Hands-on Smart Card User Interface Research, Development, and Testing

Markus Ullmann* † and Ralph Breithaupt*
* Federal Office for Information Security
D-53133 Bonn, Germany
Email: {markus.ullmann ralph.breithaupt}@bsi.bund.de
† University of Applied Sciences Bonn-Rhine-Sieg
Institute for Security Research
D-53757 Sankt Augustin, Germany
Email: markus.ullmann@h-brs.de

***Abstract*—The latest advances in the field of smart card technologies allow modern cards to be more than just simple security tokens. Recent developments facilitate the use of interactive components like buttons, displays or even touch-sensors within the card's body thus conquering whole new areas of application. With interactive functionalities the usability aspect becomes the most important one for designing secure and popularly accepted products. Unfortunately, the usability can only be tested fully with completely integrated hence expensive smart card prototypes. This restricts severely application specific research, case studies of new smart card user interfaces and the optimization of design aspects, as well as hardware requirements by making usability and acceptance tests in smart card development very costly and time-consuming. Rapid development and simulation of smart card interfaces and applications can help to avoid this restriction. This paper presents a rapid development process for new smart card interfaces and applications based on common smartphone technology using a tool called SCUID^{Sim}. We will demonstrate the variety of usability aspects that can be analyzed with such a simulator by discussing some selected example projects.**

***Keywords*–Smart Card; Smart Card User Interface Design, Interactive Smart Card Applications; Rapid Prototyping; Simulation; Testing; Usability.**

I. Introduction

Today, smartphones belong to the most widely used electronic consumer devices for communication, infotainment and entertainment. High end systems are equipped with a powerful processor, graphics processor, camera, high resolution display as well as a lot of sensors and support various communication technologies, thus providing a powerful combination of convenience and versatile possibilities. That is why more and more applications that require a high level of security like banking or electronic shopping are becoming tools for everyday use. Therefore, a secure element on the device is included to store the cryptographic keys and to perform cryptographic operations. Unfortunately, it is very challenging to combine high security elements with complex consumer electronics optimized for convenience with their different open APIs, communication interfaces and update mechanisms, as jailbreaks and other successful attacks on employed security measures have shown in the past.

If security is the dominant issue, e.g., for personal identification, authentication, access control, banking, pay-tv, crypto services, etc., another system progression takes place based on smart cards, which consists mainly of a secure element embedded in a smart card body. Smart cards are superior in the area of low cost, mobility and especially certifiably high security. However, common smart cards are simple security tokens and have no user interface. Every additional authentication process requires external devices like, e.g., keypads for entering passwords, which by themselves are potentially vulnerable against side channel attacks, eavesdropping, etc. It would be ideal to combine the security level of a smart card with the convenience of a smartphone to keep everything "on card". Recently developed interactive components allow the integration of input devices, like buttons, keypads or touch based gesture interfaces as well as output devices like displays and LEDs directly into a smart card.

With such interactive functionalities the usability aspect becomes the most important one for designing a usable smart card and adds many new demands to the development process. Now, aspects like the adequate size of a button, the visibility of a touch interface, the resolution, contrast and speed of a display and the overall design of the card have to be addressed as well as an appropriate hardware/software-codesign to ensure clear user guidance and high overall usability. This can only be achieved by conducting extensive field tests with as many people as possible. Creating the necessary card prototypes with the complete design and full hardware and software functionality can be very expensive and time-consuming, which makes usability centered security research difficult. This is the motivation for SCUID^{Sim} (**S**mart **C**ard **U**ser **I**nterface **D**evelopment **Sim**ulator): to support the development and evaluation of smart cards with user interfaces. In this paper, we present an alternative approach to allow all the necessary testing in order to determine the requirements for design, hardware components and the software without the need to build costly prototypes. By using common smartphones as a development and evaluation platform, almost all user related aspects can be investigated by simulating the "look & feel" of a new smart card design before any real hardware integration is needed.

SCUID^{Sim} is an android application and therefore usable on a wide range of smartphones, which combine all the necessary hardware input/output components as well as communication links, cryptographic services, the processor power and memory needed for simulating a large variety of current and future smart card interfaces and applications in a single compact device. With SCUID^{Sim} the visible aspects of a multi-component smart card can be designed on the smartphone. Based on a simple SCUID^{Sim}-API, user defined card applications can be executed while SCUID^{Sim} simulates the behavioural properties of all interactive components. New requests and requirements can be implemented, simulated and evaluated instantly. This way SCUID^{Sim} supports detailed requirement engineering for software as well as hardware and the development of new user interface concepts hand in hand. This is especially useful for the design and integration of new usable user centric security algorithms in smart cards. SCUID^{Sim} was firstly published at The Fourth International Conference on Ambient Computing, Applications, Services and Technologies (AMBIENT 2014) [1]. In this enhanced journal version, the focus is on smart card user interface research, development and testing using SCUID^{Sim}.

In recent years much effort has been made in order to integrate segmented displays into smart cards. Most prototypes used electrophoretic display technology, as shown in Figure 3, known from ebook readers. But this direction was no success story to date. High costs were one reason - but the main problem was its typical update delay of 1 second, which for small displays did not get enough public acceptance. From this experience many questions emerged, like: how fast does such a display have to be in order to be acceptable for certain applications? What resolution and contrast is necessary for adequate usability? Such usability centred questions are very hard to investigate with real smart card prototypes.

SCUID^{Sim} is a framework to initially develop and test new usability approaches very quickly prior to the development of the smart card hardware. This way all user acceptance related issues can be investigated and optimized resulting in a detailed requirement list for all hardware components. It was our goal to significantly increase the speed and efficiency of the development and evaluation of new interactive smart card concepts.

The following sections of this paper are organized as follows: Section II starts with a description of related work. Section III provides a brief overview of the software architecture of SCUID^{Sim} and its functionality. Next, Section IV describes applications of SCUID^{Sim} for the design of smart card user interfaces. We focus our attention on different LED matrix based displays which are very easy to handle, cheap and flexible as an example how the possibilities and challenges of such a technology can be investigated with SCUID^{Sim}. We present the evaluation of various interaction concepts like animated symbols, scrolling text and even rapid serial visual presentation displays for long text passages in regard to the constraints of a smart card. As input mechanism, we present a one-character display input device and in addition user inputs based on touch-gestures. Finally, in Section V we summarize our results.

II. Related Work

The first research and development projects investigating the idea to integrate input and output elements in smart cards go back as far as the late 1990s, see [2]. With the advances in low power and low profile embedded technologies many different component technologies have been successfully developed and integrated in ID1-compatible smart cards during the last decade. Primarily, a variety of display types and buttons, even fingerprint scanners, are discussed for integration, see [3] and [4]. Moreover, in [5] smart cards with an integrated display as security enforcing component are introduced. A first approach to integrate a 2D on-card gesture input sensor, implemented as capacitive touch matrix, is introduced first in [6]. It has also been an important topic for public funding in many countries (e.g., the INSITO-project of the German Federal Office for Information Security (BSI) and the SECUDIS-project of the German Ministry of Education and Research, see [7] and [8]). Despite all the effort and the growing number of available components, interactive smart cards have not yet been used in many real applications. Among other reasons this is due to high production costs and the much higher complexity of such smart cards. With the recent advances in printed electronics capacitive sensors have become a widely accepted standard technology and even printed displays are available today, see [9], [10], and [11]. But the complexity issue is still a serious obstacle on the way to the final product. At least regarding the system integration issues of combining several hardware components there have been approaches for rapid prototyping tools. One of the first was the FlexCOS system suggested by Beilke et al. [12], which uses FPGAs for a very flexible and rearrangeable interface to connect separate component prototypes into one complete system. Although this approach became a standard procedure for many manufacturers and researchers, it only covers the technological aspects. Such functional prototypes are much too bulky and fragile to conduct real world tests with many people in real application scenarios outside the lab. The usability aspects that first and foremost define how the smart card should interact, and therefore, what the requirements for the hardware and software components really are can not be tested without fully integrated and designed card prototypes. Unfortunately, each version of real prototypes to test for user acceptance requires huge expenses of time and money. This lack of end-user centered rapid prototyping tools was the starting point for the development of the SCUID^{Sim} tool. Simulation of user interfaces was very popular in the beginning of ubiquitous computing. One approach was the iStuff toolkit to support the development of user interfaces for the post-desktop age for multiple displays, multiple input devices, multiple systems, multiple applications and multiple concurrent users, see [13]. Alternative technologies were developed by the Stanford Interactive Workspaces project for multi-person and multi-device collaborative work settings, see [14]. To the best of our knowlege, SCUID^{Sim} is the first approach to model, simulate and analyze user interfaces for (contactless) smart cards.

III. SCUID^{Sim} Architecture

SCUID^{Sim} consists of two modules: a card designer that enables a flexible but simple arrangement of smart card layouts based on preconfigured components and a card simulator. In the card simulator, such a card layout can be paired with a smart card application in a real time simulation. It was a design decision to separate the card design process and the card simulation process in two independent software modules. Figure 1 illustrates the SCUID^{Sim} architecture.

A. Card Designer

The card designer is a simple tool to engineer smart card layouts. Figure 2 gives an overview of the available components in the current version of SCUID^{Sim}. Currently, the following predefined components are supported: push buttons, segmented displays (7- and 14-segments), matrix displays (RGB, greyscale and black & white), LEDs, n × m LED-matrixes, 2D-touch sensors, image boxes and the overlay

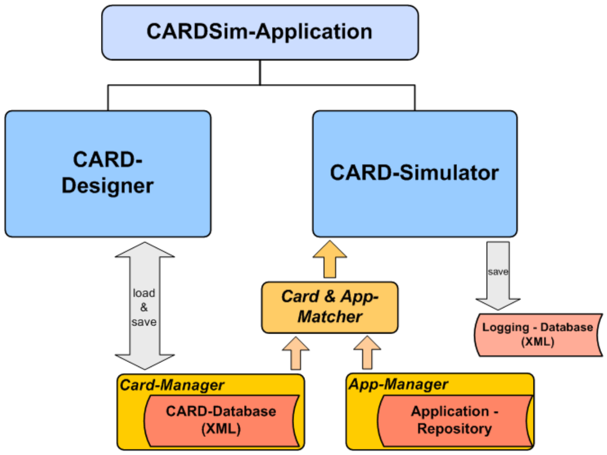

Figure 1. Overview of the SCUID^{Sim} software architecture.

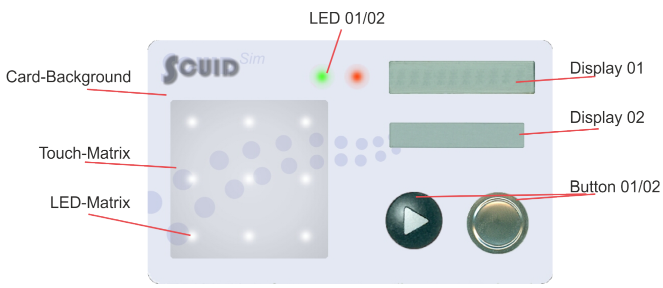

Figure 2. Available card components (in the card designer)

image of the smart card. There are also non-visible components like acceleration sensors that are automatically available to all cards if the used android smartphone is supporting it. With this initial set of predefined components, SCUID^{Sim} can already simulate a huge variety of smart card layouts. Figure 3 depicts a real card prototype opposite to a replicated design of this card within SCUID^{Sim}. This figure illustrates the very realistic replication capabilities of our tool.

Within the card designer, the properties of each component like position & size can easily be controlled via simple finger gestures commonly known from many other mobile applications. Additional properties like the appearance of the component (overlay image), a color modifier (to the overlay image, in RGB and alpha for transparency) or component specific properties like X/Y-resolution of a matrix display, or the update delay time for a display component can be set in a component property page that is dynamically generated

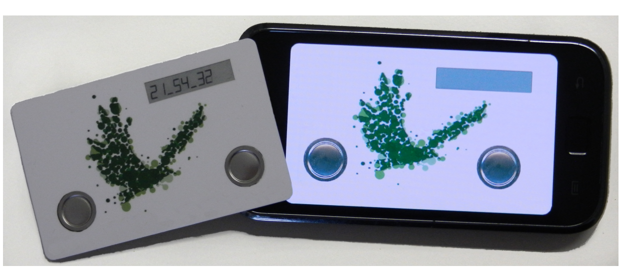

Figure 3. Confrontation real - and simulated card layout within SCUID^{Sim}

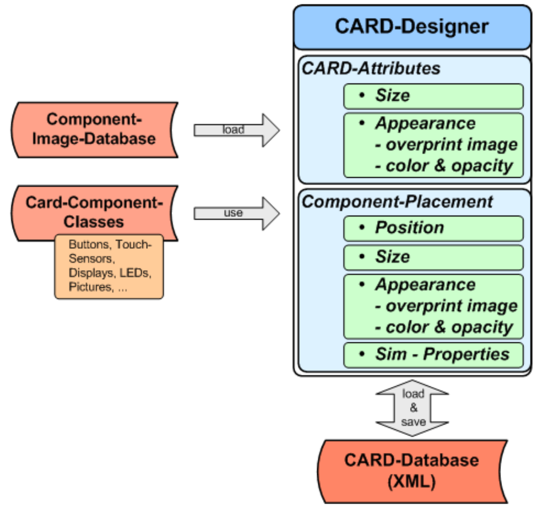

Figure 4. Software architecture of the card designer module

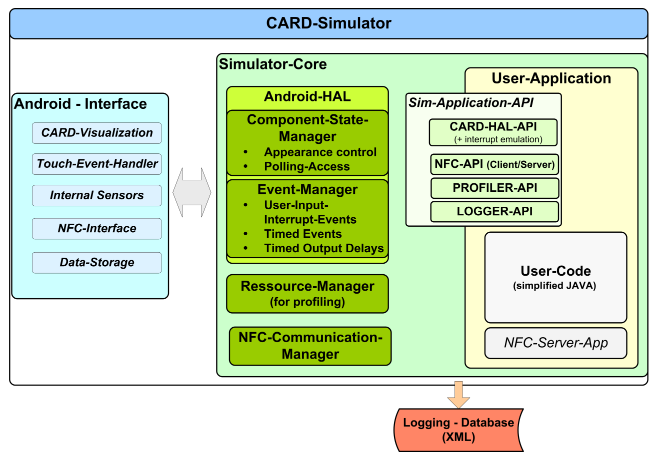

Figure 5. Software architecture of the card simulator module

based on all the properties of a selected card component. Each card component, its properties and its specific simulated behaviour (e.g., delay of the visual update) is defined in the respective class within the component library of SCUID^{Sim}. To add new components or behavioural functionality to this library, the developer simply inherits and modifies the provided component base class. All administrative support like the list of available component types and the components property page are generated “on the fly”. The complete card design can be loaded from and saved to a card library in a XML-format that can be read and edited outside SCUID^{Sim} with all existing standard XML-viewers/editors. New overlay images and even new components are easily added to the designer. Figure 4 depicts the software architecture of the card designer.

B. *Card Simulator*

The two main objectives of the card simulator are to provide a flexible framework for the development and evaluation of card applications and to simulate an user interaction with a realistic “look & feel”-experience. For creating card applications, the card simulator offers a simple API in order to access the interactive components of the simulated card. In order to keep the application as close to a real card program as possible the API allows input components to be polled and provides a simulated interrupt event handling. The concept of the API is based on the intention to shield the application developer from Android Java specific constructs in order to facilitate application code that can easily be transferred to real smart cards. In addition, the card simulator consists of a ressource manager module for simple profiling purposes as well as a flexible XML-based logging system. Since most applications for contactless smart cards imply a communication to a reader/server via NFC (ISO 14443) the card simulator offers an interface to the real NFC component of a smartphone. This way the simulated card can also be used in the targeted environment. If the real NFC component cannot be used, the framework allows the execution of NFC server applications in order to also simulate the reader functionality. A manual describing the usage and programming of applications can be found in [15]. Figure 5 depicts the software architecture of the card simulator module.

IV. Using SCUID^{Sim} for Smart Card User Interface Development and Testing

In this section, we present some examples how usability aspects of interactive smart cards can be analyzed and optimized by using SCUID^{Sim}. Typically, contactless cards follow the ISO 14443 specification, see [16]. This means that contactless smart cards usually have no battery. They are powered by the magnetic field of the terminal device. So, the available energy on real contactless smart cards for powering additional components is very limited and energy is only available if the card is in the activation distance of a terminal. The most used smart card format is ID-1 according ISOIEC 7816 [17]. This format is restricted to 85,60 mm x 53,98 mm. This makes it obvious that there is only very restricted space for additional on-card input and output components on such a smart card.

The main idea of SCUID^{Sim} is to rapidly design smart card layouts including on-card input and output devices and related applications according to this restricted dimensions and resources hand-in-hand. These capabilities can be used to design and explore new applications and to perform user studies on standard smartphones before any real smart card prototype is produced. Moreover, these capabilities of SCUID^{Sim} can be used for requirement engineering of real cards.

Our design and simulation framework, in combination with the wide availability of simulation platforms (android smartphones), made it possible for our students to quickly and easily investigate isolated usability aspects. The following brief examples of our latest studies should provide an adequate overview over the kind of analyses that can be performed with our simulation approach. Although we have performed first user studies, we do not give priority to this topic in this paper. But we include some interesting findings of these studies. The group consists of thirty unspecific persons with different levels of knowledge of information technology (in age 20 - 70: 11 female, 19 male). Obviously, there are no adequate user cross sections and no statistical relevant number of attendees.

A. Corporate Design Aspects

An important aspect in the smart card business is the combination of functional elements with the corporate design. Here the SCUID^{Sim} card designer can be easily used to design different layouts of the card body including any kind of branding as well as visual user guidance elements. These designs can be thoroughly evaluated with the simulator to explore the influence of the branding to the usability. Specifically: design of the interactive components (size, look and placement), the necessary user guidance elements and the corporate design elements to find suitable combinations in a way that the handling of the card is always clear to most of the targeted customers. Figure 3 illustrates this issue.

B. On-Card Output-Components

Most outputs on electronic devices are usually performed with optical segment- or matrix-displays. Due to the very restricted form factor of smart cards the displays itself are very restricted in their dimensions. Here physiological studies are very interesting, which were already performed in the 80's and 90's with text display formats. These display formats are rapid serial visual presentation (RSVP), in which each word is displayed sequentially at the same place on the display screen, and scrolled text, in which 13 characters are scrolled continuously from right to left or left to right across the screen. The dynamic and continuous presentation of text in this both displays requires smaller eye movements compared to usual monitors. This change in eye-movement in contrast to classical displays is believed to be responsible for higher reading rates in contrast to reading rates of usual monitors. The detailed results in the studies differ a little bit, see [18], [19], and [20], but these types of text presentation opens the perspective for displaying larger texts in restricted smart card displays.

General properties for displays are the resolution, contrast, color of the characters and background color for smart card displays, too. But besides that, the visibility of the display within the card body is an important issue. See the difference of the optical awareness of the displays between Figure 10 and Figure 12. In Figure 12, the LED matrix is included in a box and has a different background color compared to the card body. That is an important usability issue.

Besides the general display requirements and the awareness of the display, the application of the smart card display is important. Is it intended for displaying one time passwords (strings of up to 16 random characters), control instructions, short text outputs, user feedbacks, telephone numbers, long text of a few sentence, graphical symbols or others? What is the real use case of the display and which information has to be presented to the user in an adequate and readable manner? A further issue is the required speed of the display from an application perspective. So, the real display requirements are application specific and have to be tested in regard of user readability and comprehension.

1) Segmented Text Displays with Low Printing Rate: Figure 3 presents our first project in which we simulated an existing smart card prototype in order to start with ground-truth tests to determine the comparability of real and simulated cards. These cards were equipped with a standard 14-segment display component with 10 characters based on electrophoretic display technology and two buttons. The real display had good properties like high contrast and low power consumption, but unfortunately, a very low refresh rate of 1 word/update per second. It was surprising how similar the overall look and feel of the simulated card actually was. It turned out in our tests that the bigger body of the smartphone is not a big issue when it comes to usability aspects. The biggest issue of the simulated card was still the very long delay of the display. From this point on we were able to change the key parameters in the simulation: update delay, contrast, size of the display in order to find out what the minimal properties of such a display should be in order to be acceptable for most of our test subjects.

2) LED-Matrix Display for Displaying One Symbol: Complex displays, especially matrix-displays will always be a problem for smart cards. Even if the costs for the display itself can be reduced significantly, there will always be the need for display drivers, which means more complexity and costs. Also, a bigger display size would be good for usability but an increasing challenge in the integration process. Looking for alternatives, the old concept of using LEDs to build low resolution displays came to mind. LEDs are quick (even animations are possible), relatively cheap and easy to control and have the advantage that they can be mixed with other components. In that way, LED-displays can be almost as big as the card itself. On the other hand, the power consumption of LEDs is an issue and depends largely on the number of used LEDs and their brightness. In order to find out, if LED-displays could actually be a practicable alternative in smart cards, we used SCUID^{Sim} to determine the requirements regarding speed, brightness, color, size and resolution and investigated appropriate interface concepts regarding fonts, symbols, animation, user guidance and feedback.

We started with the lowest configuration humans generally can read comfortably: a 3 × 5 LED-matrix.

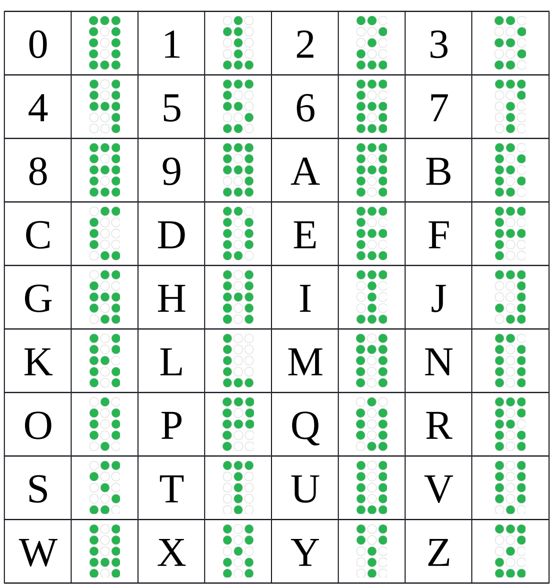

Figure 6. Used 3 × 5 LED matrix font

Figure 6 shows that our chosen font for a 3 × 5 LED matrix works quite well for digits, while on the other hand, some characters are only poorly distinguishable (like: U, W, H, M, O or Q). Lower letters worsen the problem even more. This means if only digits are processed a 3 × 5 LED matrix seems to be sufficient. But, if letters should be processed higher resolutions, like in a 4 × 5, 5 × 5, 4 × 7, or 5 × 7 LED matrix, displays are needed to achieve better character readability. That has to be analyzed.

But, which character representation should be chosen? There are no standard fonts and tests are needed to achieve distinct human readability. Figure 7 shows this difficulty for the number nine in a 5 × 7 LED matrix setting. But this holds for the whole font and has to be analyzed seriously.

Figure 7. Digit 9 in five different illustrations in a 5 × 7 font

Next, the principle illustration facilities of a 3 × 5 LED matrix display are presented. Static characters:

1) Characters, e.g., alphabet shown in Figure 6
2) Special characters, e.g., dice symbols shown in Figure 8
3) Symbols, e.g., arrows, rectangle, box, horizontal and vertical lines, etc.

Animated symbols:

4) Special characters, e.g., falling dice symbols
5) Symbols, like a falling arrow (picture frequency 200 ms), curtain up (picture frequency 200 ms), curtain down (picture frequency 200 ms) and rotary dots (dot frequency 200 ms) shown in the first row from left to right in Figure 9 and helix construction (sequentially build up dot by dot with dot frequency 200 ms), helix destruction (sequentially build up dot by dot with dot frequency 200 ms), o.k. symbol (sequentially build up dot by dot with dot frequency 200 ms) and fail symbol (sequentially build up dot by dot with dot frequency 200 ms) shown in the second row from left to right in Figure 9.

Not surprisingly, animated symbols like falling arrows and rectangles, dynamic curtains, circling dots, etc. seem to be very intelligible to the user and compensate for the low resolution to some degree. Animated symbols seem to be a suitable

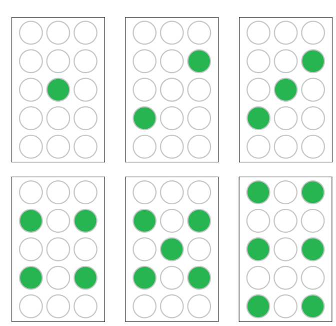

Figure 8. Digits 1 up to 6 as dice symbol

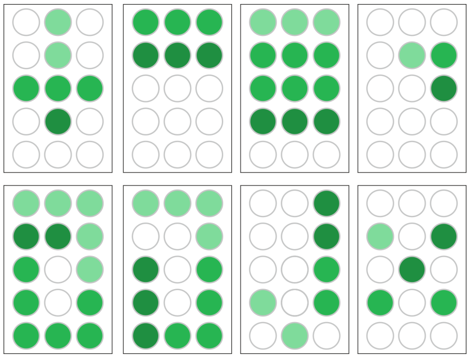

Figure 9. Animated symbols

alternative to text output to indicate card states and to give feedback information to the user.

This project showed that even a very restricted 3×5 LED-matrix display enables the presentation of a large range of characters and symbols, especially, when static and dynamic (animations) effects are exploited. Our first test results indicated that even symbols need to be chosen very carefully and have to be explained to the user in detail in order to achieve a high recognition rate. If it is possible to use symbols which are intelligible to all, they should be applied in any case.

Additionally, we tried to output short words (e.g., on, off, etc. ...) by sequentially displaying the characters of the word (rapid serial visual presentation of characters). The test users had enormous problems to read and identify even very short words depicted as sequence of letters, when they did not know the displayed word beforehand. In consequence, this approach does not seem to be suitable for displaying words. A known alternative for displaying text in a LED-matrix is of course based on scrolling text.

3) LED-Matrix Display for Srolling Text: The lower bound from a human readability perspective seems to be more or less displaying at least 2 clearly separated characters at a time. For a 5×5 font a 11×5 LED matrix display is needed to fulfil this requirement. Such a card layout is shown in Figure 10. This card is intended to perform user tests of scrolling text. Therefore, this card is equipped with a slider component and buttons for card configuration. Especially, the slider component provides user controlled repeatability of already shown text (here by swiping the finger to the left).

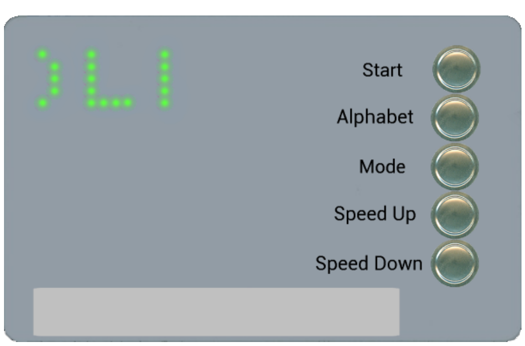

Figure 10. Layout of a smart card with 11×5 LED matrix display and slider component to perform user tests of scrolling text

Concerning the real application of the LED-matrix display numerous questions arise: Which font with which size should be used, what is the adequate speed for the displayed information and how many characters are to be displayed each time? These are specific questions, which we started to investigate in our latest studies.

Due to readability issues the majority of our testing group would prefer higher resolution fonts like 4×7 or 5×7 instead of 3×5, 4×5, or 5×5 and conceive the slider component as very helpful especially if a sequence of digits (e.g., Tel-Nr.: “0228999582”) or random characters (e.g., “H7FZ84Q2H07”) is shown. Furthermore, the impression is given that short texts (e.g., “PIN”) or longer semantic texts (e.g., “BITTE PIN EINGEBEN FUER ANMELDUNG”) can be read quite well. It turned out that a scroll speed of 52 characters per minute (cpm) instead of 95 or 38 cpm is comfortable for most of our test subjects.

If longer texts have to be displayed, scrolling text with LEDs has its limits. For such applications, rapid serial visual presentation seems to be much more adequate as we will present in the following example.

a) Matrix-Display for Rapid Serial Visual Presentation: Rapid Serial Visual Presentation (RSVP) uses the phenomenon that presenting a text word after word while keeping the centre of each word on the same spot in the display reduces the necessity for eye-movements and thereby can speed up the reading speed significantly. Studies have shown that with little training reading speeds of 1000 words per minute (wpm) and more are possible. Hence, this technique could be the ideal solution for smart cards with a one word display in applications where it could be necessary to read longer legal or technical instructions.

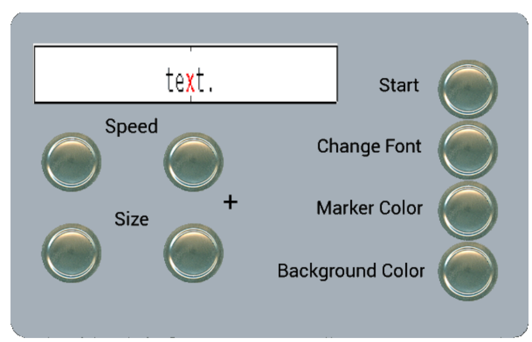

Figure 11. Layout of a smart card with a display for rapid serial visual presentation to perform user tests

Figure 11 shows an example for user tests of rapid serial visual presentation. The simulated smart card is equipped with a fast matrix display and buttons to configure the parameters of the test.

Concerning the real application of a display for RSVP, precise requirements have to be analyzed. This includes the useable font and the size, adequate speed for the displayed information, color of the central character of a word to fix the eyes to this position etc.

Result of the case study with the test group: The majority of the group enjoyed the following text configuration: largest font of 20 point instead of 16 or 12 point and a printing rate of about 150 words per minute (wps) instead of 140 wps or 270 wps, font latin instead of courier and red color for the character marker, see Figure 11. Long text (e.g., "DAS IST EIN TEST TEXT. WIR KOENNEN KURZE WOERTER UND LANGE WOERTER WIE DONAUDAMPFSCHIFFFAHRTSKAPITAEN NUTZEN. WIR KOENNTEN UNS AUCH EINEN BESSEREN TEXT AUSSUCHEN") is very well conceivable. But a problem arises if words do not fit completely into the display (e.g., "DONAUDAMPFSCHIFFFAHRTSKAPITAEN"). Then the word has to be split into parts and the parts have to be displayed sequentially. Humans can read complete words in a RSVP very well. But this is not the case if only parts of words are sequentially displayed very rapidly.

b) Matrix-Display for Graphics: The most flexible (and technically most challenging) display type for a smart card is of course a high resolution matrix display. ID-Cards or driver licence cards are typically equipped with a printed photo of the face of the legitimate user. This static photo binds the card permanently to one user and can only illustrate one perspective. A matrix display would make such a smart card reusable while also providing different views of the face and additional information about the user. This is very helpful for a manual inspection of the legitimate user. But again questions arise: which display resolution is necessary and what is the required update rate? Another application of a matrix display is to show bar codes. Bar codes can be used for optical data transmission from a smart card to a terminal for automatic data recognition and data processing. Which display size and which resolution is needed? Again, this can be easy implemented and tested with SCUID^{Sim}.

C. On-Card User Input Components

A complete on card interface needs input components as well as displays. Since in early studies almost all known button technologies with a sensible pressure point or any other kind of haptic feedback share the disadvantage to reduce the physical integrity of the smart card body, capacitive sensors are an almost ideal alternative. They are cheap and easy to integrate into a smart card and can even be used as 2D-touch sensors for complex gesture inputs - they just do not provide any kind of feedback. With such input components the aspect of visual design and user guidance is the most important issue. Also, parameters like the reading rate and resolution can be a crucial factor for the success of a specific application. Is the input component intended for the use of numbers like PINs or one time passwords (strings up to 16 random characters) or control instructions for applications? So, the real input requirements are again application specific and have to be tested in regard of user understanding and awareness.

1) Buttons: The test card in Figure 10 uses the preconfigured buttons of SCUID^{Sim}. But what is an adequate button design regarding size, distance between buttons for precise operation, color, or the user feedback in case the button is pressed? Again these design issues have to be analyzed in future studies.

2) One Character Display Input: In Section IV-B2, we have shown that a 3×5 LED matrix can be sufficient if only digits are used. Figure 12 shows a smart card with a 3×5 LED matrix for illustrating only one-character and a slider component for user input. With wiping gestures the user can scroll through the characters of the alphabet and a long touch (e.g., $\geq$ 1 second) selects the currently displayed character for input. User control and feedback (e.g., about the current position within a PIN, error or success messages, etc.) can be given with a variety of generally accepted static or even animated symbols. Such control and feedback concepts can again be implemented and tested based on SCUID^{Sim} with users under conditions that are in many aspects very similar to real cards. In [1], a case study on user authentication based on a PIN is given for the card configuration shown in Figure 12.

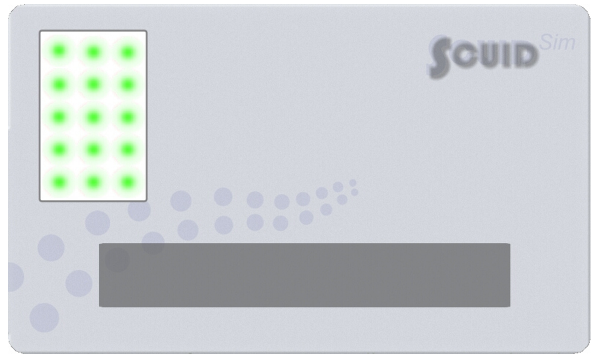

Figure 12. Layout of a smart card with a 3×5 LED matrix display and slider component to perform user inputs

3) One Character Gesture Input: In recent years, 2D finger-gestures became the favoured control concept for operating (smart-)phones. This very flexible, intuitive and therefore widely accepted input interface has also the huge advantage to be integrable into a smart card in a relatively cheap and easy way. In [6], a first real smart card prototype with such a gesture input component is presented. The sensor is a capacitive touch matrix with the size of 40 x 40 mm and is able to calculate the position of a finger-sensor contact with a resolution of 6-7 bits (which results in 64 to 128 distinguishable positions for each axis, or about 80 DPI) with a sampling rate of one point every 16ms. Gestures are recognized by the touch sensor as a time series of touch point coordinates within the active area. In that way, stroke directions and complex gestures can be detected. In contrast to a keypad a gesture interface is not restricted to input digits or characters. But it requires some form of online character recognition. On-line character recognition has to process and recognize the handwriting in real-time, ideally while the writing is still ongoing [21], in order to reduce delays. The process of character recognition can be divided into three general steps:

1) pre-processing of the input character information
2) extraction of the character features and
3) classification of the input character

Due to the primary application of smart cards, the recognition of digits (for the input of a PIN or a OneTimePassword) and control commands to operate a card are very important. But again it has to be analyzed which specific pre-processing, feature extraction and classification mechanisms are adequate for the mentioned characters and can be implemented on a very resource restricted device like a smart card. This task becomes particularly difficult if the card should detect the input of as many people as possible. In the master thesis [22],

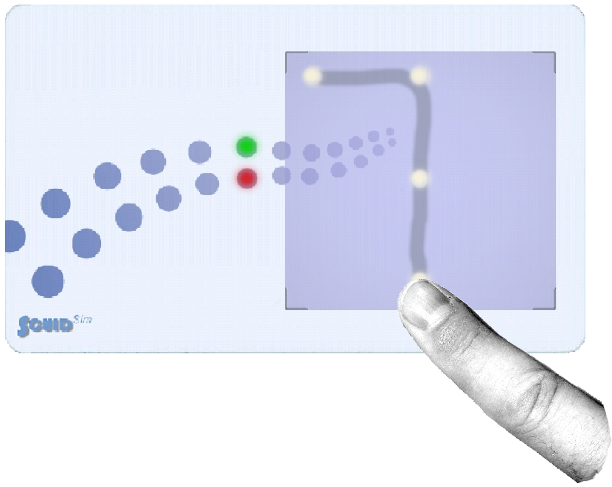

Figure 13. Layout of a smart card with a gesture component to perform user inputs

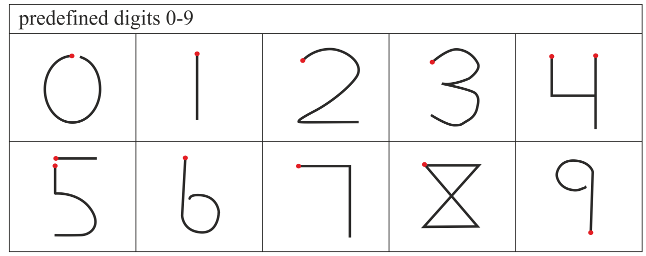

Figure 14. Predefined digits 0 to 9 for easier PIN recognition. A red dot marks the beginning of an episode

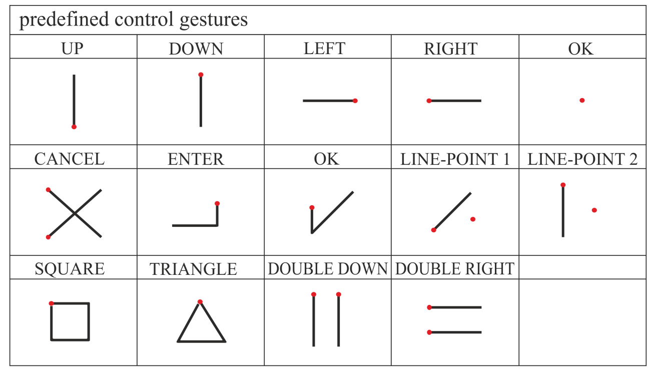

Figure 15. Predefined control gestures. A red dot marks the beginning of an episode

our simulation approach was used to develop and compare suitable gesture recognition algorithms with the focus on the reduced computing capability and memory resources of a smart card with good first results for small and middle sized sets of character and command gestures. Figure 14 shows an example for a set of predefined digits, while Figure 15 shows a subset of predefined control commands that have been used in this study. The concept of predefining the allowed alphabet for simplifying handwriting recognition was firstly introduced by Palm Inc. in their handwriting recognition software called Graffiti. The set of predefined digits [0 - 9] introduced here is similar to a simplified version of Graffiti.

Pre-Processing: The main purpose of pre-processing is to improve the input data to make the recognition process easier and more reliable, e.g., by removing irrelevant information from the sensor input which might have a negative effect on the character recognition [23]. The most important pre-processing techniques for online handwritten characters are: ”filtering”(e.g., ”noise reduction”), ”rotation”, ”size normalization” and ”filling”. Noise produced by the touch sensor may result in duplicate or erratic data points [24]. Filtering is applied to remove this kind of input data [25]. Rotation is needed to obtain an aligned representation of the inserted gesture. Users do not put in handwritten characters in exact the same size. Especially, when pixel based feature extraction is applied, this can have a negative affect on the character classification. So, size normalization is performed to obtain characters of uniform size [26]. Touch sensors have only a limited sampling rate. Depending on the user’s writing speed touch points may differ in the distance. The pre-processing technique ”filling” eliminates small gaps and holes. There exists a lot of algorithms to deal with filling, e.g., Bresenham’s line algorithm [27]. Further complex pre-processing techniques needed for off-line character recognition such as character isolation, line and word detection, etc., very often have high demands for memory and/or a capable CPU and, therefore, are not suitable in our context.

Feature Extraction: Feature extraction is the process of identifying essential characteristics in the representation of the given characters. Two classes of feature extraction methods are distinguished: structural characteristics and statistical characteristics. Usually, feature extraction methods based on structural analysis provide a high tolerance to distortions and style variations [28]. A statistical analysis extracts statistical distribution of points, e.g., in multiple zones by dividing the character image into several overlapping or non-overlapping sub-images. Next, e.g., the percentage or density of black points in each sub-image [29] is calculated. As an alternative, the distance of black points from a given boundary, such as the upper and lower portion of the character, can be used as statistical feature [30].

Classification: During classification an unknown input character image is assigned to its corresponding character class based on a metric [26]. Due to the needed memory and CPU requirements only following character recognition classes are analyzed.

- Pixel Matching: In general, pixel matching determines to which degree the pixel representation of the given input character image corresponds to a pixel representation of a character of the defined character set [31]. For large character sets pixel matching will not provide high recognition rates because the character classes overlap [32].
- Decision Tree: Decision trees are tree-like graphs constructed of multiple decisions and their possible outcome [33]
- Random Forest: Basically, random forest are a combination of several random trees. The idea goes back to Leo Breiman [34]
- k-Nearest Neighbors Algorithm (k-NN): The k-nearest neigbors algorithms calculates the distance of the

feature vector of the input character image to all sample feature vectors in a character set [35].

Due to the necessary training effort and memory requirements artificial neural networks and support vector machines (SVM) often used for such classification purposes are generally problematic and have not been investigated in this study. To correctly compare recognition results and to identify the most suitable recognition system a fixed test procedure is used. For the class of predefined digits $0 \ldots 9$ the detection rate differs from 65.4% (pixel matching) to 98.1 % (random forest). For control gestures the detection rate differs from 60.8% (pixel matching) to 90% (random forest). Moreover, estimations are given concerning memory consumption and calculation time for the mentioned feature extraction and character recognition algorithms.

Now, SCUID^{Sim} can be used to analyze and optimize feature extraction and character recognition algorithms as well as to determine the needed resources for a given target platform (CPU speed, RAM, etc.), if specific character recognition rates have to be assured. Moreover, SCUID^{Sim} can again be utilized to test the usability aspects and the general acceptance of gesture based on-card authentication concepts with a number of applicants in order to test for aspects like: acceptable size of a touch-interface on a smart card, necessary resolution and speed of a 2D-touch sensor and also visual design aspects for clear user guidance.

D. Technical Implementation of Real Smart Card Prototypes

Obviously, not all analyzed user requirements for input and output components are directly realizable in regard to the current state of technology and the energy and cost constraints. In these cases, some further investigation is necessary in order to find a feasible trade-off. We intentionally analyzed different LED matrix displays instead of modern OLED-displays or printed electronics. Contrary to bendable OLED displays and printed displays, real LEDs are available at the market and our prototypes are technically implementable today.

V. Conclusion

In this paper, we present how the tool SCUID^{Sim} can be used for rapid development and simulation of smart card user interfaces and applications. It is utilizable for early considerations of user handling requirements and overall user acceptance of user interfaces before a time-consuming and costly prototype development has to be started. Especially, card designs and application modifications are performed very quickly in software without any hardware modification. This reduces the need for development of smart card prototypes for early considerations and speeds up the whole development process.

Within this paper different specific input and output approaches are presented. First, different LED $n \times m$ matrix displays are described. Surprisingly, even a very restricted 3×5 LED matrix display enables the presentation of digits and a large range of symbols especially when static and dynamic (animations) effects are exploited. Next, a 5×11 LED matrix is presented for scrolling text. This restricted display enables reading words and short sentences although only two characters were presented at each time in this display format. Surprisingly, this form of text presentation was acceptable for the majority of test users. This is a very interesting result and we will further investigate this type of presentation. For long text passages, a display for rapid serial visual presentation is illustrated. RSVP means that words of a text are displayed sequentially at the same place at a display. Users can read even long text in this kind of display. These results show that even long texts can be displayed in very restricted displays if necessary. Regarding user inputs, we introduce a one character display input based on a 3×5 LED-matrix display and one character input based on flexible gestures. The latter enables the input of control commands as shown in Section IV-C3, too. This concept opens totally new usability concepts of smart cards if acceptable character recognition rates can be achieved. Especially, it is shown that even the analysis of feature extraction and character recognition algorithms can be supported by the tool SCUID^{Sim}.

VI. Acknowledgement

The authors would like to thank our students Anton Buzik, Tim Ludemann, Alexander Kreth and David Sosnitza for their support implementing SCUID^{Sim}, Martin Klöckner for the analysis of feature extraction and character recognition algorithms as well as our colleagues Sven Freud and Christian Wieschebrink for valuable remarks. Also thanks to the anonymous reviewers for the valuable comments.

References

[1] M. Ullmann and R. Breithaupt, "Scuid^{Sim}: A platform for smart card user interface research, development and testing," in The Fourth International Conference on Ambient Computing, Applications, Services and Technologies. AMBIENT 2014. IARIA, www.thinkmind.org, August 24 - 28, 2014 - Rome, Italy, pp. 82–87.

[2] Gerald V. Piasenka, and Thomas M. Fox, and Kenneth H. Schmidt, "Solar cell powered smart card with integrated display and interface keypad," 1998, US patent US5777903.

[3] J. Fischer, F. Fritze, M. Tietke and M. Paeschke, "Prospects and Callenges for ID Documents with Integrated Display," in Proceedings of Printed Electronics Europe Conference, 2009.

[4] Bundesdruckerei, "RFID Security Card with a One-Time Password and LED Display," 2013, www.rfidjournal.com/articles/10512.

[5] M. Ullmann, "Flexible visual display unit as security enforcing component for contactless smart card systems," in Firth International EURASIP Workshop on RFID Technology (RFID 2007), 2007, pp. 87–90.

[6] M. Ullmann, R. Breithaupt, and F. Gehring, "On-card user authentication for contactless smart cards based on gesture recognition," in Proceedings GI Sicherheit 2012, ser. Lecture Notes In Informatiks, no. 108, 2012, pp. 223–234.

[7] BDR, "Secudis Project," 2012, http://www.bundesdruckerei.de/en/684-innovative-high-security-solutions.

[8] "Thin chips for document security," in Ultra-thin Chip Technology and Applications, J. Burghartz, Ed., 2011.

[9] Taybet Bilkay, and Kerstin Schulze, and Tatjana Egorov-Brening, and Andreas Bohn, and Silvia Janietz, "Copolythiophenes with Hydrophilic and Hydrophobic Side Chains: Synthesis, Characterization, and Performance in Organic Field Effect Transistors," Macromolecular Chemistry and Physics, vol. 213, September, 26 2012, pp. 1970–1978.

[10] P. Andersson, R. Forchheimer, P. Tehrani, and M. Berggren, "Printable all-organic electrochromic active-matrix displays," Advanced Functional Materials, vol. 17, no. 16, 2007, pp. pp. 3074–3082. [Online]. Available: http://dx.doi.org/10.1002/adfm.200601241

[11] PaperDisplay, "Printed Display Products," 2013, http://www.paperdisplay.se.

[12] K. Beilke and V. Roth, "Flexcos: An open smartcard platform for research and education," in Proceedings of the 6th International Conference on Network and System Security, NSS 2012, ser. Lecture Notes In Computer Science, no. 7645, 2012, pp. 277–290.

[13] R. Ballagas, M. Ringel, M. Stone, and J. Borchers, “istuff: a physical user interface toolkit for ubiquitous computing environments,” in Proceedings of the SIGCHI conference on Human factors in computing systems. ACM, 2003, pp. 537–544.

[14] J. Borchers, M. Ringel, J. Tyler, and A. Fox, “Stanford interactive workspaces: a framework for physical and graphical user interface prototyping,” Wireless Communications, IEEE, vol. 9, no. 6, 2002, pp. 64–69.

[15] BSI, “$SCUID^{Sim}$ manual, version 0.3,” 2013.

[16] ISO/IEC, “ISO/IEC 144443 contactless Integrated Circuits Cards, Part 1-4: Physical Characteristics (1), Radio Frequency Power and Signal Interface (2), Initialization and Anticollision (3) and, Transmission Protocol (4),” 2000.

[17] ISO, “ISO-IEC 7816-X Identification Cards - Integrated Circuit Cards with Contacts,” 2011.

[18] M. C. Potter, “Rapid serial visual presentation (rsvp): A method for studying language processing,” New methods in reading comprehension research, vol. 118, 1984, pp. 91–118.

[19] H.-C. Chen, “Effects of reading span and textual coherence on rapid-sequential reading,” Memory & cognition, vol. 14, no. 3, 1986, pp. 202–208.

[20] G. S. Rubin and K. Turano, “Reading without saccadic eye movements,” Vision research, vol. 32, no. 5, 1992, pp. 895–902.

[21] A. Drissman et al., “Handwriting recognition systems: An overview,” 1997.

[22] M. Klöckner, “Gesture Based On-Card User Authentication for Contactless Smartcards,” Master Thesis, University Bochum, Bundesamt für Sicherheit in der Informationstechnik (BSI), 2014.

[23] B. Q. Huang, Y. Zhang, and M.-T. Kechadi, “Preprocessing techniques for online handwriting recognition,” in Intelligent Text Categorization and Clustering. Springer, 2009, pp. 25–45.

[24] Charles C. Tappert, Ching Y. Suen and Toru Wakahara, “The State of the Art in On-Line Handwriting Recognition,” IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 12, no. 8, August 1990.

[25] N. Arica and F. T. Yarman-Vural, “An overview of character recognition focused on off-line handwriting,” Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on, vol. 31, no. 2, 2001, pp. 216–233.

[26] L. Eikvil, “Optical character recognition,” citeseer. ist. psu. edu/142042. html, 1993.

[27] J. E. Bresenham, “Algorithm for computer control of a digital plotter,” IBM Systems journal, vol. 4, no. 1, 1965, pp. 25–30.

[28] C. Y. Suen, M. Berthod, and S. Mori, “Automatic recognition of handprinted charactersthe state of the art,” Proceedings of the IEEE, vol. 68, no. 4, 1980, pp. 469–487.

[29] D. Hunt, “A feature extraction method for the recognition of handprinted characters,” Machine Perception of Patterns and Pictures, London, England: The Institute of Physics, 1972, pp. 28–33.

[30] R. M. Brown, “On-line computer recognition of handprinted characters,” Electronic Computers, IEEE Transactions on, no. 6, 1964, pp. 750–752.

[31] J. LaViola, “A survey of hand posture and gesture recognition techniques and technology,” Brown University, Providence, RI, 1999.

[32] R. Watson, “A survey of gesture recognition techniques,” Trinity College Dublin, Department of Computer Science, Tech. Rep., 1993.

[33] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, Classification and regression trees. CRC press, 1984.

[34] L. Breiman, “Random forests,” Machine learning, vol. 45, no. 1, 2001, pp. 5–32.

[35] R. D. Short and K. Fukunaga, “The optimal distance measure for nearest neighbor classification,” Information Theory, IEEE Transactions on, vol. 27, no. 5, 1981, pp. 622–627.