Visualization of Hash-functions

Diplomarbeit von Timo Kilian aus Darmstadt
Juni 2012

TECHNISCHE UNIVERSITÄT DARMSTADT

Fachbereich Informatik
Theoretische Informatik - Kryptographie und Computeralgebra

Visualization of Hash-functions

Vorgelegte Diplomarbeit von Timo Kilian aus Darmstadt

1. Gutachten: Prof. Melanie Volkamer
2. Gutachten: Prof. Johannes Buchmann

Tag der Einreichung:

Erklärung zur Diplomarbeit

Hiermit versichere ich, die vorliegende Diplomarbeit ohne Hilfe Dritter nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, den 26.06.2012

(T. Kilian)

Contents

List of Figures

Summary

There are several cases in which long sequences of characters have to be compared. This thesis features the use cases of browser certificates and the Helios voting system, where 160-bit SHA1-values have to be compared. In this thesis a scheme will be developed to exchange character strings with pictures to make it easier to find non-matching values/pairs. There are different schemes in existence which have the disadvantage of only 25 bits of entropy per picture and bad scalability. The final scheme called CLPS is able to encode 60 bits of entropy by generating a picture with objects with four different attributes: colors, letters, patterns and shapes. This scheme is faster than the equivalent Base32-scheme with the same success rate of approximately 97%. Several studies have been conducted which helped to continuously improve the scheme. The study in the last chapter with 45 participants has shown the scheme is not useful for the Helios voting system. People are not able to remember the high entropy of 60 bits over a longer duration. The success rate for the scheme is only at 80% at a simulated Helios environment.

1 Introduction

By ‘surfing‘ on the World Wide Web you will eventually come upon websites starting with ‘HTTPS://‘ at the address bar. Those websites are communicating with the corresponding server over a secured and encrypted connection. On newer browsers this is sometimes marked with a green pictogram of a closed lock. Each website, and thus the underlying server, claiming to be the website displayed in the address bar has to prove this statement. This is done by certificates. A certificate states the website shown is the one listed in the certificate and has not been tampered with. A website may of course be hacked, altered or modified to do harm. A certificate only guarantees the identity of a website/server. If someone uses web pages like web shops or homebanking portals where confidential data must be entered, the user needs to be 100 percent sure that the website displayed in the browser is the website it claims to be. He could do so by comparing the fingerprint of the certificate of that website with the certificate of the company, which was sent to him by email or postal mail. He would have to do that every time he uses that website. Most companies don’t even offer that service. The user has to rely solely on the information given to him. A fingerprint is a short sequence of 20 bytes which is derived from the public key of the certificate by a hash or compression function. This fingerprint makes it easier to compare it to another one because of the short length. But still no one actually does it. For 20 bytes, 40 characters have to be compared, which is time-consuming and error-prone. Nearly everyone trusts the browser or is just careless.

Homebanking and web shops are used extensively, confidential data is exchanged without much ado. On the contrary, Internet voting is publicly well-known but seldom implemented and used, because of the security issues and fears that arise with the usage of the Internet. Voting machines currently in usage are black-box systems, which need to be trusted, but give no verifiable proposition, if the votes have been cast as intended, stored as cast and finally tallied as stored. The voter has to trust the machine has not been tampered with and his vote will be used as he intended. Several successful attempts to compromise a voting machine have been made [1] with an attacking technique called return-into-programming (based on buffer overflows) with the result of stealing or altering votes. Also Man-in-the-Middle attacks have proven successful [2]. Those are software-attacks so there is not even the need of physical access to the machine. But the hardware of a voting machine could also be altered, be it as design errors or by intention. This is of course a fiasco regarding the trustworthiness of electronic voting.

There are a few new promising concepts of voting schemes. One of them called *Helios* is a cryptographically verifiable voting scheme. But with all of them there is one major goal to achieve: User-friendliness. It has to be taken into account that most users will not be technically experienced. They will not understand which security problems might arise and how they can verify, or more blatantly, that they have to verify their vote has been dealt with as supposed. They want to rely on a secure and trustworthy process without any technical issues. As a study [16] on the improved Helios Voting System has shown, the interface is easy to use but people have difficulties understanding the concept of verifiability. They are most of the time not motivated to verify their vote. In a test run, only 59% of the participants have verified their vote before actually sending it. In a second test run where it was strongly suggested to verify, now 94% of the participants did do it. Surprisingly, even with an urgent call on verification, a remainder of 6% did not do it. It is shown later in this study that most participants did not understand the reason of re-encryption and thus a changing verification code.

The fingerprint of a browser certificate and the verification code of the Helios voting system have one thing in common, long chains of numbers and letters need to be compared. This is time-consuming, error-prone and simply not up to date anymore. The goal of this thesis will be to exchange the characters of a hash string by pictures. A lot more information can be encoded into a picture, pictures are 'better' to look at by people with no technological background and differences in pictures can be detected faster and easier. In the related work section (chapter 3) there are schemes which encode values to pictures. But the maximum amount of encoding-entropy of all these schemes is 25 bits per picture, which is not enough by far regarding security. The goal of this thesis will be to reach at least 50 bits (double the amount) of entropy while maintaining a low error rate. A few ideas ideas will be tested in chapter 6. A working scheme will be developed and evaluated by several user studies in chapters 7 to 9.

2 Background

In the introduction chapter two use cases for visualization of hash values are mentioned: Fingerprints of browser certificates and the Helios voting system. These will be shown and analyzed in detail in the application areas chapter. Chapter 2.2 will give an insight how hash functions work and which security problems may arise. The last chapter shows how chains of numbers are displayed and why this is not satisfactory in terms of usability and speed.

2.1 Application areas

As an example the certificates of the Google website are used here. The certificate was issued by a root authority to the Google company and must be kept secure by the owner. If an attacker is able to fake or steal a certificate, he can claim to be that particular website. When using Google this might not be a problem but if the certificate of a banking portal was compromised, this could be disastrous. An attacker might be able to lure a victim to a prepared website which claims to be and looks similar to the original banking homepage and the victim would enter confidential data without even noticing. In case of the Google certificate SHA1 [3] is used.

Figures 1 and 2 display the certificate of the Google website in detail.

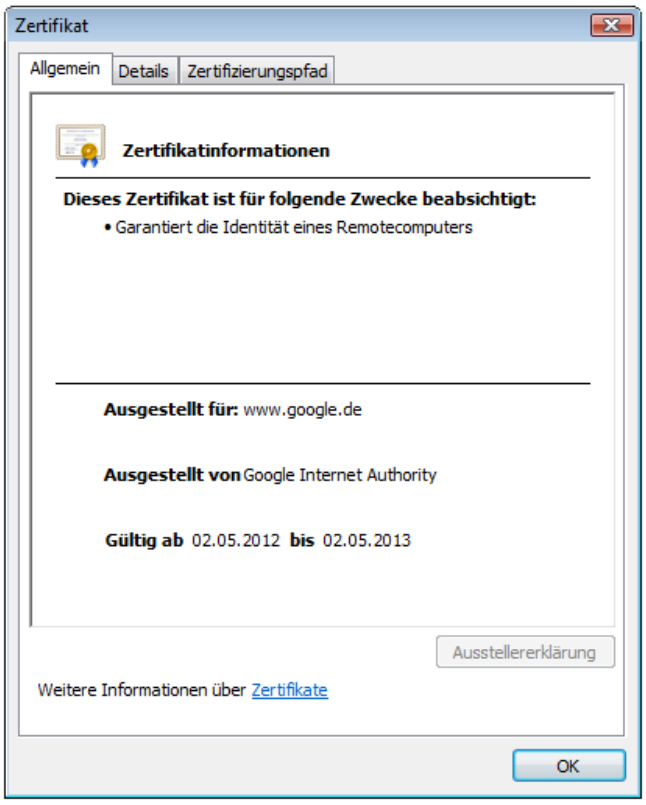

Figure 1: Certificate of the Google website

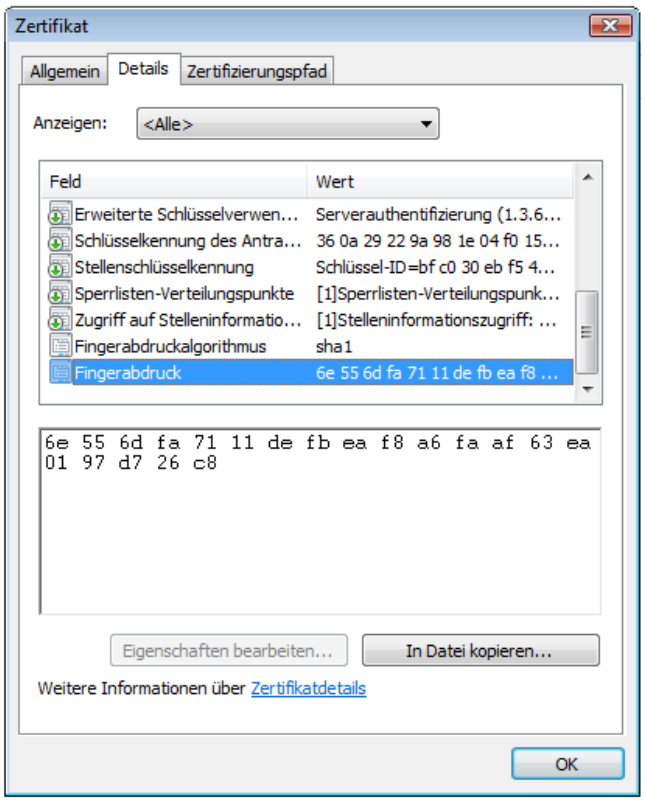

Figure 2: Fingerprint of the Certificate

The Helios Voting System [7] is an open-source web-based e-voting system. It offers cryptographically verifiable voting with SSL-secured communication and voter authentication. Public observers can act as auditors and verify the voting process and that all votes are properly tallied, while voter secrecy is provided. This is called *universal verifiability*. Helios can be used by

everyone to set up an election, compute a tally and show the outcome which can be audited by an observer. The implementation of Helios guarantees, even if Helios becomes compromised, that the integrity of an election can be verified. The Ballot preparation and casting is based on Benaloh's *simple verifiable voting protocol* [9]. The voting process works as follows (simplified):

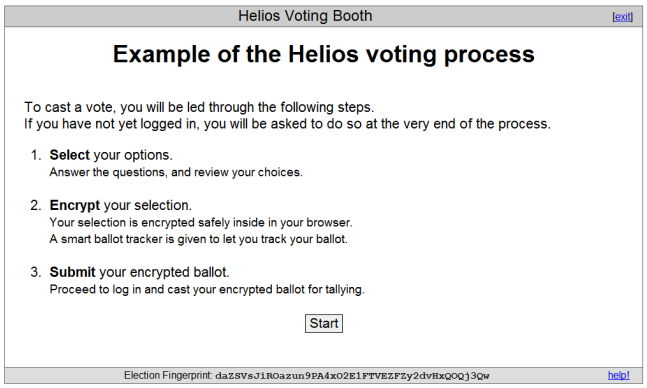

Figure 3: Helios - Information page

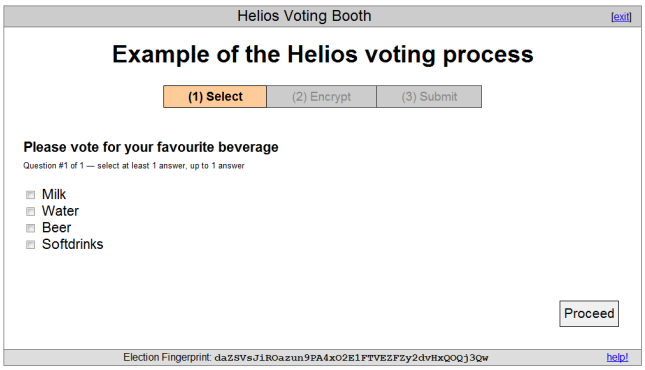

Figure 4: Helios - Voting page

Figure 3 shows the information page, where the voting process is described in detail. Voters fill out a ballot for an election by choosing their answers to one or more given questions (figure 4).

Figure 5: Helios - Review the ballot

Figure 6: Helios - Selection to cast or audit the vote

After the voting is done, the ballot can be reviewed as can be seen in figure 5 and the encryption process can be started. Figure 6 shows the possibility to either cast or audit/verify the vote. If the voters opt to verify their vote, an independent entity receives the hash value and plain text and generates the hash value again on its own. The voters can check if the hash values match and the vote was correctly encrypted. They can encrypt and verify their vote as often as they want to until they are satisfied. If a vote was verified, it needs to be re-encrypted to ensure voter privacy. Otherwise an entity might be able to link a vote to an actual person.

When the voters are satisfied, they need to authenticate and the encrypted vote will be cast (figure 7). At the end of the election period, the voters can visit the election web page where all verification codes are displayed to confirm their vote has been stored as cast. It is crucial to note that one cannot cast a verified vote. So the verification code of the final vote can be used to verify, if the vote has been stored as cast. An user guide to Helios can be found here [4].

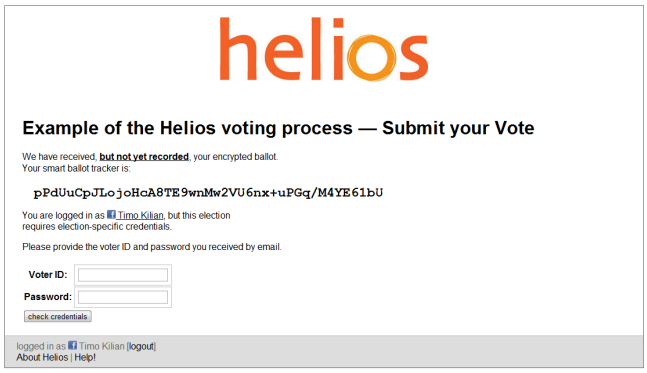

Figure 7: Helios - Authentication

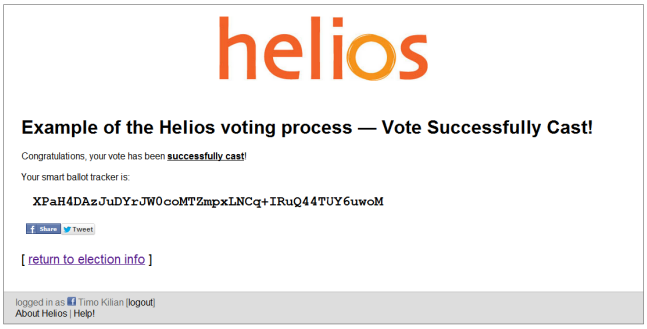

Figure 8: Helios - Success notification

2.2 Hash functions

A hash function is an algorithm that maps a large input of data to a smaller output of fixed length. The output is called message digest or hash value. A hash function is always a one-way function, meaning the input cannot be re-computed by using the output data. If, as an example, a book is taken as input, every letter of every page feeds into the input and a much smaller output, usually 40 letters, is computed. If a single letter on any page changes, the resulting output value also changes. In a 'good' hash function, a minor change on the input has a huge impact on the output resulting in a completely different value. Since the input can be nearly infinite and the size of the output value is fixed, it is possible that two different inputs produce the same output value. This is called a *collision*. A successful attack on a hash function would be finding values leading to a collision.
This leads to the three desired properties of a hash function $h(x) = y$:

- Collision resistance: It is computationally 'difficult' to find two distinct inputs x and x' which will compute the same output $h(x) = h(x') = y$
- Preimage resistance: It is unfeasible to find an input x which computes to $h(x) = y$ for any pre-specified output y
- Second-preimage resistance: It is unfeasible to find an input x' with $h(x) = h(x') = y$ for any given input x

The most commonly used hash functions today (2012) are SHA-1 - *Secure Hash Algorithm One* and SHA-2. In 2005, a theoretically mathematical weakness in SHA-1 has been found, so it is most likely SHA-1 will soon be replaced by SHA-3 or a better algorithm.
Details on the functionality of SHA-1 can be found here [3].

The length of the output (digest) of SHA-1 is 160 bits. An attacker can always find two corresponding input- and output messages by brute force in 2^{160} evaluations. This is called a preimage attack. Because of the birthday paradox [19], a collision can be found in $2^{160/2} = 2^{80}$ evaluations on an average. So 160 bits are thought secure, because it is impossible to do that much evaluations with today's computation power.

2.3 Text representation

The Helios-system, like any web browser, offers a fingerprint of a certificate represented by a hash value of arbitrary length. For the hash value to be secure (see chapter 2.2), it has to have a length of at least 160 Bit. If a user wants to compare two hash values, he needs to compare 20 characters two times - two characters representing one *byte* or 8 *bits*.

A short example is given to show how it works: the random SHA1 hash value ‘b8e457129a7f0159aa569137023ccef059cdfea3‘ is in hexadecimal notation and of 160 bits length. A given value of 160 bits length means, one has to compare ten times four digits resulting in a comparison of 40 digits. If this value is converted to a Base32 [15] value, which is used later on, the output would look like this: ‘XDSFOEU2P4AVTKSWSE3QEPGO6BM43PVD‘. So, still 32 characters have to be compared. This is time-consuming and error-prone.

The comparison of the values is crucial because the user needs to verify, in case of Helios, if his vote was encrypted, cast and stored as he intended. In case of a server-certificate an user wants to ensure, that the server is really who it claims to be. Minor changes in the values could mean an attack has happened and the server/system could be compromised. People are not good at remembering or quickly comparing many letters or symbols. They are prone to make errors on similar digits like ‘8‘ or ‘0‘ and on flipped characters [18].

There has also been an approach (see chapter 3) using an encoding of numbers to plain text. But even here the problem of bad comparisons arises. As an example the words ‘clam‘ and ‘calm‘ which have ‘a‘ and ‘l‘ flipped are hard to distinguish. In a test run contestants had to compare a lot of sentences containing similar words, which lead to high error rates.

3 Related work

There has been a lot of effort in visualization techniques lately. There have been propositions of textual [10], audio [13] and visual representations of hash values. In this thesis, the focus will be on visual techniques. A few interesting approaches will be presented which show the possibilities so far. There has been a study [14], which covers nearly all approaches of visualizing hash functions as of today and aims on finding the best scheme. The nine different schemes will be presented in the following chapters. Afterwards the results of the study will be presented and analyzed.

3.1 Random Art

Random Art was developed in 1996 by Andrej Bauer and is based on the idea of automatic generation of images. It uses a binary string as a seed for a random number generator. A function generating the image maps each image pixel to a color value. The pixel coordinates reach from -1 to 1 in x and y dimensions. The image resolution defines the sampling rate, thus an image sized 100 x 100 pixels needs 10.000 samples taken. The function is constructed by choosing rules from a fixed grammar. Since the hash value feeds the pseudo random generator, two identical inputs will give two identical pictures as output. For a listing of the algorithm, take a look at [8, page 4]. The idea in this work [8] was to generate visual fingerprints of binary fingerprints of root keys to show if they differ.

Even though *Random Art* is capable of encoding large amounts of entropy (e.g. 160 bits for SHA-1), an analysis has been made [14] to see if the perceptual entropy can reach the actual encoded entropy. The tool *PerceptualDiff* [20] was used to measure the perceptual difference of two images. After randomly generating 3700 pictures with *Random Art*, 50 pictures were detected by the tool to be similar and after manual inspection, four nearly identical pictures have been found. An estimation was given of the perceptually different pictures *Random Art* can generate: $3709^2 \,/\, 4 = 3439170.25 \approx 2^{21.71}$ bits. And only a 8.6% probability of carrying more than 23.71 or less than 19.71 bits.
The pictures give a demonstration on how different values look like with *Random Art*.

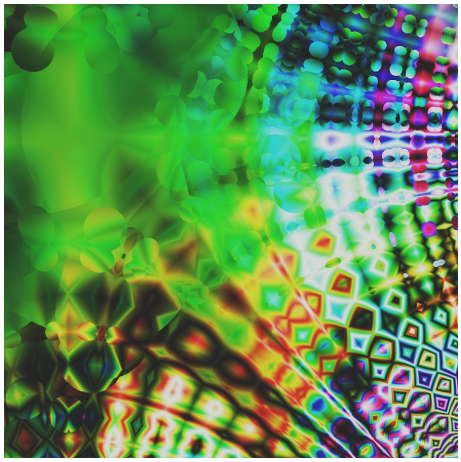

Figure 9: Random Art example 1 (Taken from [5])

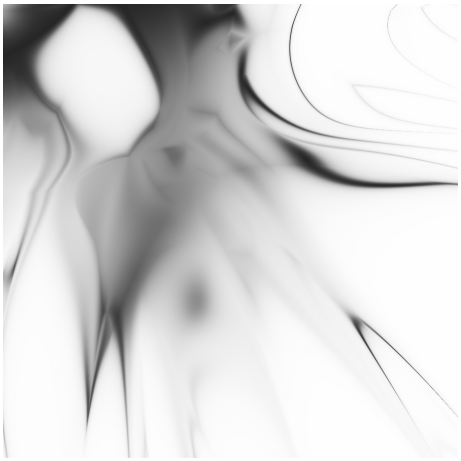

Figure 10: Random Art example 2 (Taken from [5])

3.2 Base32, Base64, Base-N

Base-N [15] is used to encode a byte into a printable alpha-numeric character. Since a byte can have a value of 0 to 255, not all of them are printable (e.g. value 13 corresponds to *carriage return* and 20 to a *space*-character and so on).
Encoding works as follows. A Base32 alphabet consists of 32 symbols (see figure 11). A group of five ASCII bytes (five times 2^8 sums up to 40 bits) is taken and converted to a group of eight Base32 characters. If the input consists of less than five bytes, zero-bits are attached to the input stream.

The advantages of Base32 are the small alphabet and no usage of similar characters like '8' and '0' or '1' and 'l', which could lead to mistakes by confusing those numbers. However, a 160 bit hash value as input string leads to an output string of 32 characters, which are also unrelated to each other and thus will be difficult to compare or remember.

Base64 and Base85 work respectively. Base64 uses 64 characters to encode the input leading to an output length of 28 characters. Base85 is a proprietary scheme. Both schemes will not be featured here.

Value	Encoding	Value	Encoding	Value	Encoding	Value	Encoding
0	A	9	J	18	S	27	3
1	B	10	K	19	T	28	4
2	C	11	L	20	U	29	5
3	D	12	M	21	V	30	6
4	E	13	N	22	W	31	7
5	F	14	O	23	X		
6	G	15	P	24	Y	(pad)	=
7	H	16	Q	25	Z		
8	I	17	R	26	2		

Figure 11: Base 32 encoding table (Taken from [15])

3.3 English words

An approach which seems to be simple and obvious is the usage of a language with no special characters like accents or umlauts. In this case a dictionary of english words is chosen and the hash value is encoded using words of this dictionary. A large dictionary contains a lot of similar words like 'moon' and 'moan' or 'ball', 'bull' and 'bill', which are difficult to distinguish if read quickly. Oxford dictionaries states there are approximately 177.000 english words in usage [12], which leads to an entropy of 17-18 bits per word, if all of the words in the dictionary are used.

3.4 Flag, T-Flag, Flag Extension

Flag is a simple visual hash representation using colors in striped form which resemble a flag. A flag consists of four stripes with a possible 2^n colors each. Because humans have difficulties on detecting minor variations in colors, the amount of different colors is limited. Also the orientation of the flag is not marked which could lead to a 'wrong' picture when rotated on a mobile or handheld device. *Flag* was proposed by Ellison et al [11].

T-Flag [17] extends *Flag* to eight strips of eight colorblind proof colors. This leads to a fixed entropy of 8 * 3 bits = 24 bits.

The authors of the study [14] proposed *Flag Extension* which introduces shapes into *T-Flag*. The amount of usable colors is limited, older people might have difficulties distinguishing color variations and color blindness is also an aspect that needs consideration. To increase the amount of entropy of a picture, shapes are added. An image consists of four colored strips (eight colorblind proof colors usable) and eight different shapes. This sums up to an entropy of ($2^3 + 2^3$) * 4 = 24 bits.

Figure 12: Flag example (Taken from [14])

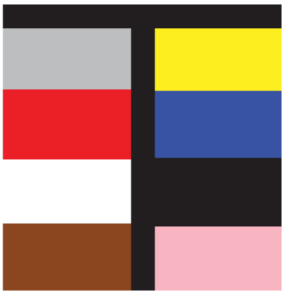

Figure 13: T-Flag example (Taken from [14])

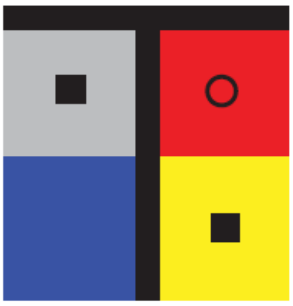

Figure 14: Flag Extension example (Taken from [14])

3.5 Asian characters

Three asian-character-based schemes are proposed by the study [14]. Asian characters contain a much higher entropy than standard ASCII characters.

The subset of commonly used traditional and simplified chinese characters consists of nearly 10.000 entries. This means an approximate entropy of 13.3 bits per character. In this study two characters of this subset are chosen resulting in an entropy of 26.52 bits.

The Korean character set called Hangul contains roughly 11.000 characters, which leads to 13.5 bits of entropy per character. Even though the characters are assembled of smaller fractions called Jamo which often lead to meaningless combinations, they can be used to encode hash values. However, native Korean speakers might be irritated by the meaningless construct and give a false-negative answer. Two characters are chosen for the study with an combined entropy of 26.90 bits.

The third scheme consists of fractions of Hiragana, one of the Japanese phonetic alphabets. One unit offers an entropy of approximately six bits. For the study four Hiragana characters are taken which hold an entropy of 24.52 bits combined.

There exist a few concerns with those 3 schemes. First of all, all characters need to be stored as pictures. This might be an issue with mobile devices and slow network connections. The Japanese Hiragana however is part of Unicode. Also Asian characters - specifically Chinese - are complex characters which sometimes only differ in a missing dot or line. This could be difficult to detect by non-native speakers.

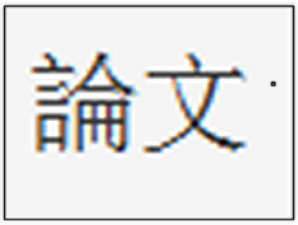

Figure 15: Chinese character example

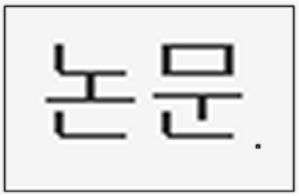

Figure 16: Korean character example

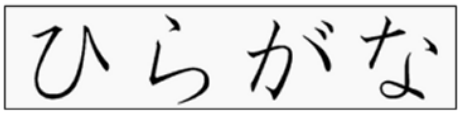

Figure 17: Japanese character example

3.6 Study on nine visualization schemes

This study has participants from the US, Japan, Korea and Taiwan. This should show if the geographical background and knowledge of the english and asian languages has any impact on the speed of recognition and the overall results. Gender, age and language abilities are taken into account.

The study is designed to have each scheme containing the same amount of entropy. Due to the composition of some schemes (*Base 32* has an entropy of 5n, so only a multiple of five is possible), each scheme can contain from 22 to 28 bits of entropy/information.

For the study three pre-generated sets of two pictures of each scheme are taken:

- One picture is equal to another. This is called an easy pairing.
- One picture is obviously different to the other. This is also an easy pairing.
- One picture is similar to the other, but has been altered in some way:
  For *Base 32* and *English words* a similar word is generated by exchanging two adjacent letters (e.g. 'moons' and 'moans').
  For *Random Arts* a perceptual similar picture is used. This means one picture looks like the other but has been altered by a small fraction.
  For *Flag* and *T-Flag* the intensity of the color of one strip is in-/decreased by one level.
  For *Flag ext.* the colors or shapes of one block are changed.
  For the asian languages minor changes like left out lines or dots are used.
  All those pictures are called hard pairs.

One of the pictures is rotated by a random angle of ± 30 degrees to simulate misalignment. Each participant has to compare 27 pairs of hash visualizations (9 schemes and 3 pictures each). The order of the schemes are random. After comparing the 3 pairs of one scheme, the participant encounters the next scheme, but all schemes only once. The pairs of pictures are pre-generated and shown with a fixed probability:

- probability $\frac{1}{2}$: an easy pair of 2 identical pictures
- probability $\frac{1}{4}$: an easy pair of 2 obviously different pictures
- probability $\frac{1}{4}$: a hard pair of 2 distinct but similar looking pictures

The participant has to decide whether the pictures are identical or not by clicking on a 'Same" or 'Diff' button. The answer and the time needed to compare the pictures are recorded.

Table 1 shows the entropy of each scheme. The entropy of Random Art is an estimation as explained in the Random Art chapter.

Scheme	Base32	English	Chinese	Japanese	Korean	Rand. Art	Flag	T-Flag	Flag Ex
Entropy	25	27	26.52	24.52	26.90	19.7-23.7	24	24	24

Table 1: Table of the entropy of each scheme

3.7 Findings of the study

The study has found no impacts on time or accuracy regarding gender or age of the participants. On hard pairs, the youngest age group was slightly faster than the 26-40 and 41-60 age groups.

As can be seen in table 2, the accuracy on easy pairs is really high with an average of 98%. The average accuracy on hard pairs drops from a 94% with Random Art to 50% with Flag which is just the probability of guessing or pressing the same button for all pairs.

Random Arts shows very good results but has the disadvantage of needing high computation power and the maximum entropy of about 24 bits per picture.

The language related schemes do not offer a very good performance on hard pairs.

The scheme with the best results and usability combined is Flag Extension as proposed by the authors of the study.

Base32 offers good results because people are accustomed to characters of the alphabet and are fast at reading and comparing them. It is also very easy to implement and can be used even on low-level handheld devices. In chapter 7.2 a second study on a Base32 related scheme was conducted and came up with even better results. So Base32 is highly recommended.

Category	Avg. accuracy on easy pairs	Avg. accuracy on hard pairs	Min. accuracy on a hard pair	Avg. time for easy pairs(s)	Avg. time for hard pairs(s)
Random Art	98%	94%	60%	4.77	3.21
Flag Ext.	98%	88%	62.7%	3.93	4.02
Base32	97%	86%	71.1%	3.39	3.51
T-Flag	98%	85%	70.8%	3.99	4.00
English words	94%	63%	42.5%	4.80	4.63
Chinese	97%	59%	51.4%	4.89	5.01
Japanese	98%	57%	39.1%	4.64	5.07
Korean	98%	54%	25.7%	4.61	4.92
Flag	98%	50%	15.1%	3.70	4.28

Table 2: Summary of all schemes, sorted by average accuracy on hard pairs

The different schemes have a few things in common which could be improved. Both Base32 and Flag Extension gave very good results but have a low entropy. This low entropy of approximately 25 bits can not easily be extended because in case of Flag Extension the number of colors and shapes is limited. Both schemes combined could give a much higher entropy while maintaining the accuracy.

4 Visualization tool

A small HTML5 program using HTML and javascript combined was used to display the hash function. Web pages have the benefit of an existing GUI - the browser - and the usability on a wide range of devices. A web page can be displayed on a PC as well as on modern smartphones or electronic kiosks. HTML5 introduces the canvas object. The canvas can be painted on using different geometrical functions like lines, circles and rectangles. Even though a three-dimensional space could be used, the 2D canvas is a better option because of the bad implementation of 3D routines by now. The disadvantage of using HTML5 is the need of a modern browser. Also the implementations differs slightly from browser to browser. The best usability is offered by Google's Chrome (Version 17), the worst by Internet Explorer (Version 9). This is because of not all functions have been implemented in IE9. The result is the same on all browsers regarding colors, size and looks. So it really does not matter which browser is used, the program has just to cover up for different functions. The web page offers the functionality of entering a hash value and generating and displaying the result. The web page lets the desired hashsize in bytes be entered ranging from six bytes (24 bits) up to 21 bytes (168 bits). Since SHA-1 is of 160 bits length, the program covers up to the maximum size. The *Random hash* button generates a random hash of the desired size. It is also possible to enter a hash value 'by hand'. The *Compute* button then starts the computation and displays the result. Beneath the menu there are two pictures side by side. Minor changes in the hash value can be easily observed by computing the different hash values and displaying them side by side.

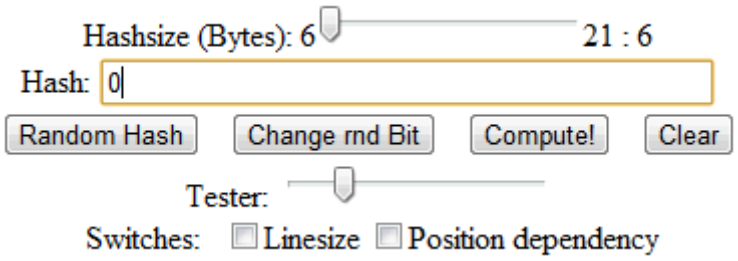

Figure 18: Tool - menu

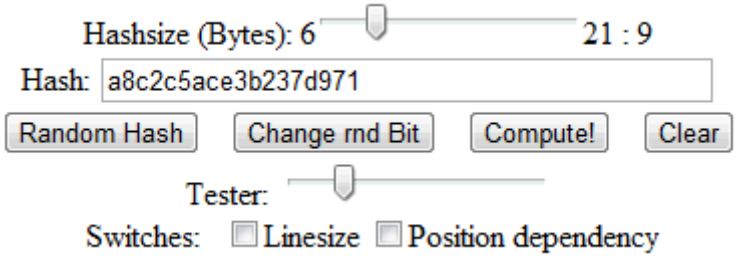

Figure 19: Tool - menu with filled-in values

Figure 20 shows a later version of the tool where saving the pictures as *JPG* or *PNG* is possible. The buttons *SaveL* and *SaveR* have been added to save the left and right pictures respectively. The hash size has been set to 64 bits in the latest version. The right picture shows one difference to the left, a black heart instead of a blue. With the *Change rnd bit* button, only one random bit in the hash value is changed and the result is displayed in the second picture.

This tool is able to produce similar pictures of arbitrary hash values and save them for later usage in a fast and easy way. A lot of pre-generated pictures will be used in the studies in the next chapters.

The source code of the tool can be found on the enclosed CD.

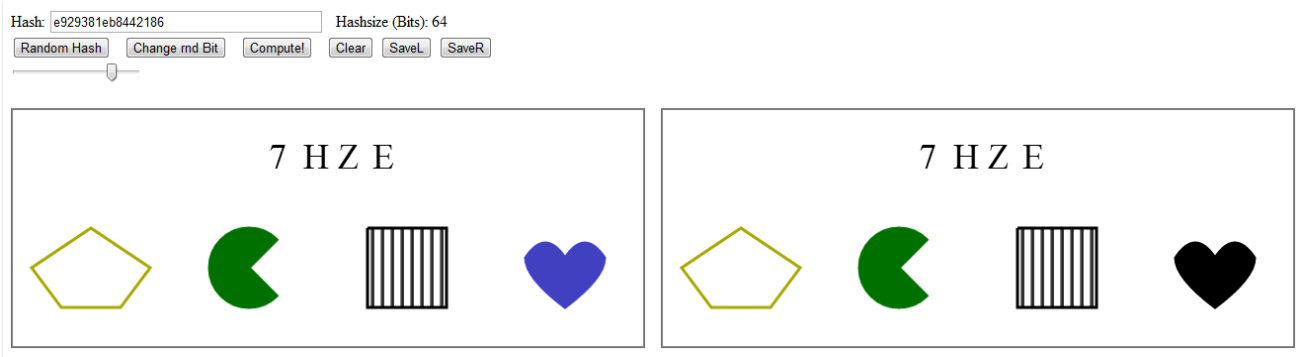

Figure 20: Tool - two pictures side by side

5 Tools for the studies

There have been two options to do the studies. Either using a webpage or using a selfmade tool. A webpage has the benefit of easy accessibility. People from all over the world can use a browser to get on a given website and do the study there. Also the browser provides a front-end and a graphical user interface (GUI), which is easy to program with HTML and Javascript. But there are three major issues which kept me from using a webpage.

The first being the loss of control. You can never be sure what the participant does while doing the study. He [1] might be trying to cheat or he could reload the webpage if he made an error. Or perhaps he can ask friends to help him on the study to gain a benefit.

The second problem is the issue of time. To get usable results, the times each participant needs for a particular part of the study have to be taken. With websites, there are problems like slow connections and unpredictable behaviours on different kinds of browsers and computers. Although it is possible to do the timing with Javascript on a webpage, the values might not be trusted.

The third problem is the accumulation of data. A server-side script language like Perl or PHP needs to be used to get the values collected on the website to a database. Afterwards the data needs to be extracted and analyzed.

There are tools online which do automated studies [6], but they do not fulfill the special needs of this thesis.

Another more uncommonly approach will be used for the studies in this thesis. A *Powerpoint* presentation has been developed, which will display the study and automatically put the results into an Excel datasheet. This is done by an underlying *Visual Basic for Applications* script (short: VBA). In the Excel sheet, the data can be sorted and analyzed by another VBA script. This approach makes it easier to maintain control of the whole execution of the study. A participant can be supervised during the process and possible errors can be solved directly. The usage of a script gives reliable values which can be utilized later on. The setup of the presentation remains the same for all studies and will be shown in detail.

The first page of the presentation gives an explanation of the study and what the participant has to do. It also shows how the pictures are built up and which different cases may arise. On the next page there is a demographic survey where people needed to specify age, gender and activities in graphic design or similar experiences. Also the grade of knowledge of computers and software was asked by answering one of these four questions:

- I need help to install new software on my computer
- I install new software on my computer all by myself
- Others ask me to help them installing software on their computers
- I do not install software on my computer

Finally by pushing the *Start* button, the interactive part of the study starts. The participant will be shown one of 25 slides with two pictures each. He then has to decide if the pictures are either identical by pushing a *green* button (true) or different by pushing a *red* button (false). This is shown in figure 21.

[1] Throughout the text '*he*' is used when speaking in third person. This was done to make the text easier to read

Figure 21: A random presentation - One slide with identical pictures

The 25 slides are completely randomized to counter effects like adaption to a certain type of slide. When the participant has finished all of the slides, he can rate the explanation at the beginning and how he got along with the study on an evaluation page. Afterwards he will be shown the final score respectively how many right answers he gave.

The hash values of the pictures along with the answers (true or false) and the times for each slide are put directly to an Excel sheet. After the participant has finished the Powerpoint presentation, all the data of the Excel sheet will be saved to a file on the local harddisc. When the next participant starts the presentation, the data will be appended to the file, so a virtually infinite amount of participants can do the study. The Excel sheet offers a few VBA functions to sort and order the accumulated data. It is also possible to compute the amount of errors per slide or per type of picture. Another function computes the minimum/average/maximum time of one or all slides. Nearly all of the values shown in the various studies are computed by these VBA functions. They took some time to develop but can be reused in later projects and offer a fast and reliable way to process raw data.

6 Ideas of visualization of hashes

In this chapter a few ideas for a new scheme to visualize hash values will be implemented and tested. The first approach in chapter 6.1 is called ‘Line-Matrix‘ and is able to display 168 bits of entropy, which was the goal to achieve at the beginning, but is useless with a high number of bits because of the complexity of the picture. The second approach in chapter 6.2 features the same entropy but introduces objects which replace the dots of the first approach. Still the picture is too complex and cannot easily be compared to another picture if certain conditions occur.

6.1 Line-Matrix

The approach is called Line-Matrix because it features a matrix, where dots are connected by lines. A matrix can hold a lot of entropy because the position of one entity alone uses one byte for the positioning information. A prior version of the tool shown in chapter 4 is used to generate pictures of hash values.

6.1.1 Description of Line-Matrix

The algorithm works as follows. The whole hash is divided into byte-long fractions. A byte is taken and a position in a 16 x 16 matrix is computed. The position is derived from the byte value where the first four bits are used for the x-axis and the second four bits are used for the y-axis.

At the computed position, a dot is drawn. A curved line is drawn to the next position and so on until all bytes of the hash value are used up. The lines start with a black color and get to a bright red color to make it easier to follow the path from the first dot to the last. The curves are right-handed all the time. Figures 22 and 23 on the next pages show matrices with six bytes and 12 bytes respectively.

The graph starts with the first byte (hex value: 3f) in row four and column 16. It ends in row 11 and column 15 (hex value: ae). The colors of the lines start with black and fade to a dark red. The values of the hash are well distributed resulting in an easy-to-compare picture. This is the hash value of the visualized picture: 3f4280ec1cae.

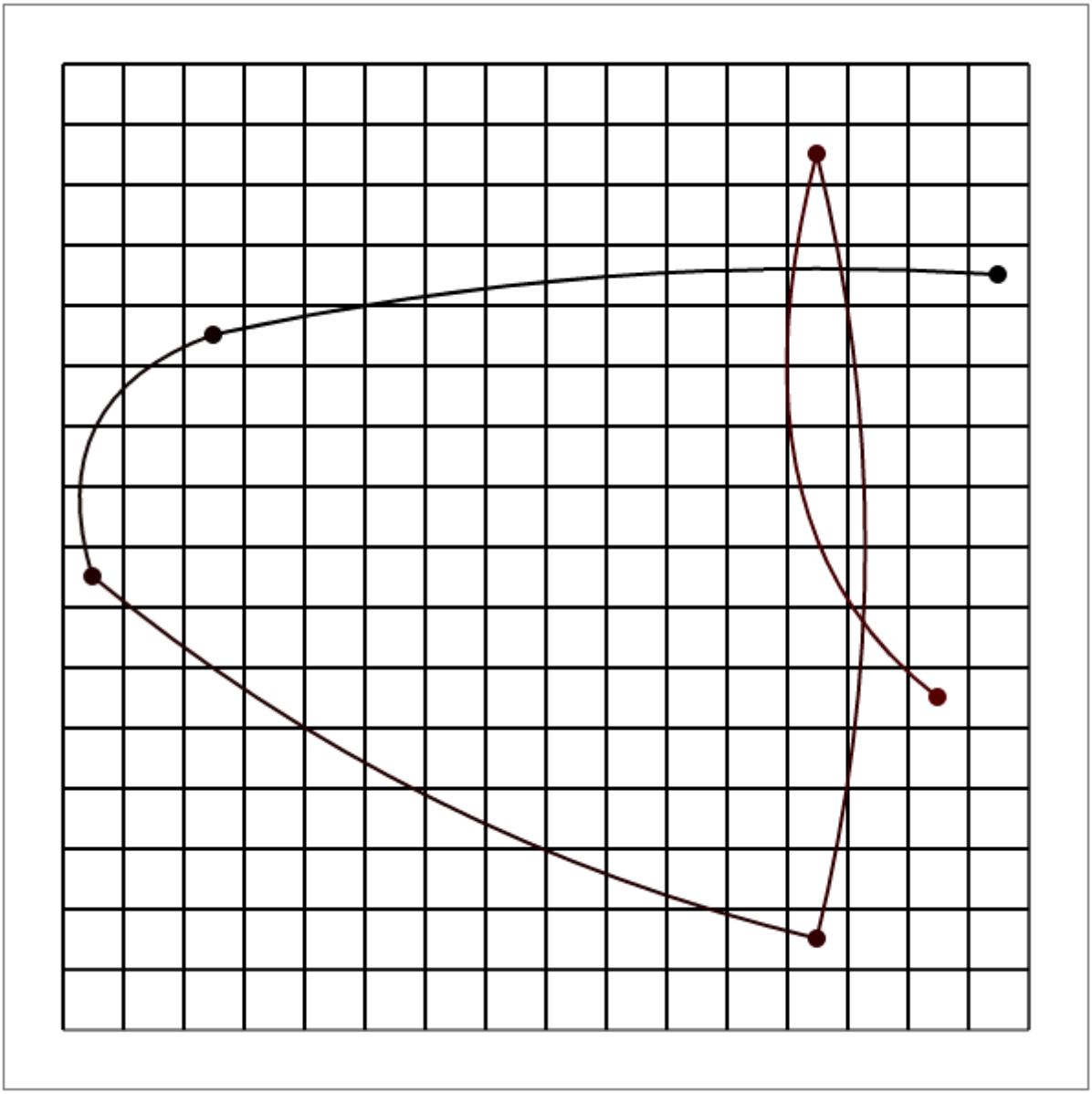

Figure 22: Line-Matrix - matrix with six bytes visualized

In figure 23, 12 bytes (96 bits) are visualized.
The hash value used is 95303382396cacf874d3c0b5. It gets harder to compare it to another picture. In this case the values are distributed. In other cases, where byte values are similar or even equal to their neighbouring values, the path of lines is hard to follow. This is shown in figure 24 on the following page.

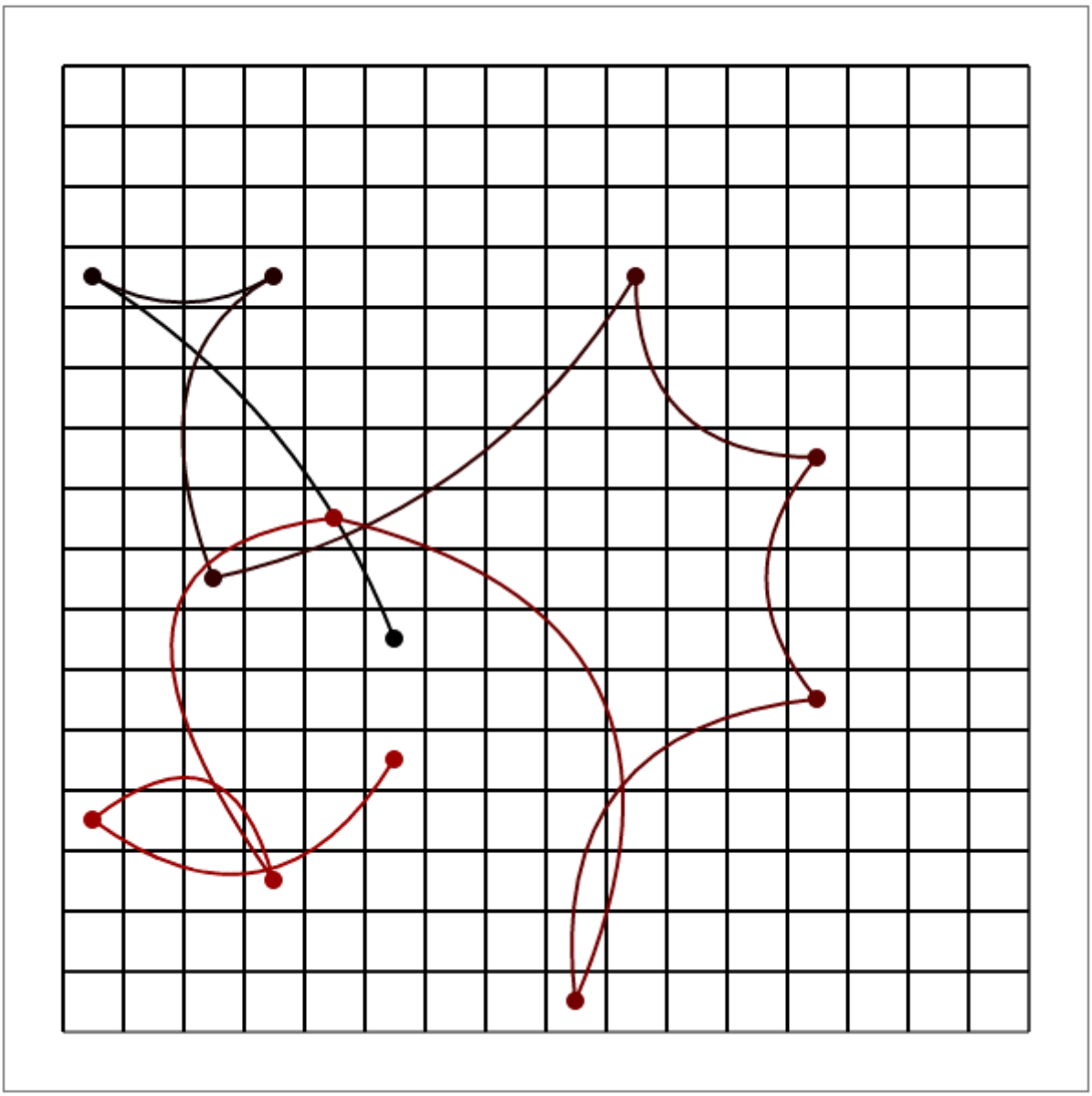

Figure 23: Line-Matrix - matrix with 12 bytes visualized

Although in figure 24 now 160 bits have been visualized, it is evident that the path of the lines is too hard to follow. The hash value is: 3e0dff21ca87762759645ec6ec630408143ae4d9.

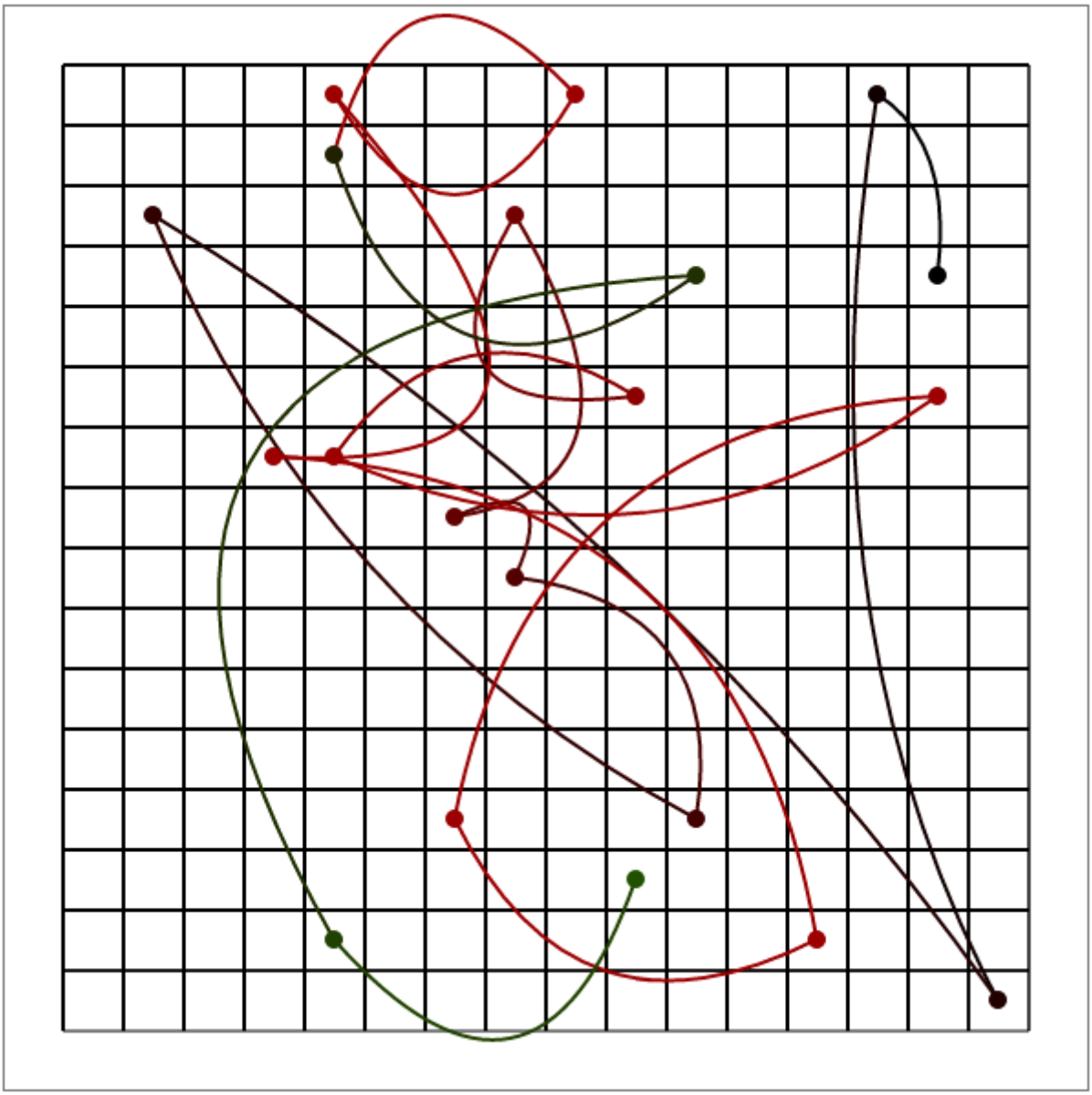

Figure 24: Line-Matrix - matrix with 20 bytes (160 bits) visualized

6.1.2 Evaluation and analysis

- The graph doesn't scale very good. The more the number of bytes that are used to display, the harder it gets to follow the path from the starting dot to the end.
- The colors of the lines start with black and fade from dark to bright red, ending in green when using up to 20 bytes. Those colors are difficult to see and they could look different on other displays or devices.
- Equal bytes in a hash value resulting in equal positions in the matrix are not considered.

6.2 Object-Matrix

The first approach identified a few issues which needed to be dealt with. If the amount of entropy in a single object (a point in the first approach) rises, the sum of objects can be lower while maintaining the size of the hash value. This generates less chaotic pictures. The occurences of two or more equal values leading to the same space in the matrix have to be considered. The path from the starting point to the last point or value has to be easy to detect and follow.

6.2.1 Description of Object-Matrix

In a first attempt objects consisting of two colored shapes are introduced. The shapes are of two different sizes (large and small) and the smaller is painted onto the larger shape. One object uses three bytes of a hash value. This is depicted in table 3.

Byte/Bits	7	6	5	4	3	2	1	0	
0	x	x	x	x	y	y	y	y	x, y = positions in matrix
1	s	s	s	s	c	c	c	c	s = shape, c=color of bigger object
2	s	s	s	s	c	c	c	c	s = shape, c=color of smaller object

Table 3: Table of the buildup of a single object

One shape is built of one of four different objects which are rotated in either north, west, south or east directions. It can have one out of 16 colors. The four different shapes are shown in figure 25. Figure 26 displays one shape rotated in four directions. The shapes also do have numbers now to easier track the path from one to another.

Figure 25: Object-Matrix - four different shapes

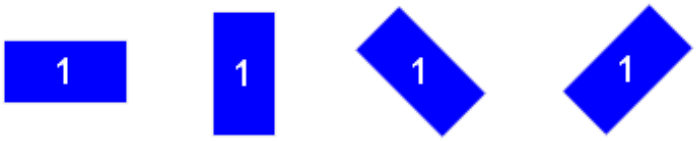

Figure 26: Object-Matrix - four rotations

Figure 27 shows the final composed objects. A smaller shape drawn upon a bigger shape and a number. The inner shapes are outlined to be better distinguishable against the bigger underlying shape.

Figure 27: Object-Matrix - four diverse composed objects

As can be seen in figure 28, the matrix has also been changed to make it easier and faster to determine the correct position of the object. The curved lines leaping from the starting to the last object are also still in existence but could be left out. The numbering of the objects shows the way from start to destination. The example shows four objects computed from this hash value of 96 bits of length : f185a3df8ab60e0362e65a1f.

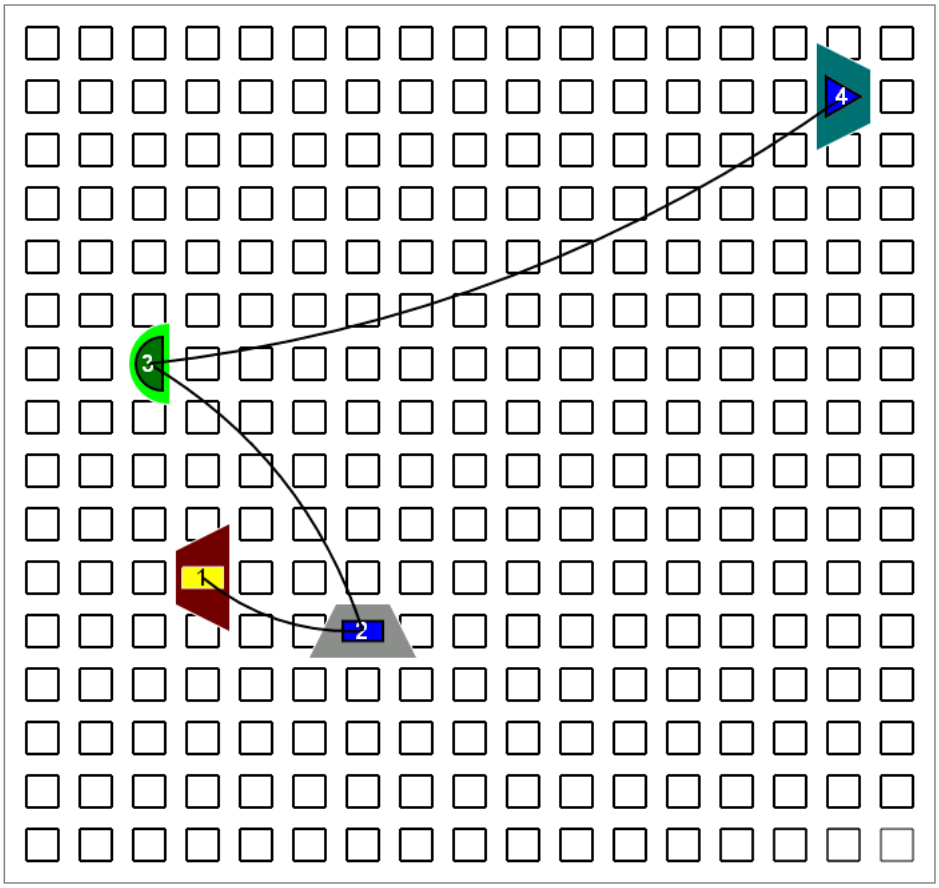

Figure 28: Object-Matrix - matrix with four objects

To cope with the problem of two objects having the same position in the matrix (the position bytes of the hash value are identical), a 17th column was attached to the matrix. The algorithm checks if a position is already occupied by an object. If this is the case, the next object will be moved to the first place (second, third and so forth) in the 17th column. This behaviour is no threat regarding second pre-image resistance because repositioning doesn't give new information to a possible attacker. Figure 29 shows the maximum entropy of 168 bits encoded into the matrix. The third object is moved to the first backup location in the 17th column because of having the same position-byte as the first object (hex: 8D).
The value of the hash is: d1ce8df9f1d09b948db5e395522ee476b117acdbb9

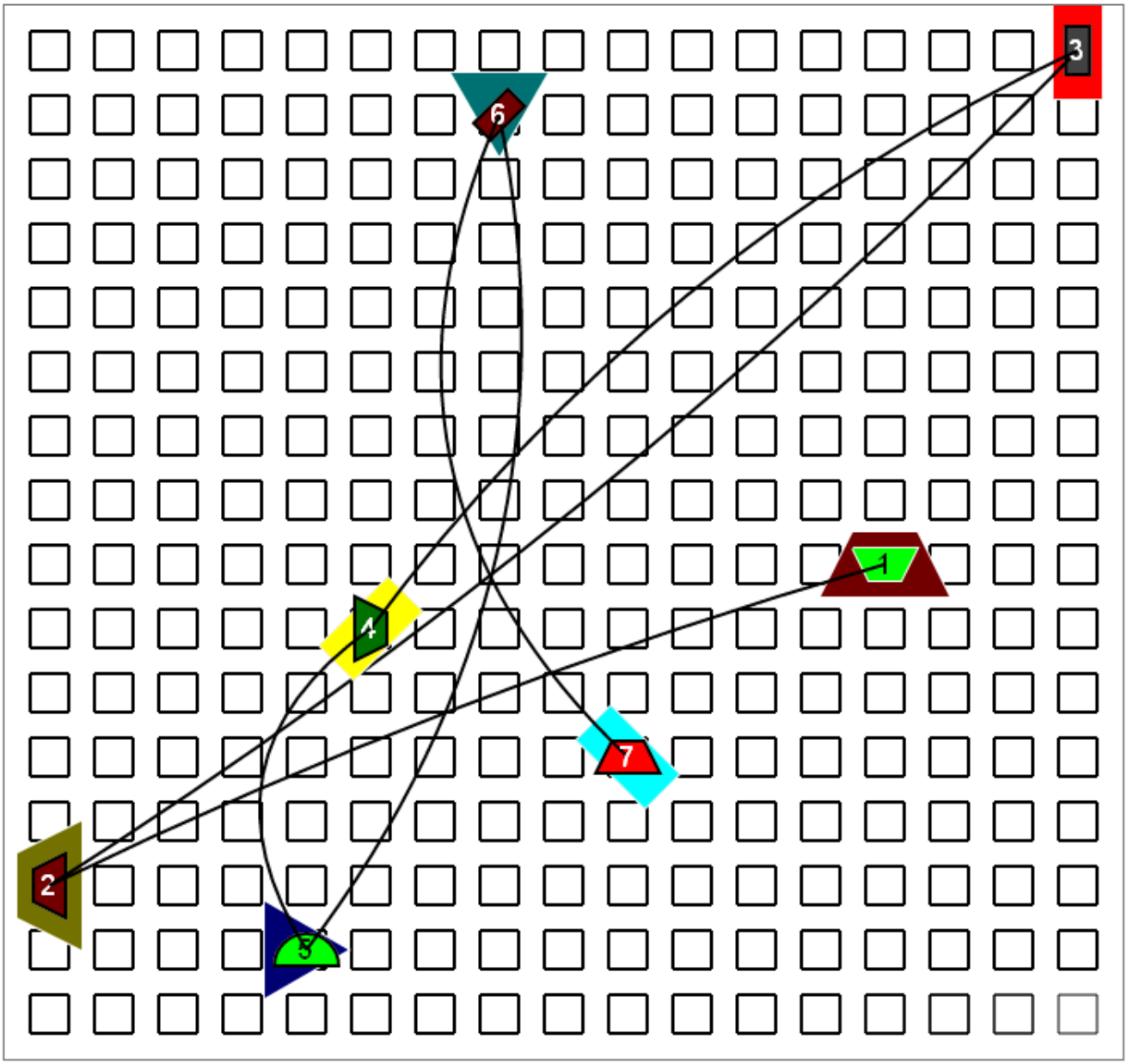

Figure 29: Object-Matrix - matrix with seven objects and an entropy of 168 bits

As can be seen in figure 30, there are still a few flaws. Objects three and six are at the same position as object one and moved to the backup position on the 17th column. The shapes of objects one and seven are of the same colors. Objects one and two as well as four and seven are very close to each other. All that can make an object hard to identify and distinguish, taking precious time and causing errors. The hash value from the previous page was slightly altered to show the problems that might occur. The value used here is: d1c18df9f18e9b948db5e3b8522ee476b18dacdcb9

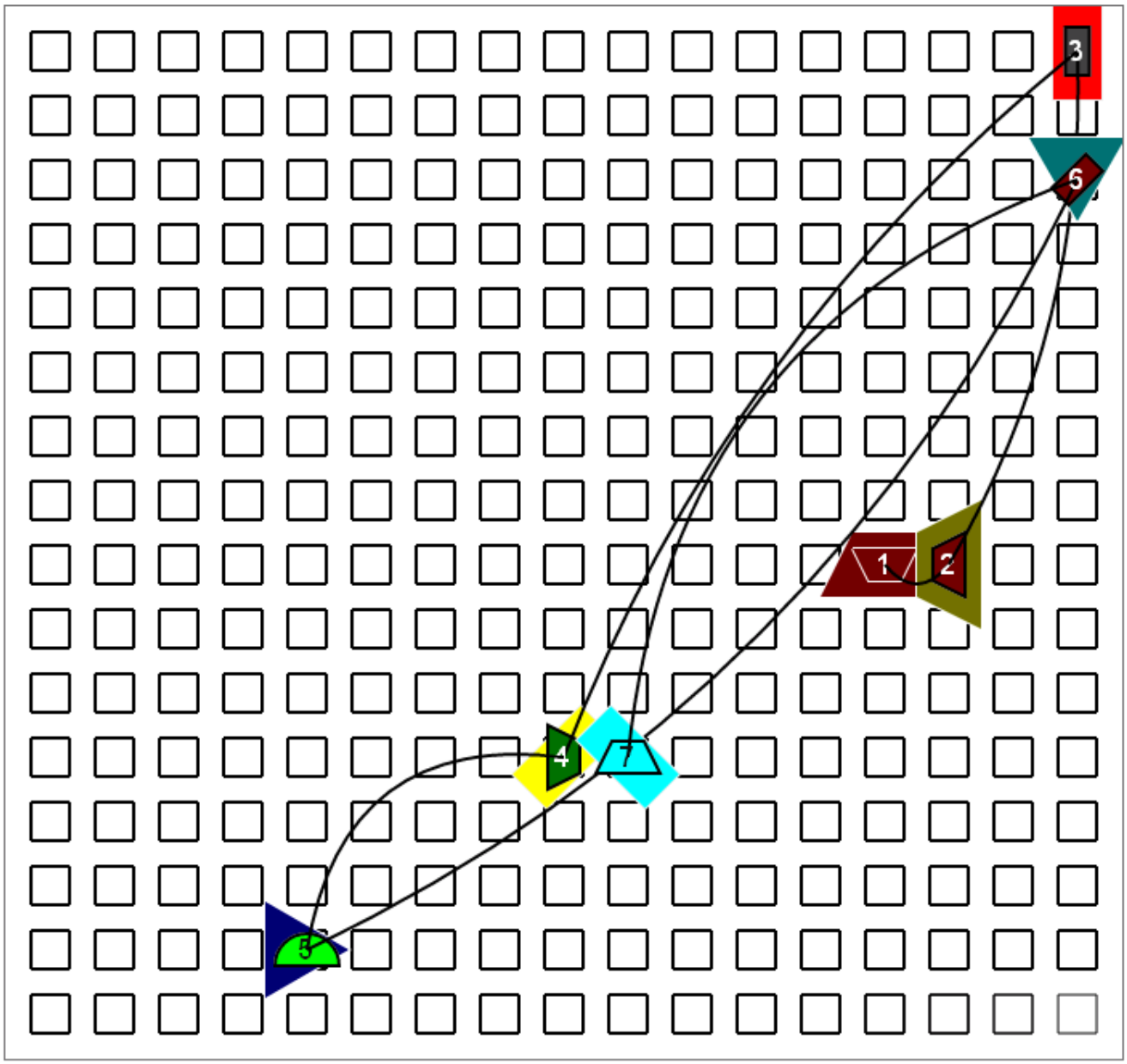

Figure 30: Object-Matrix - matrix showing flaws

6.2.2 Evaluation and analysis

- The objects have to be big enough to be identified - even by people with viewing disabilities and when viewed on smaller displays like mobile devices.
- If objects are big, they cannot be placed adjacent to each other on the matrix because they overlap. Thus the matrix has to be much smaller. If the matrix would be of size 4 x 4 then four bits of entropy would be lost for each object. The actual size is 256 squares = 2^8, a smaller size of 16 squares = 2^4 gives an entropy of only four bits.
- Two shapes forming one object can be of the same color as shown in figure 30. This renders this special object hard to identify. If two equal colors are forbidden, e.g. the color of the smaller overlaying shape is transformed to another color, there would be a security violation regarding second preimage resistance: Two different colors lead to the same hash value.
- An object can be rotated in four directions: north, east, south and west. In case of the triangle, it could be hard to recognize the direction.
- The objects look similar to each other in case of the square and the parallelogram.
- In most cases a hash value generated by a modern hashing function such as SHA-1 gives a ‘distributed‘ value, where occurances of bytes of the same value (e.g. three consecutive bytes of value 00) are not likely. But if it happens, the outcome of the algorithm could be really useless, as seen on the previous page.

7 Colors Letters Patterns Positions Shapes

This new approach is called CLPPS - ‘Colors Lines Patterns Positions Shapes‘. The entropy is reduced to 64 bits and a wider range of objects as well as patterns, colors and four letters of the alphabet are introduced. A study is conducted afterwards to find flaws and further improve it. The study also includes a test on the capabilities of people comparing Base32 values. The conclusion and the analysis will be shown in chapter 7.3.

7.1 Description of the approach

Taking the findings of chapter 6.2 into account, a new version was devised. The entropy could not be maintained and has been lowered to 64 bits per picture. The amount of colors has been reduced to eight (see figure 31). The matrix has been discarded. There are now only two positions where an object can be placed: up and down. This will become clearer on the figure showing the objects assembled. The objects have changed a lot. One object is now composed of only one shape but also a pattern which the shape is filled with. The patterns are shown in figure 32 and the objects in figure 33.

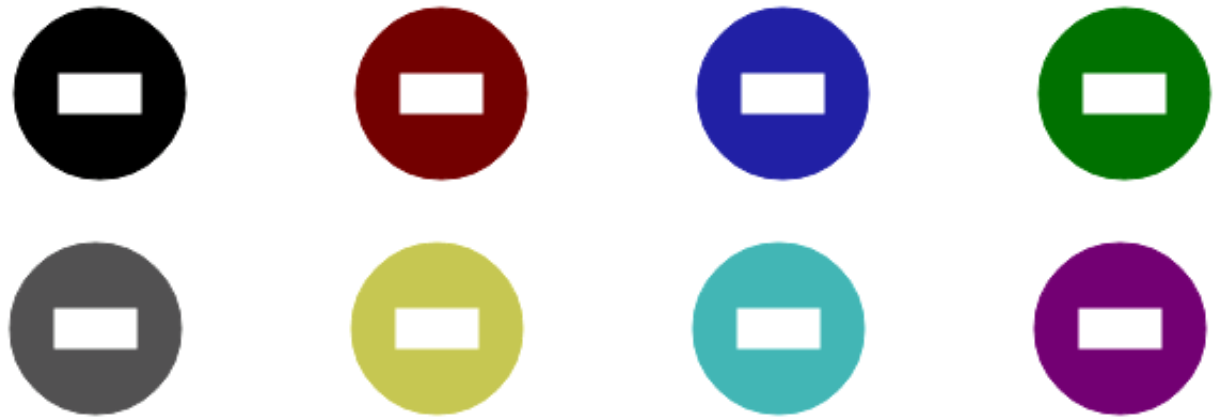

Figure 31: CLPPS - All 8 possible colors

Figure 32: CLPPS - All 4 different patterns

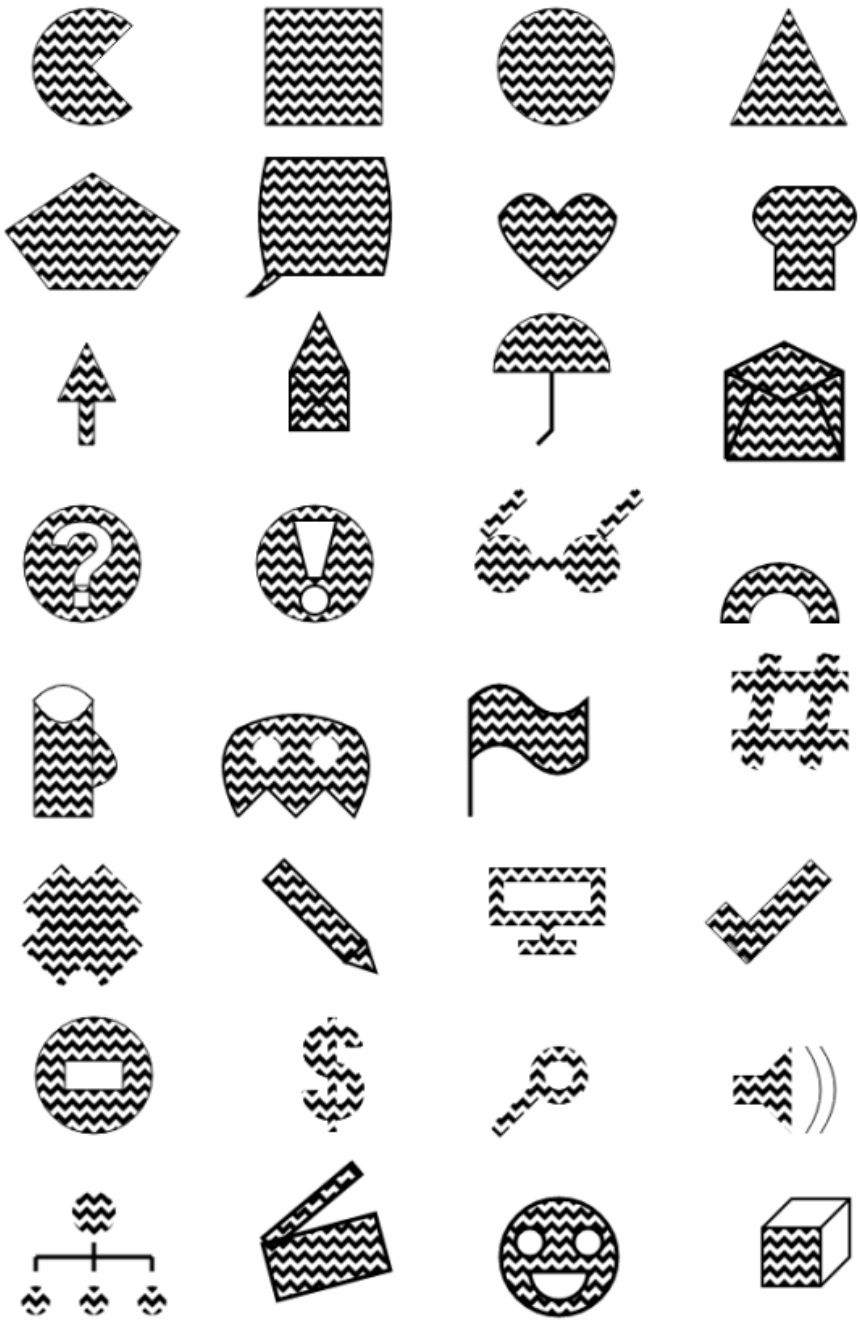

Figure 33: CLPPS - All 32 shapes with same color and pattern

Finally a word consisting of four letters of a Base32-alphabet is added to the picture. Table 4 gives an overview of the letters that can be used. The letters ‘I‘ and ‘O‘ and the numbers ‘0‘ and ‘1‘ are left out to avoid confusions when comparing the letters.

The truetype font ‘Arial 36px bold‘ was used for the letters because truetype fonts are scaleable and look much better when zoomed in than monospace fonts.

Letters	ABCDEFGHJKLMNPQRSTUVWXYZ23456789

Table 4: Table of the letters

Figure 34 shows an example of the final composed picture including four shapes with different colors, patterns and positions and the letters. The first two objects are in an *up* position while the objects three and four are in a *down* position. The patterns of the last two objects are equal. The *dotted* pattern was not used in the example.

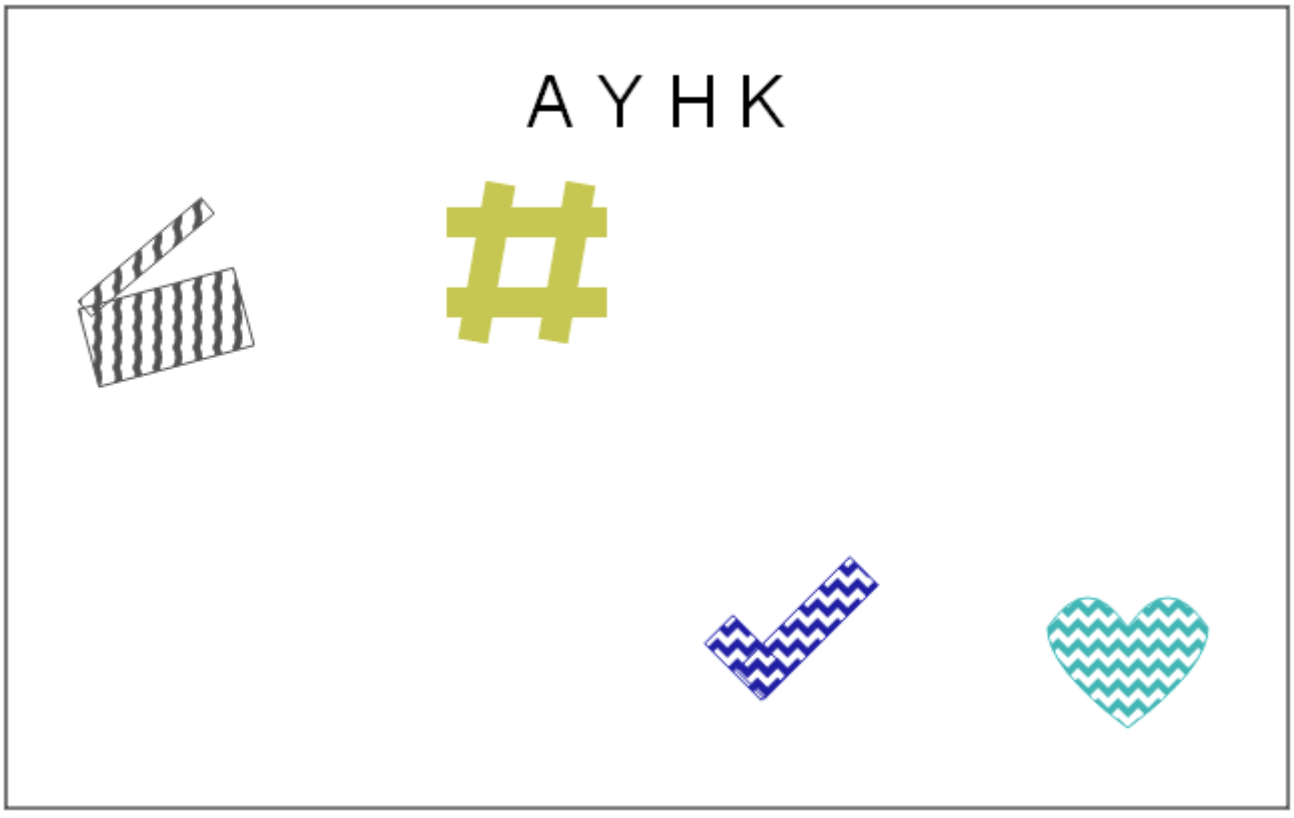

Figure 34: CLPPS - example of final picture

7.2 Description of user study: Comparison of CLPPS and Base32

To test the improvements made with CLPPS and to compare it against the best scheme in existance so far, a study with 16 participants was conducted. The study took place in a medium sized company with employees mostly doing office-work. This was done to get a broader range of age and IT-knowledge. Recruitment was done by simply walking from office to office and asking people while trying to keep the number of women and men equal. The environment was mostly quiet but sometimes disturbed by telephone calls or collegues asking questions. The Base32 part of the study was done to check if the results regarding Base32 of the study [14] in chapter 3 can be confirmed. There, Base32 had a success rate of 97% on easy and 86% on hard pairs. The Powerpoint presentations as shown in chapter 5 are used for the visualization of CLPPS and Base32 respectively. The participants did both presentations consecutively but with alternating starting presentations.

The results of the demographic survey are shown in the following tables.

	Participants	Age (average)
Women	8	40,375
Men	8	37,75

Table 5: Table of participants and age

Grade of computer knowledge	Amount
Novice	5
Advanced	6
Expert	4
No proposition	1

Table 6: Table of grade of IT knowledge

	Persons into ...
Graphic arts/Painting	2
Design	1

Table 7: Table of experience in graphic or design

The grade of IT knowledge of the participants is average and has no influence on the results, as well as the knowledge or profession of painting or design.

The slides are randomized to avoid problems that might occur due to the positions of a set of slides. This means if the participants always see a fixed order of slides they may adapt to those slides and the results may vary from another random order of slides. Randomization ‘mixes‘ the results so they are valid for all kinds of orders. For the CLPPS setup, there are 10 slides with identical pictures (see figure 35) and five slides with apparently different pictures (figure 36). Those pairings are called ‘easy pairs‘. There are also 11 slides with pictures with only one difference - so called ‘hard pairs‘ (figure 37). The 11 slides with hard pairs are built up as follows: One slide with an object changing its shape, two slides with one letter changing, two slides with a change of colors, two slides with an object changing positions from up to down and vice versa, four slides with a pattern change. The hard pairs are generated by using similar attributes (e.g. with colors - ‘blue‘ is changed to ‘purple‘, with letters - ‘S‘ is changed to ‘5‘ and so on).

Figure 35: CLPPS - example showing two identical pictures

The slides with apparently different pictures are used to resemble reality. A hash function is built in such a manner that if the input changes by only a bit, the output changes drastically. These pairs of pictures are called ‘easy pairs‘ because they are really easy to detect. In the study the participants often hesitated for a short moment when they first came upon such a pair, because they did not expect such a picture so easy to distinguish. They became faster on the next appearances of those pairs.

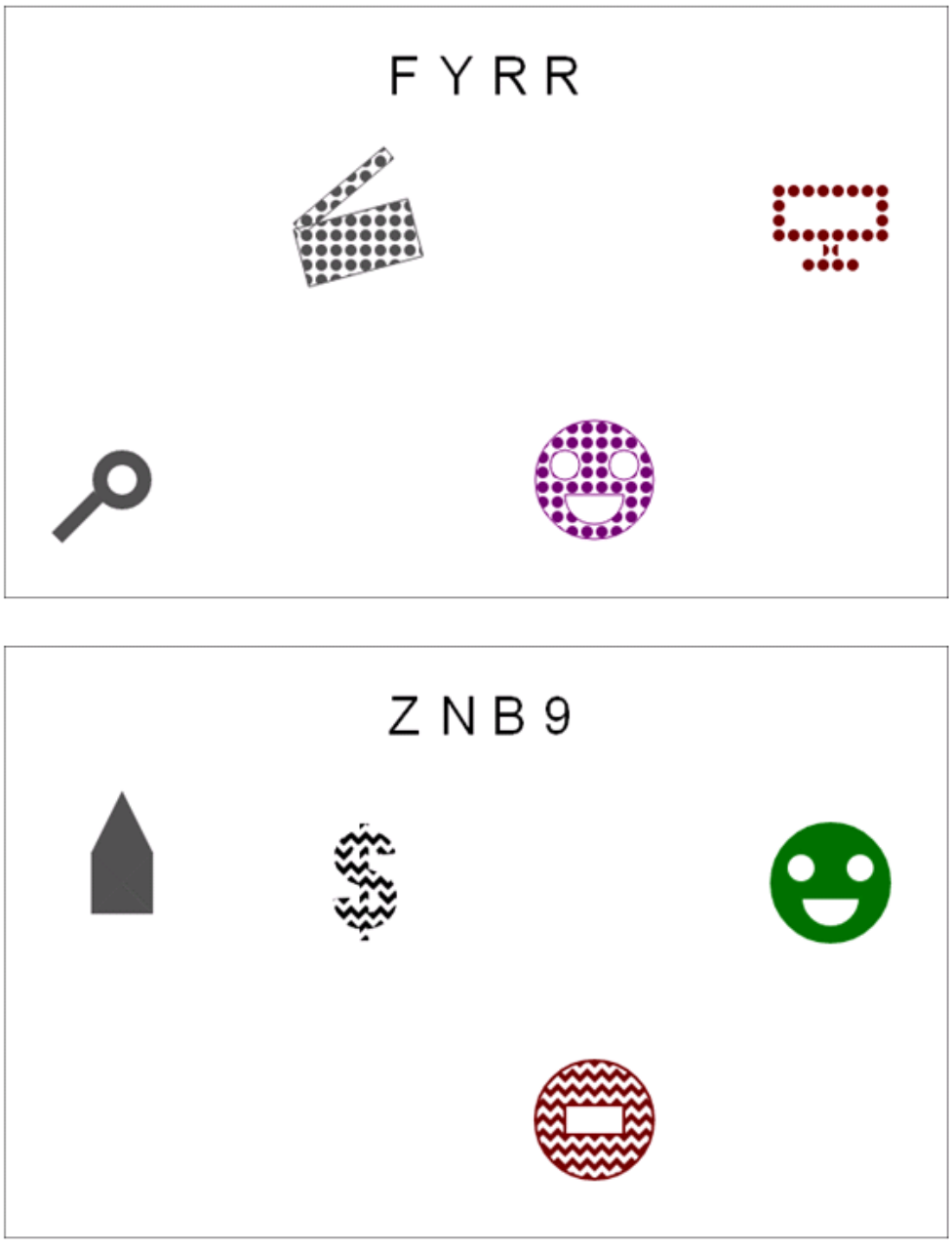

Figure 36: CLPPS - example showing two apparently different pictures

In this picture, the first object changes its position from *up* to *down*. Changing of positions is a major cause for errors as can be seen later in the results of the study. There are four 'hard pairs' with one object changing its pattern. This was done to find out, which of the patterns (dotted, filled, horizontal lines, vertical lines) are difficult to compare.

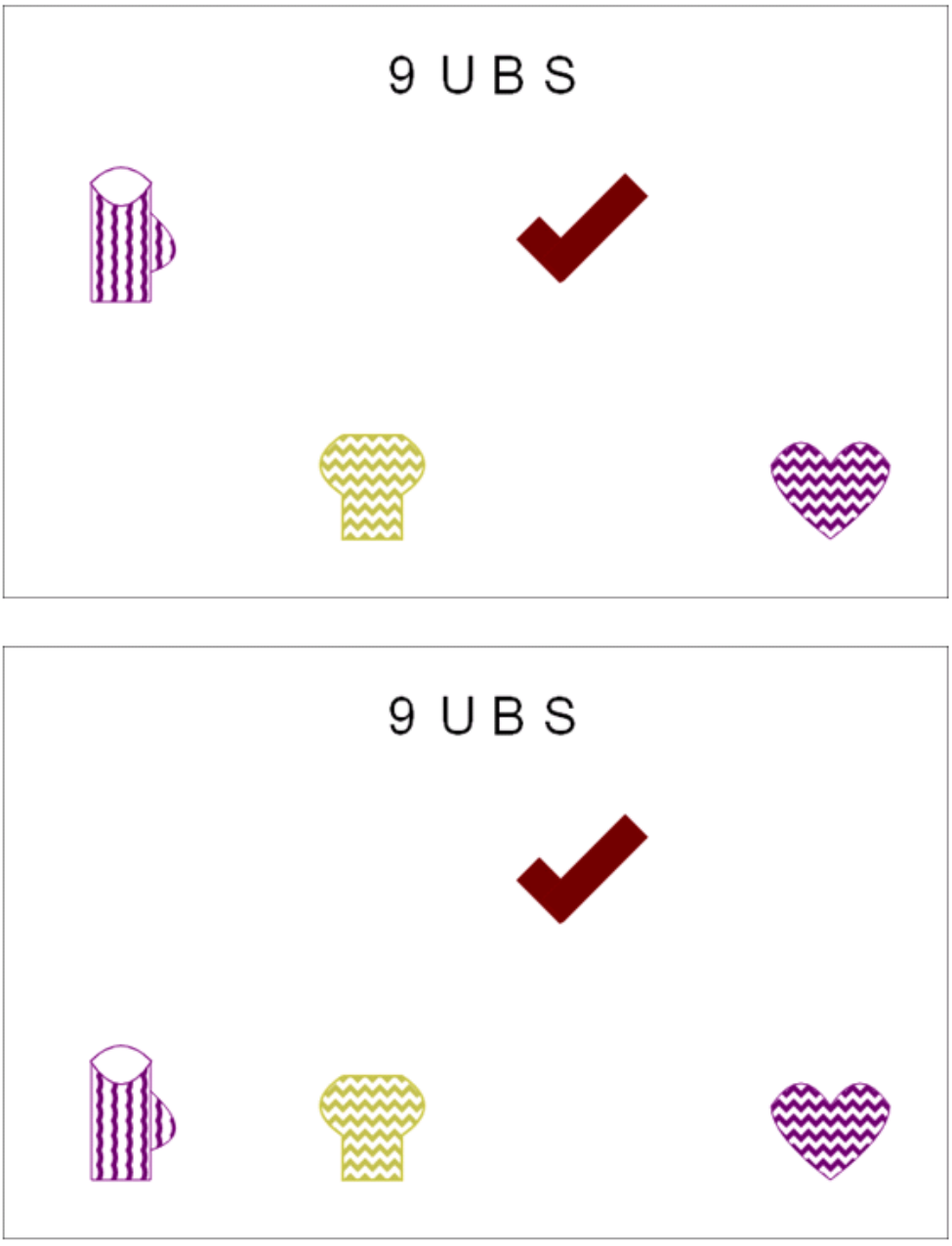

Figure 37: CLPPS - example showing two pictures with one difference, a 'hard pair'

For Base32, the 25 slides are divided into 10 slides showing two equal pictures, five slides showing two apparently different pictures and 10 slides with only one or two characters changing. The setting was changed to a word length of 65 Bits, which equals 13 Base32-characters, to be better comparable against CLPPS. One Base32 character carries an entropy of five bits. The font used for the characters is ‘36px monospace bold‘. A TrueType-font offers a much better quality but the characters are of different sizes. As an example, the ‘W‘ takes more than double the space as the ’I’. Figure 38 shows a slide with a hard pair. There are two pictures with one picture having a minor difference to the other. To make the comparison a little bit more difficult, the vertical distance between the two strings of Base32-characters is about 10 centimeters.

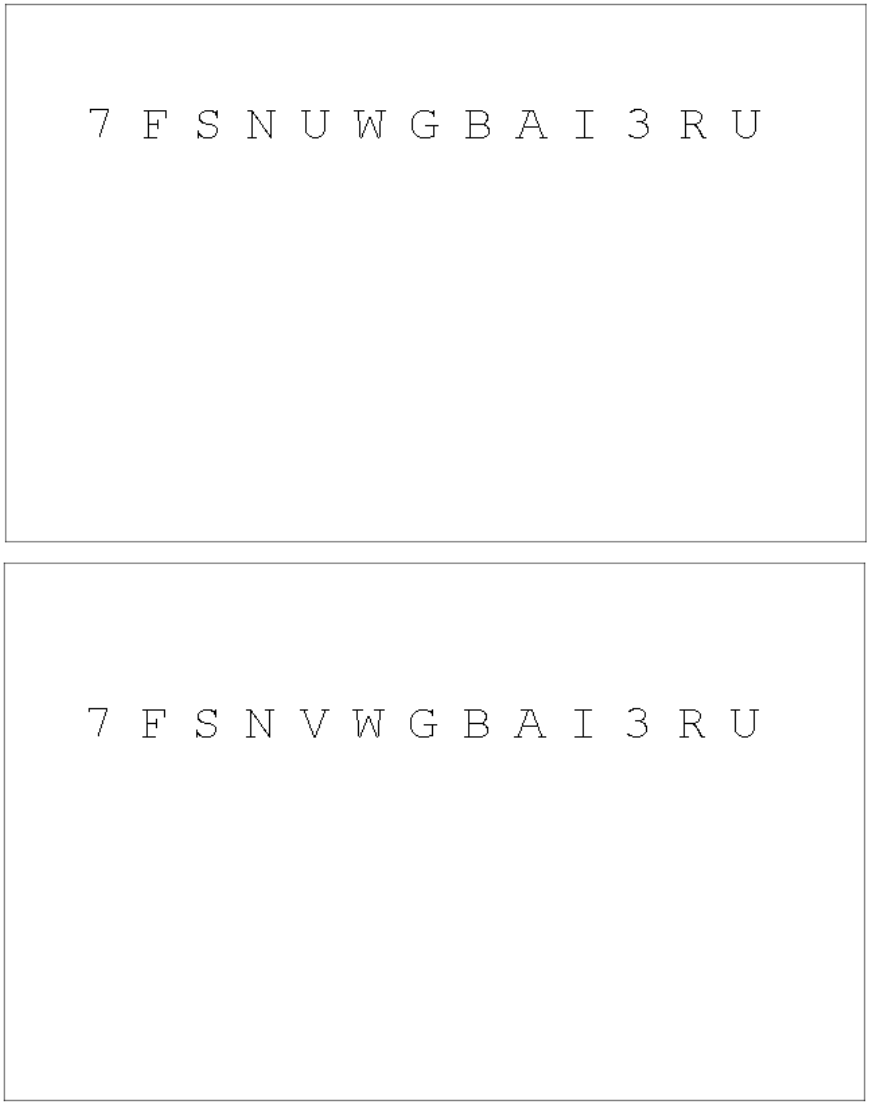

Figure 38: Base32 - example of picture showing a difference in the fifth character

7.3 Findings of the study

The results of the study are shown in the following tables. For the CLPPS part, only the 10 slides where the participants made at least one mistake are featured in table 8. The other 16 slides causing no mistakes are left out. The first three slides which caused an unusual high rate of errors are discussed explicitly. For Base32, the eight slides where at least one error occured are displayed in table 9. Because there are no high error rates regarding the Base32 slides, those will not be discussed. Two participants clicked the wrong button on a slide which lead to false negatives and explains the errors on easy pairs. Also one participant thought that one object in CLPPS has moved its position a tiny bit compared to the other picture, which also lead to a false negative on an easy pairing. The percentage of those could be decreased by a larger number of participants.

Slide	Cumulative time	Average time	Errors	Percentage of success	Category
Figure 39	2:16:230	0:08:514	11	31	hard
Figure 40	1:46:276	0:06:641	9	44	hard
Figure 41	1:45:541	0:06:596	7	56	hard
—	1:17:576	0:04:848	3	81	hard
—	1:39:107	0:06:194	2	88	hard
—	2:31:817	0:09:489	1	94	easy
—	1:45:796	0:06:612	1	94	hard
—	2:22:489	0:08:905	1	94	easy
—	1:26:377	0:05:398	1	94	hard
—	2:13:247	0:08:328	1	94	easy

Table 8: Results of CLPPS, each entry is per slide, sorted by errors

Cumulative time	Average time	Errors	Percentage of success	Category
1:43:714	0:06:482	2	88	hard
2:26:754	0:09:171	2	88	hard
1:44:642	0:06:539	2	88	hard
1:36:555	0:06:035	1	94	hard
1:52:523	0:07:032	1	94	hard
2:16:107	0:08:506	1	94	easy
2:42:026	0:10:127	1	94	easy
1:07:882	0:04:242	1	94	hard

Table 9: Results of Base32, each entry is per slide, sorted by error

The study has shown, with CLPPS, there are major problems of detecting the *up* and *down* positions of the objects - this will be shown in figures 39 and 40. Also some patterns are difficult to compare to other similar-looking patterns. This will be addressed in figure 41. The first issue is the position changing of one object from up to down or vice versa. 11 of 16 and nine of 16 did not detect this difference on the respective slides. Because the position adds one bit per object to the entropy and the summed-up entropy of 64 bits per picture should be maintained, another way needs to be found to make the importance of the position clearer and discrepancies easier to detect. The second issue are the patterns of the objects. The 'dotted' pattern looks too similar to the 'horizontal-waved' pattern, so 44% of the participants could not distinguish between those two. The patterns need to be unique to be better comparable. Objects changing shape or colors did not cause many errors as well as changing of the letters. When asked, people answered to always start with the four letters and thus immediately detecting differences.

Tables 10 and 11 give an overview of the accuracy and the speed of the comparison of CLPPS and Base32 respectively. The average accuracy with CLPPS on easy pairs is good and only 0.36% lower than Base32. The average accuracy on hard pairs could be better much better with a difference of 14% and the minimum accuracy with 31% is really bad. The average time is ok with about 7 seconds for both schemes, given the amount of entropy, but needs to worked on. The results of this Base32 study are much better compared to the related work, even though a much higher entropy was used - 65 instead of 25 bits. The average accuracy of easy and hard pairs combined is 97.25%, which is by far the best result of all schemes tested. This will be the goal to achieve in the next approach of my work.

	Easy pairs	Hard pairs	Combined
Average accuracy	98.80 %	80.72 %	91.15 %
Minimum accuracy	94.00 %	31.00 %	67.35 %
Average time per slide	07:052 s	05:860 s	06:548 s

Table 10: CLPPS - Accuracy and times

	Easy pairs	Hard pairs	Combined
Average accuracy	99.16 %	94.375 %	97.25 %
Minimum accuracy	94.00 %	88.00 %	91.6 %
Average time per slide	06:962 s	05:876 s	06:528 s

Table 11: Base32 - Accuracy and times

On this slide, the last object changes its position from *up* to *down*. 11 out of 16 people failed to detect this. This is a really high error rate which will be confirmed in the next slide.

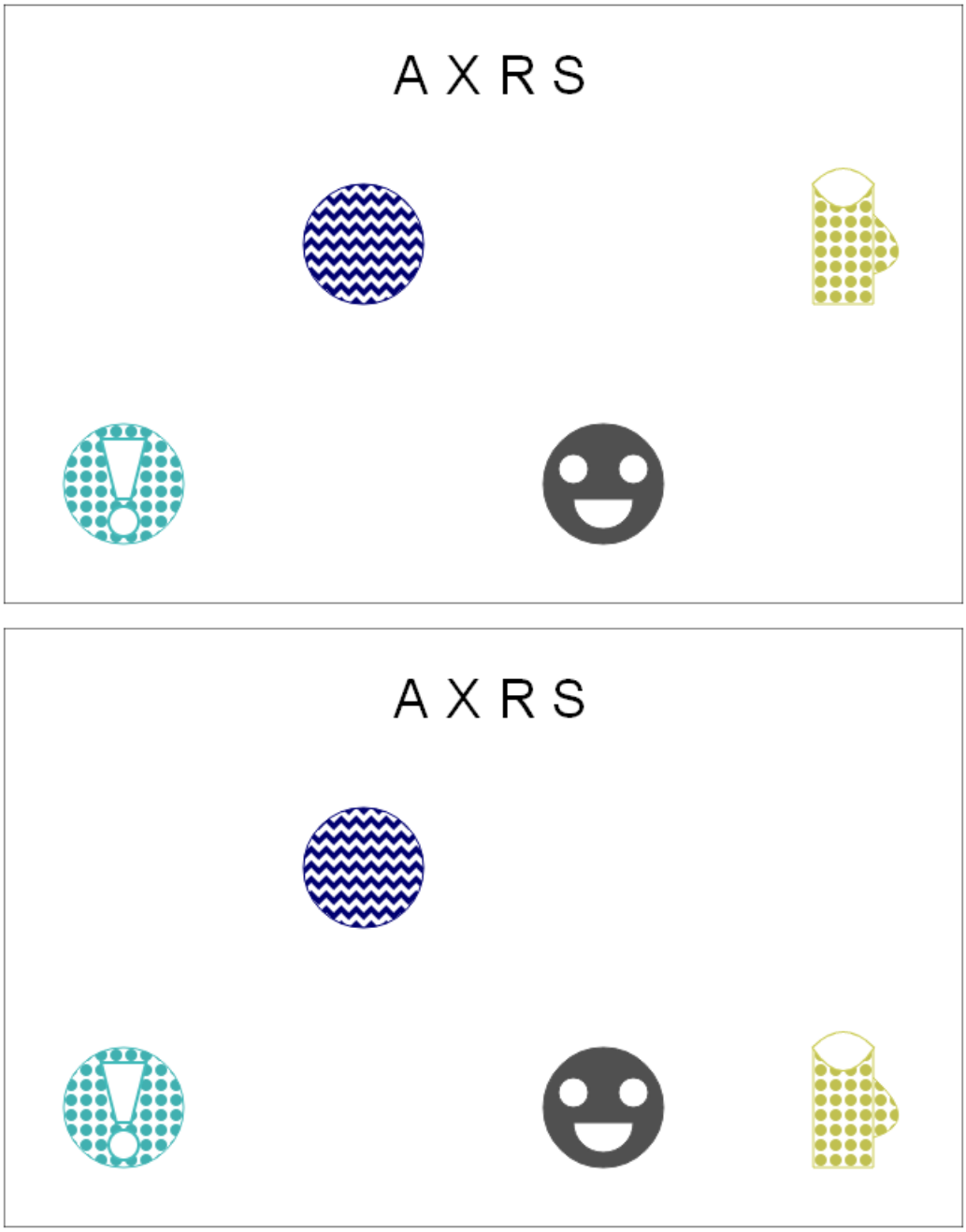

Figure 39: CLPPS - slide with one position difference

This is the second slide with a position changing. Nine out of 16 participants failed to detect the first object has changed its position from *up* to *down*. The slightly lower error rate of nine to 11 from the previous slide could be explained by being the first object instead of the last as in the previous slide. If the first of the four objects changes, it can be detected faster because usually people start comparing from left to right.

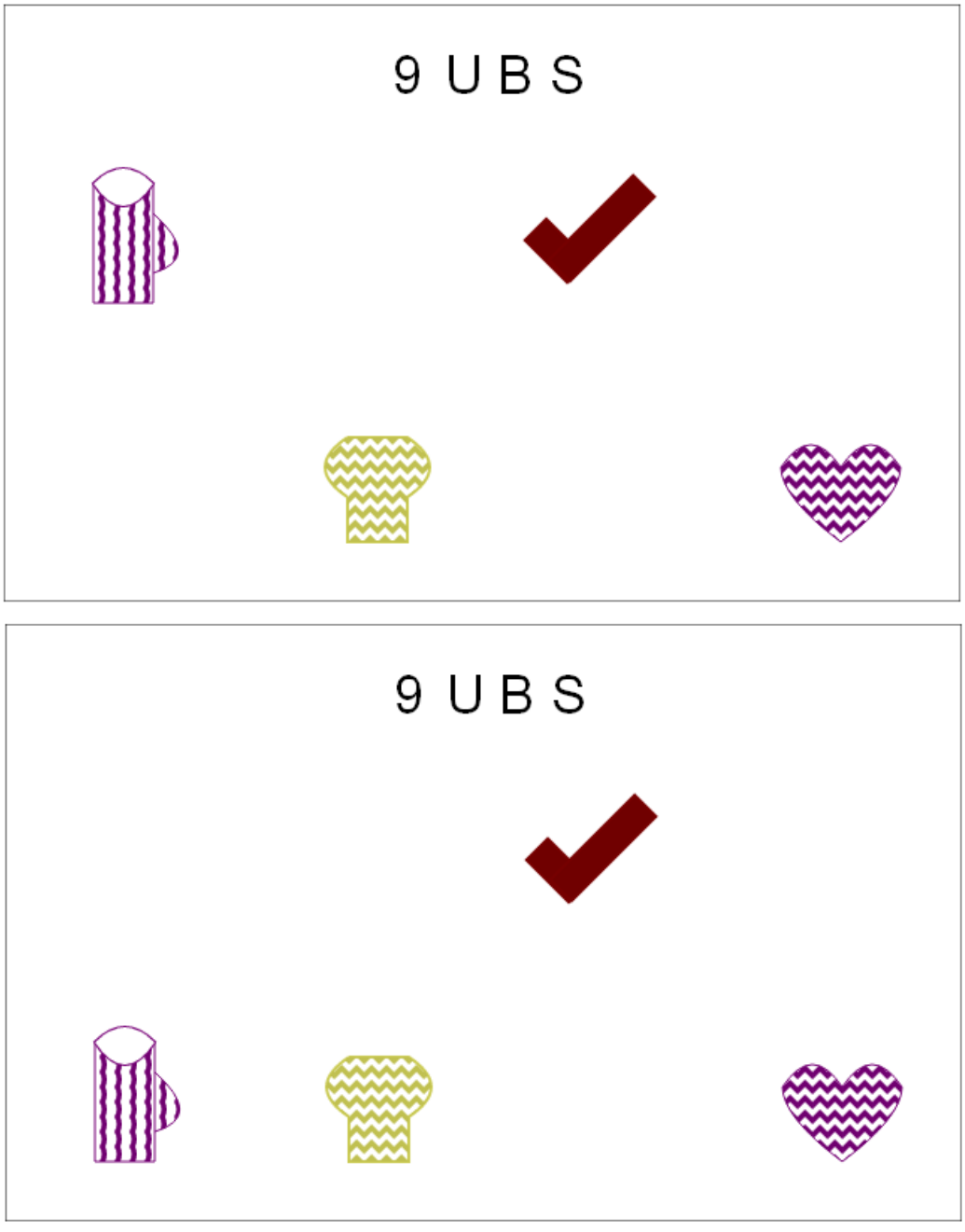

Figure 40: CLPPS - second slide with a position difference

The third slide shows a change of pattern in the third object. The pattern of the arrow-like object is of waved horizontal lines while the pattern in the picture below is of a dotted type. Seven out of 16 participants did not detect this difference.

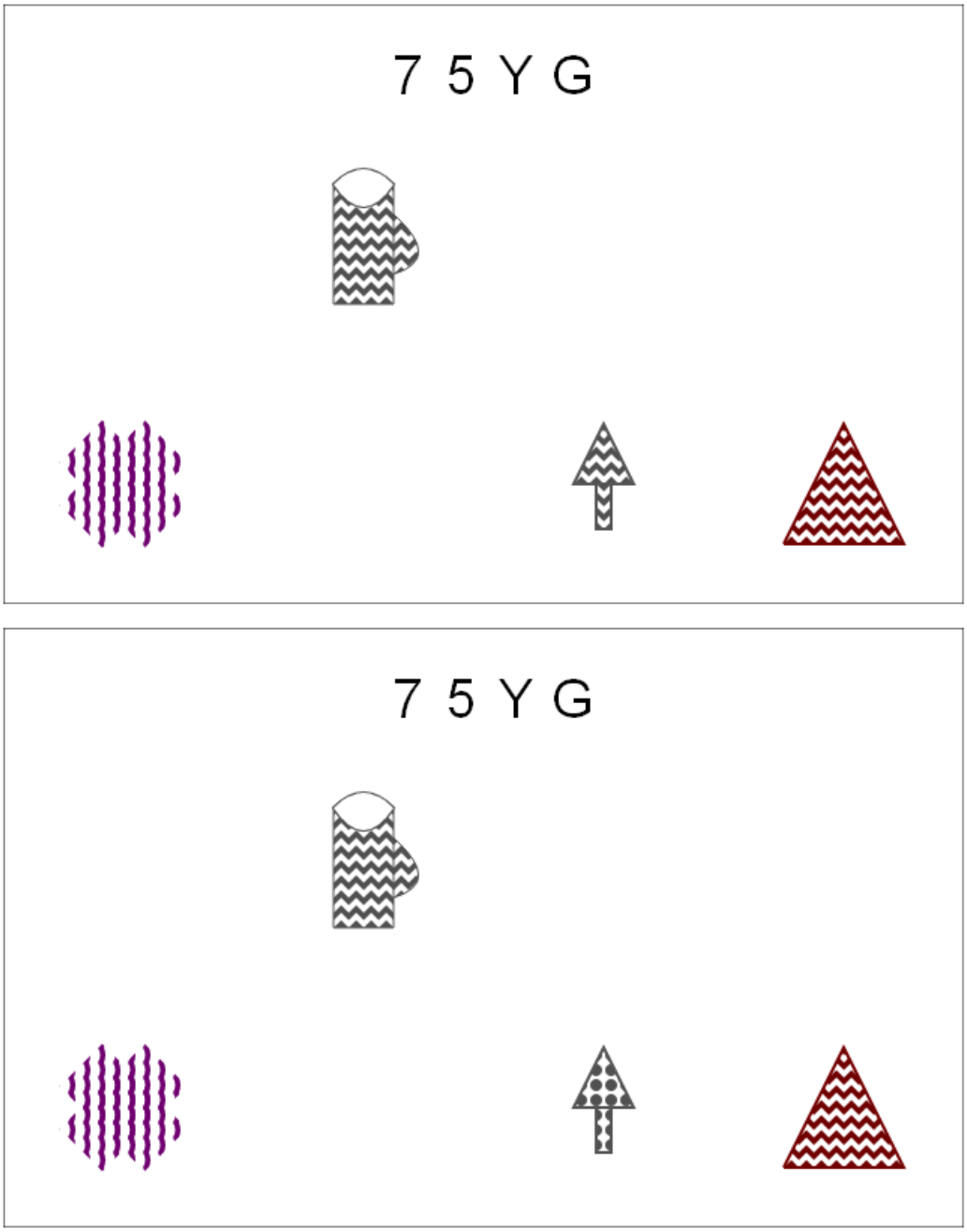

Figure 41: CLPPS - third slide with a pattern change

8 Improved version of CLPPS

The results of the first study have shown two major sources of errors. The first being the *up*- and *down* positions of the object and the second being the patterns. This approach introduces a few improvements to cope with the findings of the last study. Another study will be conducted to test if the improvements are useful.

8.1 Description of the approach

A horizontal line will be introduced between the objects to distinguish between the *up* and *down* positions.
The patterns are changed to straight instead of waved lines. The dotted pattern was replaced by an empty or non-filled pattern.
Some of the shapes in the previous version have been of different sizes which led to confusion while comparing the objects. Although clearly mentioned in the explanation, a few people thought one object has moved by a millimeter and this is a distinction which led to false positives in one case.
The font has been changed to ‘charter 36px bold‘ because some letters are better to identify. Figure 42 shows all the changes while figure 43 shows the new patterns and figure 44 gives an overview of the changes to the shapes. Some shapes have been replaced to make it easier to detect the patterns used.

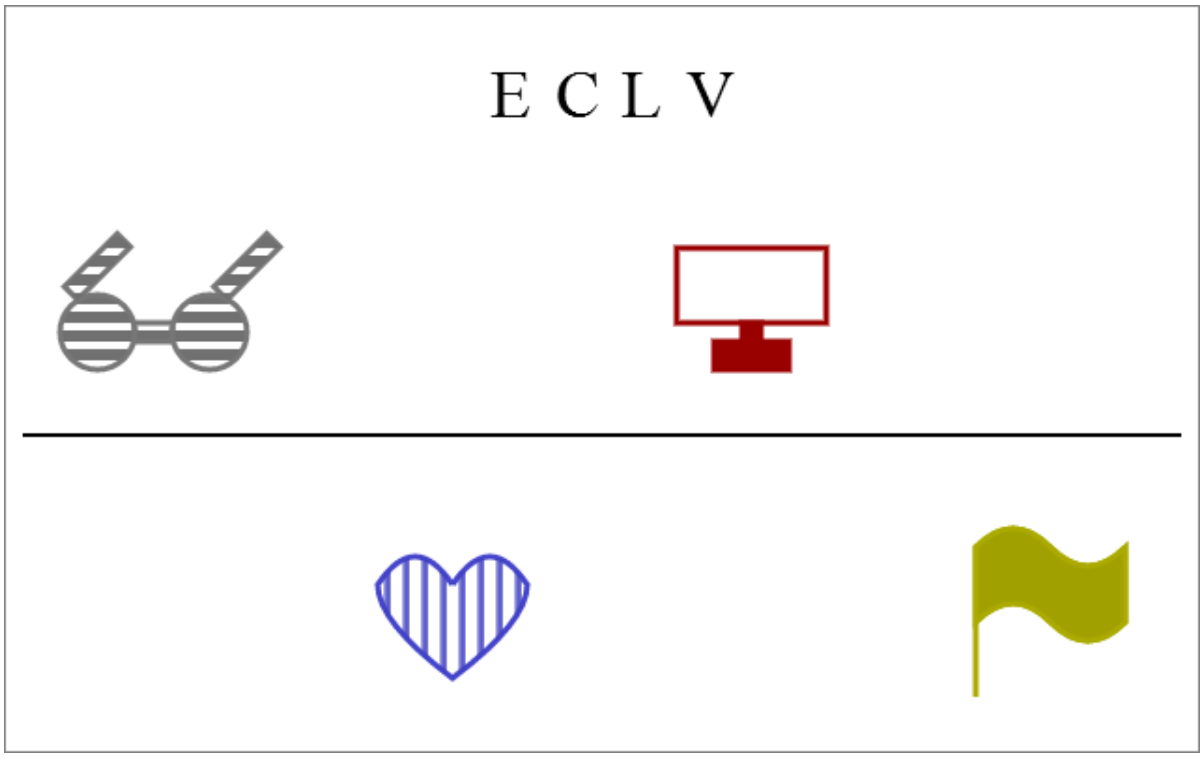

Figure 42: CLPPS(imp.) - A picture showing the separating line and the other changes

Figure 43: CLPPS(imp.) - New patterns

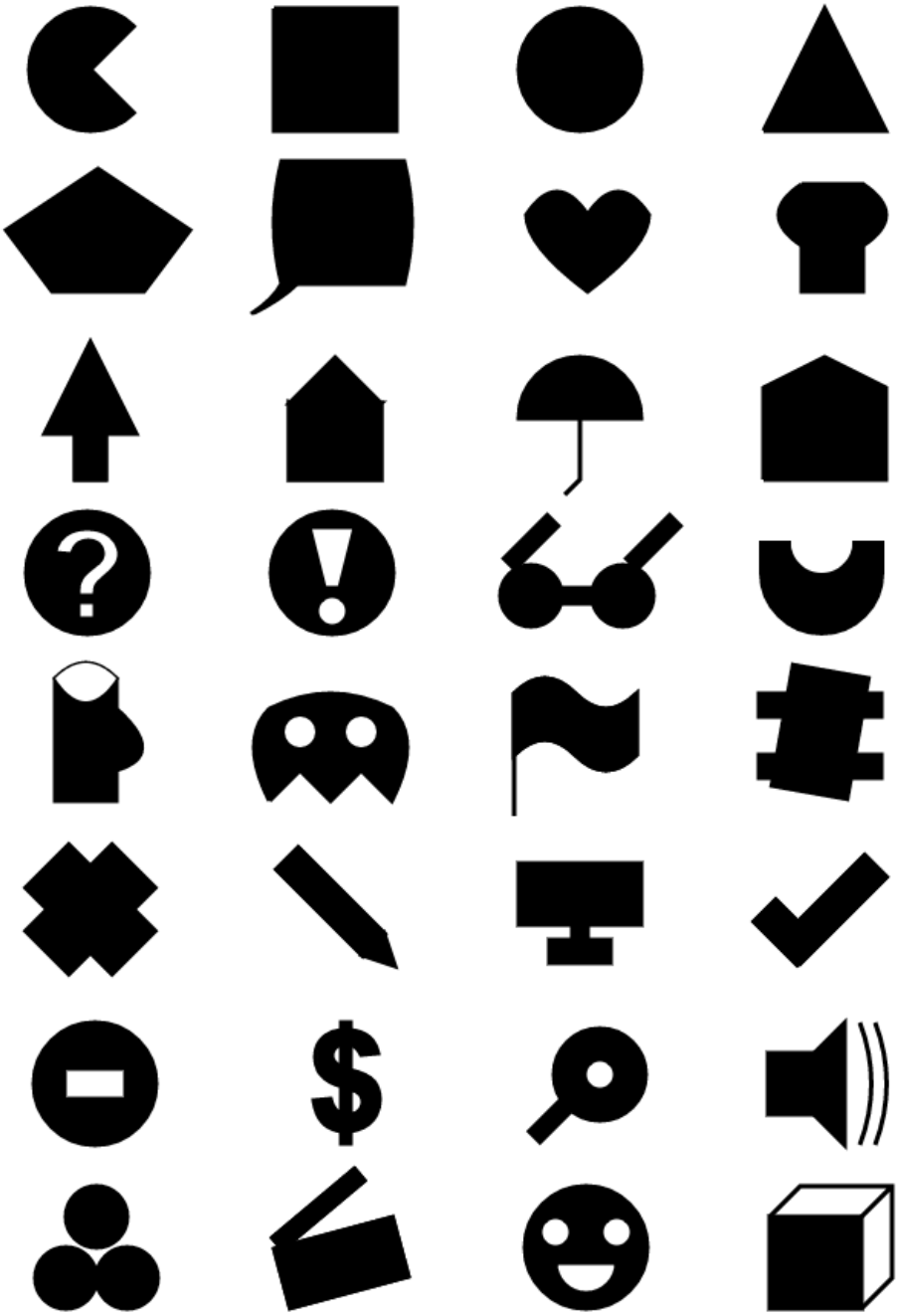

Figure 44: CLPPS(imp.) - New shapes with black color

8.2 Description of user study: Test of the improvements

The framework from the last study was used and the improvements shown are added. Also the explanation at the beginning was adapted to introduce the separating line and the types of patterns. The setup remains the same but the amount of slides decreased to 25. There are still 15 ‘easy‘ slides with 10 equal and five completely different pictures.
There are now 10 ‘hard‘ slides with only one difference consisting of :

- Two slides with shapes changed in one object
- Two slides with letters changed or exchanged (‘5‘ changed to ‘S‘, ‘8‘ and ‘B‘ are swapped)
- Two slides with colors changed or exchanged (black changed to grey, two colors swapped)
- Two slides with positions changed or exchanged (up changed to down, two objects swapping positions)
- Two slides with patterns changed or exchanged (vertical to horizontal lines and vice versa)

The study was conducted in the same company as the first study, but with different participants. The average age is 25 years, which is much younger than in the previous studies, but age is of no concern which this and the previous studies have shown. The youngest participant was 23 and the oldest 47. Again, designing experiences have no influence on the results. The results of the demographic survey are shown in the following tables.

	Participants	Age (average)
Women	8	25,25
Men	8	36,38

Table 12: Table of participants and age

Grade of computer knowledge	Amount
Novice	5
Advanced	9
Expert	1
No proposition	1

Table 13: Table of grade of IT knowledge

	Persons into ...
Graphic arts/Painting	2
Design	1

Table 14: Table of experience in graphic or design

8.3 Findings of the study

The results are shown in table 15. The four slides causing the most errors will be handled separately and shown on figures 45 - 48. Those four slides are again pattern- and position-related. The two errors with easy pairs are caused by false negative answers where participants either clicked the wrong button or thought they had seen a difference in the pictures.

Slide	Cumulative time	Average time	Errors	Percentage of success	Category
Figure 45	1:58:842	0:07:427	7	56	hard
Figure 46	1:38:248	0:06:141	6	62	hard
Figure 47	1:39:600	0:06:224	5	69	hard
Figure 48	1:16:046	0:04:752	4	75	hard
—	2:04:843	0:07:802	1	94	easy
—	1:18:405	0:04:900	1	94	hard
—	1:14:284	0:04:643	1	94	hard
—	2:29:987	0:09:373	1	94	easy

Table 15: Results of the study, each entry is per slide, sorted by error

	Easy pairs	Hard pairs	Combined
Average accuracy	99.16 %	85.00 %	93.5 %
Minimum accuracy	94.00 %	56.00 %	78.8 %
Average time per slide	06:306 s	05:303 s	05:905 s

Table 16: Table of times and accuracy of the second study

Compared to the first study (chapter 7.2), the amount of errors has decreased significantly. The average accuracy on hard pairs rised by 5% and the minimum accuracy increased from 31% to 56%. The combined average time per slide goes down by half a second to 5:905 s, compared to the first study. The introduction of a separating line into the pictures gave better results.

But regarding the results of Base32 of the first study, these values are still too low. The combined average accuracy is 93.5% which is 4% lower than the corresponding results of Base32. The pictures with hard pairs of patterns are still error-prone and the results have not improved very much. The goal is now to solve the problems with position- and pattern-related hard pairs to get a better accuracy than Base32.

The next four pages will show the slides with high errors in detail.

The first object moves from *down* to *up* position. Seven out of 16 participants failed to recognize this. Even though the horizontal line has some influence and the error rate decreased compared to the first study, the positions are still error-prone. This can be seen on the next slide which confirms the assumption.

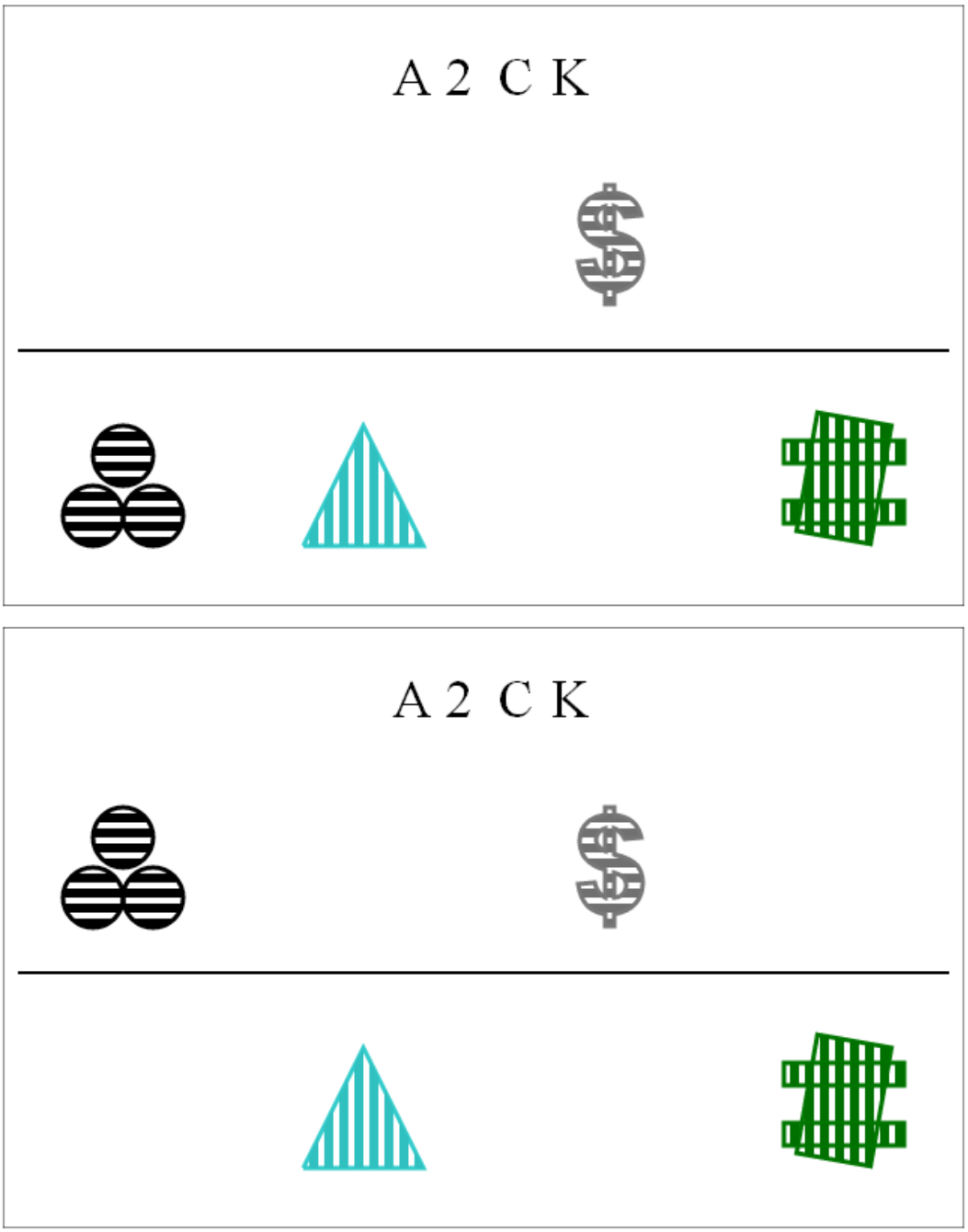

Figure 45: CLPPS(imp.) - Slide with high error rate on positions

On this slide with the second-highest error rate, the two objects in the middle change positions. Six out of 16 people did not detect this. The slightly lower error rate can be explained by now two objects changing positions instead of only one. The more objects change, the easier they are detected by a participant. But this is still not a good result.

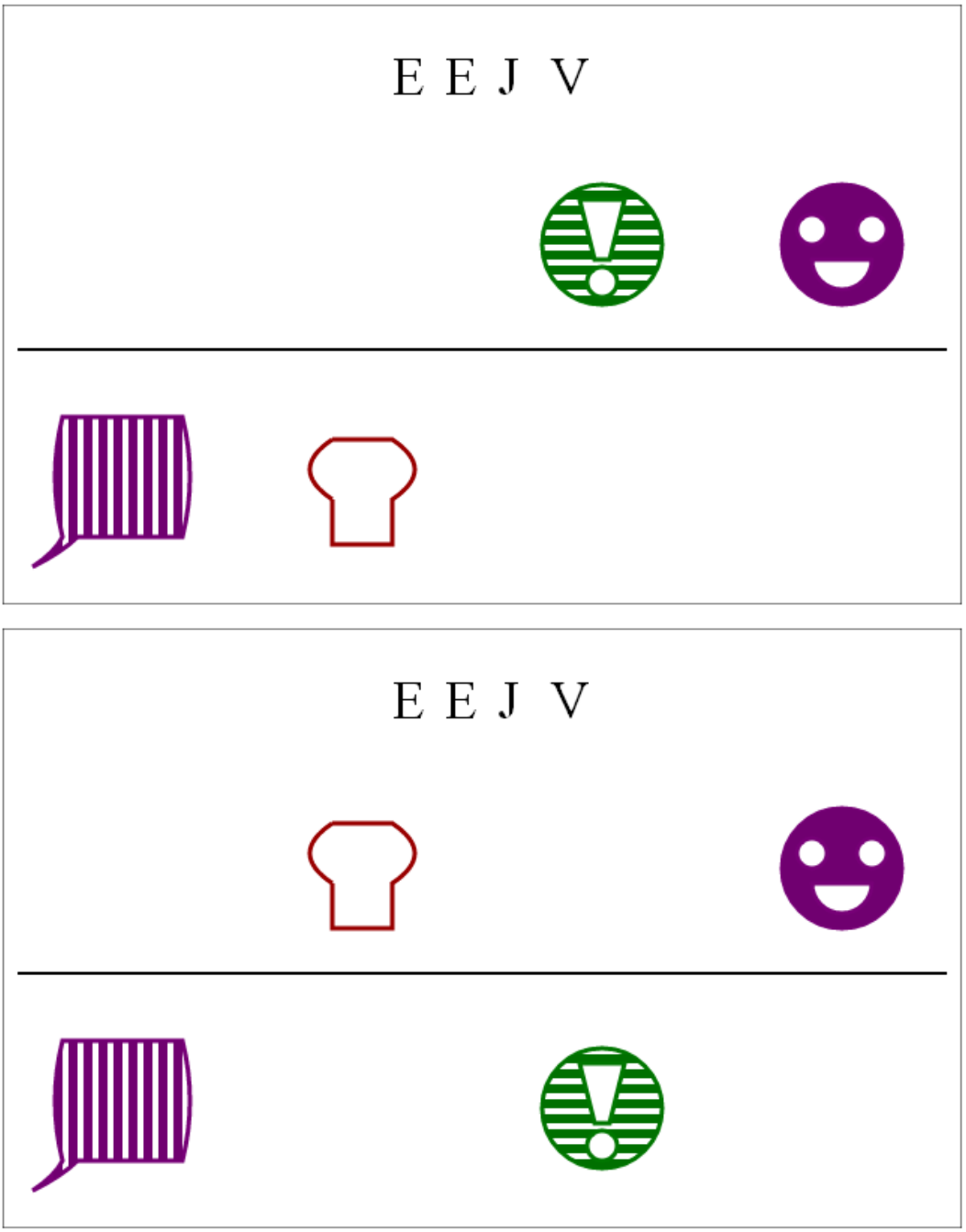

Figure 46: CLPPS(imp.) - Second slide with high error rate on positions

The pattern in the third object changed from vertical to horizontal lines. Still Five out of 16 made a mistake here. The patterns look too ‘equal‘ when comparing them too fast. This is no single case as can be seen on the next figure.

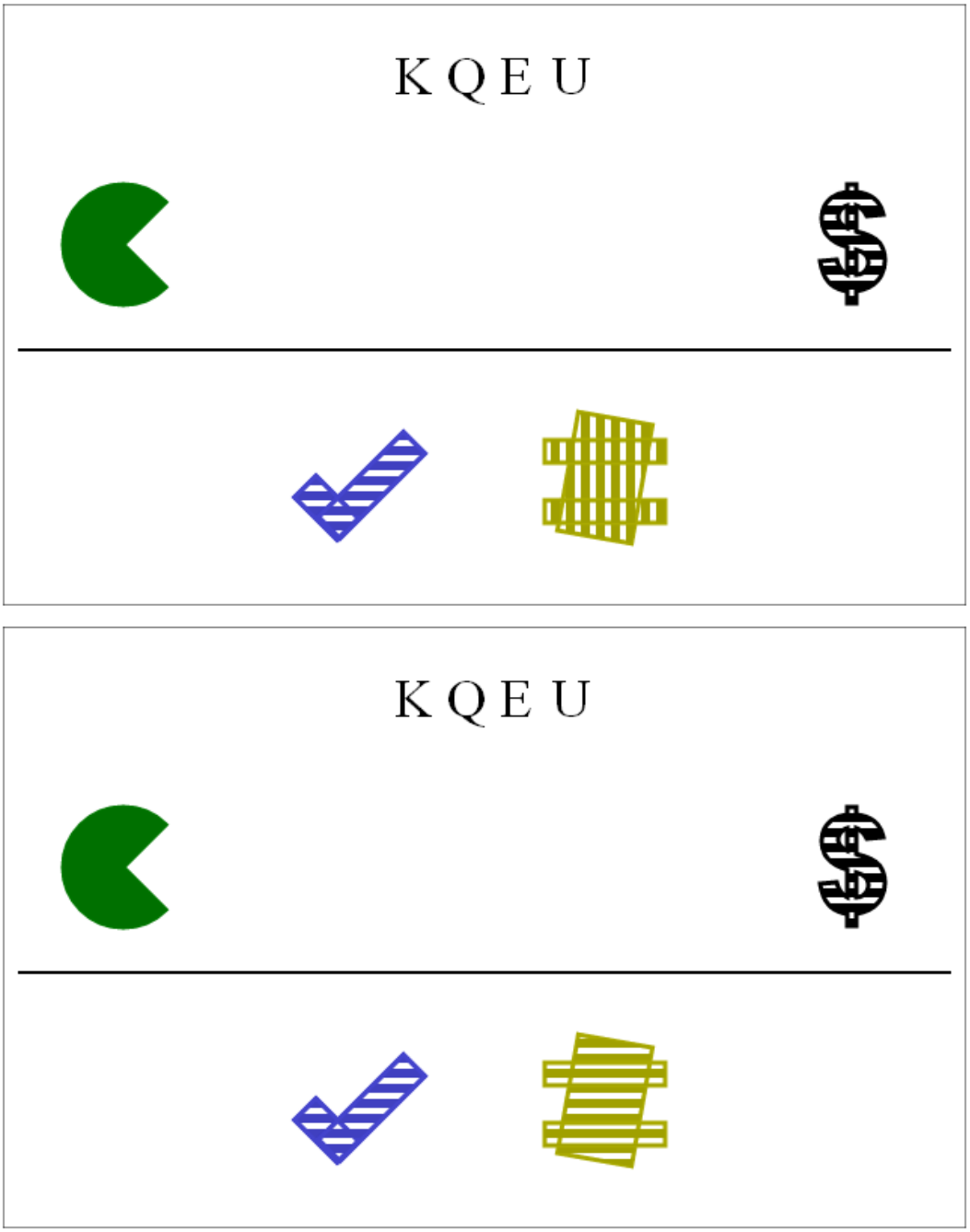

Figure 47: CLPPS(imp.) - Slide with pattern change

On this last slide the first two objects change their patterns. The first from vertical to horizontal lines and the second vice versa. Here four of 16 participants failed. This is according to the first two slides where two objects changing gave a better result. But patterns are still error-prone.
The other patterns that were tested - from blank to completely filled and the other way around produced no errors and were all detected.

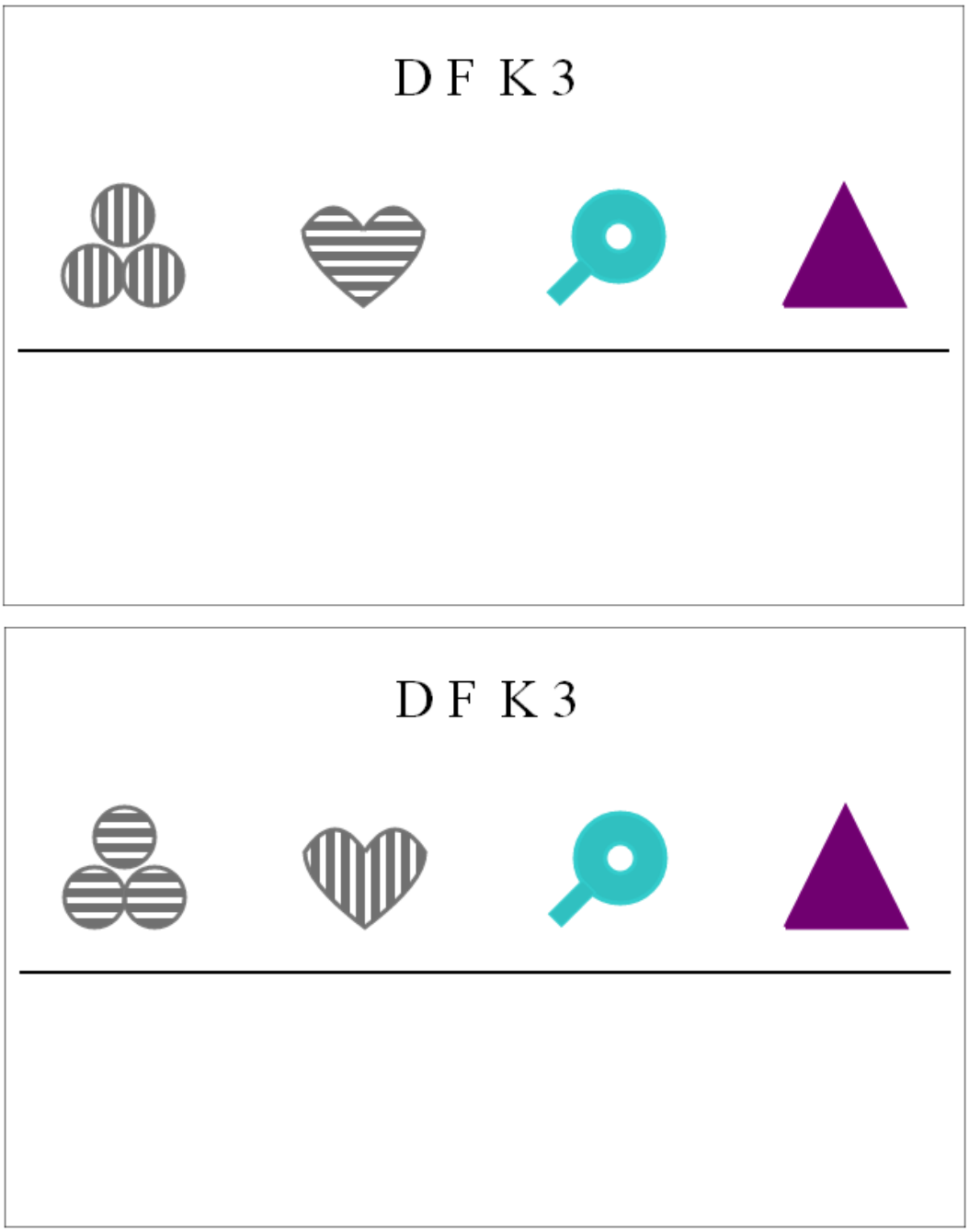

Figure 48: CLPPS(imp.) - Second slide with two patterns changing

8.4 Description of user study: CLPPS with side by side pictures

In this study, the pictures are the same as in the second study (chapter 8.2), but side by side. This will show if people are better at comparing the positions of the objects, if the pictures are side by side and not one being below another. It should also show, if this induces side effects like worse recognition of colors, shapes or patterns of the objects as well as letters. Tables 17 and 18 show the demographic informations.

The study was conducted in the company again with different participants and the same setting as the other studies. The average age was 34,5 years with 25 being the youngest and 55 the oldest participant. There are more people with an advanced grade in IT knowledge now, but as in accordance with the other studies, this had no impact on the results.

	Participants	Age (average)
Women	7	33,125
Men	7	35,875

Table 17: Table of participants and age

Grade of computer knowledge	Amount
Novice	1
Advanced	10
Expert	1
No proposition	2

Table 18: Table of grade of IT knowledge

The table regarding the experience in graphic or design is dropped, because the last studies have shown no relation between these experiences and the amount of errors a participant has made. There are not enough values to make a reliable proposition, if designing or painting helps in comparing pictures, since only five participants answered the questions about these experiences.

8.5 Findings of the study

Even though there are only 14 participants, the results show a variety of errors which have not been an issue before. The error rate concerning the positions of the objects has improved but there are errors of all kinds now. The combined accuracy of 96% is still very good though and only 1.25% lower than the Base32 result.

There are too few participants to actually make plausible propositions, but the results have shown two things. First of all the success rate on position related pictures increased from 59.38% from the second study to 85.71% in this study. So the up and down positions are much better to compare, when pictures are side by side. Secondly the average time per slide increased by one second, compared to the study without positions (chapter 9.2).

Nevertheless there are errors in all types of pictures now (positions, patterns, colors, letters) except changes of the shape of an object. Eight of the 14 errors were caused by two participants alone. This is a high deviation which could be ruled out of the results, if there were more participants.

Slide	Cumulative time	Average time	Errors	Percentage of success	Category
Figure 49	1:16:093	0:05:435	2	86	hard
Figure 50	1:09:502	0:04:964	2	86	hard
Figure 51	1:07:637	0:04:831	2	86	hard
—	1:18:804	0:05:628	2	86	hard
Figure 52	1:14:553	0:05:325	2	86	hard
—	2:09:680	0:09:262	1	93	easy
—	2:27:166	0:10:511	1	93	hard
—	0:59:286	0:04:234	1	93	easy
—	1:49:800	0:07:842	1	93	hard

Table 19: Results of the study with side by side pictures, each entry is per slide, sorted by error

	Easy pairs	Hard pairs	Combined
Average accuracy	99.05 %	91.43 %	96.00 %
Minimum accuracy	93.00 %	86.00 %	90.20 %
Average time per slide	07:036 s	06:328 s	06:753 s

Table 20: Results regarding accuracy and times

As the Figures show, the participants had most problems with objects exchanging attributes like colors, patterns or even letters. These slides will be shown and analyzed on the following pages. The fourth slide in the upper table with two errors is also position-related and will not be shown additionally.

There are two out of 14 participants not detecting the position changing of the first object. Combined with the other slide with position related pictures (not shown here) there are four errors in sum. The two participants not detecting the position change on the first slide also didn't detect it on the other slide. There is an accuracy of 85.7% on these slides. It rises to 92.85% though if the errors of both participants are counted as one.

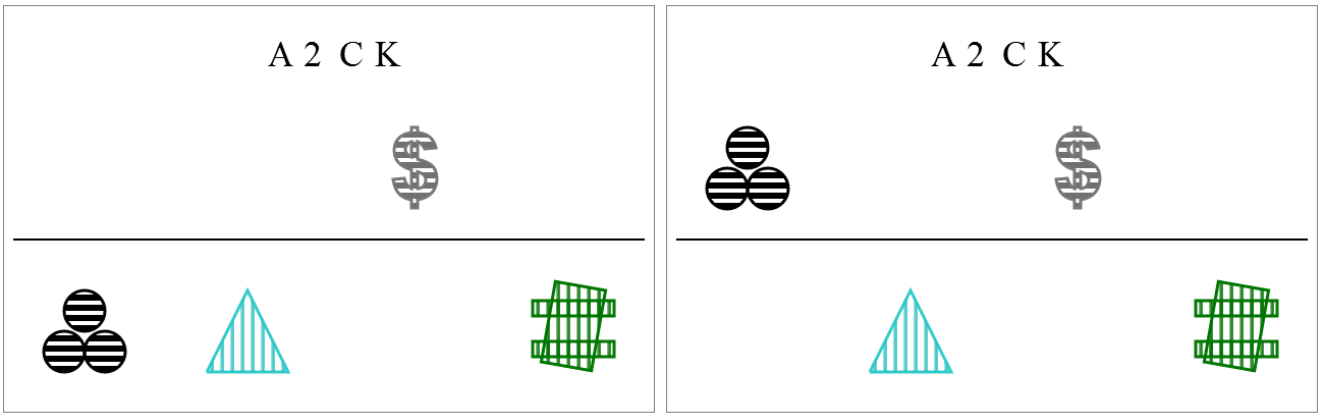

Figure 49: CLPPS(imp.) - First object moves from *up* to *down*

The two participants not detecting the position changes as mentioned above also did not detect the exchanged letters. Since these two participants caused eight of the 14 errors combined they could be ruled out under normal conditions. Due to the fact there are only 14 participants, this is not possible. A much higher number of participants might have given far better results because high deviations are not counting as much to the result as they do here.

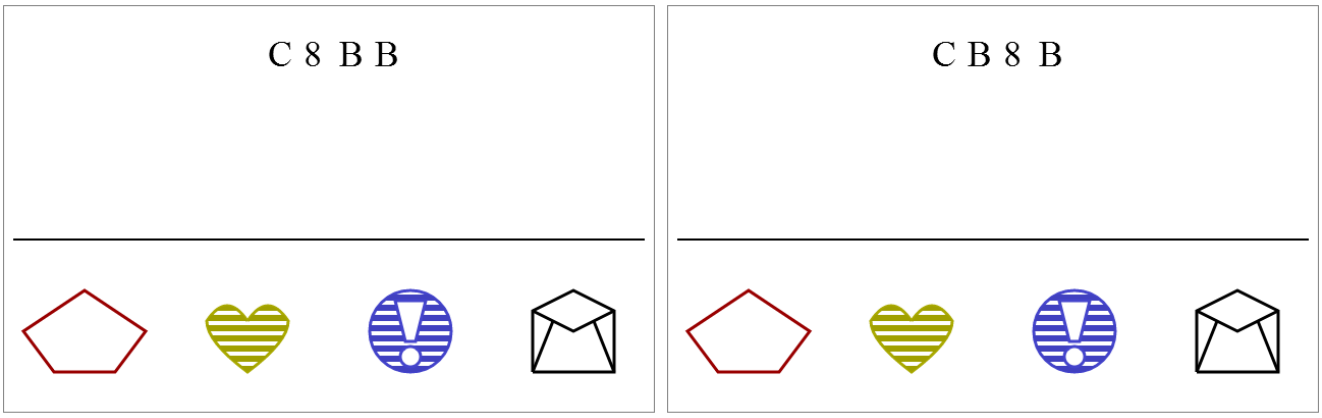

Figure 50: CLPPS(imp.) - Letters two and three exchanged

Two of the 14 participants failed to detect pattern changes in the objects. There are two other slides with only one object changing patterns which caused no errors at all. So if the same patterns are used at exchanged positions, people have difficulties on detecting the difference. This is only an assumption which cannot be proven by this insufficient number of participants, but will be worth looking into.

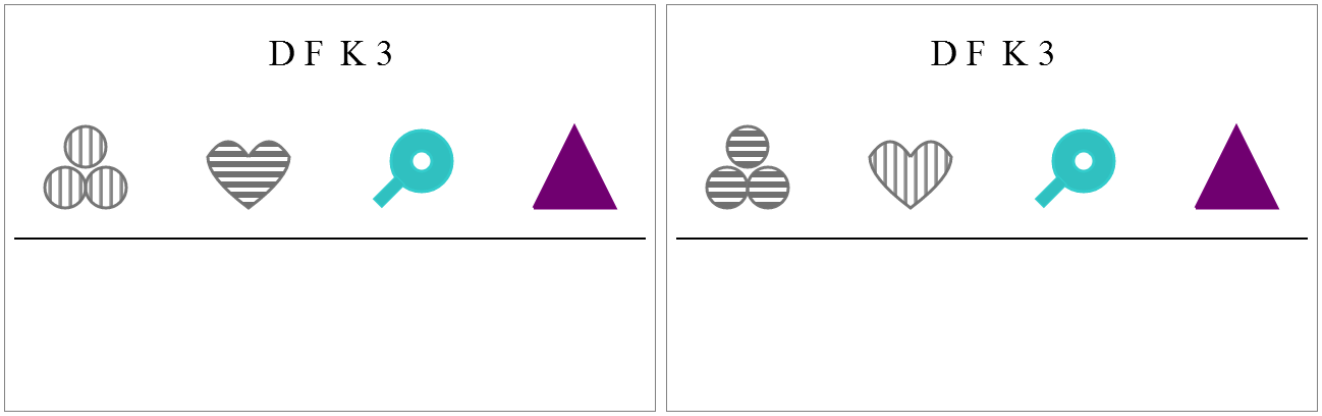

Figure 51: CLPPS(imp.) - Patterns of first two objects exchanged

The two participants not detecting the differences on the previous page (Figures 49 and 50) also did not detect the color changes in the objects below. Color changes have never been an issue before and it can be assumed this is not the normal case because of the amount of errors the two participants made.

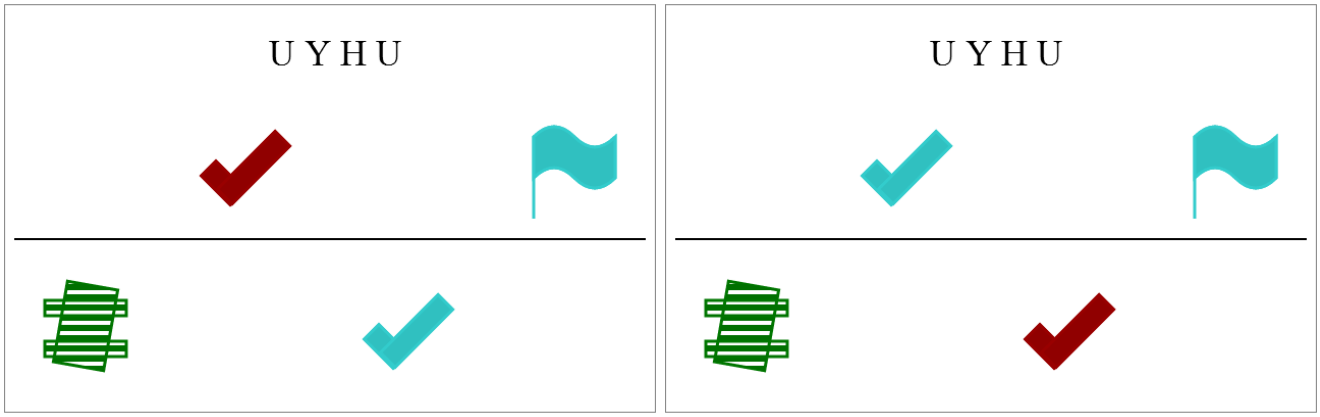

Figure 52: CLPPS(imp.) - Colors of objects two and three exchanged

9 Colors Letters Patterns Shapes

The study in chapter 8.2 has shown there are still two sources of errors. The first being the positions. Introducing a horizontal boundary line has improved the error rate but is still not satisfactory. The second issue was the two patterns with horizontal and vertical lines. These problems will be solved in this final approach, thus it is called ‘CLPS - Colors Letters Patterns Shapes‘. A study will then be conducted which will show the performance of this scheme.

9.1 Description of the approach

To cope with the problem of positions, the entropy was now cut by four bits to 60 bits and the positions of the objects are discarded. All objects are in one ‘line‘ now which should make them much easier to compare.

The line-strength of the pattern with vertical lines has been changed to a smaller size compared to the pattern with horizontal lines. This also should make these two patterns easier to compare.

The figure below gives an oversight of the changes made.

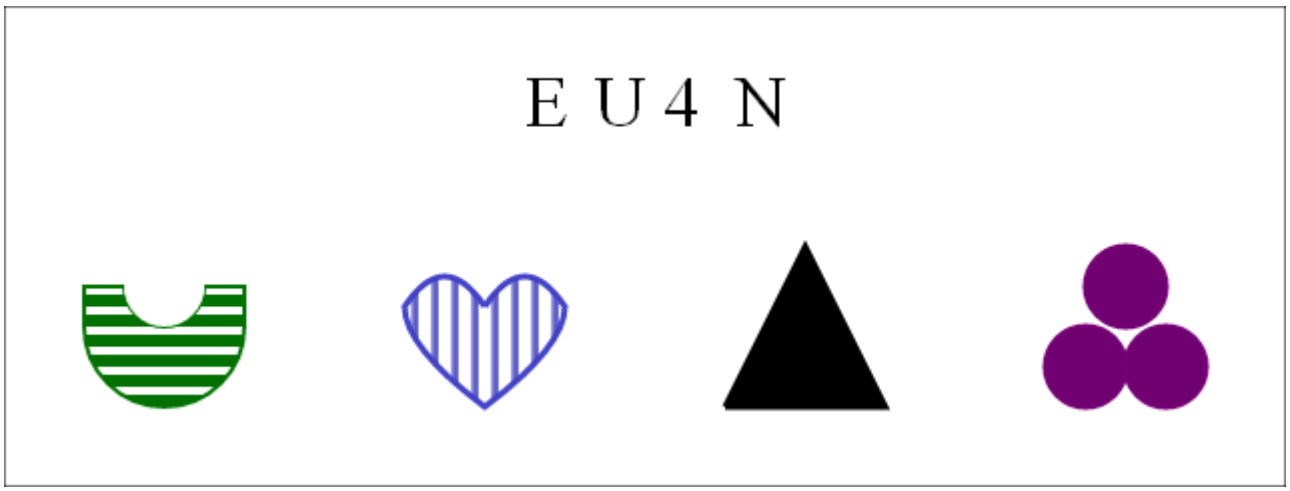

Figure 53: CLPS - Example of final changes

9.2 Description of user study: CLPS without positions

This study was conducted with 30 participants to get plausible results. The setup is the same as in the last study and the changes are applied as described. The study was conducted in the same company again with different participants. In addition 14 employees of the CASED (Center for Advanced Security Research Darmstadt) participated. Because of this, the grade of IT knowledge rises and the amount of experts is the highest of all studies.

The average age has increased slightly, the youngest participant being 24 and the oldest 43.

There are still 15 ‘easy‘ slides with 10 equal and five completely different pictures. The 10 ‘hard‘ slides with one or two differences are build as follows:

- Two slides with shapes changed in one object
- Two slides with letters changed (‘Z‘ changes to ‘2‘, ‘5‘ changes to ‘S‘)
- Two slides with colors changed (‘black‘ changes to ‘grey‘, ‘turquoise‘ to ‘blue‘)
- Four slides where patterns changed in one object (two slides) or two objects exchanged patterns (two slides)

There are four slides with patterns changing now, because there are no more positions and patterns caused a lot of errors in the last studies. All the pattern-related hard pictures have vertical and horizontal lines swapped because these caused most of the pattern-related errors in the previous studies. The following tables show the demographic survey.

	Participants	Age (average)
Women	14	38,29
Men	16	31,56

Table 21: Table of participants and age

Grade of computer knowledge	Amount
Novice	3
Advanced	16
Expert	10
No proposition	1

Table 22: Table of grade of IT knowledge

9.3 Findings of the study

As in the previous studies, the first three slides with errors will be shown and analyzed in detail on the following pages.

Slide	Cumulative time	Average time	Errors	Percentage of success	Category
Figure 54	2:20:773	0:04:692	6	80	hard
Figure 55	2:22:542	0:04:751	3	90	hard
Figure 56	2:54:134	0:05:804	2	93	hard
—	2:43:394	0:05:446	2	93	hard
—	2:05:792	0:04:193	1	97	hard
—	4:10:308	0:08:343	1	97	easy
—	1:33:417	0:03:113	1	97	hard
—	2:37:616	0:05:253	1	97	hard
—	3:39:158	0:07:305	1	97	easy
—	3:19:623	0:06:653	1	97	easy
—	4:00:923	0:08:030	1	97	easy
—	4:10:237	0:08:340	1	97	easy

Table 23: Results of the study, each entry is per slide, sorted by error

	Easy pairs	Hard pairs	Combined
Average accuracy	98.88 %	94.66 %	97.20 %
Minimum accuracy	97.00 %	80.00 %	90.20 %
Average time per slide	06:299 s	04:639 s	05:635 s

Table 24: Results regarding accuracy and times

Table 24 shows the minimum accuracy on hard pairs has increased from 31% of the first study to 80% by now. The average combined accuracy is 97.20% which is better than Random Art of the study [14] in chapter 3.6, which had the highest accuracy achieved so far. It is also as good as the result of Base32 (97.25%) of the study in chapter 7.2. The combined average time of 5:635 seconds per slide is the best time so far regarding all studies. It is also faster by one second compared to the Base32 times.

There is one negative aspect though. One slide with patterns changing in the first two objects (figure 54) still caused six errors. This is a much too high percentage which will need to be taken care of by further adaption of the patterns. The horizontal lines could be exchanged by diagonal lines to make the pattern be better distinguishable from the pattern with vertical lines.

There are also 14 participants of the CASED. These took the study after it was conducted in the company with 16 participants. Despite the fact there are naturally much IT-experts, the results degraded rapidly. Of the 16 participants of the company (mostly advanced in IT knowledge)

eight errors have been made. The 14 participants of the CASED (mostly experts) made 12 errors. This can be explained by the foreign exchange students as mentioned above but also by over-confidence. One participant did not even read the explanation and did in fact score badly. It is sometimes the case when people are professionals, they tend to get ‘sloppy‘. The human factor always plays a role in the results of a study.

As the results turned out to be very good and there could not be much done to further improve them, a second goal needs to be achieved. The study in the following chapter will be done to test, if people can actually remember pictures and point out the differences, when they see the picture again (or a similar) after a period of time.

The recognition rate increased from 69% of the second study to 80% concerning slides with the highest error rate on patterns. But still six out of 30 did not detect the first two objects exchanging patterns.

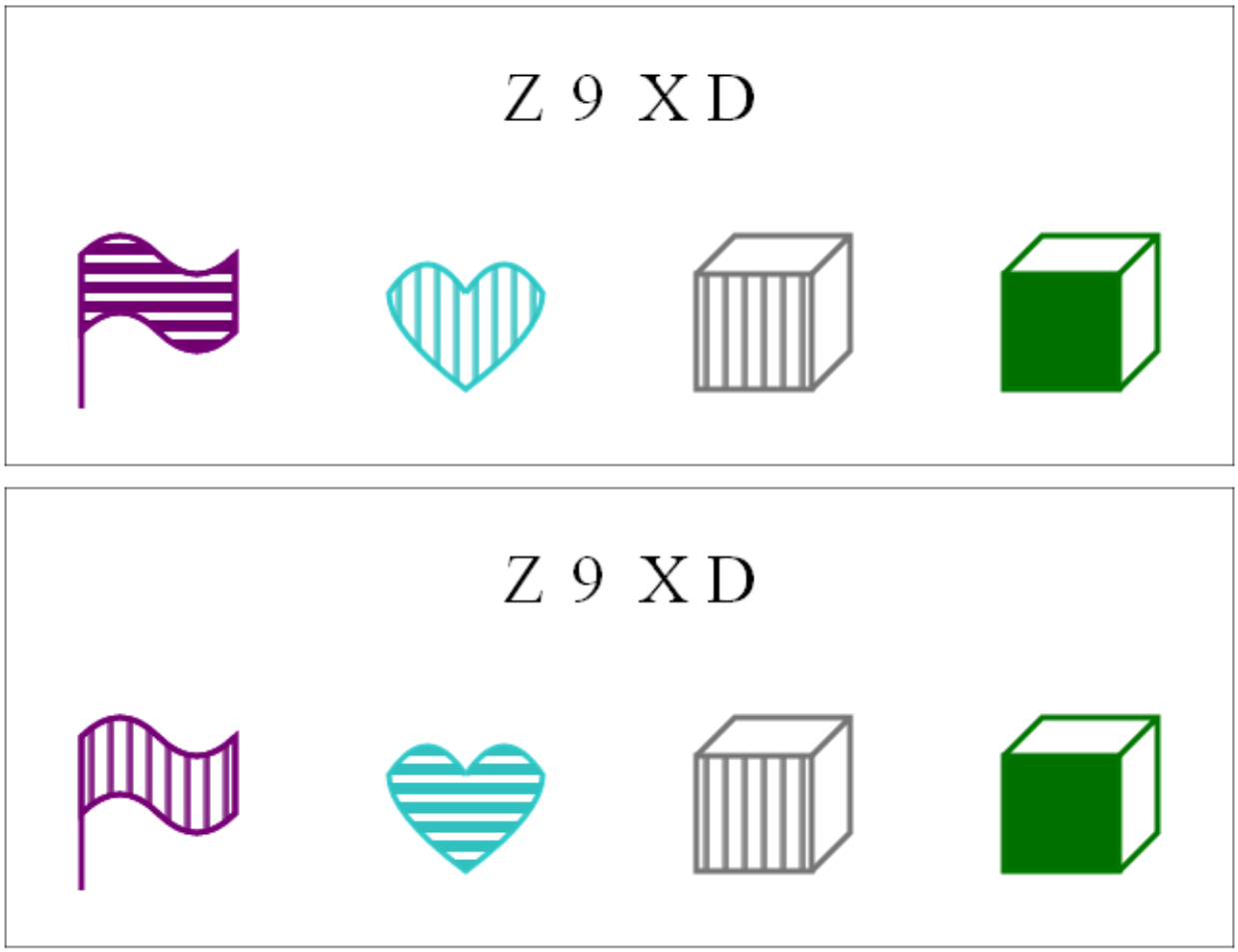

Figure 54: CLPS - Slide with two patterns changing

A few exchange students from asian countries, two from China and one from Japan, were asked to participate in the study. This was to test if there are issues regarding the four letters. This was proposed by the study in chapter 3.6, where people from asian countries had a lower recognition rate on english words and vice versa. Two of them actually made errors regarding the letters. Since pictures with letters exchanged have never been a problem in the first two studies, this can be seen as an issue when using the charset of the standard alphabet in other countries which have other charsets (e.g. Greece, Russia and various asian countries). Even though it is a far too small value to give a reliable proposition, this needs to be taken into account in further studies. In figure 55 three and in figure 56 two participants did not detect the letters changing. But an accuracy of 90% and 93% respectively is a good result, given the circumstances mentioned above.

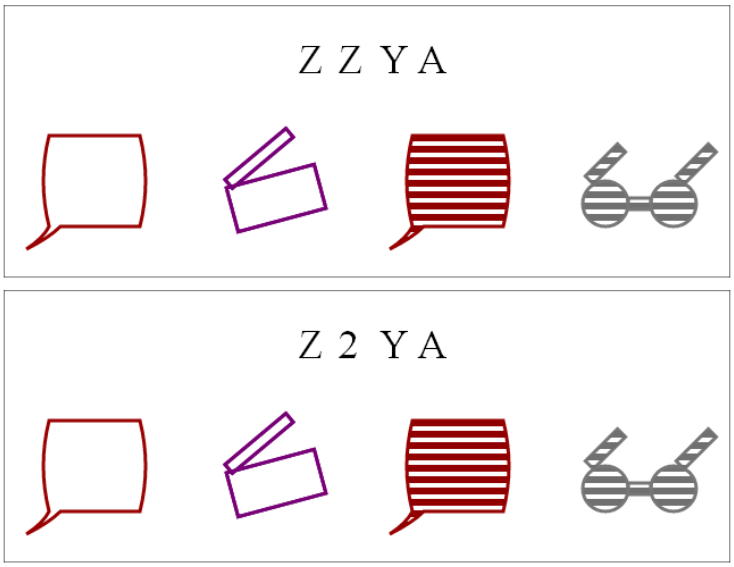

Figure 55: CLPS - First slide with letter changing

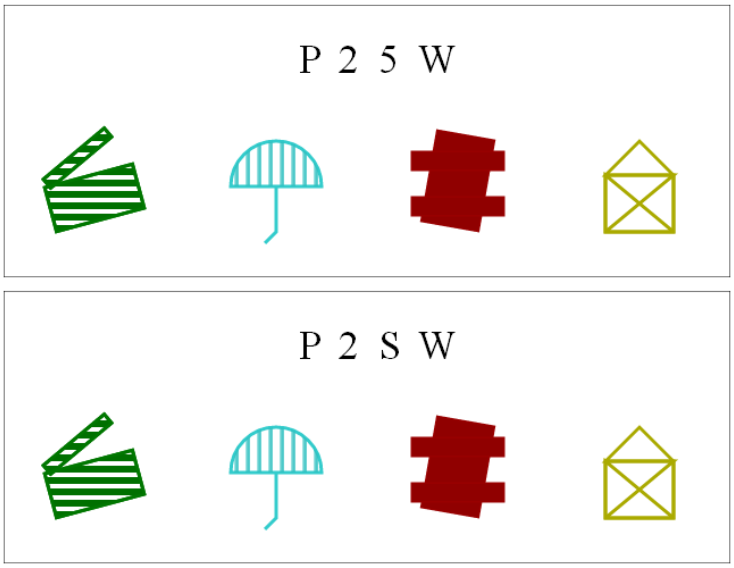

Figure 56: CLPS - Second slide with letter changing

10 Helios-simulating study of the CLPS approach

This study was designed to simulate a ‘real-world‘-environment of the Helios voting system [7]. It should show if people are actually able to remember a picture with a high entropy and state if the next picture seen is the same or different as the first one after a short period of time. This was realized by showing a picture to a participant on a slide. On the next slide the participant was asked a multiple choice question which he had to answer. On the following slide another picture was shown to the participant who had to decide if this picture was equal to the first. The whole process was repeated five times for each participant.

10.1 Design of the study

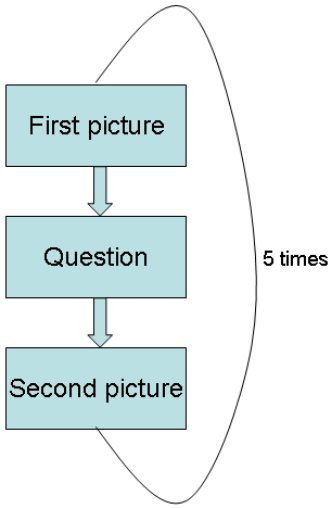

Figure 57: CLPS/Helios - Diagram of the process flow

There are 16 pictures of which one is chosen randomly as the starting picture. Each participant will see as second picture one of the following with a fixed percentage:

- 40% for an equal picture
- 20% for an obviously different picture
- 40% for a ‘hard‘ picture

The ’hard’ pictures have one or two differences in one attribute (colors, shapes, patterns, letters) to the corresponding first picture. For all four attributes there are four pictures summing up to 16 pictures. The pictures are built the same as in the last study without ‘positions‘ and contain an entropy of 60 bits.

The hard pictures are built as follows:

- Four slides with shapes changed (one object changes its shape, two objects swap shapes)
- Four slides with letters changed (‘Z‘ to ‘2‘, ‘5‘ to ‘S‘, ‘4‘ to ‘7‘, ‘F‘ to ‘G‘)
- Four slides with colors changed (‘black‘to ‘grey‘, ‘turquoise‘ to ‘blue‘, ‘purple‘ to ‘red‘, ‘yellow‘ to ‘green‘)
- Four slides with patterns changed (vertical to horizontal lines and vice versa, blank to filled)

The hard pairs are generated regarding the results of the last study. Colors and letters are used which could be easily confounded. Patterns are used which caused most of the errors in the previous studies.

The study was conducted partly in the company and partly with IT students which were asked in a computer pool room at the University of Darmstadt. Because of the fact that academic studies of IT-related subjects are dominated by men, there were much more male participants. The average age also decreased drastically because of the many students.

	Participants	Age (average)
Women	17	26,00
Men	28	27,32

Table 25: Table of participants and age

As table 26 indicates there is a mixture of novice, advanced and experts participating in the study now.

Grade of computer knowledge	Number
Novice	11
Advanced	14
Expert	14
No proposition	6

Table 26: Table of grade of IT knowledge

Since this is a totally new approach, the slides and errors made will be analyzed in detail. Every participant was shown five random pictures out of 16, so the participants cannot easily be compared. Instead there will be an analysis of the type of pictures and which pictures caused the most errors.

10.2 Findings of the study

As table 27 shows, all 45 participants made 45 errors in sum leading to an average success percentage of 80%. Three participants made three errors meaning they had an accuracy of only 40%, which is very low.

	Number of participants	Amount of errors made
	14	0
	20	1
	8	2
	3	3
sum	45	45

Table 27: Table of number of errors

Even though a picture should be easy to detect if it is completely different to the corresponding, two people made an error on a sequence with obviously different pictures leading to a success rate of 95,6%.

	Equal pairs	Hard pairs	Obviously different pairs	Combined
Average percentage	83,3%	68,9%	95,6%	80%

Table 28: Table of success rate of the different slides

If we assume that obviously different pictures are easy to detect with a probability of 100% and 'hard' and 'equal' pictures are detected with a rate of 50% if the participant always just guesses the answer, then we have a guessing rate of: $\frac{4}{5} \cdot 50\% + \frac{1}{5} \cdot 100\% = 60\%$. The success rate of 'hard' pairs only is just 8,9% higher than the rate for guessing. The average success rate of 80% for all types of pictures is 20% higher than the guessing rate but still far from good.

Table 29 gives an overview of the errors made on a special type of slide and the average times for each type of slide. While the amount of errors are only counted for slides on which actually errors occured, the times are listed for all slides of that type, whether erroneous or not. If the entry is zero, no errors have been made for that particular type of slide. It can be seen and is in accordance with the last studies, that slides with exchanged letters give the best recognition rate. The rate for slides with 'hard' pictures shape- or color-related is very low. Pictures with these attributes will be analyzed on pages 68 and 69.

Slides related to...	Number of slides	Amount of errors	Percentage	Time per slide
Colors	50	18	64%	21,420s
-equal	17	6	64,7%	20,072s
-hard	29	11	62%	23,243s
-different	4	1	75%	17,538s
Letters	26	5	80,8%	18,110s
-equal	17	3	82,4%	18,807s
-hard	9	2	77,8%	19,102s
-different	0	0	0%	19,079s
Patterns	46	11	76,1%	22,552s
-equal	22	3	86,4%	24,143s
-hard	24	8	66,6%	22,501s
-different	0	0	0%	17,916s
Shapes	35	11	68,6%	20,249s
-equal	16	3	81,3%	18,259s
-hard	15	7	53,3%	21,688s
-different	4	1	75%	21,417s

Table 29: Table of picture types and errors

In accordance with the study in the related works section (chapter 3.6), age has an impact on the results. As can be seen in table 30 the group of age 36-57 has a much higher error rate.

Age	Amount	Errors	Rate
19-35	34	30	0,88
36-57	11	15	1,36

Table 30: Table of age - error coherence

In this study men had a better performance. This is not confirmed by the other studies where the results are balanced.

Gender	Amount	Errors	Rate
Female	17	22	1,29
Male	28	23	0,82

Table 31: Table of gender - error coherence

IT knowledge has a minor influence on the results. The better a person handles computer related tasks and is adapted to the usage of computers the less likely he is to be nervous and make errors. But also the age has an influence since most experts are students which are younger than the employees in the company.

IT-Knowledge	Amount	Errors	Rate
Novice	11	13	1,18
Advanced	14	13	0,93
Expert	14	11	0,79
No proposition	6	8	1,33

Table 32: Table of IT knowledge - error coherence

The following table shows the more errors a participant has made the less time he needed. The average time needed for one slide is 11 seconds which is not very fast. But 60 bits of information have to be remembered.

Errors made	Average time per slide
0	11,628 s
1	11,029 s
2	07,200 s
3	07,918 s

Table 33: Table of time - error coherence

There were no indications of other time related issues. Participants who took more time above average for the first slide didn't get better results than the others. Also participants who needed an extreme amount of time for each slide didn't score better than the rest on an average.

On figure 58, the color of the first object changes from purple to red. This was not detected by four out of nine. The change of color of the third object in figure 59 has not been seen by three out of seven participants. Even though there are not enough values, the results give a hint that color changes might be difficult to remember when the colors are alike.

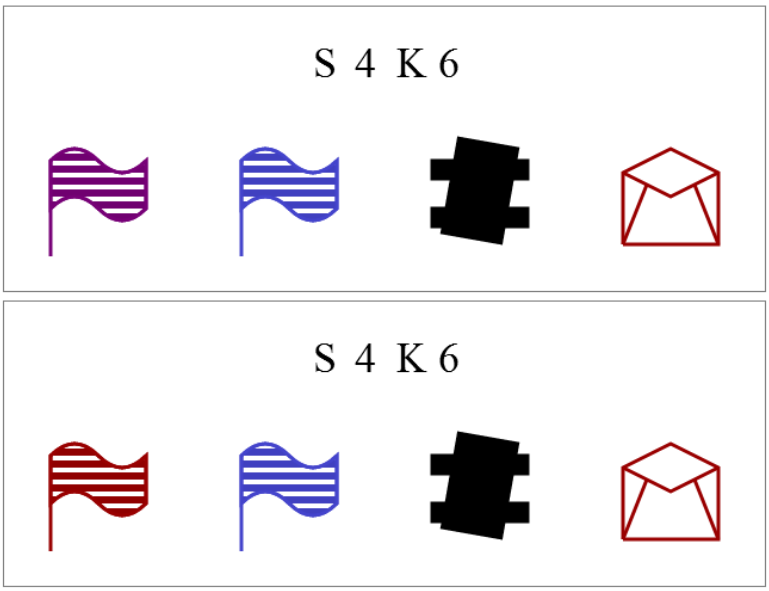

Figure 58: CLPS/Helios - Picture with color changing in the first object

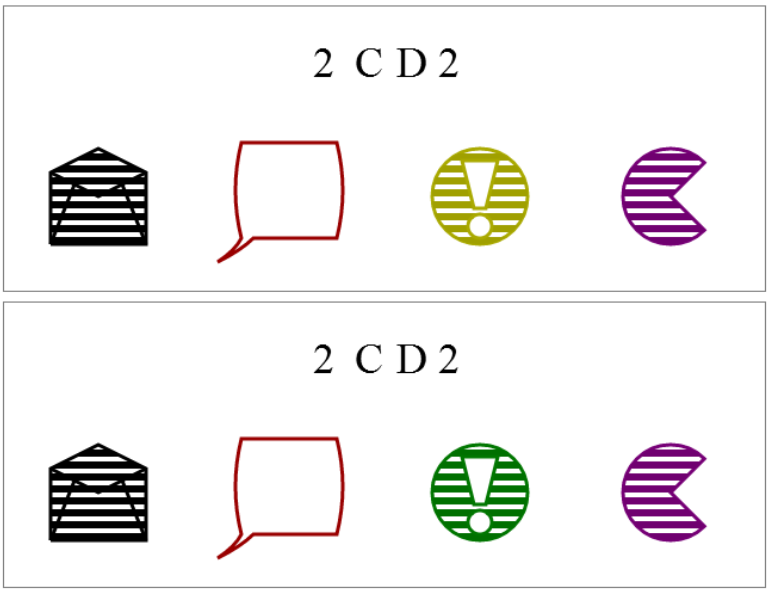

Figure 59: CLPS/Helios - Picture with color changing in the third object

Even worse results came with shapes changing in one object. These two pictures depict the worst cases. In the first case (figure 60) three of five did not remember the change in the fourth object, in the second (figure 61) three of six failed to remember and identify the picture correctly. However, the third slide (not shown here) with the shape change in the first object was answered correctly by three of four participants and the fourth slide, where objects three and four exchanged places had a 100% success rate.

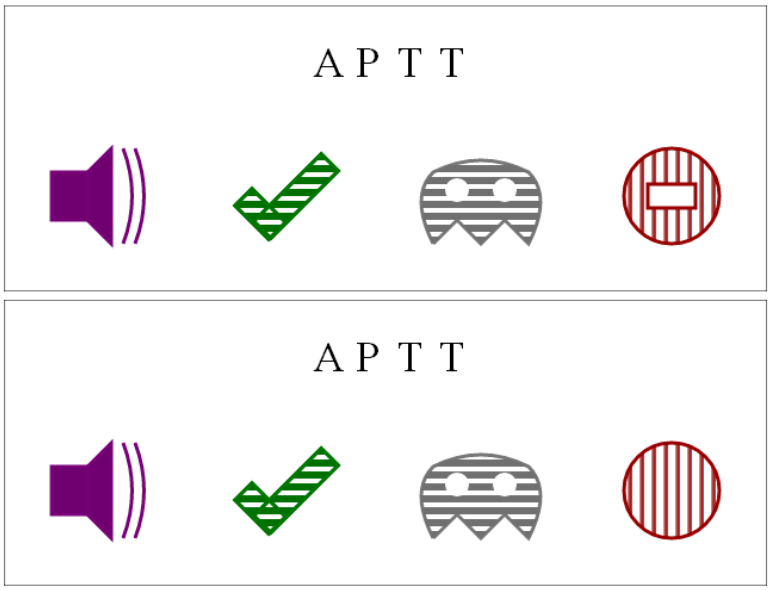

Figure 60: CLPS/Helios - Picture with shape changing in the last object

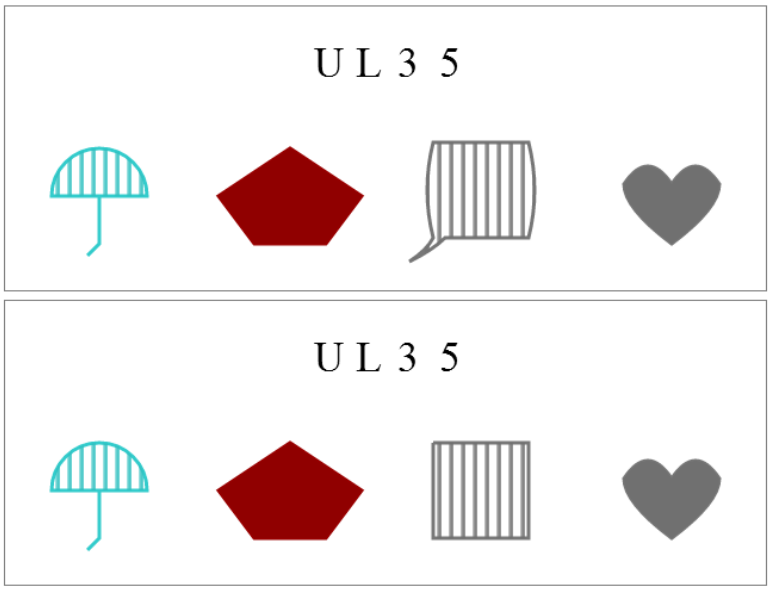

Figure 61: CLPS/Helios - Picture with shape changing in the third object

The study has shown that people have problems remembering pictures with a high entropy. Even though 14 participants did not make an error and remembered all five pictures, 31 did at least make one mistake. It seems really hard to remember 60 bits of entropy in a picture in a stressful situation and under the pressure of time. Some participants stated that they did not remember what the first picture looked like after answering the question. It can be proposed that a higher entropy would lead to more errors and an even lower success rate.

Also the average time needed per slide is at about 20 seconds. It would be faster to write down the corresponding hash value or to print out the picture than trying to remember it.

Age is also a major concern. The results have shown that the younger the group of participants, the better the results. Younger people are better and faster at remembering things. If the findings are transformed to a 'real' application, the age needs to be considered. More and more elder people are using the Internet and given the findings of age-error coherence, this might be a problem.

11 Conclusion and future work

The goal of the thesis was to visualize the whole 160 bits of entropy of SHA1. The first and second approaches have shown that this is hardly to achieve, so the entropy has been lowered to 60 bits for the third approach.

The studies have shown that it is possible to find a scheme, where hash values can be substituted by pictures. In the study of the CLPS approach (chapter 9.2) an average accuracy on hard pairs of 94.66% was achieved which is nearly the same accuracy (94.375%) as in my Base32 study in chapter 7.2 and a better value than the results of the study in chapter 3. Also the pictures in my work carry 60 bits of entropy which is more than twice the amount compared to the 25 bits of the other study.

The last study (chapter 10) simulating the Helios environment did not come up with good results. People do have problems remembering this much information in a short period of time. The average time per slide was 11 seconds. It would be faster to print or even write down the information of the first picture and compare it against the other. The average success rate of 68.9% for hard pairs is only slightly better than guessing. It has to be taken into account, that there is a 80% rate of hard pairs.

People are very accustomed to numbers and letters of the alphabet. It is easier to compare short sequences of hexadecimal bytes and find differences than comparing pictures which one has never seen before. If people would ‘learn‘ how pictures in a scheme look like and what differences could be between them, the whole process would be faster and more reliable.

When it comes to remembering information, there is a point where people cannot digest more information in only a short period of time. I think the difficulty is the same, whether one needs to remember 60 bits of information in form of sequences of numbers or encoded in pictures.

For future work in regard to my thesis, there can be done a lot to improve the pictures. I propose to use ‘real‘ pictures like photographs instead of drawn ones, because there is a higher recognition rate when seeing pictures one has seen before. The colors used can be maintained, since the studies have shown no abnormalities or higher error rate regarding the colors. The problem will be the amount of entropy the pictures provide. 1000 pictures only carry 10 bits of entropy ($2^{10} = 1024$). If colors and patterns are added to the picture, the entropy might rise to 16 bits. So at least four pictures are needed to get an combined entropy of 64 bits. All pictures have to be completely different which rules out pictures of (e.g. famous) persons because one will always find persons who look alike.

Another way would be to encode the hash into pictures with a ‘story‘ or ‘sentence‘ (connected pictures). As an example: ‘A man‘ (picture one, noun), ‘walks‘ (picture two, verb), to a ‘big‘ (picture three, adjective), ‘house‘ (picture four, noun), at the ‘sea‘ (picture five, noun), and so on ... This needs more research than could be provided in this diploma thesis.

Also worth consideration would be pictures changing over time. Objects could change positions or colors after a given period of time. But there is a lot of research to do if people are able to remember those differences and if they do not get confused with the correct order of the pictures.

Another thing to research is to inject data the user already knows (like a password) into the encoding of the hash. This will increase the entropy while maintaining the accuracy.

As was mentioned before, an increasing entropy will affect times and accuracy to a point where the cost-benefit ratio is too bad and the scheme will become useless.

References

[1] Last visited on 02/05/2012. [Online]. Available: http://www.jacobsschool.ucsd.edu/news/news_releases/release.sfe?id=873;http://www.physorg.com/news169133727.html

[2] Last visited on 02/05/2012. [Online]. Available: http://www.securityweek.com/man-middle-attacks-voting-machines-vote-early-often-and-why-not-vote-remotely

[3] Last visited on 02/05/2012. [Online]. Available: http://www.itl.nist.gov/fipspubs/fip180-1.htm

[4] Last visited on 02/05/2012. [Online]. Available: http://usg.princeton.edu/index.php?option=com_content&view=article&id=230:guide-to-helios&catid=78:elections&Itemid=115

[5] Last visited on 02/05/2012. [Online]. Available: http://www.random-art.org

[6] Last visited on 02/05/2012. [Online]. Available: http://www.onlineumfragen.com/

[7] B. Adida, "Helios: Web-based open-audit voting," in *SS'08 Proceedings of the 17th conference on Security symposium*, 2008, pp. 335–348. [Online]. Available: http://dl.acm.org/citation.cfm?id=1496734;http://www.heliosvoting,org

[8] D. S. Adrian Perrig, "Hash visualization: a new technique to improve real-world security," Tech. Rep., 1999. [Online]. Available: http://www.cs.berkeley.edu/~dawnsong/papers/randomart.pdf

[9] J. Benaloh, "Simple verifiable elections," in *EVT'06 Proceedings of the USENIX*, 2006. [Online]. Available: http://dl.acm.org/citation.cfm?id=1251008

[10] e. a. Bryan Ford, Jacob Strauss, "Persistent personal names for globally connected mobile devices," in *OSDI '06 Proceedings of the 7th symposium on Operating systems design and implementation*, 2006, pp. 233–248, last visited on 05/25/2012.

[11] S. D. Carl Ellison, "Public-key support for group collaboration," in *ACM Transactions on Information and System Security*, vol. 6, no. 4, 2003, pp. 547 – 565, last visited on 02/05/2012.

[12] O. dictionaries, 2012, last visited on 02/05/2012. [Online]. Available: http://oxforddictionaries.com/words/how-many-words-are-there-in-the-english-language

[13] M. T. Goodrich and M. S. et al, "Loud and clear: Human-verifiable authentication based on audio," in *Proceedings of the 26th IEEE International Conference on Distributed Computing Systems*. IEEE, 2006, p. 10, last visited on 05/25/2012.

[14] A. Hsu-Chun Hsiao; Yue-Hsun Lin; Studer, "A study of user-friendly hash comparison schemes," in *Computer Security Applications Conference, 2009. ACSAC '09. Annual*, 2009, pp. 105 – 114. [Online]. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=5380523

[15] S. Josefsson, 2006, last visited on 06/05/2012. [Online]. Available: http://www.ietf.org/rfc/rfc4648.txt

[16] F. Karayumak, M. Olembo, M. Kauer, and M. Volkamer, “User study of the improved helios voting system interfaces,” in *Socio-Technical Aspects in Security and Trust (STAST), 2011 1st Workshop on*, 2011, pp. 37 – 44. [Online]. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=6059254

[17] Y.-H. Lin, A. Studer, H.-C. Hsiao, J. M. McCune, K.-H. Wang, M. Krohn, P.-L. Lin, A. Perrig, H.-M. Sun, and B.-Y. Yang, “Spate: small-group pki-less authenticated trust establishment,” in *Proceedings of the 7th international conference on Mobile systems, applications, and services*, ser. MobiSys ’09. New York, NY, USA: ACM, 2009, pp. 1–14. [Online]. Available: http://doi.acm.org/10.1145/1555816.1555818

[18] E. Uzun, K. Karvonen, and N. Asokan, “Usability analysis of secure pairing methods,” In Usable Security (USEC, Tech. Rep., 2007.

[19] P. C. van Oorschot and M. J. Wiener, “Parallel collision search with application to hash functions and discrete logarithms,” in *CCS ’94 Proceedings of the 2nd ACM Conference on Computer and communications security*, vol. 2, 1994, pp. 210 – 218, last visited on 05/25/2012.

[20] H. Yee, “Perceptual image diff,” last visited on 02/05/2012. [Online]. Available: http://pdiff.sourceforge.net/