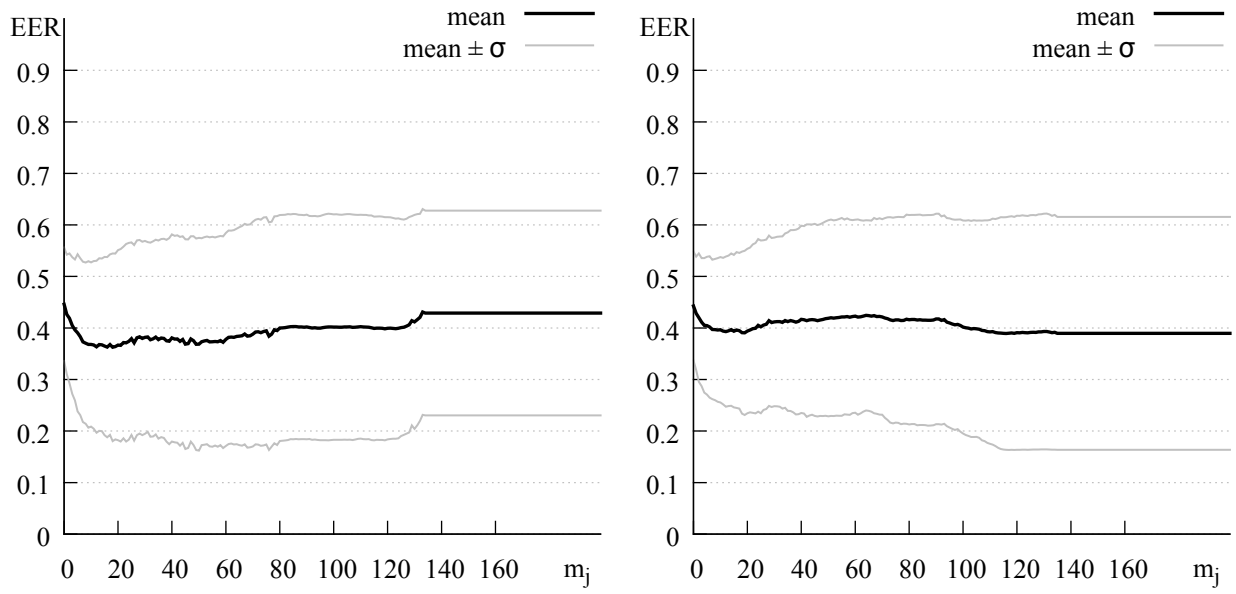

Figure 46 Development of the EER, $E^- \overset{?}{\leftarrow} E^+$, *D* on the left, *A* on the right

of the results is also unacceptably high; its value is on average less than σ of the previous experiment 11.1, but it is still too big.

11.3 Discussion and summary of reusing data sets

Two experiments were designed and carried out in order to explore how tuned features are reusable in various environments. The first experiment 11.1 explored what happens if test samples are from different data source (i.e. driver or API, see chapter 6.4.2). The second experiment 11.2 explored how successfully the entities are identified in samples taken on a different computer and with a different mouse.

Both experiments failed to produce acceptable results. Replacing the source of data worked only in non-accelerated environment where the result was expected. Replacing the environment did not work in any case—the best EER achieved was approximately 0.35.

The feature selection algorithm was able to form good entities, but when these entities were matched with data that the entities were not tuned for, the entities were not identified.

This result could be explained for example with:

- Data sets from different environments are incompatible.

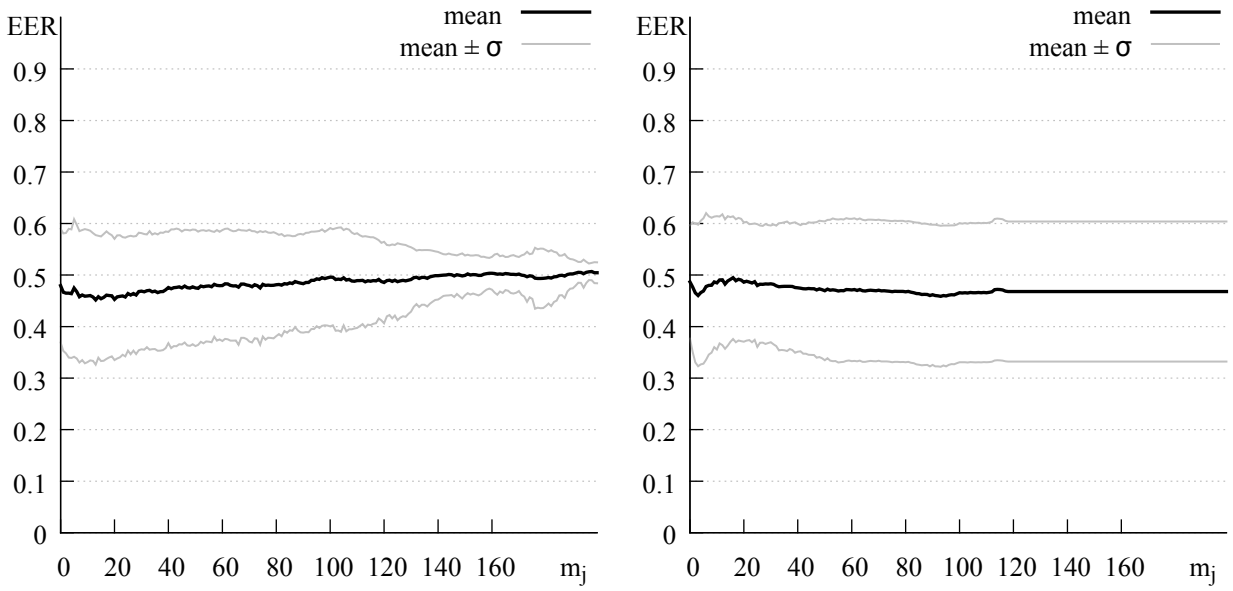

Figure 47 Development of the EER, $E^{-} \overset{?}{\leftarrow} E^{s}$, *D* on the left, *A* on the right

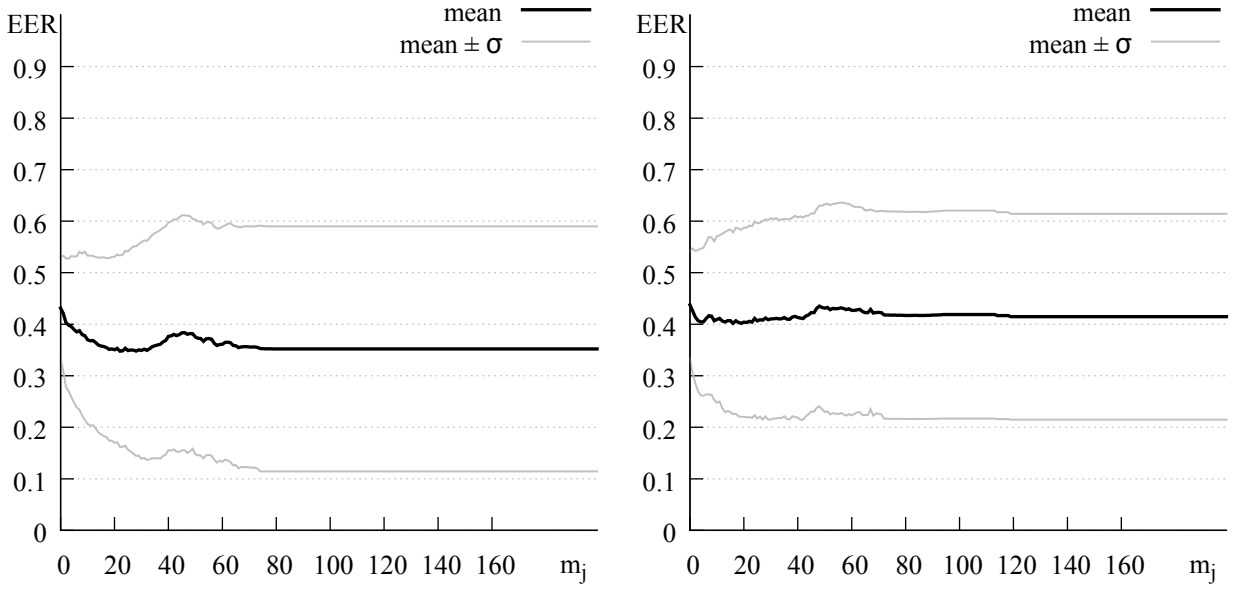

Figure 48 Development of the EER, $E^{+} \overset{?}{\leftarrow} E^{-}$, *D* on the left, *A* on the right

The information content of data sets is probably not the cause of this incompatibility because referenced prior research shows that data from various environments can be mixed (at least in some cases).

A possible source of incompatibility may be the feature selection. Compared to [4], this dissertation uses similar features, but the data processing process differs.

In order to confirm or disprove this explanation, a thorough analysis of features' forming and selection would help.

- The used feature selection algorithm might have strong tendency to search for local extremes.

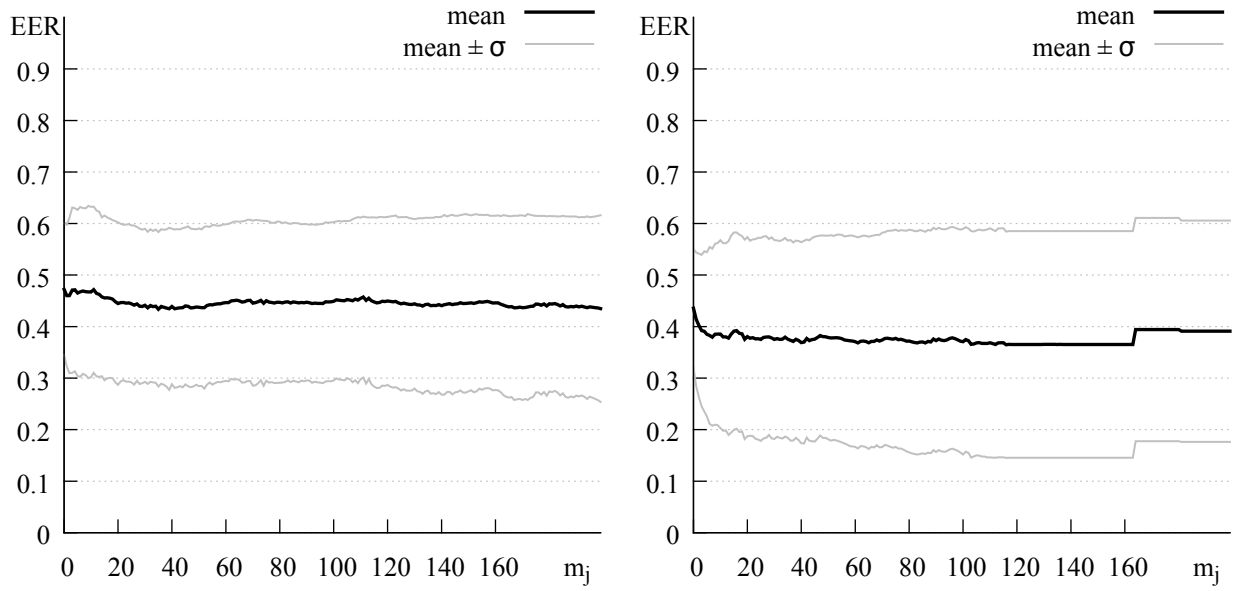

Figure 49 Development of the EER, $E^{+} \overset{?}{\leftarrow} E^{s}$, *D* on the left, *A* on the right

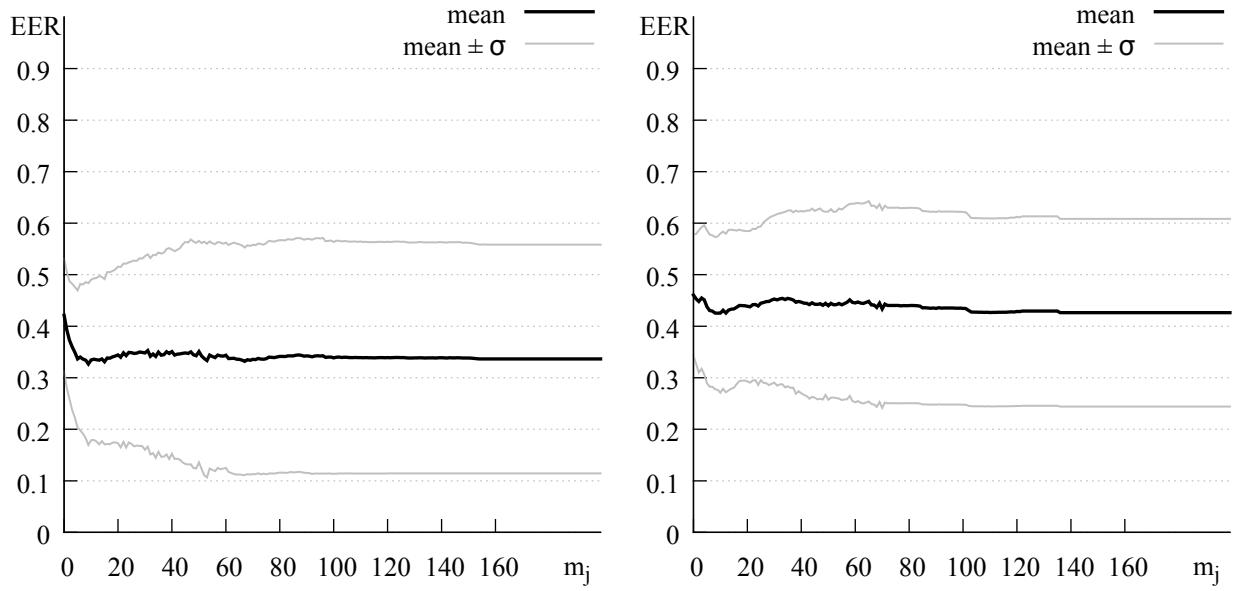

Figure 50 Development of the EER, $E^{s} \overset{?}{\leftarrow} E^{-}$, *D* on the left, *A* on the right

In such case, features that perfectly fit the given training data set would be selected, but some important subtle details could be passed over. If these subtle details were not passed over, it could lead to better feature reusability, because the feature selection could take advantage of these subtle details to find better (perhaps global) extreme.

The observation might be supported with the results of experiment 10.3, see the discussion in chapter 10.3.6. Repeated feature selections frequently found appropriate solution that was different from other appropriate solutions.

Further experiments could either confirm or disprove this explanation by focusing on the structure of the feature space and on its local and global minima.

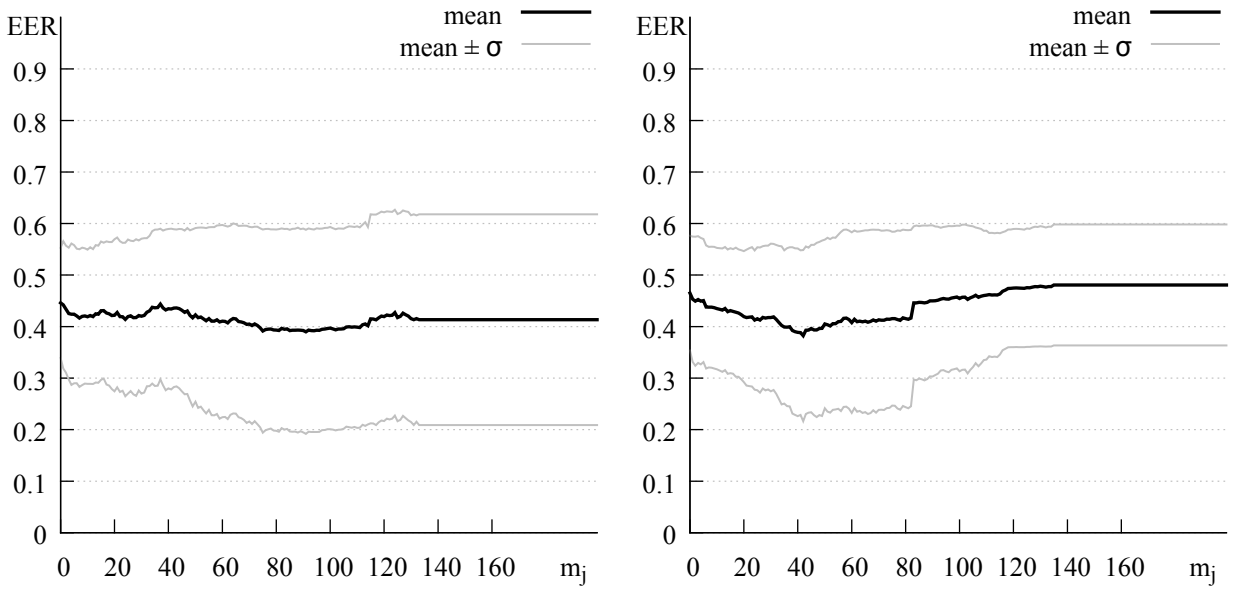

Figure 51 Development of the EER, $E^s \overset{2}{\leftarrow} E^+$, *D* on the left, *A* on the right

- The environments and the used data sets contained too little information for proper tuning.

  This idea might be supported with results from experiment 10.6, displayed in figure 42 R. This figure shows the best ever result achieved, even though the result was obtained for combined environment E^s that consists of unrelated data sets. The heterogeneity of information content of E^s could be the reason for achieving such a good result.

The overall result of cross-data set comparison is that the identification system used is not able to identify people across different environments.

12 DISSERTATION OUTCOMES

The outcomes of this dissertation are summarized in this final chapter, together with a top level discussion of the achievements. There are also references to the corresponding detailed discussion relating to the each goal.

12.1 Discussion of goals

Deep analysis of feature selection and metrics

Feature selection algorithms and metrics were researched in detail in chapter 10.2 and in entirety in chapters 10.3 and 10.5. Two algorithms were analyzed (SFS and SFFS, see chapter 5.3.2), and a new metric d_{EER} was developed in chapter 5.2.5 for faster comparison of the quality of systems and system variants.

The main results are:

- Measured data representing mouse-like device movements is sensitive to the nesting effect. The simple SFS algorithm is not sufficient and more advanced algorithms need to be used like SFFS.
- The newly developed metric d_{EER} can replace the traditional computing ERR and it has potential to replace the EER at all.
- The combination of the SFFS algorithm and the d_{EER} metric is, in principle, capable of searching features that describe a person well. Unresolved problems persist so the feature selection did not find the solution in all situations.
- The chosen approach of selecting features has big computational time complexity and for a large number of people it may be less usable.

This particular dissertation goal has been successfully fulfilled.

Chapters containing discussions relating to this goal are: 10.2.1, 10.2.3.

Enhancement of former work for unrestricted movements

The former work [4] was selected due to the fact that it used similar procedures and methods. [4] expects users to direct the mouse along a pre-determined path, this dissertation does not restrict user's movements in any way. The enhancement of [4] was discussed in sections throughout the entire content of this dissertation, for example in chapter 10.3.6.

The main results are:

- It is vital to extend the principle of the approach used in [4] to allow free movements. The identification system used in this dissertation is generally able to identify people.
- Experiments in this dissertation revealed that allowing unrestricted movements may require longer pieces of information than needed in [4].

This particular dissertation goal has been fulfilled.

Chapters containing discussions relating to this goal are: 10.3.6, 10.5.3.

Comparison of two methods of obtaining mouse-like device data

The operating system offer information from mouse-like devices in two forms, raw and adjusted. Both forms were described in chapter 8.2 and compared in experiment 10.6.

The main results are:

- Both data sources are sufficient for identifying people, but they are not interchangeable.
- Measured, adjusted data read from the API data source gives better results than raw data read from the driver data source. This finding means that the *movement-eye* model of controlling the mouse, explained in chapter 6.4.2, is more likely.

This particular dissertation goal has been successfully fulfilled.

Chapters containing discussions relating to this goal are: 10.6.2, 11.3.

Exploration of relationship and influence of various user environments

In order to analyze the influence of various environments, each user was instructed to produce data in three different arrangements. These were described in chapter 8.1 and analyzed in experiment 11.2.

The main result is that the data from different environments is unrelated in this dissertation. This observation in principle disallows usage of the dissertation's identification system.

This particular dissertation goal has been fulfilled, but the obtained results are unsatisfactory.

The chapter containing discussion relating to this goal is: 11.3.

12.2 Contribution to science and praxis

Evaluation of the importance of the selected features

According to experiment 10.4, the following features have greater importance than others (for the definitions of quantities, refer to chapter 9.2.6):

- The features relating to turning. These are, for example, normal acceleration a_n or jerk d^2v.
- The features relating to the straightness. These are, for example, inverted straightness r or jitter j.

Overall, the finding means that mouse-like device movements are distinguishable due to changes of direction rather than due to changes of speed. In other words

it is probably more important for the brain to control the final movement target, rather than to control the progress of the movement.

This finding may help with further research into principles of controlling mouse-like device movements.

Improvement of the feature selection process

This contribution is linked to the first goal of this dissertation which is *deep analysis of feature selection and metrics*. The main outcome regarding contributing is an invention of a novel metric d_{EER} that can replace the traditional EER.

d_{EER} is a measure that effectively measures the quality of the identification system, and which is equivalent to the EER. d_{EER} has two advantages over the EER:

- It can be computed directly from measured genuine similarities and impostor similarities. This means no curves need be constructed nor intersected in order to compute the d_{EER}.
- The computation of d_{EER} is fast. This means that repetitive tasks requiring comparison of performance of system(s) can be significantly sped up.

You can find more details in chapter 5.2.5.

General contribution

This dissertation uses the classical approach to resolve the feature selection and the classification problem. This might be considered regressive because current general research preference is given to modern artificial intelligence and soft-computing techniques (like in [12] and onwards).

The author believes that both approaches—classical and soft-computing—do not contradict, that they complement one other, and that both benefit from each other's achievements.

This dissertation adheres to this idea and attempts to bring out deeper understanding of behavioral identification systems utilizing the classical approach, with the hope that discovered problems and bottlenecks, and also results, can be addressed and/or overcome with soft-computing methods.

12.3 Proposals for further research

This dissertation has discovered difficulties and various problems during experimentation that have not yet been resolved. In order to continue research, the following topics need further exploration:

- The reliability of measurements. According to experiments, the data is scattered and likely undersampled. Better filtration and extraction would be useful.
- The suitability of selected features. Some information about features importance has been discovered, but thorough statistical and/or sensitivity analysis would discover more.
- The quality of the used probabilistic model of features. Quality improvement could either mean improvements in random variables, or improvements in the modeling probability distribution itself.
- The replacement of feature selection algorithm with different and/or a stochastic algorithm that can overcome existing big computational complexity.
  Some works utilizing soft-computing techniques was already presented, e.g. [12], but these works used artificial intelligence for the entire identification algorithm itself, not only as a solution for a particular partial task (like the feature selection).
- The space of the features, from which the feature selection selects the relevant features. The feature selection was able to find many correct solutions in many experiments, but these results only weakly correlated. Consequently, the repeatability and the convergence of feature selection is not good.
- The common information of the entity that is shared across various environments. The experiments in this dissertation have not succeeded in discovering this common information. Revealing this information is needed in order to enable the used identification system to work in real conditions.

13 CONCLUSION

How to identify people using mouse-like input devices has been researched for more than 12 years. During this time, many works have tested a few vital approaches and have proven that mouse-like device can be used to identify people.

After analyzing various published papers, this dissertation identified some general methodological omissions, and has not found any attempt to reproduce results. Two existing methodological problems are prominent: at the first, identification results are insufficiently tested on more various computers, so it is not known if a particular person would be identified on all computers and, for the second, operating system offers multiple methods of obtaining mouse coordinates and it is not known which method works well and which method works badly.

Analysis of research also revealed that only one paper (as far as the author knows) has tried to evaluate selected identification features with the aim of uncovering dependencies in the data and their hidden meanings.

Aware of this, this dissertation provides brief theoretical backgrounds and proposes experiments and research which would help improve the above-mentioned weaknesses. In particular, one prior work was chosen for enhancement and two areas of experiments were designed and carried out: experiments aimed at analyzing and improving feature extraction and feature selection processes, and experiments aimed at comparing identification results obtained in various environments. All experiments were designed and carried out by the author.

Experiments aimed at analyzing and improving feature extraction and feature selection improved the processes by overcoming the nesting effect and by developing a new measure for comparing the EERs. Simultaneously, the experiments proved that enhancements applied to the prior work are possible and that this derived enhanced method is also in principle able to identify people.

Experiments aimed at comparing various environments revealed that the identification system used is not able to overcome differences in environments, and that further research into the field is needed.

This is not the only finding which needs continuing research. Due to this, there are all discovered and yet unresolved problems summarized at the end of this dissertation, and proposals for the further research are suggested.

The general overview concerning this dissertation's goals is that three out of four particular goals (the analysis of feature selection, the enhancement of former work and the comparison of two sources of data) were positively fulfilled, and that one goal (exploring the relationship of various environments) was fulfilled negatively without achieving the presumed estimated results.

REFERENCES

1 WAYMAN, J., JAIN, A., MALTONI, D. and MAIO, D., 2005. *Biometric Systems, Technology, Design and Performance Evaluation*. ISBN 978-1-85233-596-0.

2 RAK, R., MATYÁŠ, V. and ŘÍHA, Z., 2008. *Biometrie a identita člověka*. Praha: Grada Publishing, a. s.. ISBN 978-80-247-2365-5.

3 JORGENSEN, Z. and YU, T., 2011. *On mouse dynamics as a behavioral biometric for authentication*. Raleigh: North Carolina State University.

4 GAMBOA, H. and FRED, A., 2004. A behavioral biometric system based on human computer interaction. *Proceedings of SPIE*. Vol. 5404, pp. **381–392**.

5 OEL, P., SCHMIDT, P. and SHMITT, A., 2001. Time prediction of mouse-based cursor movements. *Proc. Joint AFIHM-BCS Conf. Human-Computer Interaction IHM-HCI, 2001*. Vol. 2, pp. **37–40**.

6 FITTS, P. M., 1954. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology*. Vol. 47, iss. 6, pp. **381–391**.

7 IKEHARA, C. S. and CROSBY, M. E., 2003. User identification based on the analysis of the forces applied by a user to a computer mouse. *System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on*. Pp. **(7)**.

8 HASHIA, S., 2004. CS 297 Report. *Authentication by mouse movements*. San Jose: San Jose University.

9 GUTIERREZ-GARCIA, J. O., RAMOS, F. and UNGER, H., 2007. User Authentication via Mouse Biometrics and the usage of Graphic User Interfaces: An Application Approach. *Proceedings of the International Conference on Security and Management*. Pp. **76–82**.

10 GAMBOA, H. and FRED, A., 2003. An identity authentication system based on human computer interaction behaviour. *3rd workshop on pattern recognition in information systems PRIS 2003*. Vol. 3, pp. **49–55**.

11 KUMAR, S., SIM, T., JANAKIRAMAN, R. and SHENG, Z., 2005. Using continuous biometric verification to protect interactive login sessions. *Computer Security Applications Conference, 21st Annual*. Iss. 3, pp. **440–450**.

12 AHMED, A. A. E. and TRAOR'E, I., 2005. Detecting computer intrusions using behavioral biometrics. *Privacy, Security and Trust*. Victoria: University of Victoria.

13 PUSARA, M. and BRODLEY, C. E., 2004. User Re-authentication via Mouse Movements. *Proceedings of the 2004 ACM Workshop on Visualization and Data Mining for Computer Security*. Washington. Pp. **1–8**.

14 SCHULZ, D. A., 2006. Mouse Curve Biometrics. *Biometric Consortium Conference 2006, Biometrics Symposium: Special Session on Research at the*. Pp. **1–6**.

15 HASHIA, S., POLLETTB, C. and STAMP, M., 2005. *On using mouse movements as abiometric*. San Jose: San Jose University.
16 AHMED, A. A. E. and TRAOR'E, I., 2007. A new biometric technology based on mouse dynamics. *IEEE Trans. Dependable and Secure Computing*. Vol. 4, iss. 3, pp. **165–179**.
17 REVETT, K., JAHANKHANI, H., MAGALHES, S. T. de and SANTOS, H. M. D., 2008. A survey of user authentication based on mouse dynamics. *Proceedings of 4th International Conference on Global E-Security (Communications in Computer and Information Science)*. London: Springer. Vol. 12, pp. **381–392**.
18 EUSEBI, C., COSMIN, G., DEEPA, J. and MAISONAVE, A., 2008. A Data Mining Study of Mouse Movement, Stylometry, and Keystroke Biometric Data. *Proceedings of Student/Faculty Research Day*. White Plains: Pace University.
19 AJUFOR, N., AMALRAJ, A., DIAZ, R., ISLAM, M. and LAMPE, M., 2008. Refinement of a Mouse Movement Biometric System. *Proceedings of Student/Faculty Research Day*. White Plains: Pace University.
20 NAZAR, A., TRAOR E, I. and AHMED, A. A. E., 2008. Inverse Biometrics For Mouse Dynamics. *International Journal of Pattern Recognition and Artificial Intelligence*. Vol. 22, iss. 3, pp. **461–495**.
21 NAKKABI, Y., TRAOR'E, I. and AHMED, A. A. E., 2010. Improving mouse dynamics biometric performance using variance reduction via extractors with separate features. *IEEE Transactions On Systems, Man, And Cybernetics, Part A : Systems And Humans*. Vol. 40, iss. 6.
22 ZANDIKARIMI, H., LIN, F., CARLOS, C., CORREA, J. and DRESSNER, P. et al., 2014. Design of a Mouse Movement Biometric System to Verify the Identity of Students Taking Multiple-Choice Online Tests. *Proceedings of Student/Faculty Research Day*. White Plains: Pace University.
23 HAMID, N. A., SAFEI, S., SATAR, S. D. M., CHUPRAT, S. and AHMAD, R., 2011. Mouse movement behavioral biometric systems. *User Science and Engineering (i-USEr) 2011, International Conference on*. Pp. **206–211**.
24 FEHER, C., YUVAL, E., MOSKOVITCH, R., ROKACH, L. and SCHCLAR, A., 2012. Information Sciences. *User identity verification via mouse dynamics*. Raleigh. Vol. 201, pp. **19–36**. ISSN 0020-0255.
25 CHAO, S., ZHOMING, C., MAXION, R. A., XIANG, G. and XIAOHONG, G., 2012. Comparing classification algorithm for mouse dynamics based user identification. *Biometrics : Theory, Applications and Systems (BTAS) 2012, IEEE Fifth International Conference on*. Pp. **61–66**.

26 MUTHUMARI, G., SHENBAGARAJ, R. and PEPSI, M. B. B., 2014. Mouse gesture based authentication using machine learning algorithm. *Advanced Communication Control and Computing Technologies (ICACCCT) 2014, International Conference on*. New York. Pp. **492–496**.
27 ZHENG, N., PALOSKI, A. and WANG, H., 2011. An Efficient User Verification System via Mouse Movements. *Proceedings of the 18th ACM Conference on Computer and Communications Security*. Chicago: ACM. Pp. **139–150**.
28 CHIEN-CHENG, L., CHIN-CHUN, C. and DERON, L., 2012. A New Non-intrusive Authentication Approach for Data Protection Based on Mouse Dynamics. *Biometrics and Security Technologies (ISBAST) 2012, International Symposium on*. Pp. **9–14**.
29 SAYED, B., TRAOR'E, I., WOUNGANG, I. and OBAIDAT, M. S., 2013. Biometric Authentication Using Mouse Gesture Dynamics. *IEEE Systems Journal*. Vol. 7, iss. 2, pp. **262–274**.
30 JAIN, A. K. and ROSS, A., 2008. *Handbook of Biometrics*. Raleigh: Springer. ISBN 978-0-387-71040-2.
31 JOHNSON, N. L., KOTZ, S. and BALAKRISHNAN, N., 1994. *Continuous Univariate Distributions*. 2nd issue. New York: John Wiley. Vol. 1. ISBN 978-0-471-58495-7.
32 MathWave Technologies, Inc., 2014. *EasyFit—Distribution Fitting Made Easy*. http://www.mathwave.com/, cited 2014/10/6.
33 ŠTROSOVÁ, J., 2012. *Zobecněné a normální inverzní Gaussovo rozdělení pravděpodobnosti*. Brno: Masarykova univerzita, přírodovědecká fakulta. http://is.muni.cz/th/357381/prif_b/, cited 2014/10/8.
34 JUSTUS, C. G., HARGRAVES, W. R., MIKHAIL, A. and GRABER, D., 1978. Journal of Applied Meteorology. *Methods for Estimating Wind Speed Frequency Distributions*. Atlanta: School of Aerospace Engineering, Georgia Institute of Technology. Vol. 17, pp. **350–353**.
35 FILLIBEN, J. J., 2003 (revised 2012). *NIST/SEMATECH e-Handbook of Statistical Methods*. Albany: NIST, U. S. Commerce Department. http://www.itl.nist.gov/div898/handbook/, cited 2014/11/03.
36 WAYMAN, J., 1998. *A Generalized Biometric Identification System Model*. San Jose: U. S. National Biometric Test Center, San Jose State University.
37 SCHUCKERS, M. E., 2010. *Computational Methods in Biometric Authentication*. London: Springer. ISBN 978-1-84996-202-5.
38 MANIVANNAN, P., 2011. Comparative and Analysis of Biometric Systems. *International Journal on Computer Science and Engineering*. Kanchipuram. Vol. 3, iss. 5, pp. **2156–2162**. ISSN 0975-3397.
39 GUYON, I., GUNN, S., NIKRAVESH, M. and ZADEH, L. A., 2006. *Feature Extraction*. Berlin: Springer. ISBN 978-3-540-35487-1.

40 Amanita Design, s.r.o., 2009. *Machinarium game*. http://machinarium.net/, cited 2014/10/8.
41 Microsoft, Corp., 2014. *Change mouse settings*. Redmond. http://windows.microsoft.com/en-us/windows/change-mouse-settings, cited 2014/12/10.
42 Antigen, U., 2007. *Mouse Optimization Guide*. http://www.overclock.net/t/173255/cs-s-mouse-optimization-guide, cited 2014/12/11.
43 Hoppan, U., 2010. *Tutorial: How To Customize Windows Accel*. http://www.esreality.com/index.php?a=post&id=1945096, cited 2014/12/11.
44 Hoppan, U., 2010. *Link to download CustomCurve tool*. http://esreality.com/download.php?file_id=103413, cited 2014/12/11.
45 E. SHANNON, C., 1949. Proc. Institute of Radio Engineers. *Communication in the presence of noise*. Vol. 37, iss. 1, pp. **10–21**. Reprint as classic paper in Proc. IEEE, vol. 86, no. 2, 1998, online: http://www.stanford.edu/class/ee104/shannonpaper.pdf.
46 CATMULL, E. and CLARK, J., 1978. *Catmull-Clark subdivision surface*. https://en.wikipedia.org/wiki/Catmull-Clark_subdivision_surface, cited 2014/10/31.
47 POLLOCK, D. S. G., 1993. *Smoothing with Cubic Splines*. London: Queen Mary University of London. http://www.physics.muni.cz/~jancely/NM/Texty/Numerika/CubicSmoothingSpline.pdf, cited 2014/10/5.
48 L'Ecuyer, P., Meliani, L. and Vaucher, J., 2002–2015. *A Framework for Stochastic Simulation In Java*. Montreal: IEEE Press. Pp. **234–242**. http://simul.iro.umontreal.ca/ssj/indexe.html, cited 2014/10/24.
49 Kolařík, M. and Jašek, R., *Fast method of comparing performance of identification systems*.
50 Price, K., Storn, R. M. and Lampinen, J. A., 2005. Differential evolution: A Practical Approach To Global Optimization. *Natural computing series*.

LIST OF AUTHOR'S PUBLICATION ACTIVITIES

Conference papers

1. KOLAŘÍK, M., JAŠEK, R., 2011: Identification of persons using gait—some problems of technical realization of measuring using consumer electronics devices. *Proceedings of the International Workshop: Methods and Applications of Artificial Intelligence*. Bielsko-Biala. Pp. **13**–**19**.
2. JAŠEK, R., VÝMOLA, T., KOLAŘÍK M., 2013. APT detection system using honeypots. *Proceedings of 14th WSEAS International Conference on Automation & Information*. Valencia. ISBN 978-960-474-316-2.
3. KOLAŘÍK, M., JAŠEK, R., KOMÍNKOVÁ OPLATKOVÁ, Z., 2014. Maximizing Vector Distances for Purpose of Searching—A Study of Differential Evolution Suitability. *Proceedings of the Fifth International Conference on Innovations in Bio-Inspired Computing and Applications IBICA 2014*. Ostrava. ISBN 978-3-319-08155-7.
4. KOLAŘÍK, M., JAŠEK, R., KOMÍNKOVÁ OPLATKOVÁ, Z., 2014. Maximizing vector distances using differential evolution—relation to data redundancy. *AIP Conference Proceedings: ICNAAM 2014*, 1648, 550019. Rhodes. DOI: `http://dx.doi.org/10.1063/1.4912774`.
5. JAŠEK R., KOLAŘÍK, M., 2015. Features preferred in identification system based on computer mouse movements. *AIP Conference Proceedings: ICNAAM 2015*, accepted for publication.
6. KOLAŘÍK M., JAŠEK, R., 2015. Fast method of comparing performance of identification systems. *ICCWS 2016*, accepted for publication. Boston.
7. JAŠEK R., KOLAŘÍK, M., 2015. The most relevant features selected from computer mouse moves and their distributions. *ICCWS 2016*, accepted for publication. Boston.

Journal articles

1. JAŠEK R., KOLAŘÍK, M., 2015. Features of computer mouse moves suitable for intelligent systems. *Journal Of Intelligent Systems*, accepted for publication. ISSN: 0334-1860.

CURRICULUM VITAE

Identity

Martin Kolařík, born August 7th, 1973

E-mail

martin.kolarik@email.cz

Education

2007, Tomas Bata University in Zlin, Faculty of Applied Informatics
Bachelor's degree, Information Technologies, diploma with excellence

2009, Tomas Bata University in Zlin, Faculty of Applied Informatics
Master's degree, Information Technologies, diploma with excellence

2009–, Tomas Bata University in Zlin, Faculty of Applied Informatics
doctoral studies, Engineering Informatics

Professional profile

combine professional analytical, innovative and developing skills to design, code, test, deliver and support large scale of applications from device level programming to enterprise solutions

Technological skills

C++/C#/COM, J2EE/J2ME/J2SE, deep Win32 API, embedded devices, XML/XSLT/XSD/XSL:FO, CAN/OpenCAN/MBUS/HART, KNX/LON, SQL

Language skills

Czech—native speaker, English—advanced, Russian—passive

Work history

1992–2006, Moravské přístroje, a.s.
many positions, particularly: analysis, developing, head of project groups

2006–, Edhouse, s.r.o.
senior software engineer, J2EE projects; project manager, electron microscopes group; lead software engineer

2006–, SmartControl, owner
development of custom software solutions for industry and building automation